**Jimma University**

**Institute of Technology**

**School of Computing**

**Department of Computer Science**



**Data mining Assignment: Reinforcement Learning in data mining**

**Group Member**

**Name**  **Idno**

1. Miniyahil Kebede…………………………………Ru0572/08
2. Bizuayehu Lemmi………………………………….Ru0568/08

# Reinforcement Learning in Data Mining

1. **Introduction**

* Reinforcement learning is a type of machine learning. It allows the machine or software agent to learn its behavior based on feedback received from the environment.
* In machine learning, learner knows which action is to be taken. On the other hand, in reinforcement learning, the learner doesn't know the action and it tends to discover which action will give the most reward signal.
* It is different form supervised learning, since it is not adequate for learning from interaction. In this case, the agent (user) has to act and get learned through experience.
* Reinforcement learning is based on goal-directed learning from interaction.
* It is about taking suitable action to maximize reward in a particular situation.

**Example:**

* In chess game, player makes a move based on planning move, expecting possible replies of opponents. Then, the player takes immediate and spontaneous judgment and plays the move.
* In this example, the player uses his experience to improve the performance and evaluate positions to improve his play over the period of time.

#### **Elements of Reinforcement Learning**

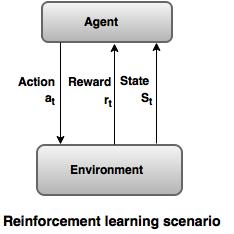
Other than the **agent** and the **environment**, one can identify four main sub elements of RL

1. **Policy —**is a mapping from perceived states of the environment to actions to be taken when in those states. The policy is the core of a reinforcement learning agent in the sense that it alone is sufficient to determine behavior. It maybe stochastic, specifying probabilities for each action.
2. **Rewards —**on each time step, the environment sends to the reinforcement learning agent a single number called reward. The agent’s sole objective is to maximize the total reward it receives in the long run. The reward signal thus defines what are the good and the bad signals for the agent. It may be a stochastic function of the state and action.
3. **Value Function —**specifies, roughly, the value of a state is the total amount of reward an agent can expect to accumulate over the future, starting from that state. Whereas rewards determine the immediate, intrinsic desirability of the environmental states, values indicate the long-term desirability of states after taking into account the states that are likely to follow and the rewards available in those states. For example, a state might always yield a low immediate reward but still have a high value because it is regularly followed by other states that yield high rewards, or the reverse could be true.
4. **Model of the environment —**this mimics the behavior of the environment that allows inferences to be made about how the environment will behave. For example, given a state and an action, the model might predict the resultant next state and next reward. Methods for solving reinforcement learning problems that use models are called model-based methods, as opposed to simpler model-free methods, trial and error learners.

Rewards are in a sense primary, whereas values, as predictions of rewards, are secondary. Without rewards there could be no values, and the only purpose of estimating values is to achieve more reward. Nevertheless, it is values which we are most concerned when making and evaluating decisions.

## **Reinforcement and environment function**

* It uses the knowledge acquired while exploring the action leads to learning through rewards or penalties.
* Rewards are related to specific actions and value function is the collective effect.
* Environment needs to be modeled to receive the correct responses so that it can accept the inputs from changing scenarios and finally produce the optimized value.



1. **Basic algorithms of Reinforcement learning**

Reinforcement Learning algorithms are widely used in gaming applications and activities that require human support or assistance. Usually, an RL setup is composed of two components, an agent, and an environment. The environment refers to the object that the agent is acting on, while the agent represents the RL algorithm. The environment starts by sending a statement to the agent, which then based on its knowledge to take an action in response to that state. After that, the environment sends a pair of next state and reward back to the agent. The agent will update its knowledge with the reward returned by the environment to evaluate its last action. The loop keeps going on until the environment sends a terminal state, which ends the episode.

Some of commonly used RL algorithms are:

**Q-Learning:**

This algorithm is most used and basic reinforcement algorithm, this uses the environment rewards to learn over time, the best action to take in a given state. In the above implementation, we have our reward table “P” from where the agent will learn from. Using the reward table it chooses the next action if it’s beneficial or not and then they update a new value called Q-Value. This new table created is called the Q-Table and they map to a combination called (State, Action) combination. If the Q-values are better, we have more optimized rewards.

For example, if the taxi is faced with a state that includes a passenger at its current location, it is highly likely that the Q-value for pickup is higher when compared to other actions, like dropoff or north.

