A New Framework of Memory for Learning

John Tan Chong Min

19 Dec 2022

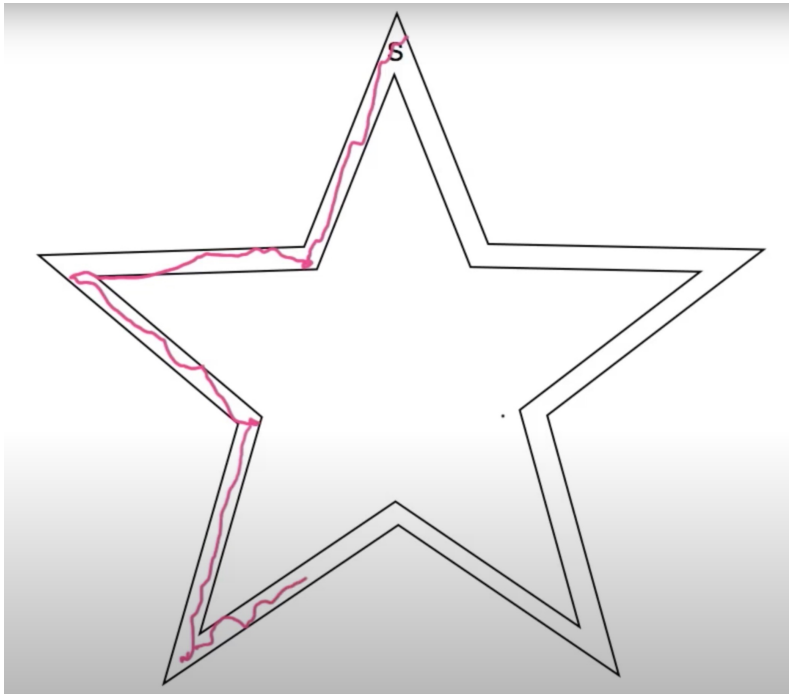
Special thanks for discussing and improving my ideas in NeurIPS 2022:
Wu Shuchen (Abstraction, Hashtable and overall discussion)
Joseph Campbell (Reinforcement Learning)
Markus Hiller (Continual Learning)

Aim

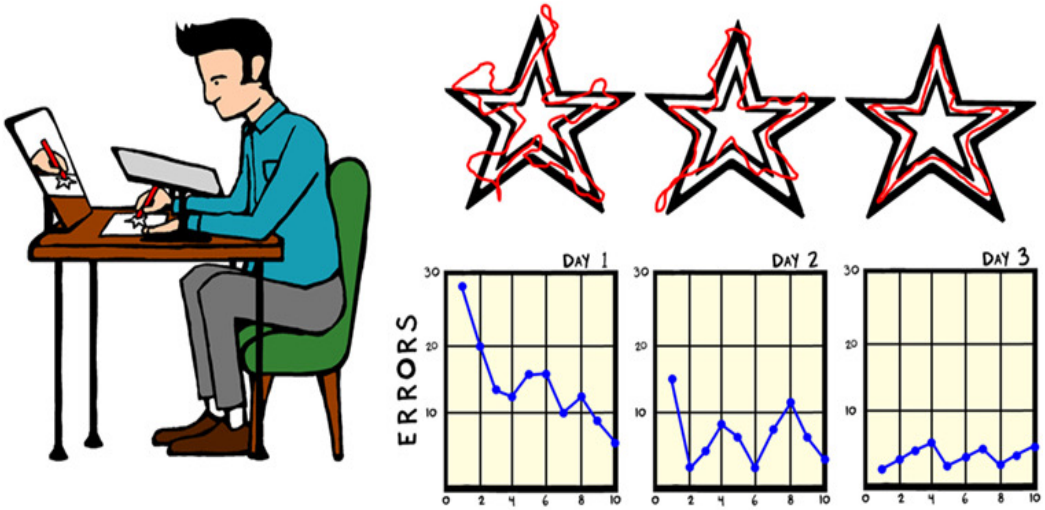
- To seek biological / neuroscience and psychology inspirations for how humans learn fast, and model it into AI
- Do not need to replicate faithfully the detailed mechanisms behind human thinking
- Just need to take broad concepts and apply it

Star-tracing task

- https://www.youtube.com/watch?v=wFoIvB-04YY



The case of Henry Molaison



- HM did not have a hippocampus
- He did not have the ability to form new memories, and had no episodic or semantic memories
- Although he could not recall having done the star tracing task, he could still get better at it (procedural memory)
- Subjects with hippocampus could learn the star tracing fast much faster than HM (not shown)

Image Extracted from: https://kids.frontiersin.org/articles/10.3389/frym.2018.00045

HM and Modern Neural Networks

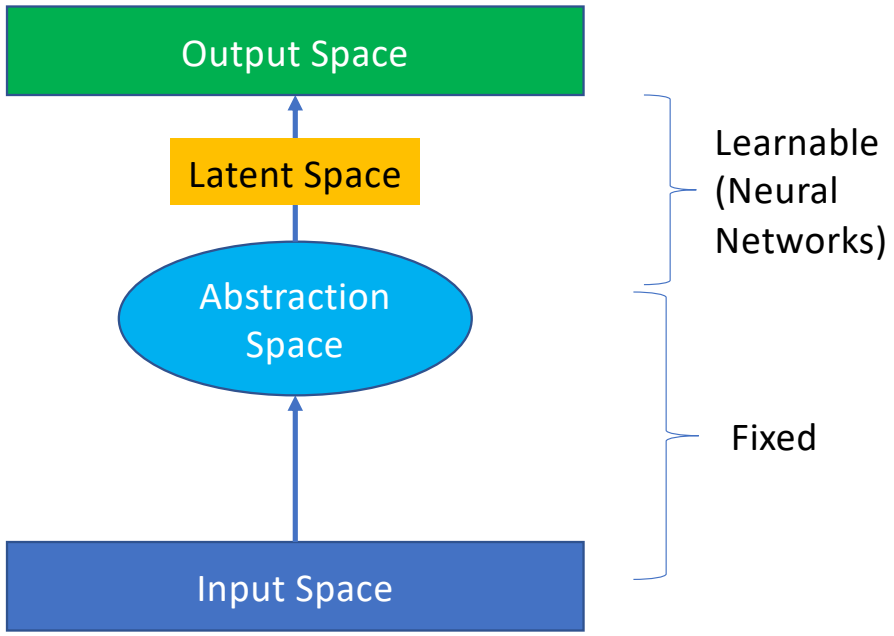
- Modern Neural Networks are like HM
- There is no memory storage nor retrieval other than updating the weights
- Weight updating is akin to procedural memory
  - Gradual learning over time, slow and stable
- Memory formation and retrieval is akin to episodic/semantic memory
  - Fast learning over time, variable and constantly changing
- **Memory is useful for learning!!**

