CS 8803 DRL Midterm Report - Decision Transformers (Repo)

Aman Garg, Niki Vasan, Ethan Haarer, Karthik Kothuri, Yee Ching Lau

October 24, 2025

1 Introduction

In recent years, transformers have revolutionized the AI landscape, showing incredible ability to scale in various language and vision generation tasks. The Decision Transformer paper reframes reinforcement learning as conditional sequence modelling, forgoing the need for traditional value estimation or temporal difference learning methods.

The core architecture of decision transformers relies on their trajectory representation, where each output action is generated from the ternary input of return, state, and action. This sequence enables the transformer to learn meaningful patterns and conditionally generate new actions at test time. The returns for each triple in the trajectory are captured as returns-to-go, or the sum of the future rewards as opposed to past rewards. The last K timesteps are fed into the transformer, yielding 3K total context tokens at training (2K at inference). Each element of the triplet is passed through a linear layer to project the input to its respective embedding dimension. If the states are visual, a convolutional layer is used instead. These are passed through a standard GPT-based architecture, and a final linear decoder layer maps the latent representation output to the best action for the agent. This approach enables direct credit assignment through self-attention layers, avoiding slow Bellman backups and is robust to sparse or distracting rewards.

During model training, the authors train their transformer using a sequence modelling objective using the collected experiences from the data so there is no need to bootstrap for long-term rewards. Using offline trajectory data, the predicting head that corresponds to each state token is trained to predict the action within the triplet. This is done with either MSE for continuous actions or crss-entropy for discretized action spaces. Where traditional RL paradigms often use Bellman backups, transformers' attention heads help to avoid distractor signals given from the state, and help the model to focus on training core behavior instead of following distracting signals.

The Decision Transformer obtains competitive results, either matching or exceeding other offline RL baselines like BEAR, CQN and QR-DQN. On long-

horizon credit assignment tasks like Key-to-Door, the Decision Transformers significantly outperforms traditional TD methods like CQL, with a 94.6% success rate vs 13.3%. However, there are some limitations in the design. The supervised loss is incredibly simplistic, the state encoders are shallow, and the trained environments were done offline so there were limited opportunities for exploration. Although many parts of this design have claimed to scale well, their combination still requires testing to further solidify the relationship that extracted features and environmental complexity scale together.

2 Modification Description

A core aspect of the Decision Transformers architecture is the linear or convolution layer used to map the input triplet of returns, state and action to the embedding space. The convolutional layer provided in the paper a simple 3-layer CNN. At the time, this may have been a common way to map other modalities to the token space; however in the four years since this paper's debut, there has been a wealth of research on improving these methods including CLIP and ViT-based encoders.

We propose that we modify this architecture to replace the standard, small CNN used to encode visual state information and instead test the ability of other visual encoders, such as Vision Transformers (ViT) or autoencoders to connect the visual input to the decision transformer. This would enable us to update the existing architecture with modern approaches, as well as ablate these encoders to demonstrate how each affects performance and learning.

3 Motivation

The existing architecture extracts enough detail in order to sequence simpler tasks from the Atari datasets, however this may be challenged in more visually complex environments. If the goal is to optimize performance for more complex environments or real-world data, this existing architecture is bottle-necked by the current mechanism for extracting visual state information.

Furthermore, stronger models like ViTs have inductive biases that could enhance performance in offline RL, where we are heavily constrained by the dataset size. Similarly, improved visual comprehension could augment the self-attention mechanism in the decision transformer, which is the backbone for the successful credit assignment process. In general, by utilizing more updated encoding methods, we hope to observe an improvement in overall learning and performance, extending this work to real-world applications.

4 Feasibility

This modification should be relatively straightforward to implement. Most machine learning frameworks like PyTorch have packages that support the use of

black-box models. We will define a new encoder and add it to the GPT module provided in the original code, ensuring that the output dimensionality of the encoder correctly maps to the token space of the transformer.

For testing, we will use the MuJoCo hopper dataset used in the walkthrough and plan to add other environments like Atari or Half-Cheetah. Ideally, we would like to ablate multiple different encoders, including ViT, different CNN architectures, autoencoders or other models. Time-permitting, we can also test variants of a given model class (e.g. small vs large ViT); however, if a ViT is too large or computationally infeasible for our resource constraints, we can always fall back on simpler architectures like ResNets.