Q-values are initialized to an arbitrary value, and as the agent exposes itself to the environment and receives different rewards by executing different actions, the Q-values are updated using the equation:

https://cdn-images-1.medium.com/max/800/1*1Ii88l4M_Vgv9YjaiAFOEA.png

Here comes a question, how to initialize this Q-Values and how to calculate them, for that we initialize the Q-values with arbitrary constants and then as the agent exposes to the environment it receives various rewards by executing different actions. Once the actions are executed, the Q-Values are executed by the equation.

Here Alpha and Gamma are the parameters for the Q-learning algorithm. Alpha is known as the learning rate and gamma as the discount factor both the values range between 0 and 1 and sometimes equal to one. Gamma can be zero while alpha cannot, as the loss should be updated with some learning rate. Alpha here represents the same which is used in supervised learning. Gamma determines how much importance we want to give to future rewards.

Below is the algorithm in brief,

* **Step 1:** Initialize the Q-Table with all zeros and Q-Values to arbitrary constants.
* **Step 2:**Let the agent react to the environment and explore the actions. For each change in state, select any one among all possible actions for the current state (S).
* **Step 3:** Travel to the next state (S’) as a result of that action (a).
* **Step 4:** For all possible actions from the state (S’) select the one with the highest Q-value.
* **Step 5:** Update Q-table values using the equation.
* **State 6:** Change the next state as the current state.
* **Step 7:** If goal state is reached, then end and repeat the process.

#### **State-Action-Reward-State-Action (SARSA):**

#### SARSA, another popular RL algorithm, is quite similar to Q-learning. The key difference between SARSA and Q-learning is that SARSA is an on-policy algorithm. It implies that SARSA learns the Q-value based on the action performed by the current policy instead of the greedy policy.

#### The Sarsa algorithm is an On-Policy algorithm for TD-Learning. The primary difference between it and Q-Learning is that the maximum reward for the next state is not necessarily used for updating the Q-values. Instead, a new action, and therefore reward is selected using the same policy that determined the original action.

#### **Deep Q Network (DQN):**

#### DQN leverages a Neural Network to estimate the Q-value function. The input for the network is the current, while the output is the corresponding Q-value for each of the action.

#### In 2013, DeepMind applied DQN to Atari game. The input is the raw image of the current game situation. It went through several layers including convolution layer as well as a fully connected layer. The output is the Q-value for each of the actions that the agent can take.

Two essential techniques for training DQN are Experience Replay and Separate Target Network.

## **Approaches to Reinforcement Theory of Learning**

Reinforcement Learning has a number of approaches. Here, I have discussed three most well-known approaches: Value-based Learning, Policy-based Learning, and Model-Based Learning Approaches.

### **Value Based Learning Approach:**

Value-based Learning estimates the optimal value function, which is the maximum value achievable under any policy. Storing the value function (or) policy might not be possible especially if the state-action pairs are high dimensional. Thus, function approximators like linear regression, Neural networks are used. In value-based RL, the goal is to optimize the value function V(s). The value function is a function that tells us the maximum expected future reward the agent will get in each state.

The value of each state is the total amount of the reward an agent can expect to accumulate over the future, starting in that state. Then the agent uses this value function to select which state to choose at each step. The agent decides to take up the state with the biggest value.

### **Policy-Based Learning Approach:**

Policy-based Learning searches directly for the optimal policy which achieves the maximum future reward. In policy-based approach, we want to directly optimize the policy function π(s) without using a value function. The policy is what defines the agent behavior at a given time. We learn a policy function. This lets us map each state to the best corresponding action.

This approach has two types of policy:

* **Deterministic**: a policy at a given state will always return the same action.
* **Stochastic**: output a distribution probability over actions.

### **Model-Based Learning Approach:**

In Model-based RL, the environment is treated as a model for learning. This means a model of the environmental behavior is created. This is a great approach until you discover that each environment will need a different model representation.

1. **Application areas in data mining**

Data Mining is the process of extracting patterns from data. Two general avenues of research in the intersecting areas of agents and data mining can be distinguished. The first approach is concerned with mining an agent’s observation data in order to extract patterns, categorize environment states, and/or make predictions of future states. In this setting, data is normally available as a batch, and the agent’s actions and goals are often independent of the data mining task. The data collection is mainly considered as a side effect of the agent’s activities.

