Improving Reasoning and Planning Capabilities of Language Models using Reinforcement Learning

Varun Dhanraj, Chris Eliasmith {vdhanraj, celiasmith}@uwaterloo.ca

Centre for Theoretical Neuroscience, University of Waterloo

University of Waterloo

CNRG computational neuroscience research group
compneuro.uwaterloo.ca

CTN Centre for Theoretical Neuroscience

INTRODUCTION

System 1 Cognition

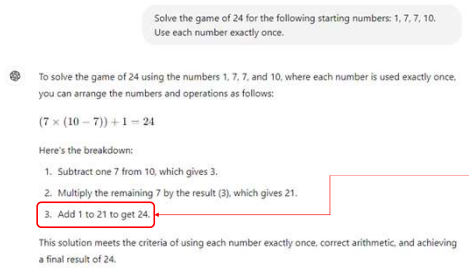
- While LLMs excel at a large number of tasks, they fundamentally correspond to the type of "System 1" thinking humans engage in
- System 1 is characterized by thinking via subconscious pattern recognition
- LLMs are effectively pattern recognition models, providing probabilities of next tokens given a context
- Just like System 1 of the brain, LLMs struggle with consistent and context independent reasoning and planning skills
- System 1 thinking struggles to identify errors, akin to LLM hallucination

Proposed Method

- This study trains a Deep Q-Network (DQN) to more accurately perform mathematical and logical reasoning in the context of the game of 24
  - The game of 24 involves combining 4 numbers together using addition, subtraction, multiplication, and division to obtain 24
- The final hidden layer of the most recent token in the LLM is the state of the DQN, and the actions are the next tokens it can predict
- The DQN is trained via a custom reward function that penalizes mathematical and logical errors while rewarding valid actions

Results Summary

- The results indicate the DQN increases the reward received by **12.46%** relative to a standard LLM
- This framework does not modify the weights of the original LLM, but instead trains a modular DQN network to only affect the behavior of the agent in a particular task
- Also, this framework facilitates continual learning, since the agent gets progressively better at the task as they interact and receive rewards from it



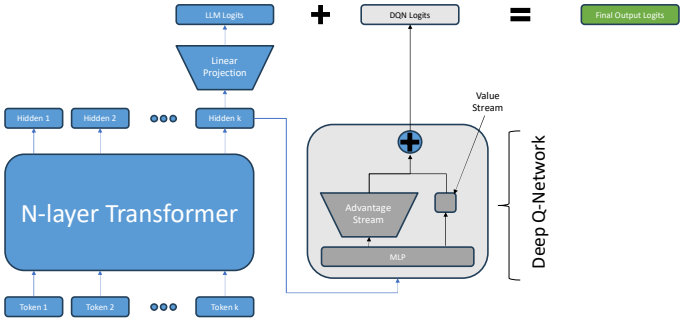
METHODS

Framework Summary

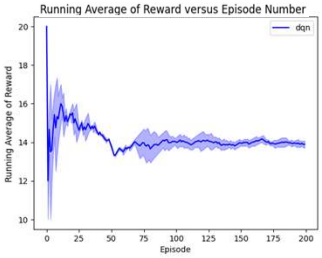
- The standard decoder-only LLM generation is modified by feeding the hidden layer of the most recent token into a DQN, which produces a set of logits which are added to the LLMs regular logits
- This process leaves the original LLM intact, while modifying the final output logits (which determine the next token)
- The DQN is initially set to have all Q-values be 0 (i.e., DQN logits will be 0)
- The LLM attempts to solve problems, and receives rewards based on its performance
- Currently, an external LLM (GPT-4o) is queried to provide a score to the entire answer, and intermediate steps are provided rewards by a custom function

Important Capabilities

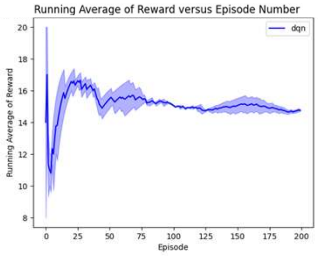
- The system maintains a replay buffer, which stores recent state-action-reward tuples that are used to provide the DQN with decorrelated and diverse training data
- The system utilizes a target network, which is a copy of the DQN that is updated at a set interval, and whose goal is to improve training stability
- Exploration is handled by sampling the next tokens via a SoftMax over the output logits with a temperature σ ($\sigma \to 0$ as the number of episodes $\to \infty$)
- Initially, the agent will observe correct solutions to problems (i.e., imitation learning), which will help the agent choose better actions initially
- The agent is gradually introduced to more difficult problems as the DQN trains (i.e., curriculum learning), allowing the agent to learn easy problems before tackling the more difficult ones



RESULTS



(a) Running average reward of LLM only inference over 200 trials



(b) Running average reward of LLM + DQN over 200 trials

- This experiment used the Llama 3 (8B parameter) model as the LLM
- A simplified version of the game of 24 was used for this problem, where 3 numbers are provided, and the agent must use a mathematical operation to combine the first 2 numbers to obtain the 3rd number
- The average reward obtained by using LLM logits only was **13.90** out of 20, and for the LLM+DQN logits the average reward was **15.65** out of 20
- This indicates that using the DQN improves the average reward by **12.56%**

Examples

```
Prompt: Use one arithmetic operation to combine 12 and 12 to obtain 24.
    → Start with 12, 12. Then multiply 12 and 12 to get 24.0.
Prompt: Use one arithmetic operation to combine 3 and 4 to obtain 12.
    → Start with 3, 4. Then multiply 3 and 4 to get 12.
```

CONCLUSION

- This study provides a novel framework to improve the planning and reasoning capabilities of LLMs
- The procedure involves training a modular DQN, trained on reward signals designed to incentivize logically and mathematically correct actions, while penalizing mistakes
- The results indicates this strategy can achieve a noticeable improvement in the performance of the agent in mathematical and logical reasoning tasks

NEXT STEPS

- Since DQNs are relatively sensitive to hyperparameters (e.g., memory buffer size, learning rate, target network update frequency, etc.), more hyperparameter tuning could result in large increases in performance
- Trying the framework on the full game of 24 (instead of the simplified version) would further showcase the ability of the DQN to prevent mathematical and logical errors, while also facilitating planning
- New techniques can be tried as well, such as model based RL
  - In this framework, the transition model is the portion of the LLM that predicts the next hidden state given the context, and the reward model is the system that determines the reward