Learning to Act by Predicting the Future

Yonatan Deloro - Amine Kheldouni

Ecole des Ponts ParisTech
https://github.com/AmineKheldouni/DirectFuturePrediction

January 14^{th}, 2019

Plan

Section 1

Introduction

Motivations

Classic approaches to learn goal-directed behavior generally falls into the reinforcement learning field.
Several limitations in complex sensory environments :

- Scalar reward signal
  vs. rich feedback at various time offsets
- Designed to learn to behave with respect to only one given goal
  vs. goal can change dynamically at test time

Motivations

In contrast, we will aim to study DFP, a supervised learning-based approach to learn how to behave in complex sensory environments. The model learns to act based on :

- **Sensory stream :** raw sensory inputs of a complex and dynamic three-dimensional environment.
- **Measurements stream :** intrisic measurements (low-dimensional variable characterizing the state)
- **Goal :** vectorial goal defined with respect to the selected measurements

Section 2

The model

Direct Future Prediction modelling

- The observations o_t are formed with the raw sensory input s_t and a set of measurements m_t.
- Goals can be expressed as linear combinations of future measurements ($u(f;g) = g^T f$). f being the differences of future and present measurements for a temporal offset τ :

$$f =< m_{t+\tau_1} - m_t, ..., m_{t+\tau_n} - m_t >$$

- At test time, the agent chooses the action maximizing the predicted outcome :

$$a_t = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\, g^T F(o_t, a, g; \theta)$$

where F is a parameterized function approximator of f to be learnt.

The architecture of f the function approximator

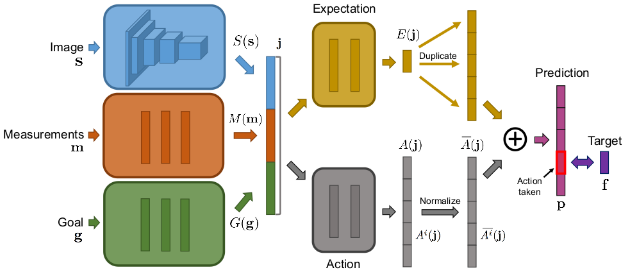


Figure – Network architecture modelling the function approximator F

The training

- At train time, goal can be fixed or generated randomly at each episode.
- Agent follows an ϵ-greedy policy, where ϵ decreases throughout time (ie. exploration decreases)
- F trained with experiences collected by the agent using experience replay : mini-batch selected randomly among a fixed-size memory of most recent experiences.

$$\text{Loss :} \quad \mathcal{L}(\theta) = \sum_{i=1}^{N} \|F(o_i, a_i, g_i; \theta) - f_i\|^2$$

Implementation used and refinements

We forked Mr. Felix Yu's repository[2] for DDQN and DFP.

Our changes :

- Moved from the DDQN implementation to a DQN model
- Adapted the sensory inputs to match the original DFP model (one sensory image)
- Added random goal training regime, and a test session
- Enabled multiple environments (D2, D3)
- Implemented a logger engine for results gathering and experiences

Section 3

Experiments

Experiment 1 : DFP against DQN

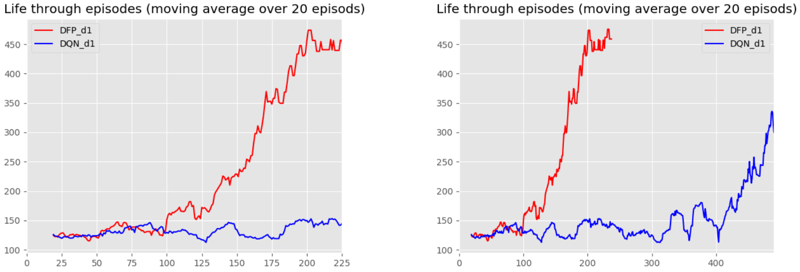


Figure – Agent's life in environment D1 ("health gathering")

Experiment 2 : Scalar reward against vectorial feedback

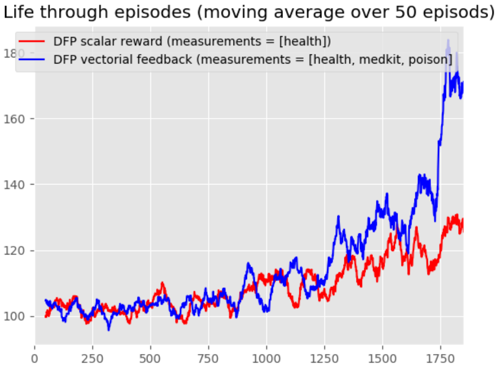


Figure – Agent's life in environment D2 ("health gathering supreme")

