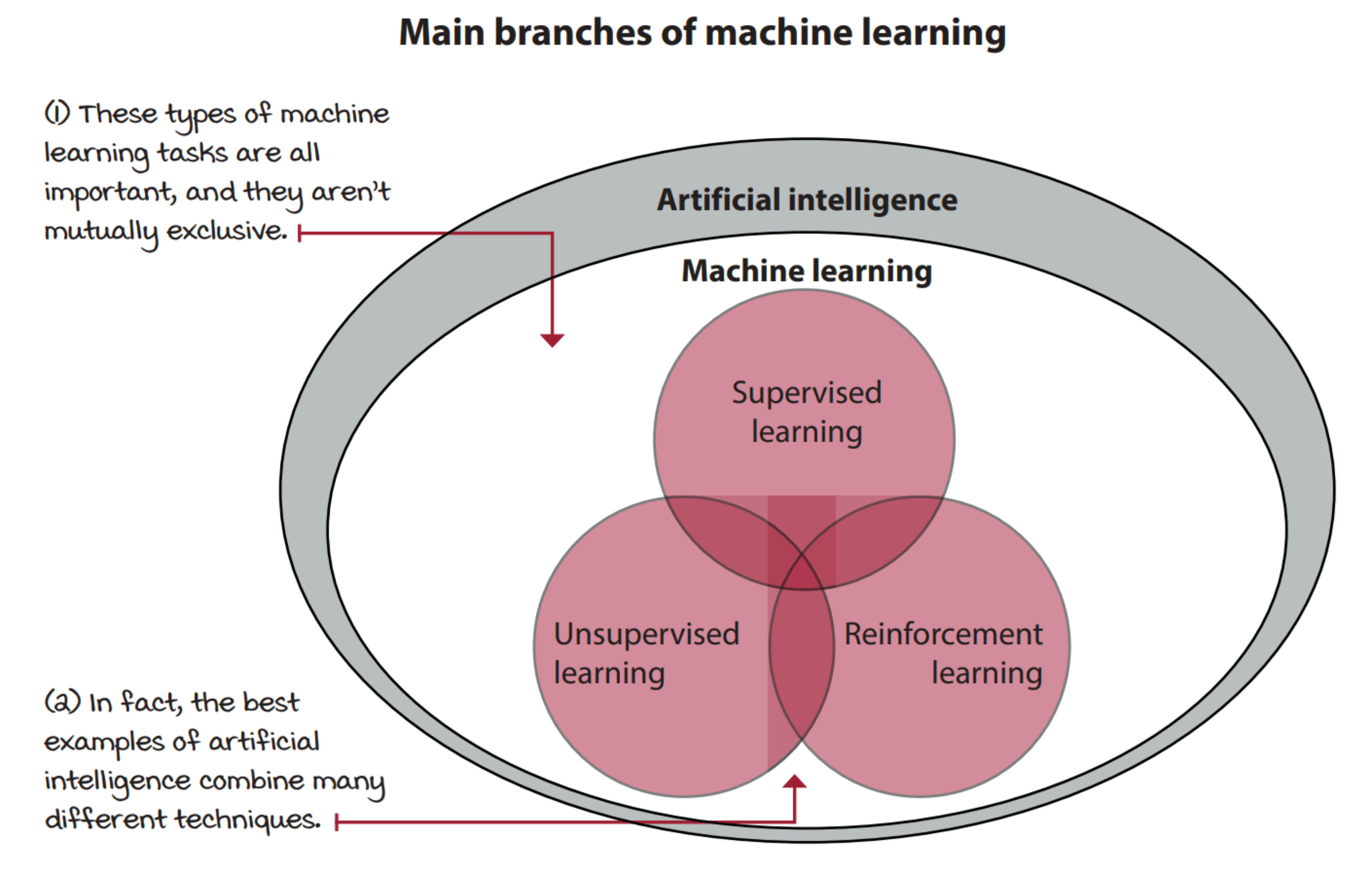
# Reinforcement Learning Training 2025

Round 1

# Where is RL in ML?



# Supervised Learning

- We know *all* the right answers (label)
- We teach machine.

# Unsupervised Learning

- We don't know the answer.
- We let machine find structure in the data.

