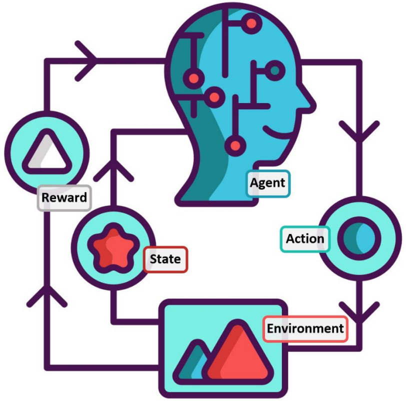
Transformers for Reinforcement Learning

Omar Bahri

Project Summary

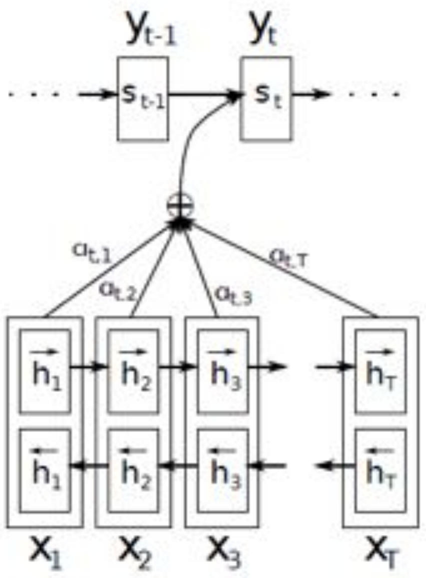
- Transformers are great for sequence modeling.
- By redefining RL as a sequence modeling problem, we can use transformers.
- Can the performance of Decision Transformer on offline benchmark datasets be achieved on a simple problem, with a custom dataset?

HOW?

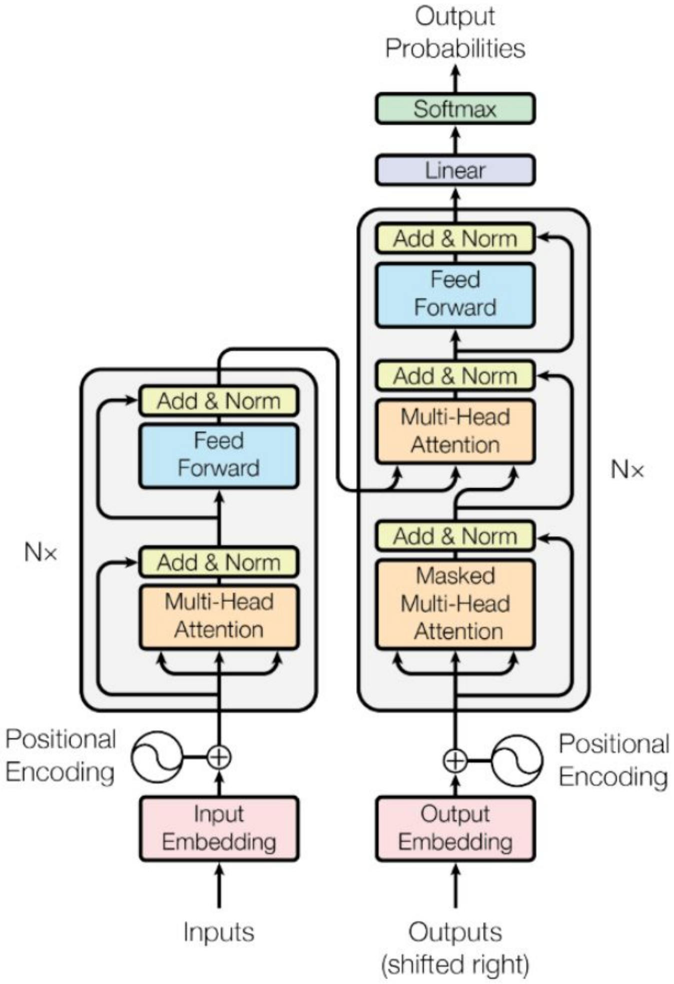


Attention

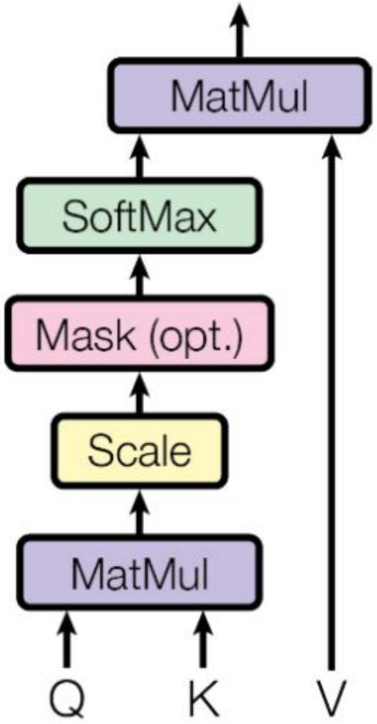
- Constructing a context vector:
  - **Normal LSTM**: take the embedding of the last layer (last hidden state).
  - **With attention**: consider all word embeddings simultaneously.



