Deep Apprenticeship Learning for Playing Video Games

Miroslav Bogdanovic[1], **Dejan Markovikj**[1], **Misha Denil**[1] and **Nando de Freitas**[1,2]
[1]University of Oxford, UK
[2]CIFAR Fellow
{miroslav.bogdanovic, dejan.markovikj}@gmail.com,
{misha.denil, nando}@cs.ox.ac.uk

Abstract

Recently it has been shown that deep neural networks can learn to play Atari games by directly observing raw pixels of the playing area. We show how apprenticeship learning can be applied in this setting so that an agent can learn to perform a task (i.e. play a game) by observing the expert, without any explicitly provided knowledge of the game's internal state or objectives.

Background

Mnih et al. (2013) recently demonstrated that it is possible to combine Q-learning with deep learning to play Atari games. Their method learns to maximize the score of the game, which is explicitly provided to the model during training.

We extend the approach of Mnih et al. (2013) to the apprenticeship learning setting, allowing our agent to learn to play without being provided with any explicit knowledge of the game score. In this paper we take a very simple approach to apprenticeship learning by simply observing the expert play and training a classifier to identify expert actions from game states.

Deep Apprenticeship Learning

Following Mnih et al. (2013), we train a convolutional neural network to play Atari games by observing only raw pixels of the playing area. However, instead of learning to maximize the game score directly, we attempt to imitate the behaviour of an expert player. By watching an expert play, our network is able to learn to map game states to actions in a way that does not require that the score of the game is provided externally. We call our method Deep Apprenticeship Learning (DAL).

Data Collection

To interact with Atari games we used the Arcade Learning Environment (Naddaf 2010). To collect training data we modified the Arcade Learning Environment to record states (video frames) and actions (button presses) while a human plays the game.

We collected human gameplay data for the *Freeway* game. *Freeway* is a classical game about trying to cross a street while avoiding cars. *Freeway* is interesting from an RL perspective, because the agent only obtains a reward after crossing the street. That is, the reward occurs only after many actions are taken, and provided that the right actions are taken. This rare reward situation is very hard for RL systems. However, supervised learning, when expert data is available, can overcome this problem as shown in this paper.

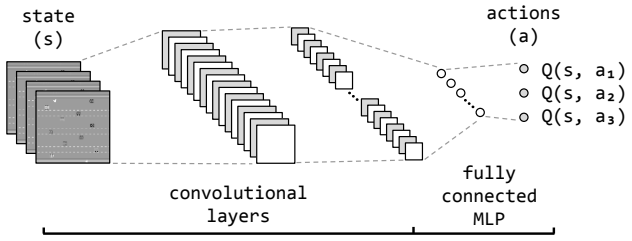


Figure 1: Architecture of our deep apprenticeship learning network.

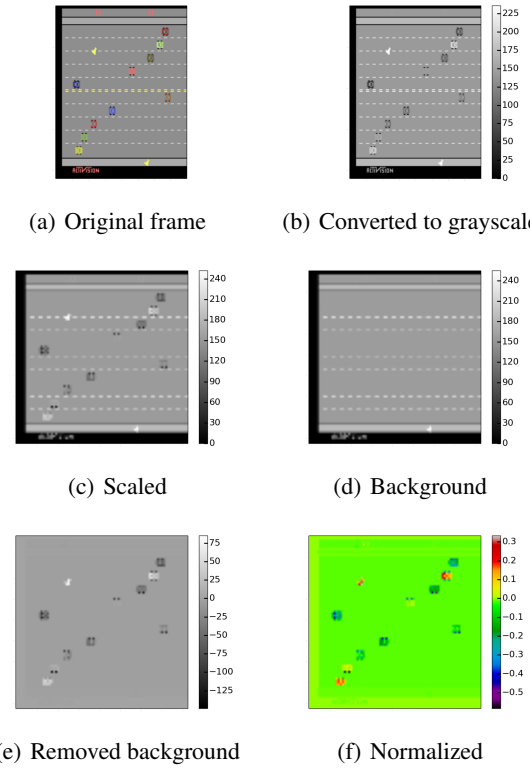
We collected 500,000 examples for *Freeway*. An example consists of a (state, action) pair, where for describing the state we used one, two or four sequential frames. However, we found that this choice did not have a significant effect on performance, which is sensible because the state of a game of *Freeway* can be entirely inferred from a single frame.

We preprocess each of the game frames by converting the images from the 128 color Atari palette to grayscale. Each frame is resized from the original 210×160 to 83×83. Our preprocessing introduces some distortion of the original image, but based on visual inspection we believe that no relevant information is lost. We also remove the background from each frame by subtracting the median image computed over a large number of sample frames. (Figure 2)

Experiments

We train a deep convolutional neural network to learn a mapping from states to actions using the expert data collected for each game. Our neural network was implemented in Theano (Bergstra et al. 2010), and its architecture is illustrated in Figure 1.