# Reinforcement Learning

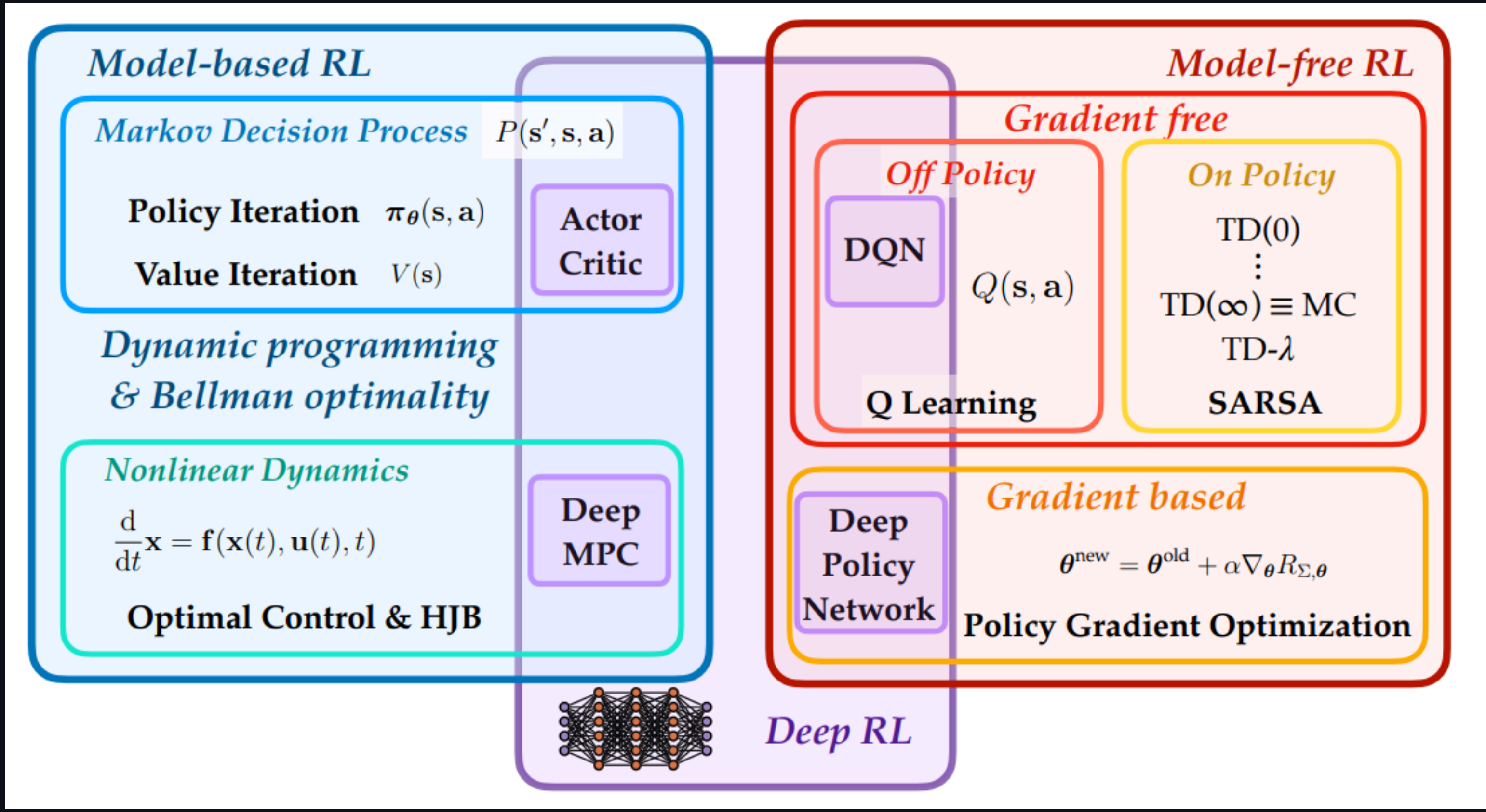
- We don't know *all* the right answer
    - but we have a way to conduct *trial-and-error* experiments.
- We let the machine *discover* the answers.

# Applications

- `ChatGPT`
  - Enhanced by reinforcement learning through a technique called Reinforcement Learning from Human Feedback (RLHF). [1] [2]
- `Spot`
  - Utilize reinforcement learning (RL) to enhance their locomotion and manipulation capabilities. [3]

# Types of RL

- Don't worry. We will come back later.