Memories as abstraction

Abstraction as generalization

Abstraction

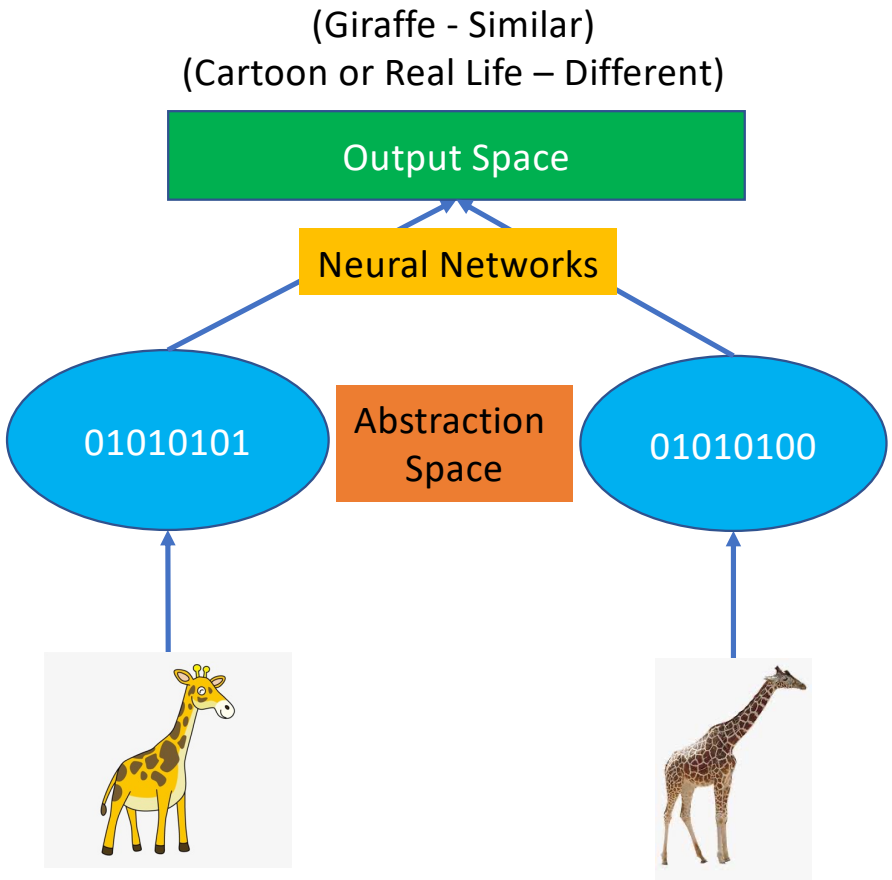
- Memories are stored in an abstract form
- The same abstraction can then referenced across contexts (generalization)
- This enables reuse of learnt memories in different contexts
- Lower level than word embeddings / latent space in Large Language Models / CLIP



Abstraction

- **Hypothesis: Abstraction process is fixed and unlearnable, which stores new memories without the need of updating ALL previously stored memories the moment the abstraction mapping changes**

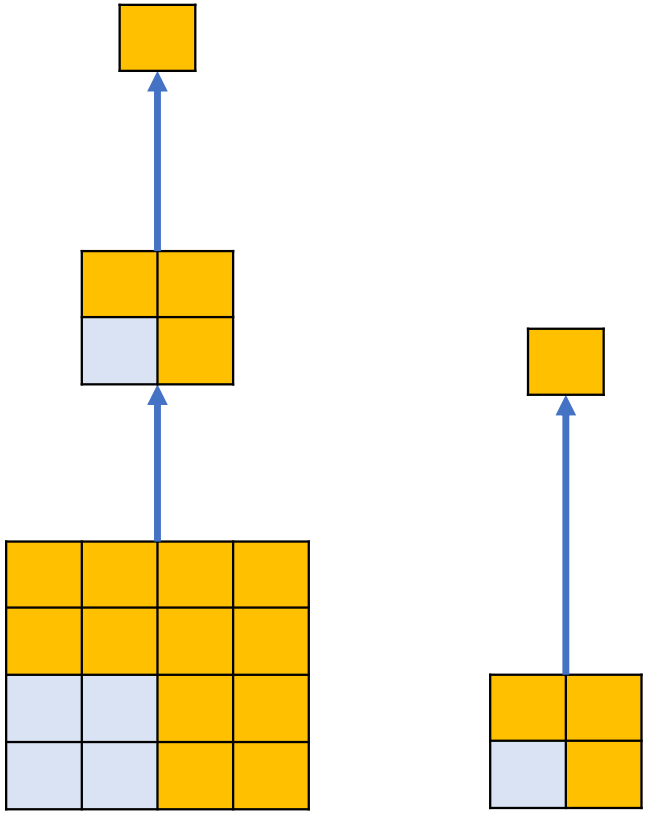
- With the abstraction space as input, neural networks can learn associations / dissociations between them for various tasks



Recursive Abstraction

- We can get inputs at different levels of details
- The abstraction procedure should be the same regardless of scale
- The abstraction should be able to be applied recursively
- Hierarchical chunking may help to get a good abstraction space

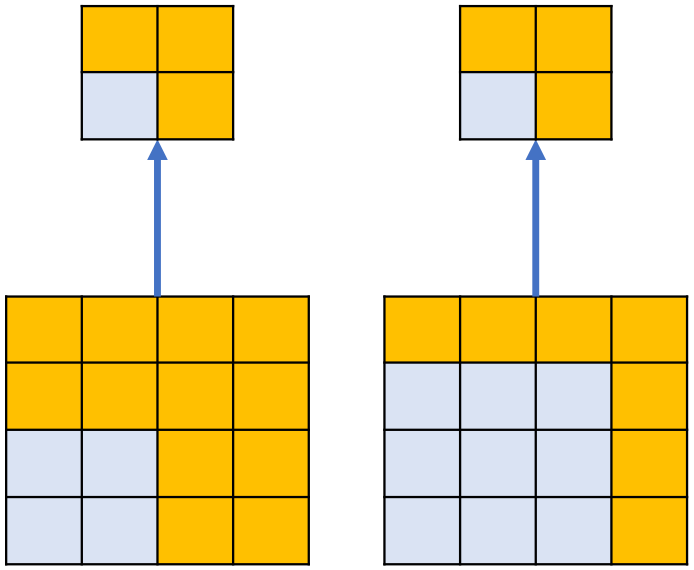
Abstraction Process:
Majority Mapping in a 2x2 square



Lossy Abstraction

- It can be hard to regenerate the complex stimuli we experience
- We store them in a very limited form (it is impossible to reconstruct accurately the original stimuli)
- Lossy representation may help with generalization
- Can think of Convolutional Neural Network filters and pooling layers

Abstraction Process:
Majority Mapping in a 2x2 square



Learnable Abstraction

- Perhaps the abstraction being fixed is too strong an assumption
- Babies generally don't remember long-term memories until they are 2 years old
- Perhaps the brain is learning how to form the right abstraction space, perhaps using techniques like autoencoder reconstruction loss?