Experiment 3 : Predicting measurements at multiple temporal offsets

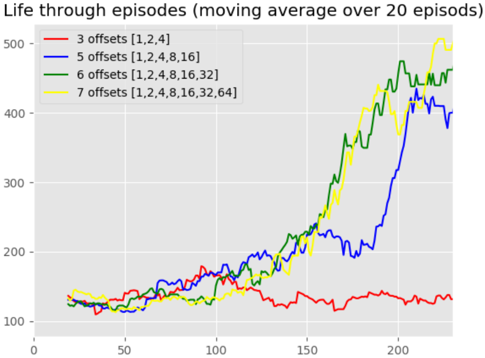


Figure – Agent's life in environment D2 ("health gathering supreme")

Experiment 4 : Goal-agnostic training

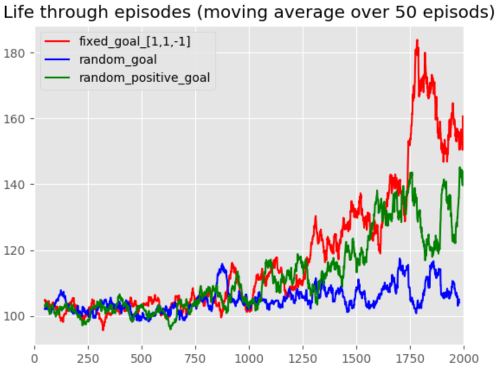
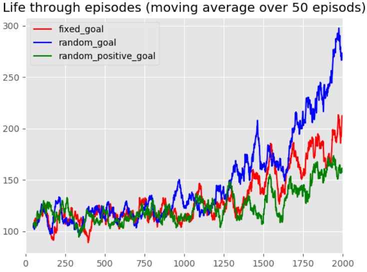


Figure – Agent's life in environment D2 ("health gathering supreme")

Experiment 4 : Goal-agnostic training



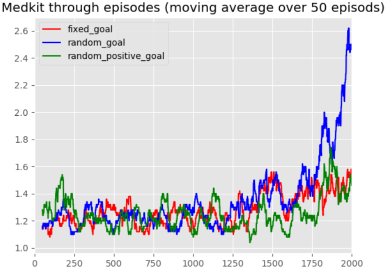


Figure – Results for agent's life (*left*) and medkits (*right*) on D3 (*red* : fixed, *green* : random $[0, 1]$, *blue* : random $[-1, 1]$)

Experiment 4 : Goal-agnostic training

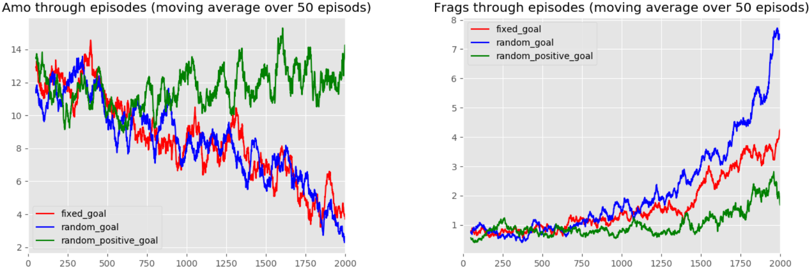


Figure – Results for agent's ammo (*left*) and frags (*right*) on D3 (*red* : fixed, *green* : random $[0, 1]$, *blue* : random $[-1, 1]$)

Experiment 4 : Goal-agnostic training

$$m = (ammo, health, frags)$$

Test goal	Fixed goal			Random goal $[0, 1]$			Random goal $[-1, 1]$		
	A	H	F	A	H	F	A	H	F
$(0.5, 0.5, 1)$	3.4	271.5	6.5	**3.7**	**283.7**	**6.9**	2.4	265	2.6
$(0, 0, 1)$	1.5	**298.1**	2	4.3	291	6.3	**5.9**	240.3	**6.9**
$(0, 1, 0)$	21.6	195	0	**23**	**210.6**	0	22.2	207	**1**

Table – Average scores in environment D3 (each group of three columns is a train regime and each row is a test-time goal)