# RL Formalism

- Entities
  - Agent
  - Environment
- Communimation
  - Actions
  - Reward
  - Observation

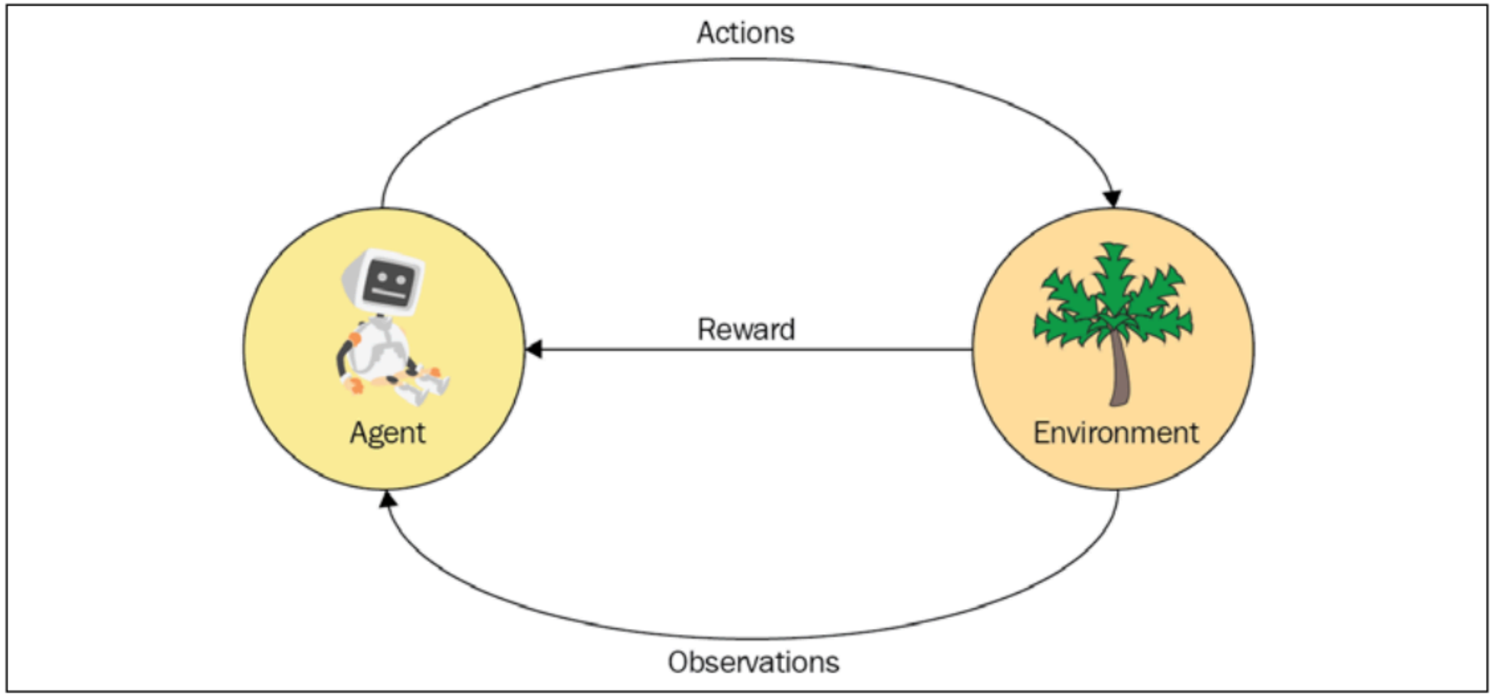


Figure 1.2: RL entities and their communication channels

# Reward

- A scalar value we obtain periodically from the environment.
    - Can be positive or negative
- Tell our agent how well it has behaved.
- Reflects the success of the agent's recent activity (local)
    - Not all the successes achieved by the agent so far.
- What an agent is trying to achieve is the largest *accumulated* reward over its sequence of actions.

# Agent

- An agent is somebody or something who/that interacts with the environment by executing certain actions, making observations, and receiving eventual rewards for this.
- In most practical RL scenarios, the agent is our piece of software that is supposed to solve some problem in a more-or-less efficient way.

# Environment

- The environment is everything outside of an agent.
- The agent's communication with the environment is limited to
  - Reward (obtained from the environment)
  - Actions (executed by the agent and given to the environment)
  - Observations (some information besides the reward that the agent receives from the environment).

# Action

- Actions are things that an agent can do in the environment.
- We distinguish between two types of actions—discrete or continuous.
  - **Discrete actions** form the finite set of mutually exclusive things an agent can do, such as move left or right.
  - **Continuous actions** have some value attached to them, such as a car's action turn the wheel having an angle and direction of steering.