We held out one gameplay episode to test on (around 2% of the data) and used all other collected data for training.



(a) Original frame (b) Converted to grayscale

(c) Scaled (d) Background

(e) Removed background (f) Normalized

Figure 2: Preprocessing pipeline for each frame.

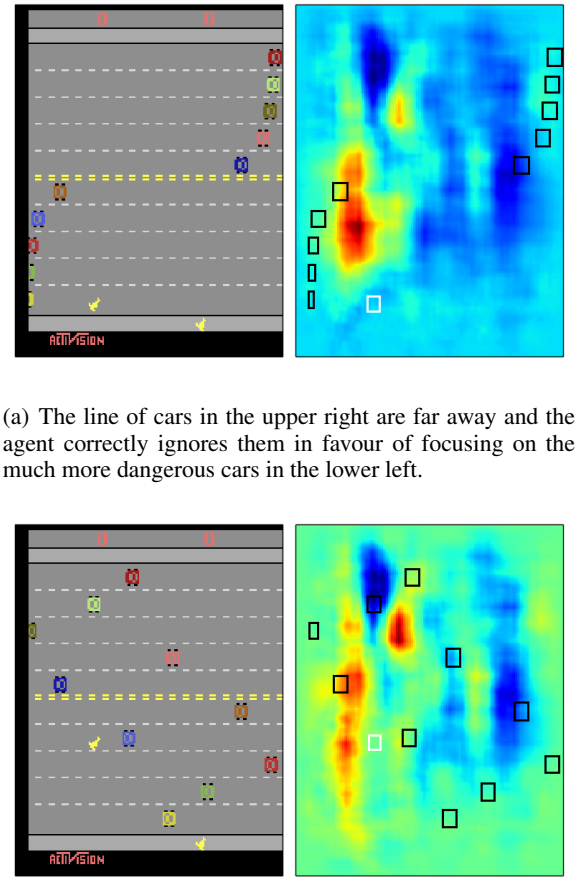
Random	0
Human	32
Sarsa	11
Deep Apprenticeship Learning	17

Table 1: Comparison of scores for the game Freeway.

The training time was fixed to 24 hours, which is between 13 and 30 epochs depending of the number of frames used in the input.

We use the best-scoring network to play the game. In Table 1 we present the comparison of the score we get in Freeway with the scores for random play, human play, and the Sarsa approach used in Bellemare et al. (2013). The method of Bellemare et al. (2013) employs existing computer vision features and finds a control policy by maximizing the score in the game. It is important to note that this approach, like that of Mnih et al. (2013), uses information about the score obtained from the engine, which is not the case in our method.

Our approach scores significantly higher than the Sarsa approach, despite the fact that Sarsa is trained explicitly to maximize the game score. Both approaches are much better than random play, which fails to score any points. From looking at the gameplay we can see that the resulting policy does not just encode the fact the we want to get to the other side, but also that we want to avoid hitting cars. There are many cases in which the player stops when the car is coming, waits for it to pass, and then moves on.



(a) The line of cars in the upper right are far away and the agent correctly ignores them in favour of focusing on the much more dangerous cars in the lower left.

(b) As the player moves across the road focus shifts to place more emphasis on the top half of the screen.

Figure 3: Saliency: The network learns what parts of the image it must pay attention to to perform well.

Analysis

Once we have trained our model to imitate expert play we can use the technique of Simonyan, Vedaldi, and Zisserman (2014) to produce saliency maps of the gameplay area, which we can use to understand the relative importance of different parts of the scene to decision making. The maps represent the gradient of the output corresponding to the chosen action with respect to the input image. Some examples are shown in Figure 3.

Cars below the yellow line move from left to right, and above the yellow line they move from right to left. Figure 3 shows that the agent has correctly learned look to its left on the bottom half of the screen and to look to its right on the top half. Regions of the screen that are far from the player are ignored, even when they contain cars. Additionally by comparing 3(a) and 3(b) we see that when the player is further across the road the agent focuses more strongly on the top half of the screen. Taking into account this example, we can infer that the agent has learned to look first left then right before crossing the street.

References

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research* 47:253–279.

Bergstra, J.; Breuleux, O.; Bastien, F.; Lamblin, P.; Pascanu, R.; Desjardins, G.; Turian, J.; Warde-Farley, D.; and Bengio, Y. 2010. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*. Oral Presentation.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Naddaf, Y. 2010. *Game-Independent AI Agents for Playing Atari 2600 Console Games*. Masters, University of Alberta.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop at International Conference on Learning Representations*.