Section 4

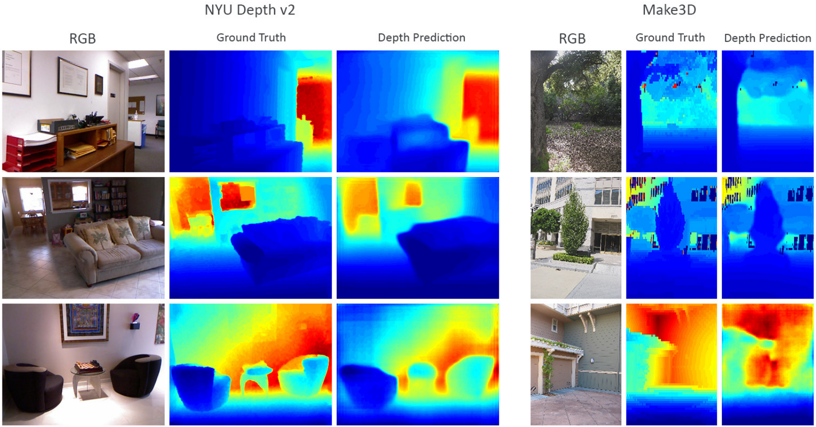
Improvements

Adding depth perception to the sensory input

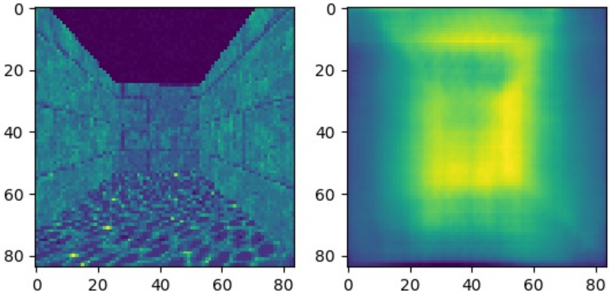
We do not have a pair of stereo images here.

- Prediction from motion : DepthNet "End-to-end depth from motion with stabilized monocular videos"
  - only for translational movement of the camera, lack of rotation, no distortion !
- Prediction from a single image : FCRN for Depth Map prediction.
  " Iro Laina and al., "Deeper Depth Prediction with Fully Convolutional Residual Networks".
  - ResNet50 where fully connected layer replaced by new up-sampling block
  - No post-processing/refinement step
  - Input : (304,228,3) / Prediction : (160,128)
  - Pre-trained on NYU-Depth-v2 dataset or Make3D, but weights only available for tf framework for NYU-Depth-v2

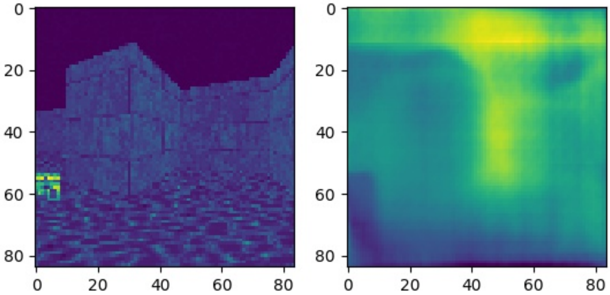
Adding depth perception to the sensory input



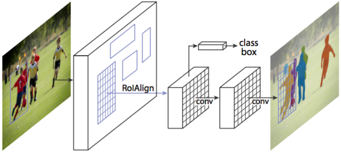
Adding depth perception to the sensory input



Adding depth perception to the sensory input



Segmentation of the sensory input



- Object detection with Fast-RCNN (Region Proposal Network and classification)
- Fully convolutional network (FCN) on Region of Interest (RoI)