# Observation

- Observations are pieces of information that the environment provides the agent with that say what's going on around the agent.
- *I am guessing it is something that agent can use to make action?*

# Markov Processes (MP)

- Also called a Markov chain
- Models a system observed through a sequence of states.
- The system transitions between states according to certain dynamics, but the observer cannot influence the system.

# MP - State Space

- The set of all possible states is called the state space.
- For MPs, the state space is finite but can be very large.
- Observations form a sequence or chain of states, known as the history.

# MP - Markov Property

- The future state depends only on the current state, not on the full history.
- Each state is self-contained and unique.
- This simplifies modeling by focusing only on the current state to predict the future.

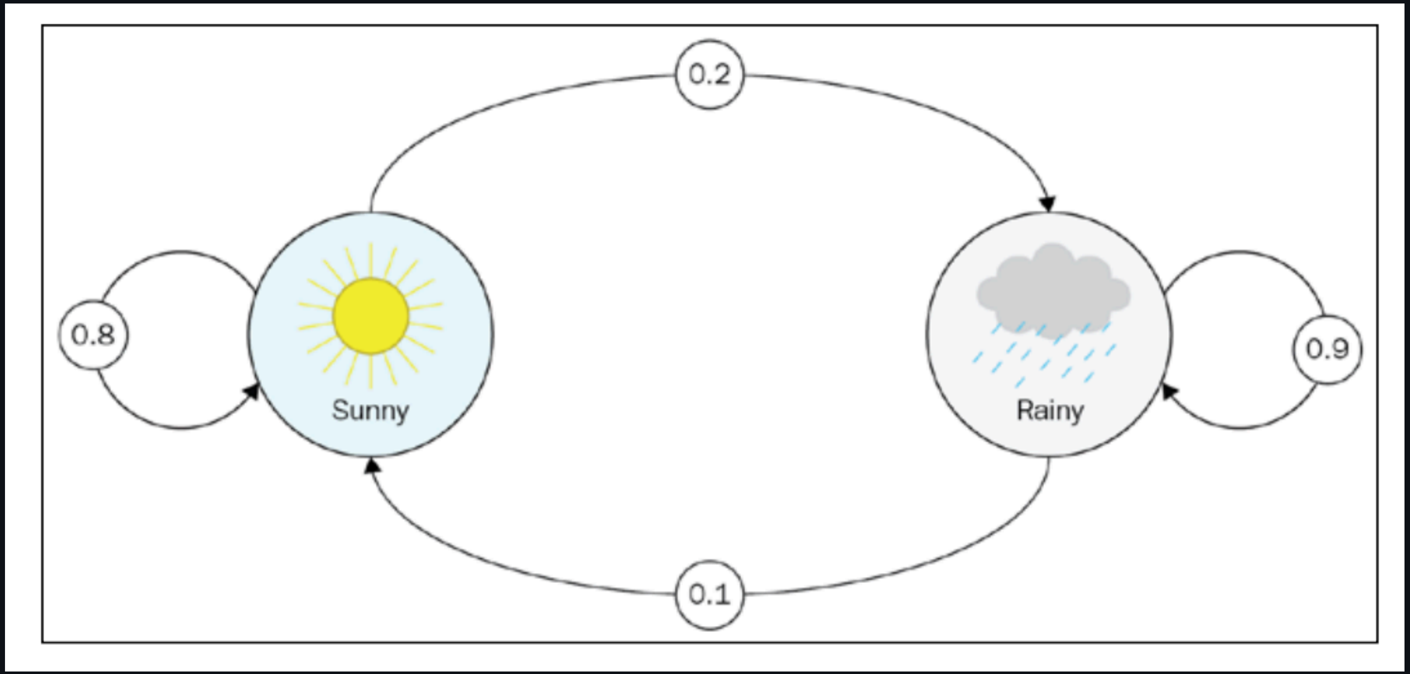
# MP - Example (Weather Model)

