Investigating Neural Constraints on Learning in Artificial Neural Networks

James Read, Janak Sharda, Xiao Zheng, Kevin Patino Sosa

***Abstract*—Neural networks are powerful models that are widely used in a variety of fields. However, their mechanisms for learning and representing information are not fully understood. Traditional neuroscience techniques revealed that a biological neural network develops constraints when learning a new task, which can be caused by the connectivity of neurons and can affect the difficulty of learning. To gain insight into the learning mechanisms underlying neural networks, we investigate whether similar constraints exist in deep Q networks (DQNs), which are a type of deep neural network trained to perform reinforcement learning tasks. We trained the DQN on a cursor-moving task that is similar to the experiment conducted on monkeys, and investigated the constraints of the task by exploring the factors influencing the learning process.**

***Index Terms*—Deep Q-Learning, Dimension Reduction, Intrinsic Manifold, Factor Analysis, Multi-Layer Perceptron**

I. Introduction

Neural networks are computational models that can perform a wide range of tasks, such as natural language processing, computer vision, and reinforcement learning. However, the mechanisms by which these networks learn and represent information are not well understood. One way to potentially gain insight into the inner workings of neural networks is to use techniques traditionally for studying neural circuits in the brain on neural networks.

One such technique in neuroscience is to examine the representations and constraints biological neural networks develop when learning a new task. These representations arise from the activity patterns of the neurons and from the nature of the task. Constraints in learning can arise from the connectivity of the neurons and can affect the ease or difficulty of learning. For example, Sadtler et al. [1] showed that non-human primates could more readily learn to control a brain-computer interface (BCI) using neural activity patterns that were consistent with the intrinsic manifold of their motor cortex, which reflects the natural patterns of comodulation among neurons. Conversely, learning was impaired when the BCI required activity patterns that were outside of the intrinsic manifold. This suggests that the existing structure of a neural network can shape learning by facilitating or impeding certain patterns of neural activity [1].

In this project, we aim to investigate whether similar constraints exist in artificial neural networks, specifically deep Q networks (DQNs) that are trained to perform reinforcement learning tasks. DQNs are a type of deep neural network that learn to estimate the optimal action for a given state, using a combination of temporal difference learning and experience replay [2]. DQNs have achieved remarkable success in learning to play various Atari games from raw pixel inputs, surpassing human performance in some cases [2]. However, it is unclear how DQNs learn and represent these complex tasks, and what factors influence their learning process. To address this question, we trained a DQN to play a simple video game that required moving a cursor to a target location on a screen and designed experiments for the DQN that parallel the ones conducted on monkeys in [1]. The results of our experiments show that the neural activity of an artificial neural network can be reduced to low-dimensional intrinsic manifold which can be used to control the cursor's movement exactly like was done by Sadtler et al with the neural activity in monkeys. Due to time constraints we were not able to complete the manifold perturbation experiments on the neural network but we believe we have sufficiently motivated the effort to continue the experiments in the future.

II. Background Information

A. Neural Constraints Imposed by Intrinsic Manifold

To study the extent of difficulty to generate a new patterns of neural activity through learning, Sadtler et al. [1] explores the role of neural constraints during learning in the context of brain-computer interfaces (BCIs). First, they trained the non-human primates to move a cursor on a screen to a goal position by modulating the neural activity of a group of roughly 100 neurons in the primary motor cortex of the monkeys. To do this, the authors used a dimensional reduction technique to reduce the activity of the neurons to a 10-dimensional mapping termed the "intrinsic manifold." The manifold captures the prominent patterns of co-modulation among the recorded neurons, which reflect underlying network constraints. After the manifold is calibrated, a Kalman filter was then applied to the manifold to control the velocity of the cursor allowing the monkey to control the cursor by modulating its neural activity. Once the intrinsic manifold was calibrated, the authors would manually perturb the manifold used for controlling the cursor and observed if the monkey could modulate their neural activity patterns to match the new manifold.

From these experiments, the authors observed that the monkey could learn to control the cursor more quickly and accurately when the perturbation manifold stayed within the mapping of the intrinsic manifolds. However, learning was severely impaired when the perturbed manifold required activity patterns that were outside of the intrinsic manifold. The results suggest that activity patterns of individual neurons in the motor cortex are constrained by the co-modulation patterns, which indicates the existing structure of the neural

network in the brain can shape learning by facilitating or impeding certain patterns of neural activity.