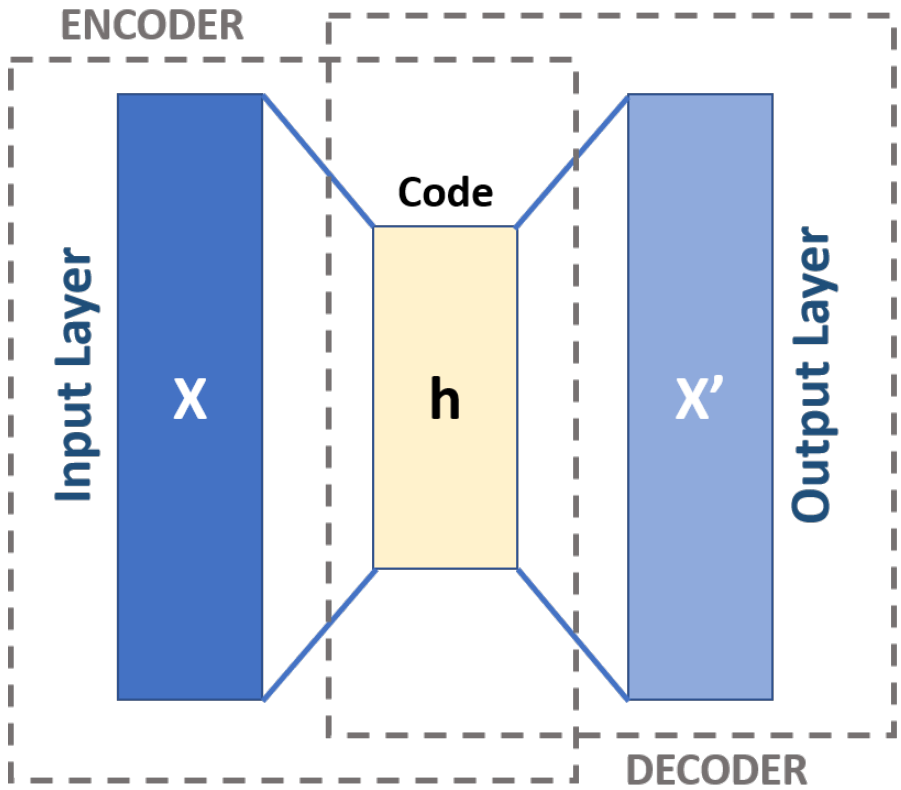


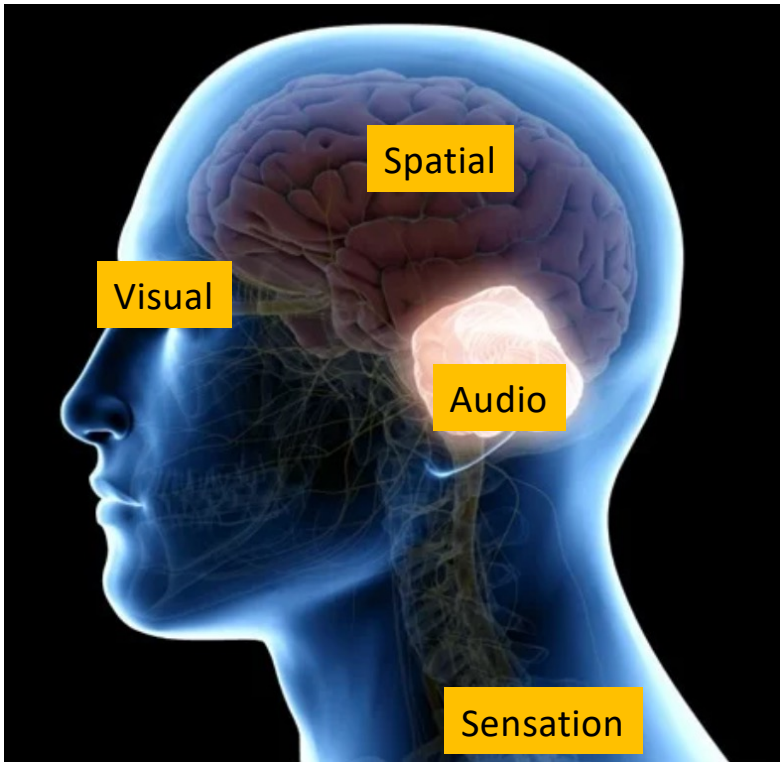
Image from: https://en.wikipedia.org/wiki/Autoencoder

Memories as a multi-modal system

Multi-modal processing

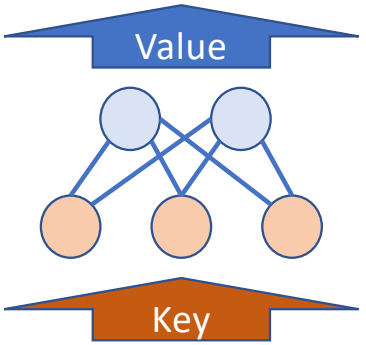
- The brain takes in multiple inputs from various sensory domains
  - Visual
  - Audio
  - Sensation (e.g. Touch, Pain)
  - Spatial (e.g. Location)

- Need to store these various inputs into a compact form for memory



Hashing

- Storing memories can be as easy as storing it into a hash table
- Retrieving the memory can be a lookup from this hash table
- In the brain, this hash table may be modelled as a neural network (though not ideal)
  - Neural network-based modelling can lead to values being changed as we learn other key-value pairs, which may cause inaccurate memory retrieval
- In AI systems, it is best modelled as the hash-table on a computer
  - No memory retrieval issues except for collisions
  - We can also allow for multiple similar keys to exist if we change the lookup mechanism to be an attentional-based one



Key (Visual)	Value
Visual A	Monica
Visual B	Joey
Visual C	Rachel
…	…

Multi-modal hashing

- With multi-modal input, the keys to the hash table are also multi-modal

- Just by referencing only a subset of the modes, we can extrapolate the situation to various unseen settings
  - Knowing that we are at Location E, which is similar to location C, we know that we have a high chance to encounter Joey
  - Knowing that we see Visual D, which is similar to visual C, we know that we have a high chance to encounter Rachel

Key (Visual, Spatial)	Value
(Visual A, Location A)	Monica
(Visual A, Location B)	Monica
(Visual B, Location C)	Joey
(Visual C, Location D)	Rachel
(Visual A, Location D)	Monica
…	…

Incomplete Hash

- We can also match with incomplete multi-modal data, that way the matching will be done only based on the subset of key modalities which match the input (aka incomplete hashing)

- Just one part of the stimulus (i.e. location) can be matched to the nearest hash in memory
  - We trigger memories of events just by looking at past photos
  - We trigger memories of events by listening to a particular piece of music
  - We trigger memories of events just by passing by a location

Key (Visual, Spatial)	Value
(Visual A, Location A)	Monica
(Visual A, Location B)	Monica
(Visual B, Location C)	Joey
(Visual C, Location D)	Rachel
(Visual A, Location D)	Monica
…	…

Key (Visual)	Value
Visual A	Monica
Visual B	Joey
Visual C	Rachel
…	…

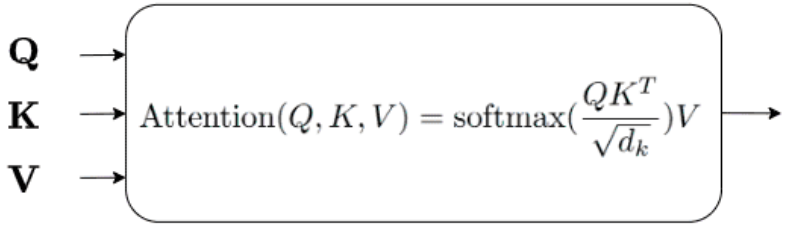
Key (Spatial)	Value
Location A	Monica
Location B	Monica
Location C	Joey
Location D	Rachel
Location D	Monica

Multiple Referencing

- We try to match a novel situation with a key that has been stored in the multi-modal hash table
- There can be multiple similar matches, and the value returned can be weighed according to the similarity of matching query with key
  - Similar to the Query-Key matching procedure in Transformer architecture
- For robustness, we can also match multiple combinations of the query with different omitted parts and average the returned value

Query: Visual A, Location C

Key (Visual, Spatial)	Value	% Match
(Visual A, Location A)	Monica	50
(Visual A, Location B)	Monica	50
(Visual B, Location C)	Joey	50
(Visual C, Location D)	Rachel	0
(Visual A, Location D)	Monica	50
…	…	