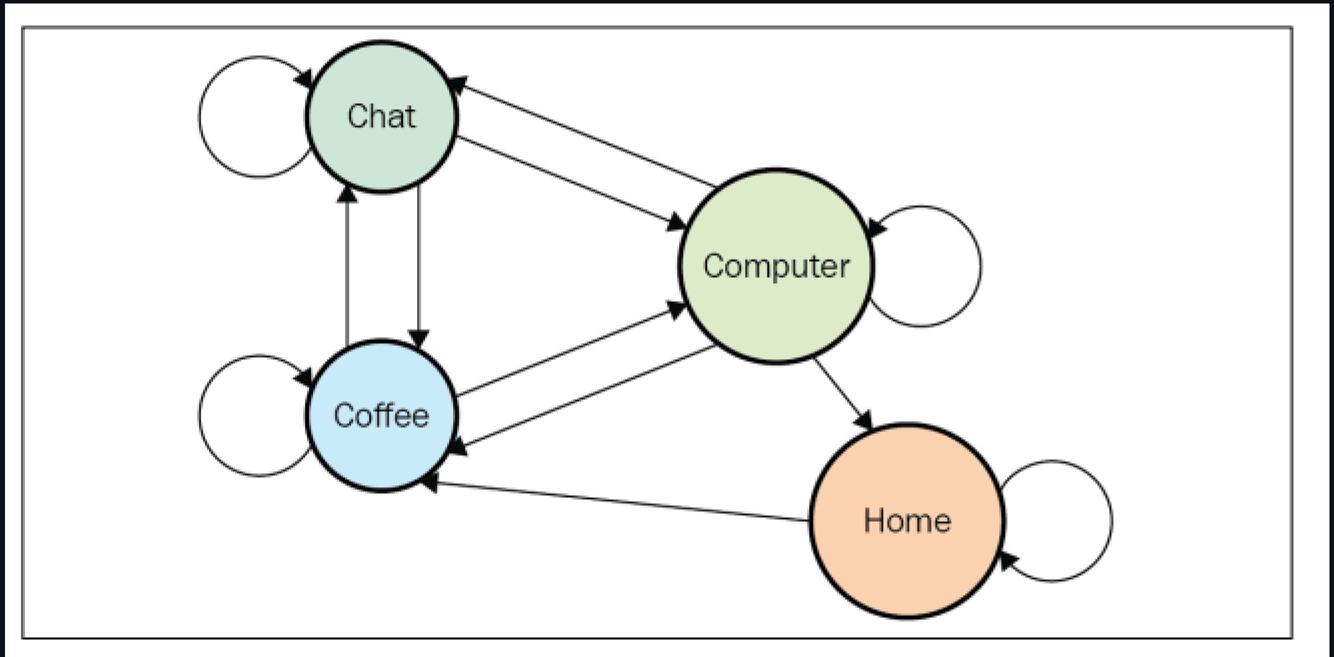
- States: `{sunny, rainy}`
- Sequence example: `[sunny, sunny, rainy, sunny, …]`
- The Markov property means the probability of rain tomorrow depends only on today's weather, not previous days.
  - This is a simplification and not fully realistic since weather depends on many factors (season, geography, solar activity).
  - To capture more dependencies, the state space can be extended (e.g., include season with weather states).