B. Deep Q-Learning

Reinforcement learning is a machine learning methodology in which an agent is taught to complete a control task. Such tasks include controlling robots, optimizing decision policies and playing video games. These algorithms work by allowing an agent to explore the task environment while providing rewards based on the agent's actions. Positive rewards are intended to encourage desired behaviors and negative rewards discourage unwanted behaviors. Deep Q-Learning is a type of reinforcement learning algorithm where the actions the agent takes are determined by a deep neural network that predicts what the optimal action is based on the current state of the agent's environment. The goal of our project was to determine if the neural network in a Q-learning algorithm trained to play a similar cursor moving game would have the same properties observed by the neural circuits studied in [1]. Specifically, we wanted to determine If the "neural activity" in the neural network could be used to construct an intrinsic manifold and if the manifold can be used as an input space to control the cursor. We define the "neural activity" of a neuron in the network as the magnitude of its output during each time step of the game. A higher magnitude corresponding to a larger firing rate in a biological neuron.

C. Dimensionality Reduction

The curse of dimensionality refers to the fact that the amount of data required to estimate a function of several variables increases exponentially as the number of features or dimensions of data increases, which can cause a variety of problems, such as overfitting, sparsity, and difficulty in visualizing and interpreting the data. For example, in our study, the neuronal output has 100 dimensions, with each dimension representing the activity of a neuron. However, the neural activity patterns might correlate with each other so that the overall characteristics can be explained in a low-dimensional space (or manifolds). Therefore, reducing the data dimensionality into the intrinsic manifold is essential to extract the neural activity patterns better.

In this project, we selected factor analysis to reduce the dimensions. Factor analysis is a procedure to explain the intercorrelations among the observed variables by constructing fewer common factors. The common factors are unobservable random variables that are presumed to exist in the population and constitute the observed variables. The unique factors in each observed variable are part of this variable that is uncorrelated with all the other observed variables, including its measurement error.

There are other popular methods to extract the manifolds from neural activities, such as Principal Component Analysis (PCA) [3]. However, we preferred factor analysis to PCA for several reasons. First, factor analysis assumes the neural activity is influenced by a set of unobservable factors, which could be interpreted as the manifolds that generate these neural signals. In contrast, PCA assumes the principal components are a linear combination of original signals and therefore are less easily interpretable. Second, factor analysis independently models variance for each neuron, while PCA assumes equal variance across neurons. This difference is vital since the variance of spiking variance is considered proportional to the mean firing rate [4]. Additionally, factor analysis allows non-Gaussian or non-linear data, while PCA assumes the data are linearly related and normally distributed. [6]

III. Methodology

A. Video Game Design

Game: The game comprise of a $n \times n$ grid, where we set one position as the target position. The target position is randomly chosen from a set of pre-defined target positions. Next, we randomly initialize the cursor position on the grid. The user can take either of the four actions: up, down, right or left (0,1,2,3) and the task is to move the cursor to the target position in least number of steps.

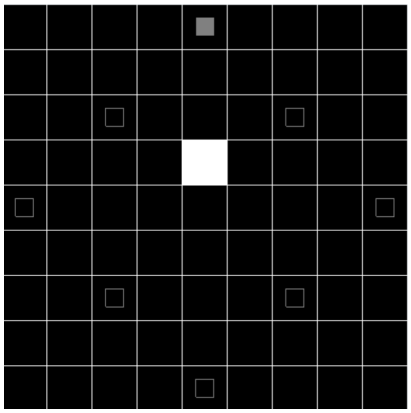

Fig. 1. Screenshot of a 9×9 version of the game at one time step. The white square represents the cursor, the small gray square is the goal position and the empty small squares are the other potential goal positions.

B. Neural Network Construction

We built a DQN in PyTorch which can take the current state of the game and predict the reward of each possible action. The agent then picks the action which the neural network predicts to have the highest reward. The current state is defined as a $n \times n$ array, where the cursor position is marked as '1', target position as '-1' and the other elements to be zero. The neural network is a single-layer fully connected network with 100 hidden neurons. We reward the agent based on the distant between the position of of the cursor and the goal position. As the cursor moves closer to the goal position its reward is increased. The reward is accumulated over each episode, where one episode is described as reaching target position from the initial position successfully. The DQN network maximizes the cumulative reward by updating the neurons in the network. For use in estimation from the intrinsic manifold, the output of each neuron is sent to estimation algorithm instead of to the final layer of the neural network. This prevents the network from selecting the action based on the network's outputs.

C. Factor Analysis

We used the python package FactorAnalysis to build the analytical model. The number of factors were estimated by the eigenvalues of the correlation matrix of factor loadings (which was 6 in this study). Then, the final factor analysis model was constructed by the selected factor numbers, with the rotation methods as "varimax" to maximize the variance among the factor loadings. Factor scores were computed as linear combinations of the observed neural activity, weighted by their factor loadings. Using these factor scores, the original high-dimensional data (100 neurons) were transformed into a 6-dimensional manifold.