Storing and Forgetting Memories

Storing memories

- There is limited room in the brain to hold memories

- Need to be very selective as to what can be stored
- Priority of storing is determined by emotion
  - Flashbulb memories are very powerful
- High emotion memories (fear, happy) will be prioritized over low emotion ones (neutral, boredom)
  - In school settings, we largely do not remember the process of studying, but the friends we made and the activities we did with them

Forgetting memories

- Memory retrieval helps to improve the strength of the memory
- Memories that are not accessed often are forgotten and make room for other memories to be stored
- Helps to make sure the memories we have are useful for the current environment
  - Less used memories are those not relevant and forgotten
  - Frequently used memories are those relevant and strengthened

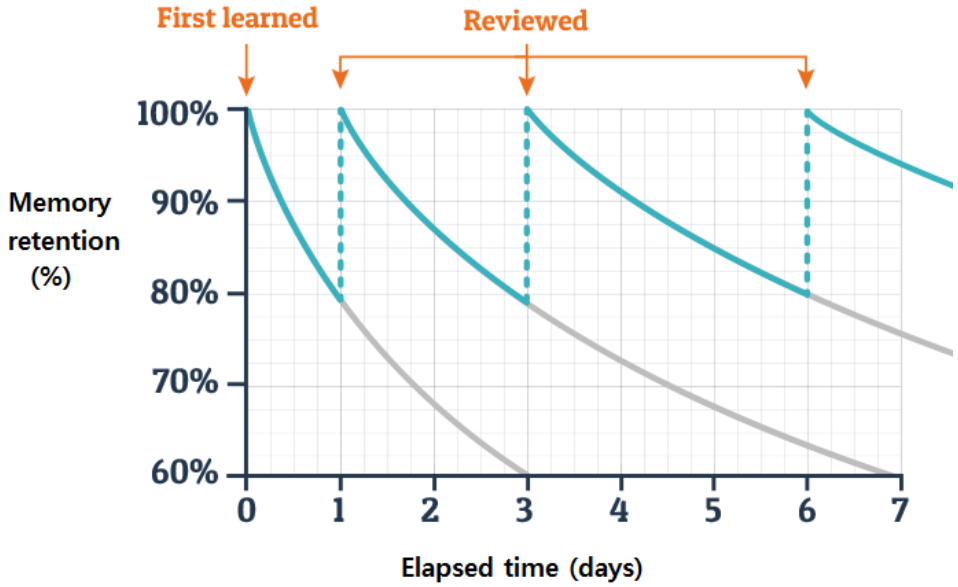


Chart from: Chun, Bo Ae & hae ja, Heo. (2018). The effect of flipped learning on academic performance as an innovative method for overcoming ebbinghaus' forgetting curve. 56-60. 10.1145/3178158.3178206.

Inter-memory linkages

Inter-memory associations

- Once memory is formed, if two memories are highly related (i.e. due to temporal association, causality), the memory will be linked together via a directional mapping
- When one memory is triggered, the subsequent links will also be activated as well
- The link to the next state could be stored as a value of the first state, or it could be a separate inter-memory linkage network

Memory in Reinforcement/Continual Learning

Typical Reinforcement Learning is Slow

- Typically updated by one-step Bellman update (Temporal Difference Error)

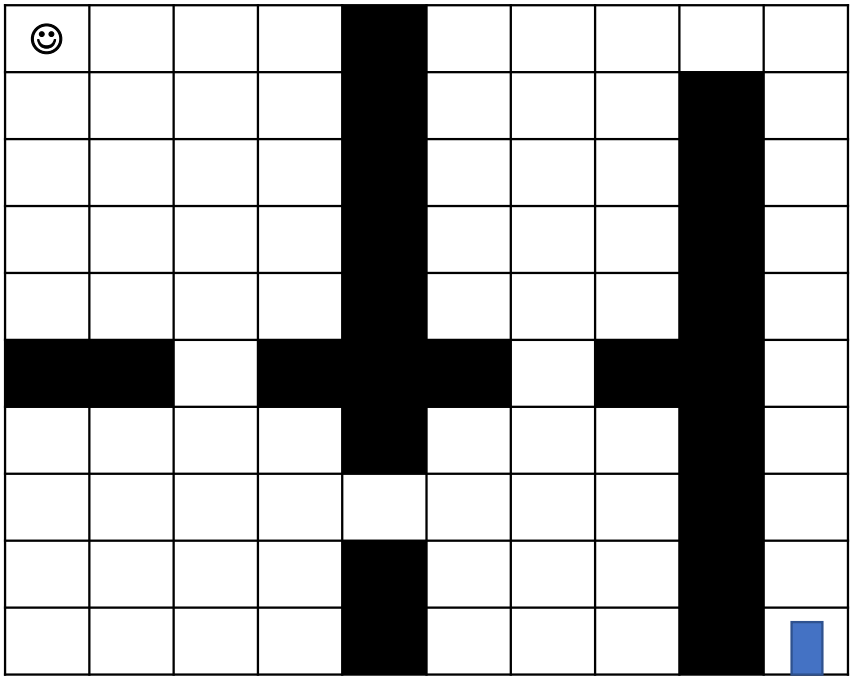
- Takes as many updates as the path length to update the states with the final value

$$V(s) \leftarrow V(s) + \alpha(\overbrace{r + \gamma V(s')}^{\text{The TD target}} - V(s))$$

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{current value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Big(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} - \underbrace{Q(s_t, a_t)}_{\text{current value}}\Big)}^{\text{temporal difference}}$$

$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{new value (temporal difference target)}}$$

Walled Maze



Legend:

☺ Agent

Door

Hippocampal Replay

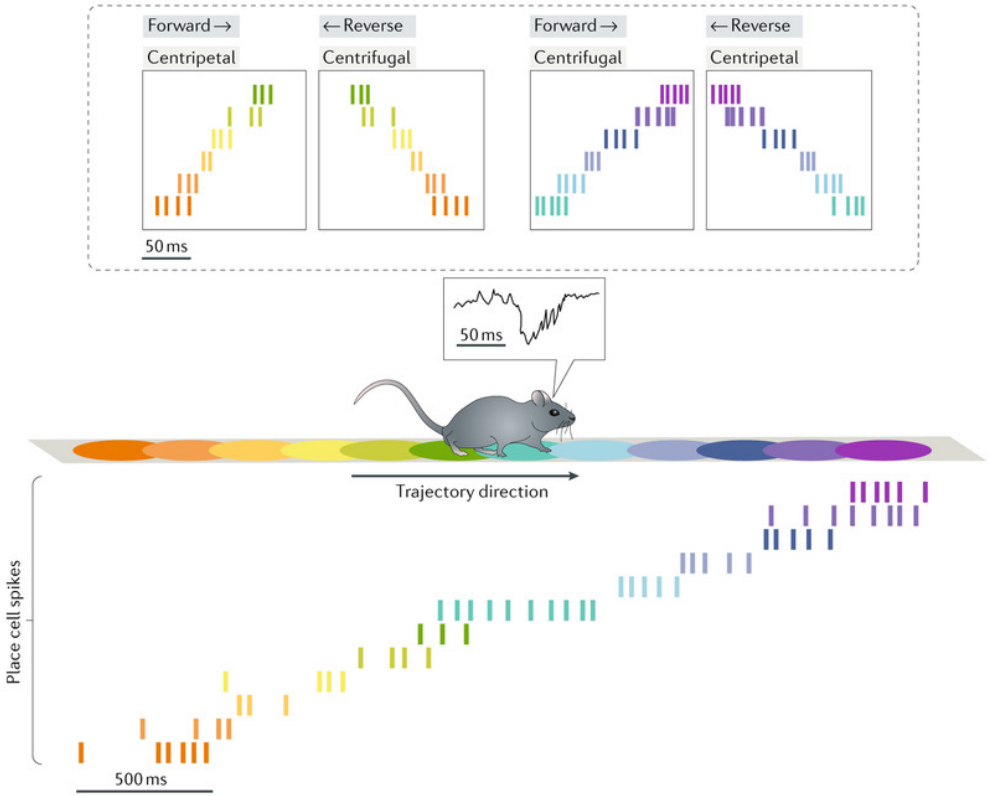


Figure extracted from Joo, H. R., & Frank, L. M. (2018). The hippocampal sharp wave-ripple in memory retrieval for immediate use and consolidation. Nature reviews. Neuroscience, 19(12), 744–757. https://doi.org/10.1038/s41583-018-0077-1

