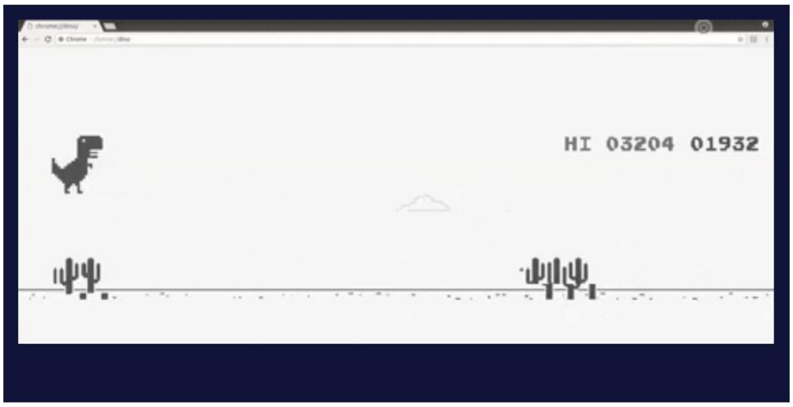
046211 - Deep Learning - Project Proposal

Name	**ID**	**Campus E-Mail**
Ari Frummer	314974023	ari.frummer@campus.technion.ac.il
Gal Kinberg	318263076	gal.kinberg@campus.technion.ac.il

Project Name: Teaching a Neural Network to Play Chrome's Dino Run Game

Short Description/Plan: In this project we will use a neural network to teach an agent to play chrome's dinosaur game, also known as Dino Run, directly from screenshots of the game.



This is an interesting task since the network needs to learn at the same time both meaningful representations of the high-dimensional input, and a good policy for playing the game.

We intend to base the project on the Deep Q-Network (DQN) architecture originally proposed by DeepMind and used by them to teach agents to play Atari games [1]. This architecture uses a convolutional part that acts as a feature extractor, and then a fully connected part that acts as a state-value estimator and trained with the Q-Learning algorithm.

We intended to explore how different methods we learned in the course effect the network's performance (such as augmentation, regularization, adding batch normalization), and if we can use a pre-trained convolutional network for the feature extraction part to get better performance. We also intend to explore newer DQN architectures, such as Dueling DQN, and compare their performance to the original method.

Since this is a RL scenario, we do not need any external dataset, as the data is the actual game as the agent plays it. We will compare the results to a previous project that used a DQN on this game [2].

(We also note that we have previous experience in reinforcement learning from the EE course 046194 Planning and Learning in Dynamic Systems)

Available Resources: [1] "Playing Atari with Deep Reinforcement Learning", DeepMind, https://arxiv.org/abs/1312.5602
[2] "Build an AI to play Dino Run", Ravi Munde, https://blog.paperspace.com/dino-run
[3] http://https//github.com/Paperspace/DinoRunTutorial

Does your project build upon previous projects? Yes, the project is based on the blog post in [2]. The previous project was implemented in TensorFlow and used only the original most basic DQN, with no augmentation or normalization. We intend to implement the algorithm in PyTorch, and explore newer DQN architectures and methods.