D. Trajectory Prediction

For predicting the trajectory using the intrinsic manifold, the project initially attempted to follow the approach used by Sadtler et al. [1] by implementing a Kalman filter. However, defining the state dynamics using the intrinsic manifold proved to be a challenging task. Furthermore, discovering and defining the state dynamics required a deep understanding of the underlying dynamics and was highly dependent on the choice of features. As a result, this approach was not feasible for the current project.

A Multi-Layer Perceptron (MLP) was proposed as an alternative approach to estimating the action from the intrinsic manifold. MLPs are known for their ability to learn complex non-linear relationships between the input and the output. Furthermore, compared to the Kalman filter, MLPs are more flexible and do not rely on a predefined dynamics model. [6] Instead, MLPs can learn the dynamics directly from the data, making them a suitable option for trajectory prediction and estimating the actions using the intrinsic manifold as input. [6]

IV. Results

A. Neural Network Training

During the training process, we use a hyper-parameter that picks between the DQN's predicted action and a random action to allow the agent to explore the environment. Initially this hyper-parameter selects the random action with higher probability to garner exploration and over time is tuned to select the output from the DQN with higher probability as the network learns. In Fig. 2 we see that as we train the network over more episodes the agent maximizes the total reward over each episode. We observe that after about 200 episodes the network converges and is able to play the game with perfect accuracy.

B. Extracting Intrinsic Manifolds by Factor Analysis

The original neural activity obtained from the neural network (II.B) can be noisy and highly correlated, which makes it difficult to interpret. Therefore, we computed the low-dimensional intrinsic manifolds using factor analysis. We identified 6 factors that explain most correlations among variables based on the Scree plot (Fig. 3). The correlation between each neuron and each factor were estimated by the factor loadings

Fig. 2. Total reward the agent receives during each episode as a function of training episodes. For each action, a negative reward proportional to the distance between the cursor and the goal position is given and a reward of $+10$ is given when the cursor reaches the goal.

(Fig. 4), from which we could observe several neurons were significantly influenced by a same factor, for example, neuron 33 and neuron 46, indicating a high correlation between these two neurons.

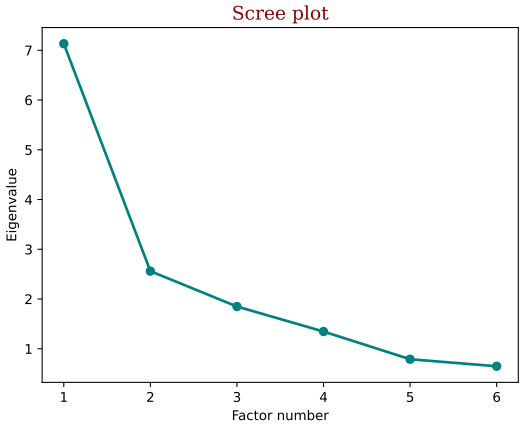

Fig. 3. Scree plot showing the eigenvalues of each estimated factor.

Then, we used the estimated factor loadings to compute the factor scores of each neuron, which represent the estimated scores of each neuron on the 6 factors (i.e. intrinsic manifold), based on their original activity (Fig. 5). Clustered factor scores indicate similar patterns underlying the actions, which were estimated by the MLP model in the next section. As observed from Fig. 6, the manifold constructed by scores of the first two factors can separate the data well. These factor scores were used for further trajectory inference.

C. Trajectory Inference

We trained a MLP to predict the agent movement trajectories, which takes the factor scores obtained from factor analysis as input, and outputs the 4 actions (up, down, right, and left) of the agent. As a result, the model can accurately

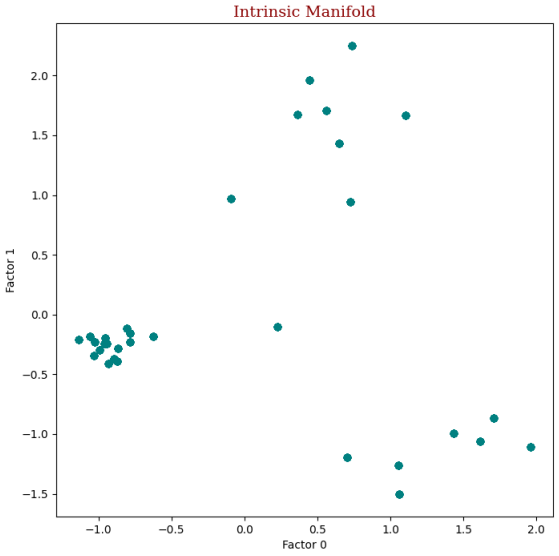

Fig. 4. The intrinsic manifolds visualized by the factor scores of the first two factors. Each point represents an observation.