Algorithm 1 Hippocampal Replay

1: **procedure** HIPPOCAMPALREPLAY(env, $trajectory$):
2: Consolidate the list of states in the successful trajectory, in chronological order ▷ Pre-play
3: Visit states in reverse order from the goal, and update the memory of each state ▷ Replay

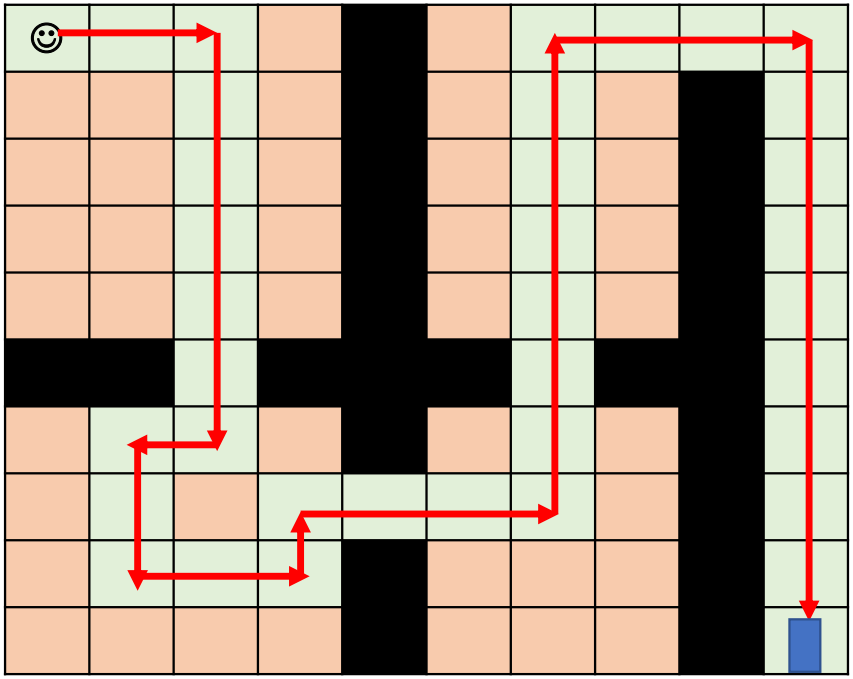
- Inspired from sharp wave-ripple in mice
- Pre-play to **retrieve states** along successful trajectory
- Replay to **update memory** along successful trajectory
- Different from random sampling of Replay Buffer in DQN and prioritized sampling in Prioritized Experience Replay!!

For one possible implementation check out my NeurIPS memARI 2022 workshop paper: Using Hippocampal Replay to Consolidate Experiences in Memory-Augmented Reinforcement Learning. Chong Min John Tan, Mehul Motani.

Hippocampal Replay to help with learning (Simple Version)

- Can do hippocampal replay to immediately learn a successful pathway by remembering the best action for a possible solution path
- If paired with some form of count-based approach, can lead to high value states being on the successful trajectory
- Leads to consistent solving of environment (though may not be optimal) if we limit the exploration by following path learnt in hippocampal replay instead of exploring

After Hippocampal Replay:



Legend:

Low Value States

High Value States

Obstacle

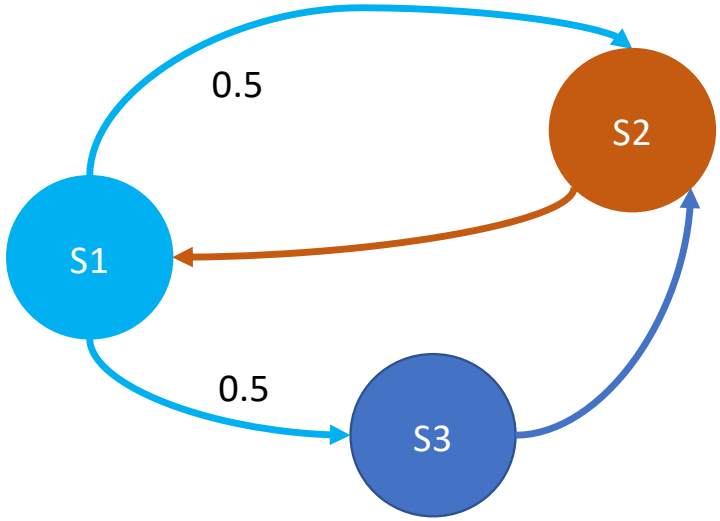
Door

Agent

Discovered Path

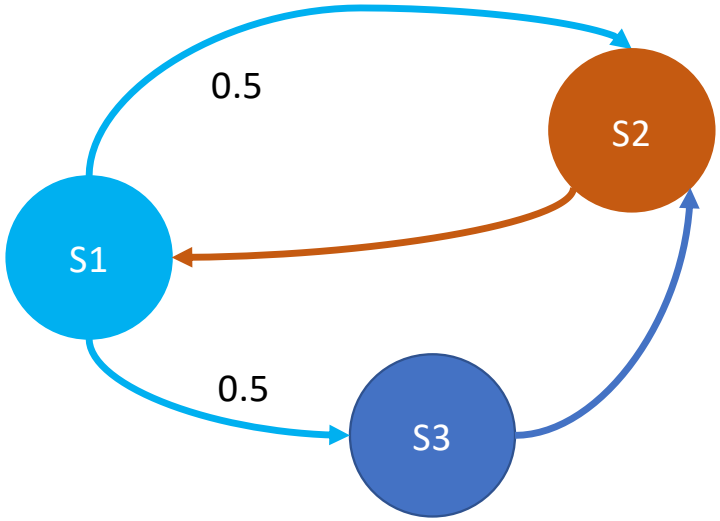
Should we model the Markov Decision Process of the world?

- Recent works have tried to utilize world models
  - MuZero
  - Dreamer v2
- Computationally Expensive
- To get probability of transition, we would need many samples to learn it accurately
- Difficult to define state transition probability when number of possible states is unbounded
- Multi-agent systems can be non-Markovian



A New RL Paradigm

- No need to model Markov Decision Process – impractical to model environments with unbounded state and action spaces
- No need to model probability of transition – just need to see how often the next state is stored based on memory
  - Most real-world short-term transitions may be deterministic
- Just need to remember experienced transitions and then retrieve it the next time we encounter a similar state
  - If probabilistic transition, we may allow for repeated keys
  - If deterministic transition, can overwrite existing key
- **No reward modelling needed.** Can lookahead using stored memory and search for the desired goal state
  - If goal state cannot be found using lookahead, go in general direction of the goal state based on action recommended by a goal-directed neural network



Key (State)	Value 1 (Next State)	Value 2 (Action)
1	2	A
1	3	A
2	1	A
3	2	A

How do humans think?