Machine learning techniques applied in such situations fall into the class of supervised learning. In contrast, the second scenario occurs where an agent is actively performing the data mining, and is responsible for the data collection itself. For example, a mobile network agent is acquiring and processing data (where the acquisition may incur a certain cost), or a mobile sensor agent is moving in a (perhaps hostile) environment, collecting and processing sensor readings. In these settings, the tasks of the agent and the data mining are highly intertwined and interdependent (or even identical). Supervised learning is not a suitable technique for these cases.

Reinforcement Learning (RL) enables an agent to learn from experience (in form of reward and punishment for explorative actions) and adapt to new situations, without a teacher. RL is an ideal learning technique for these data mining scenarios, because it fits the agent paradigm of continuous sensing and acting, and the RL agent is able to learn to make decisions on the sampling of the environment which provides the data. Nevertheless, RL still suffers from scalability problems, which have prevented its successful use in many complex real-world domains. The more complex the tasks, the longer it takes a reinforcement learning algorithm to converge to a good solution. For many real-world tasks, human expert knowledge is available.

For example, human experts have developed heuristics that help them in planning and scheduling resources in their work place. However, this domain knowledge is often rough and incomplete. When the domain knowledge is used directly by an automated expert system, the solutions are often sub-optimal, due to the incompleteness of the knowledge, the uncertainty of environments, and the possibility to encounter unexpected situations.

RL, on the other hand, can overcome the weaknesses of the heuristic domain knowledge and produce optimal solutions. In the talk we propose two techniques, which represent first steps in the area of knowledge-based RL (KBRL). The first technique uses high-level STRIPS operator knowledge in reward shaping to focus the search for the optimal policy.

Empirical results show that the plan-based reward shaping approach outperforms other RL techniques, including alternative manual and MDP-based reward shaping when it is used in its basic form. We showed that MDP-based reward shaping may fail and successful experiments with STRIPS-based shaping suggest modifications which can overcome encountered problems. The STRIPS based method we propose allows expressing the same domain knowledge in a different way and the domain expert can choose whether to define an MDP or STRIPS planning task. We also evaluated the robustness of the proposed STRIPS-based technique to errors in the plan knowledge. In case that STRIPS knowledge is not available, we propose a second technique that shapes the reward with hierarchical tile coding.

Where the Q-function is represented with low-level tile coding, a V-function with coarser tile coding can be learned in parallel and used to approximate the potential for ground states. In the context of data mining, our KBRL approaches can also be used for any data collection task where the acquisition of data may incur considerable cost. In addition, observing the data collection agent in specific scenarios may lead to new insights into optimal data collection behaviour in the respective domains. In future work, we intend to demonstrate and evaluate our techniques on concrete real-world data mining applications.

1. **What can be done by using Reinforcement learning**

**Resources management in computer clusters**

Designing algorithms to allocate limited resources to different tasks is challenging and requires human-generated heuristics. The paper “Resource Management with Deep Reinforcement Learning” showed how to use RL to automatically learn to allocate and schedule computer resources to waiting jobs, with the objective to minimize the average job slowdown.

**Traffic Light Control**

In the paper “Reinforcement learning-based multi-agent system for network traffic signal control”, researchers tried to design a traffic light controller to solve the congestion problem. Tested only on simulated environment though, their methods showed superior results than traditional methods and shed a light on the potential uses of multi-agent RL in designing traffic system.

1. **Conclusion**

Despite training difficulties, reinforcement learning finds its way to be effectively used in real business scenarios. Generally, RL is valuable when searching for optimal solutions in a constantly changing environment is needed.

Reinforcement learning is used for operations automation, machinery and equipment control and maintenance, energy consumption optimization. The finance industry also acknowledged the capabilities of reinforcement learning for powering AI-based training systems. Although trial-and-error training of robots is time-consuming, it allows robots to better evaluate real-world situations, use their skills for completing tasks, or reacting to unexpected consequences appropriately. In addition, RL provides opportunities for eCommerce players in terms of revenue optimization, fraud prevention, and customer experience enhancement via personalization.

As we amass more data, the demand for advanced data mining and machine learning techniques will force the industry to evolve in order to keep up. We’ll likely see more overlap between data mining and machine learning as the two intersect to enhance the collection and usability of large amounts of data for analytics purposes.

But some experts have a different idea about data mining and machine learning altogether. Instead of focusing on their differences, we could argue that they both concern themselves with the same question: “How we can learn from data?” At the end of the day, how we acquire and learn from data is really the foundation for emerging technology. It’s an exciting time not just for data scientists but for everyone that uses data in some form.