Exercise 3: Visual Planning with CNNs

Badhreesh, David-Elias Künstle

18/12/2017

1 Introduction

The most famous use of convolutional neural networks (CNNs) is in image classification. In contrast here we describe their application for a planning task from visual input. An agent has to find a target in a two dimensional maze of Figure 1 from a random starting position. The agent receives only partial observations (pob) of the environment.

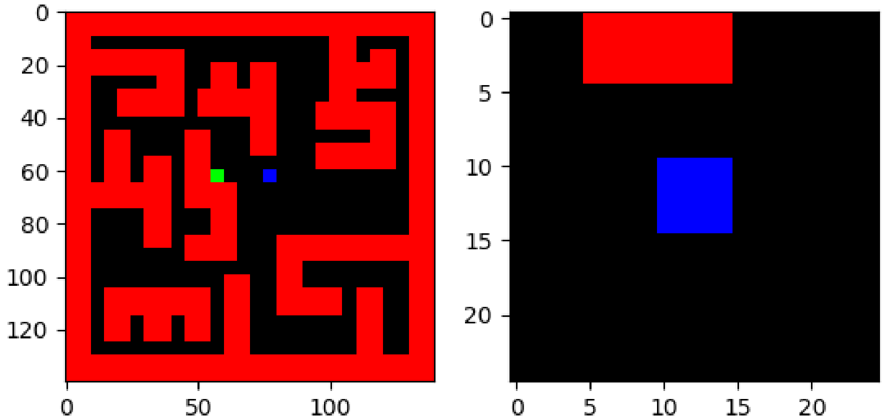


Figure 1: The agent (blue) has to find the target (green) in the maze (left, walls in red). It is only provided a history of partial observations (right).

At each time step the agent has to decide the direction (up, right, down, left) for the following discrete move by the current and some preceding pob. Pob and optimal directions for training and validation are generated via A-Star-Algorithm. For testing the agent has to maximize the number of successful runs. A run is only successful if the agent finds the target in a limited number of time. In section 3 we investigate the influence of changing target position, map, history and pob on the test performance.

2 Implementation

Our agent is implemented via stacked convolution layers like described in Table 1 using the *keras* library (*tensorflow* backend). The architecture resulted via trial and error. As default we use the provided map in Figure 1, a history of 4 pob as input data and a fixed target position. We use the input history as a flat vector (1 channel) as provided. Experiments with reshaping to two (width, height; history as channels) or three (width, height, history; 1 channel) dimensions did not result in significant better results. Trained with Adam optimization (batch size 32, learning rate 0.001) we reach over 98% accuracy on the validation set after 5 iterations over the full training data (epochs). As test setup the agent usually can find the target in less than 50 steps from any random start position. Subjective the agent seems to follow the shortest path directly to the target.

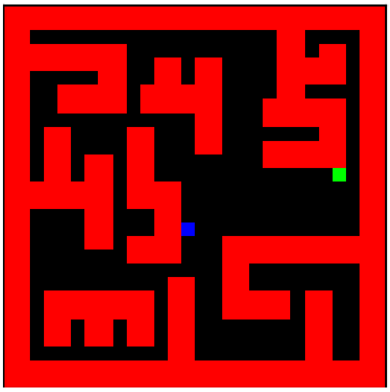
TODO: What happens if you increase the history length for the views or make the view larger ?

3 Generalization

Until this point the test situation was very close to the training situation. Here we try, how our agent performs if the task (target position) or environment (walls in maze) is changed after training.

Convolution Layer	(8 filters, size 64)
Convolution Layer	(16 filters, size 32)
Convolution Layer	(32 filters, size 16)
Convolution Layer	(32 filters, size 8)
Convolution Layer	(32 filters, size 4)
Dense Layer	(5 outputs, softmax activation)

Table 1: Network architecture. A convolution layer here includes ReLu activation function and max pooling (size 2, stride 2).



(a) randomized target position

(b) slightly modified maze

Figure 2: The agent tries using the way it learned while training to a fixed position where it expects the target. Once some pob history occurs, which the agent has not seen while training it is stuck by moving back and forth.

3.1 Target location

If the target is at the fixed position, it looks like our agent magically knows where it is and takes the shortest path. This illusion is revealed, if the target is placed on any free, random position (Figure 2a). The agent still moves straight to the middle of the map where the target was while learning. As soon as the target should appear in the pob but doesn't, the agent starts moving back and forth. Therefore it usually doesn't reach any target while testing.

The target could also be placed such, that the agent has to pass it at the usual path. Then the agent often also starts moving back and forth. We could conclude, that it hasn't even learned to go to the target if it is visible (target is in the current pob).

3.2 Map

TODO: What happens if you change the map after training (how well does your agent generalize) ?

3.3 Proposal

TODO: Can you think of ways to make the agent generalize across target locations different maps ?

TODO: For bonus points: test one of these ideas.