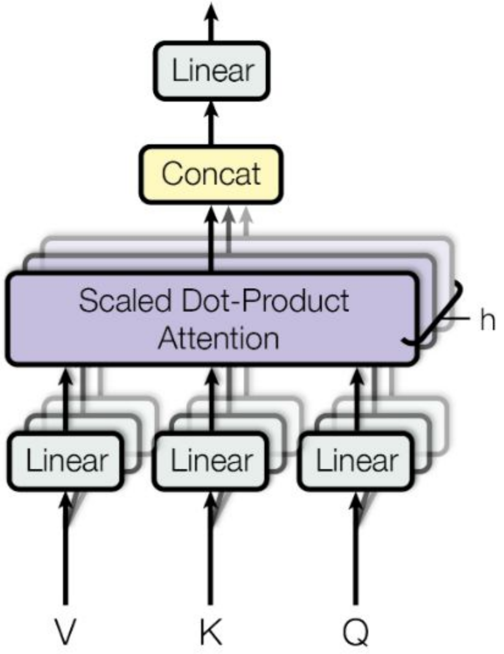
Transformers and Attention



Transformer



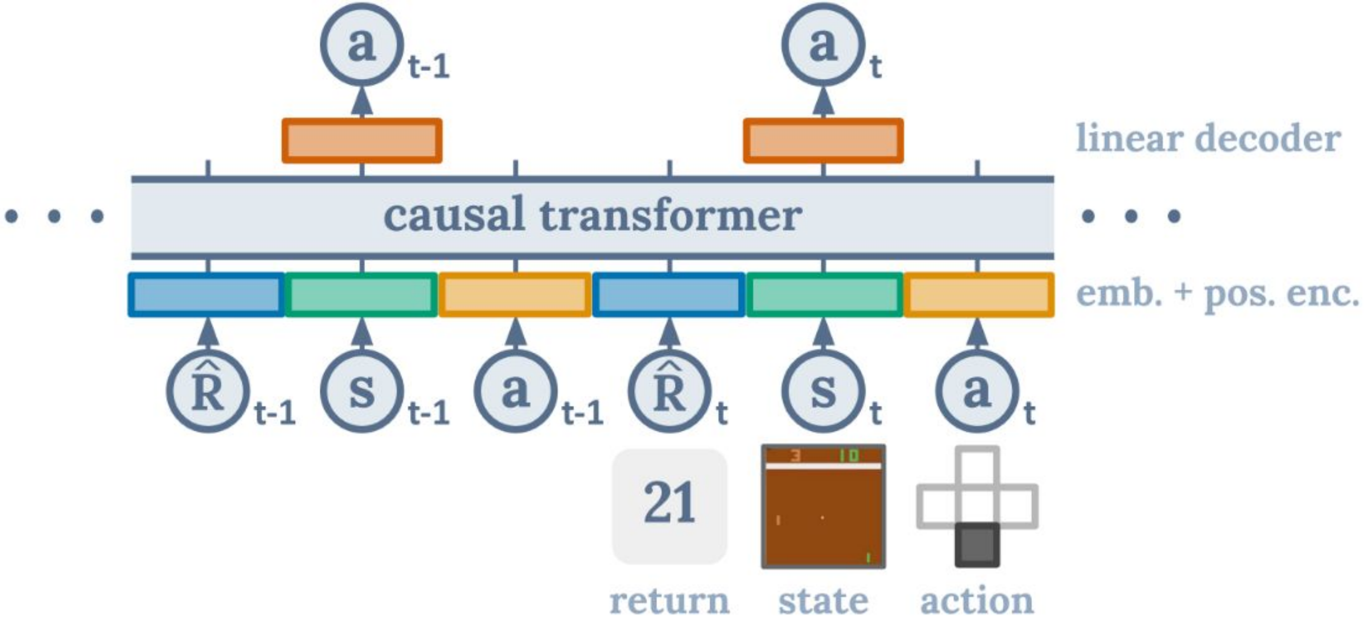
Scaled Dot Product Attention



Multi-Head Attention

Decision Transformer

$$\tau = (\hat{R}_1, s_1, a_1, \hat{R}_2, s_2, a_2, \ldots, \hat{R}_T, s_T, a_T)$$



Methods

1. Generate three datasets using the OpenAI Gym Pendulum-v0 environment, in compliance with the D4RL benchmark format.
   - **pendulum-random**: random agent, 1,000,000 steps.
   - **pendulum-medium-replay**: the replay buffer of a Soft Actor-Critic agent, 1000 steps.
   - **pendulum-medium**: trained Soft Actor-Critic agent, 200,000 steps.

2. Train Decision Transformer on the datasets and compare to Soft Actor-Critic.

Results

- Modifications to default parameters: higher learning rate, smaller embedding size.

- **pendulum-random:** the model did not learn, performance similar to random agent.
- **pendulum-medium-replay:** slightly better than random, worse than SAC.
- **pendulum-medium:** better than random, but still worse than SAC.

HERE I WILL SHOW THE LEARNING CURVES OF SAC AND THE BEST DECISION TRANSFORMER I COULD TRAIN.

I'll also show the average rewards for all models.

Conclusion