- Imagine if you were asked to choose one of two doors to walk to your office from your office lobby
- Which door would you choose?
- Do you calculate value functions for each door? Or based on memory?

Benefits of using memory

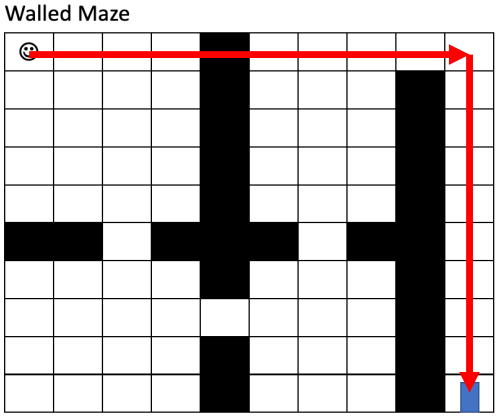
- **Hypothesis: Supervised Learning is way more efficient than Reinforcement Learning due to learning towards an unchanging target**

- Memory can serve as the target for the neural network
  - Relatively unchanging
  - Stable target to train towards

- Successful RL methods use a fixed target: Double DQN (one network fixed) and other Actor-Critic methods which train the Actor while fixing Critic and train Critic while fixing Actor
  - However, using neural network as target comes with the imperfect retrieval problem: the value stored in the hash table may change when learning new inputs

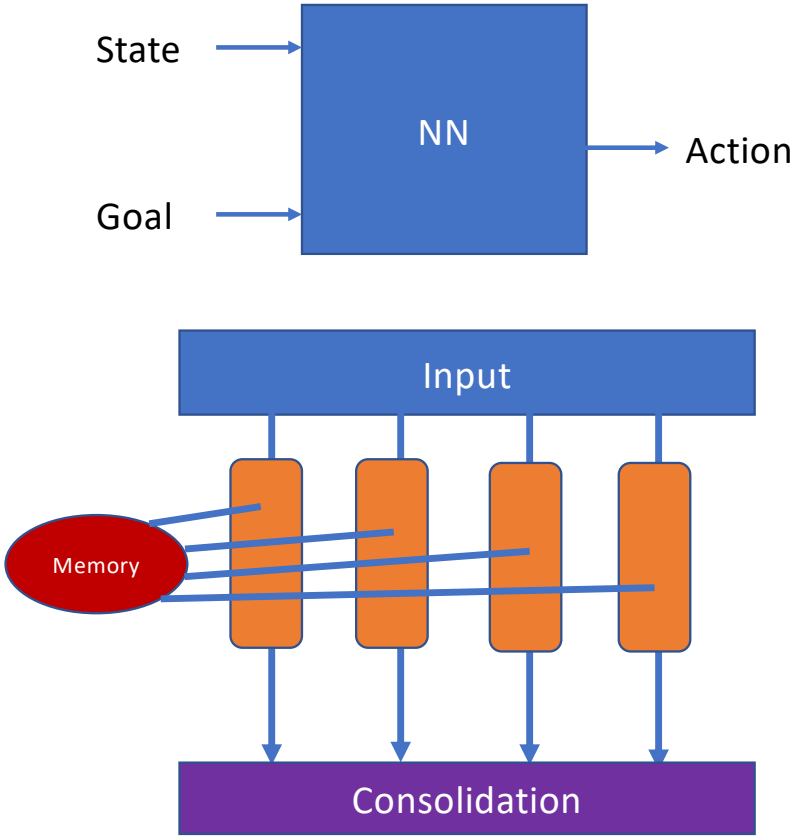
Goal-Directed Exploration

- To ensure memory only stores relevant transitions, we cannot explore everything
- We use a goal-directed exploration
- Human exploration is typically goal-driven, by some intrinsic want like food, shelter, point-to-point navigation
- Difficult problem to dissect overall goal into sub-goals for the agent to fulfil
- Difficult problem also in generating the right heuristic to head towards the direction of the goal



Two Networks – Fast and Slow

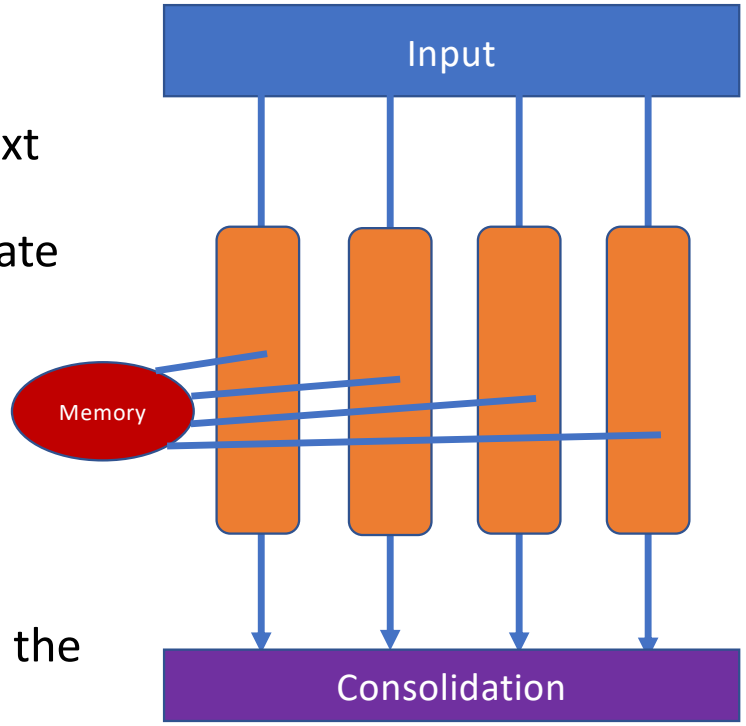
- **Fast (System 1):** Neural network predicts best action given a goal
- **Slow (System 2):** Memory retrieval network to match current state and get required action and next state from memory
  - Can also lookahead multiple timesteps to sample more possible outcomes
- Perform hippocampal replay to update experienced transition to neural network
  - Pre-play re-imagines the stored transitions and their associations from first state to last state
  - Replay updates the memory and enables the fast neural network to learn a goal-directed action



Memory retrieval network mechanism

- Uses parallel processing
- Each parallel branch is like a minicolumn in the neocortex
- Takes the starting state and then reference memory for next state
- If more than one match, randomly pick one for the next state
- If next state is goal state, break
- Continue with next state as the key to reference memory
- Repeat until X lookahead timesteps

- All parallel branches will come back with a response
- See which branch has shortest trajectory to goal state, use the first action



Continual Learning

- **Hypothesis: The way we model the environment is deterministic, short-term prediction is largely deterministic**

- If the environment changes, the memory value will not match that of the prediction
  - Add current transition to memory
  - Remove all existing memory which does not match the experience (if deterministic)

- Memory will keep storing the most updated transition of the environment

- Each time memory is accessed, memory strength is increased. Memories unused for a while will be forgotten.