# MP - Transition Matrix

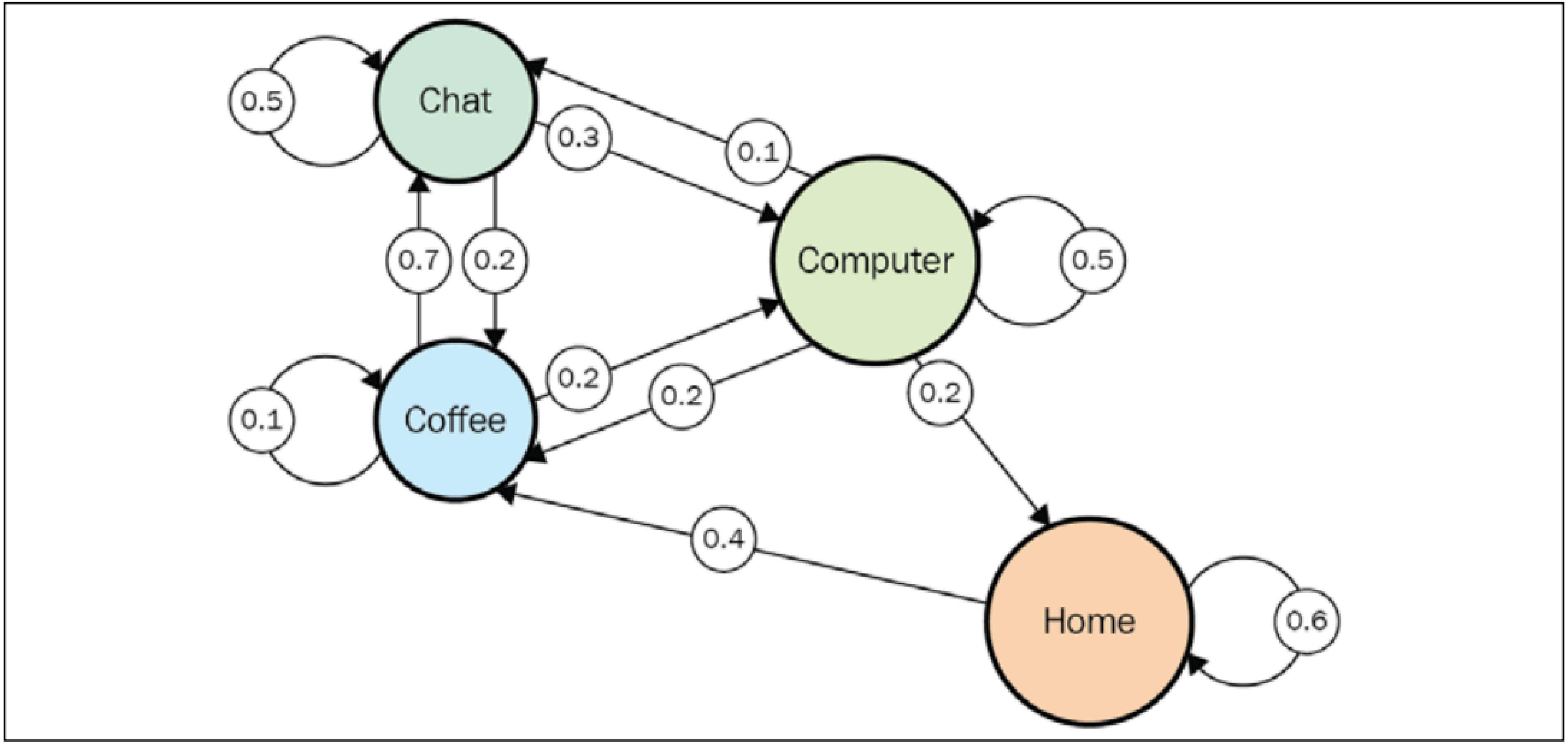
- An $N \times N$ matrix where $N$ = number of states.
- Each entry $(i, j)$ represents the probability of transitioning from state $i$ to state $j$.
- Example matrix for weather:

| | **Sunny** | **Rainy** |
|---|---|---|
| Sunny | 0.8 | 0.2 |
| Rainy | 0.1 | 0.9 |


0.2
0.8
Sunny
Rainy
0.9
0.1


Chat
Computer
Coffee
Home

| From \ To | Home | Coffee | Chat | Computer |
|---|---|---|---|---|
| **Home** | 60% | 40% | 0% | 0% |
| **Coffee** | 0% | 10% | 70% | 20% |
| **Chat** | 0% | 20% | 50% | 30% |
| **Computer** | 20% | 20% | 10% | 50% |


Chat
0.5
0.3
0.1
Computer
0.5
0.7
0.2
Coffee
0.1
0.2
0.2
0.2
0.4
Home
0.6

# Estimating the Transition Matrix

- In real-world scenarios, we typically do not know the exact transition matrix of a system.
  - Instead, we observe sequences of system states, known as `episodes`.
- How to estimate the transition matrix
  - Count all observed transitions from each state to every other state.
  - Normalize these counts so that the probabilities from each state sum to 1.
  - The accuracy of this estimation improves as more observational data (episodes) are collected.

# Markov Reward Processes (MRP)

- To model rewards, we extend the Markov Process (MP) by associating a reward value with each state transition.
- Now, each transition also has an associated reward.

# MRP - Reward

- The most general form uses a reward matrix where each entry specifies the reward for transitioning from state $i$ to state $j$.
- This matrix can be simplified if rewards depend only on the destination state, in which case only state-to-reward pairs are needed.

# MRP - Reward

- For each episode, the return at time $t$ (denoted as $G_t$) is the sum of future rewards, discounted by $\gamma$ at each step:

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+1+k}$$

- where $\gamma$ is a scalar value between 0 and 1 called a `discount factor`.

# MRP - Discount Factor

- $\gamma$ determines how much **future rewards** are valued compared to **immediate rewards**.
  - $\gamma = 1$**:** The agent values all future rewards equally, summing them without discounting. This represents perfect foresight.
  - $\gamma = 0$**:** The agent only considers the immediate reward, ignoring all future rewards—total short-sightedness.