$\Longrightarrow$ Segmentation Masks

Segmentation : first example

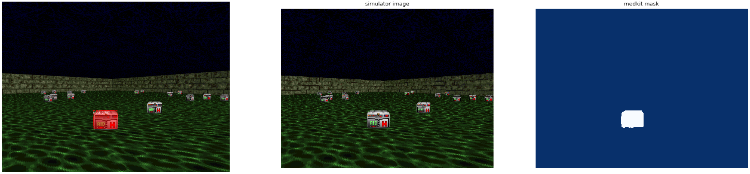


Figure – Image segmentation of a nerby medkit in VizDoom simulator

Segmentation : second example

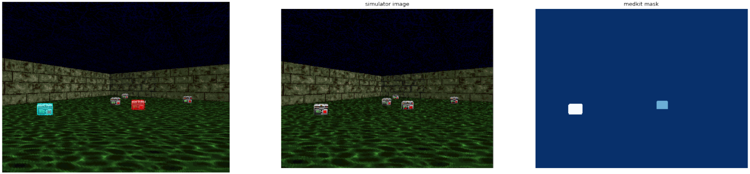


Figure – Image segmentation of multiple medkits in VizDoom simulator

Improving DFP model

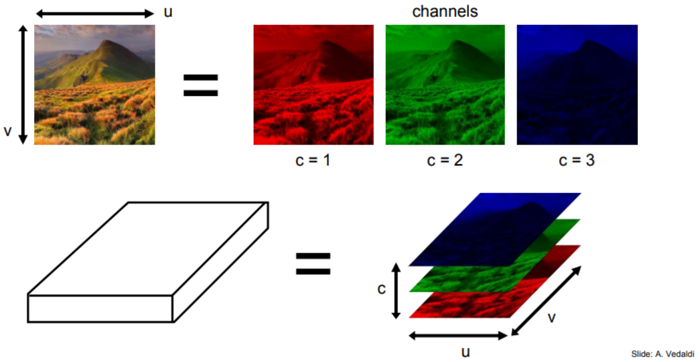
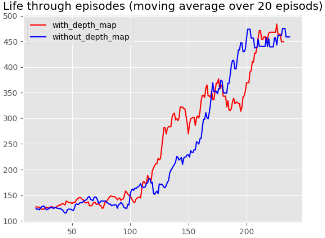


Figure – From the (R,G,B) channels to (gray image, depth map, segmentation image)

Results with Depth Map



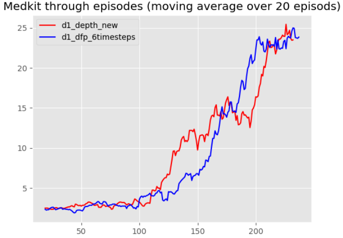


Figure – Results fore agent's life (*left*) and medkits (*right*) with and without a depth map channel in D1

CartPole environment

The CartPole environment is a basic reinforcement learning problem where the system is an inverted pendulum

Goal : Perform an action $\{left, right\}$ on the car to keep the pendulum inverted.

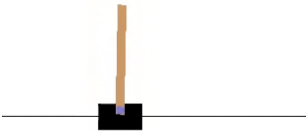


Figure – CartPole environment rendering

CartPole : DFP against DQN

- **Sensorimotor control :** An image of the environment.
- **Measurements stream :** Accumulated reward.
- **Goal :** Positive goal over the accumulated reward ($g = 1$)
- RL Coach module on Intel's GitHub repository[3].

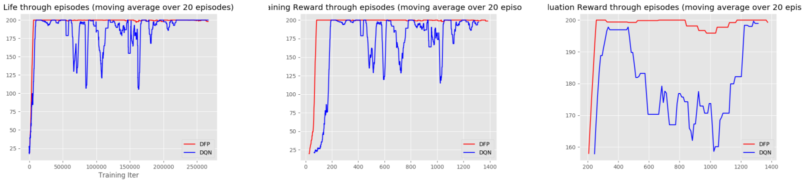


Figure – Life, train and test reward for CartPole for DFP vs DQN (*left :* Life, *middle :* Train reward, *right :* Test reward

Conclusive remarks

- Experimental illustration of DFP advantages in sensory environments :
  - vectorial reward ⇒ accelerated training.
  - learning without fixed goal at training time ⇒ dynamic goal at test time
- Improvement trial to enrich the perception module, predicting features directly from the image input.
  - Need for transfer learning
  - Computational challenge (prediction time for depth map / segmentation)
  - Open question : add directly measurements from these predicted features ? (eg. distance to the closest healthpack)

What we learnt and liked !

- Work on cutting-edge approach for behavioral learning
- Read and learn about Deep Reinforcement Learning and Computer Vision
- Practice with experimental study process and "code integration"
- Practice with computation engines : Cuda, Tensorflow, Google Cloud's Computer Engineé

This project was entertaining and genuinely interesting in a Deep Computer Vision and Learning perspective, as well as in logistics assimilation and ressources gathering ! Thanks.

Thank you !

References

- Alexey Dosovitskiy, Vladlen Koltun. Learning to Act by Predicting the Future https://arxiv.org/abs/1611.01779
- Felix Yu. GitHub open-source DFP implementation. https://flyyufelix.github.io/2017/11/17/direct-future-prediction.html.
- Intel® Nervana™ - Artificial Intelligence Products Group. CartPole DQN/DFP implementation. https://github.com/NervanaSystems/coach
- Matterport, Inc. Mask R-CNN implementation in Keras and TensorFlow https://github.com/matterport/Mask_RCNN
- Iro Laina and al. Deeper Depth Prediction with Fully Convolutional Residual Networks https://github.com/iro-cp/FCRN-DepthPrediction