Overall Procedure using Memory (Part 1)

- State and Action Prediction
  - Agent has a goal state in mind, and knows its current state
  - **System 1:** Agent queries the fast neural network to get action probabilities for the goal (exploit)
  - Get state-action visit counts via retrieval from episodic memory and choose action in explore-exploit way
    - action = $\mathrm{argmax}(\mathrm{ActionProbs} - \alpha\sqrt{numvisits})$
  - **System 2:** Agent uses the slow memory retrieval procedure to find out if there is any match in goal state in multiple lookahead simulations. If there is a match, overwrite the action from the explore-exploit mechanism

- Memory Update
  - Update the memory retrieval network with the new transition
  - Remove all memories that conflict with the current transition (if deterministic)

Overall Procedure using Memory (Part 2)

- Fast Neural Network Update

(at each time step)

  - **Previous states replay:** Update fast neural network such that conditioned on the current state as the goal state, the past history of states as the start state, the network will return that particular action for the state

  - **Future states replay (only if trajectory found):** Update fast neural network such that conditioned on all intermediate states in System 2 lookahead as goal states, the first state will return action for the current timestep

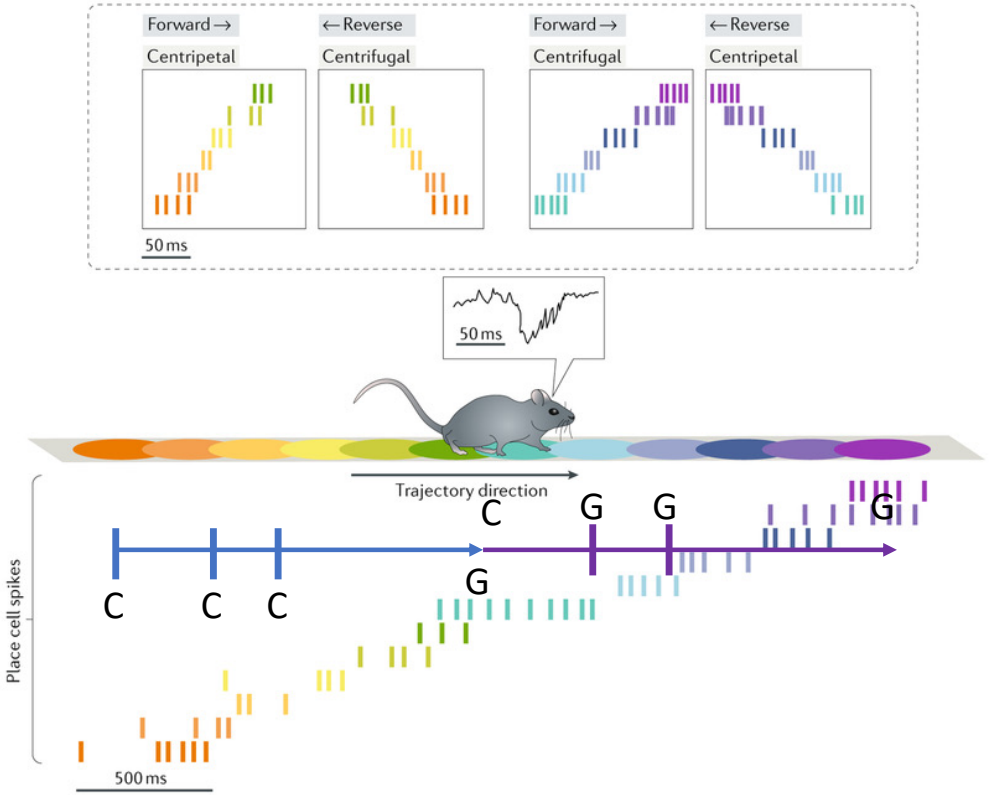


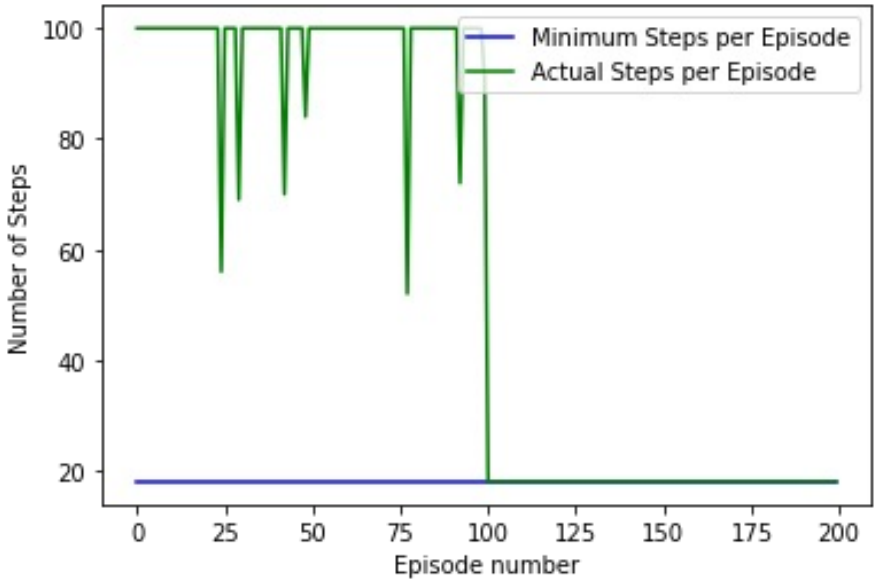
Figure extracted from Joo, H. R., & Frank, L. M. (2018). The hippocampal sharp wave-ripple in memory retrieval for immediate use and consolidation. Nature reviews. Neuroscience, 19(12), 744–757. https://doi.org/10.1038/s41583-018-0077-1

Experimental Validation

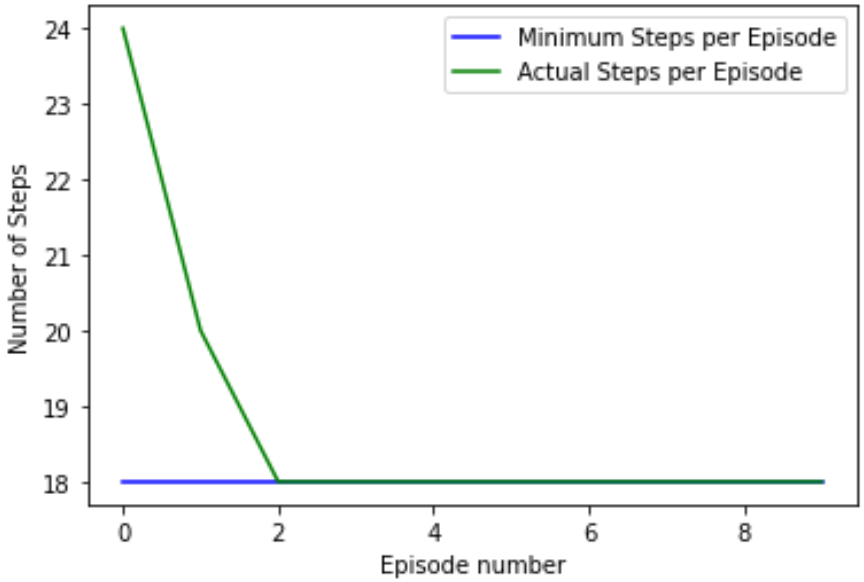
- Start State State to be (0, 0) and End State to be (9, 9) in a 10x10 grid
- Actions: Up, Down, Left, Right
- See if agent can learn to navigate to goal state efficiently
- Agent cannot see the entire grid, only knows its position and goal position
- Q-Learning vs Memory-Based Approach

Results

Q-Learning



Memory-based

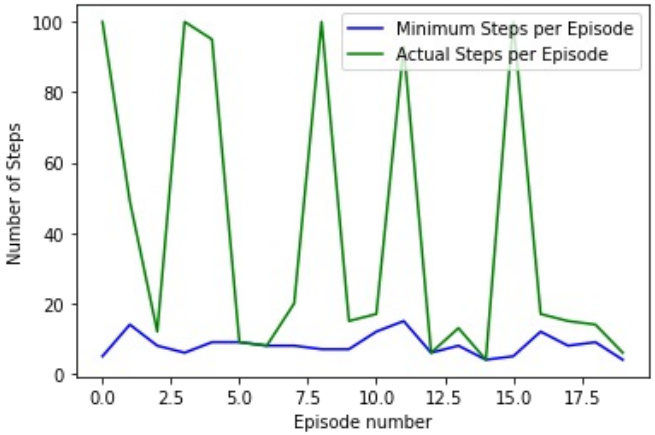


Too Easy? Harder Experiments!