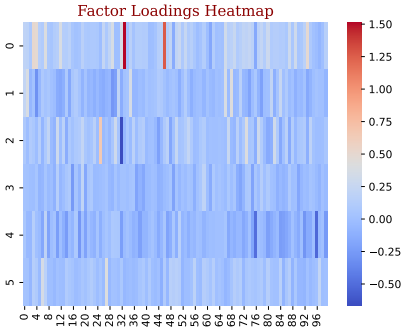

Fig. 5. Factor loadings of the estimated factor on each neuron.

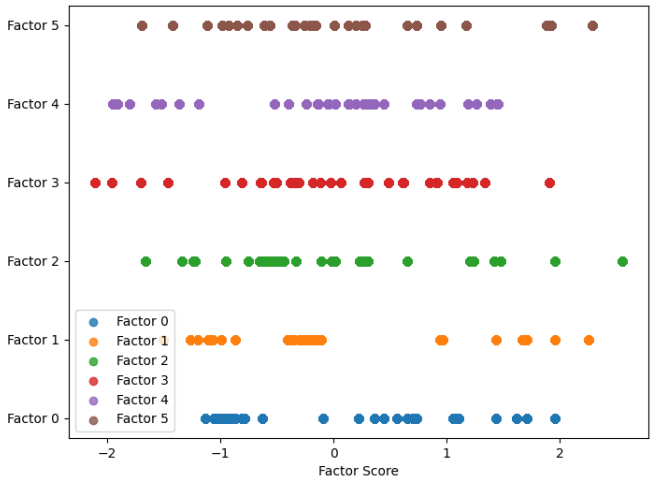

Fig. 6. The factor scores computed by factor loadings. Each dot represents an observation.

predict the agent actions (**model score: 0.99998**, **MSE: 2.48e-05**). An example of predicted trajectory is shown in Fig 7, which goes with the shortest Manhattan distance within a limited action space.

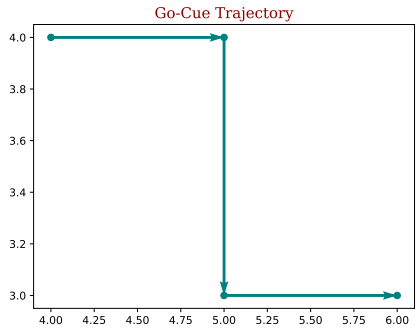

Fig. 7. Example of a predicted agent trajectory. (4, 4) is the starting point, and (3, 6) is the goal location. The trajectory takes 4 steps in total.

V. Discussion

We conclude from our investigation into the learning mechanisms underlying deep Q networks that these networks are constrained in the same ways as biological neural circuits. Our experiments on a cursor-moving task have shown that the neural activity of a DQN can be reduced to a low-dimensional intrinsic manifold, similar to what has been observed in non-human primate brains. Additionally, our findings highlight dimensional reduction techniques' critical role in extracting the intrinsic manifold and enabling accurate predictions of the go cue dynamics using a perceptron model with low layers (6,4). By reducing the dimensionality of the data, we were able to extract meaningful features that enabled us to model the complex dynamics of the go cue effectively. Furthermore, it underscores the importance of incorporating advanced data processing techniques such as dimensional reduction to enhance machine learning models' accuracy and efficiency. Overall, we believe our approach of using techniques traditionally used in neuroscience research to investigate artificial neural networks present a novel and useful way to study these powerful neural networks.

VI. Data & Code Availability

The code developed for this project is available at GitHub

References

[1] Sadtler, P. T., Quick, K. M., Golub, M. D., Chase, S. M., Ryu, S. I., Tyler-Kabara, E. C., ... Batista, A. P. (2014). Neural constraints on learning. Nature, 512(7515), 423-426.

[2] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Hassabis, D. (2015). Human-level control through deep reinforcement learning. nature, 518(7540), 529-533.

[3] Levi, R., Varona, P., Arshavsky, Y. I., Rabinovich, M. I., Selverston, A. I. (2005). The role of sensory network dynamics in generating a motor program. Journal of Neuroscience, 25(42), 9807-9815.

[4] Nicolelis, M. A., Baccala, L. A., Lin, R. C., Chapin, J. K. (1995). Sensorimotor encoding by synchronous neural ensemble activity at multiple levels of the somatosensory system. Science, 268(5215), 1353-1358.

[5] Shadlen, M. N., Newsome, W. T. (1998). The variable discharge of cortical neurons: implications for connectivity, computation, and information coding. Journal of neuroscience, 18(10), 3870-3896.

[6] Cunningham, J., Yu, B. Dimensionality reduction for large-scale neural recordings. Nat Neurosci 17, 1500–1509 (2014). https://doi.org/10.1038/nn.3776

[7] Cristian S. Calude, Shahrokh Heidari, Joseph Sifakis, What perceptron neural networks are (not) good for?, Information Sciences, Volume 621, 2023, Pages 844-857