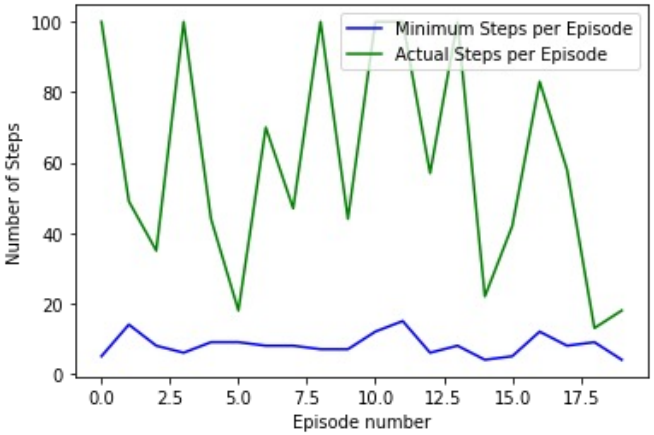
- Set Start and End State to be changing in a 10x10 grid
- Obstacles which block certain squares
- Obstacle locations change every 10 episodes
- Memory Based Approach with Ablations
  - Without Slow Memory Retrieval
  - Without Fast Goal-Directed Network
  - Without Fast Goal-Directed Network and Slow Memory Retrieval

Results

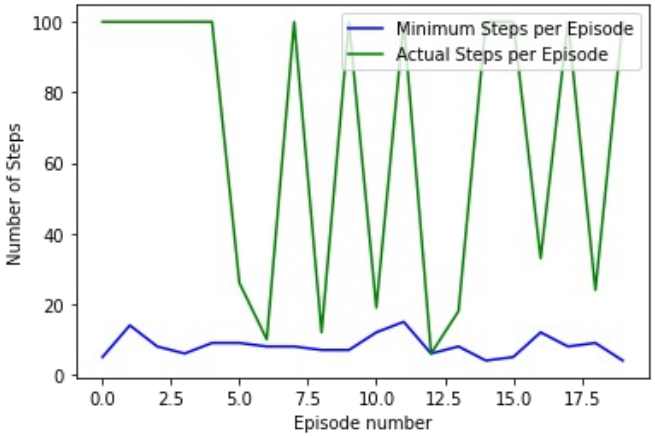
Memory Based:
(baseline)
+ 629 steps
from minimum



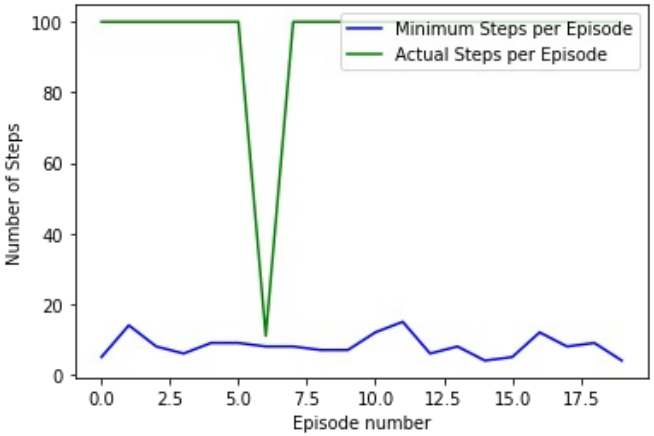
Without
Slow:
+ 1036 steps
from minimum



Without
Fast:
+ 1184 steps
from minimum



Without
Fast & Slow:
+ 1747 steps
from minimum



Results

- Memory mechanism is very robust and can learn even when environment changes
- Count-based mechanism alone is not sufficient to learn well; there is a great benefit to using memory for learning
- Value-based systems are slow to converge and not adaptable in a changing environment

A Tale of Two Pathways

Reward and Pain

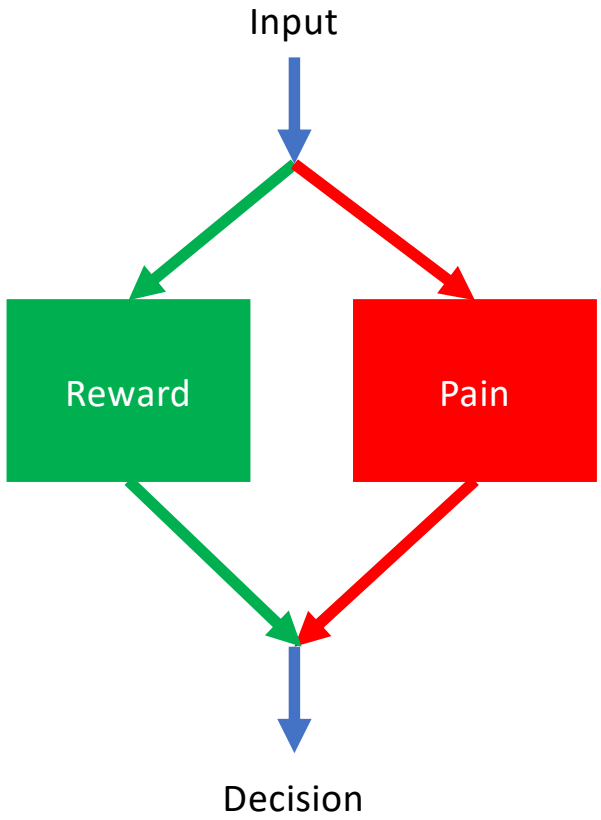
Problems with only Reward

- In real world cases, we may not want to explore everything
  - YOLO: If exploration leads to death of the agent, then it is bad
- If we were to only store the good experiences, we may forget what is dangerous and repeat them unknowingly
- Perhaps a second pathway is necessary for agent's safety

Two Pathways

- Reward Pathway
  - Generally similar to dopamine reward pathway to predict future reward
  - Agent is optimized to repeat actions which give good rewards
  - Subject to anti-reward pathway that encourages exploration
  - Reward pathway can also be the goal-driven pathway

- Pain Pathway
  - Agent remembers painful outcomes and strives not to repeat it
  - Similar to how fear is processed in amygdala and basal ganglia
  - Constrains the agent's actions to avoid future pain
  - Some behavior can already be imbued biologically
    - E.g. Gag reflex, Withdrawal reflex



Two Pathways and Memory

- Learnt experiences can be stored in both the reward and pain pathway
- While reward pathway is more for long-term planning, the pain pathway will provide the short term reflex to prevent the agent from immediate harm
- Both can have a slower memory retrieval system as well as a faster neural network predictor system

Discussion

Questions to Ponder

- How best to explore? Random or goal-directed
- How do we store continuous state and action spaces into memory?
- How do we know how similar two states are when finding out the query-key matching mechanism of retrieving memories?
- How do we define a goal?
- How do we split goals to sub-goals?