Tuning Modular Networks with Weighted Losses for Hand-Eye Coordination

Fangyi Zhang, Jürgen Leitner, Michael Milford, Peter I. Corke *†

Abstract

This paper introduces an end-to-end fine-tuning method to improve hand-eye coordination in modular deep visuo-motor policies (modular networks) where each module is trained independently. Benefiting from weighted losses, the fine-tuning method significantly improves the performance of the policies for a robotic planar reaching task.

1. Introduction

Recent work has demonstrated robotic tasks based directly on real image data using deep learning, for example robotic grasping [2]. However these methods require large-scale real-world datasets, which are expensive, slow to acquire and limit the general applicability of the approach.

To reduce the cost of real dataset collection, we used simulation to learn robotic planar reaching skills using the DeepMind DQN [3]. The DQN showed impressive results in simulation, but exhibited brittleness when transferred to a real robot and camera [4]. By introducing a bottleneck to separate the DQN into perception and control modules for independent training, the skills learned in simulation (Fig. 1A) were easily adapted to real scenarios (Fig. 1B) by using just 1418 real-world images [5].

However, there is still a performance drop compared to the control module network with ideal perception [5]. To reduce the performance drop, we propose fine-tuning the combined network to improve hand-eye coordination. Preliminary studies yet show that a naive fine-tuning using Q-learning does not give the desired result [5]. To tackle the problem, we introduce a novel end-to-end fine-tuning method using weighted losses in this work, which significantly improved the performance of the combined network.

2. Methodology

We consider the planar reaching task, which is defined as controlling a 3 DoF robot arm (Baxter robot’s left arm) so that in operational space its end-effector position \( x \in \mathbb{R}^2 \) moves to the position of the target \( x^* \) in a vertical plane (ignoring orientation). The reaching controller adjusts the robot configuration (joint angles \( q \in \mathbb{R}^3 \)) to minimize the error between the robot’s current and target position, i.e., \( \| x - x^* \| \). At each time step 1 of 9 possible actions \( a \in a \) is chosen to change the robot configuration: 3 per joint – increasing or decreasing by a constant amount (0.04 rad) or leaving it unchanged. An agent is required to learn to reach using only raw-pixel visual inputs \( I \) from a monocular camera and their accompanying rewards \( r \).

The network has the same architecture and training method to [5], but with an additional end-to-end fine-tuning using weighted losses, as shown in Fig. 2. The perception network is first trained to estimate the scene configuration \( \Theta = [x^* q] \in \mathbb{R}^5 \) from a raw-pixel image \( I \) using the quadratic loss function

\[
L_p = \frac{1}{2m} \sum_{j=1}^{m} \| y(I^j) - \Theta^j \|^2 ,
\]

error between the robot’s current and target position, i.e., \( \| x - x^* \| \). At each time step 1 of 9 possible actions \( a \in a \) is chosen to change the robot configuration: 3 per joint – increasing or decreasing by a constant amount (0.04 rad) or leaving it unchanged. An agent is required to learn to reach using only raw-pixel visual inputs \( I \) from a monocular camera and their accompanying rewards \( r \).

The network has the same architecture and training method to [5], but with an additional end-to-end fine-tuning using weighted losses, as shown in Fig. 2. The perception network is first trained to estimate the scene configuration \( \Theta = [x^* q] \in \mathbb{R}^5 \) from a raw-pixel image \( I \) using the quadratic loss function

\[
L_p = \frac{1}{2m} \sum_{j=1}^{m} \| y(I^j) - \Theta^j \|^2 ,
\]
where \( y(I^j) \) is the prediction of \( \Theta^j \) for \( I^j \); \( m \) is the number of samples. The control network is trained using K-GPS [5] where network weights are updated using the Bellman equation which is equivalent to the loss function:

\[
L_q = \frac{1}{2m} \sum_{j=1}^{m} \left\| Q(\Theta^j, a^j_t) - (r^j_t + \gamma \max_{a^j_{t+1}} Q(\Theta^j_{t+1}, a^j_{t+1}) \right\|^2
\]

where \( Q(\Theta^j, a^j_t) \) is the sum of future expected rewards \( \sum_{k=0}^{\infty} \gamma^k r^j_{t+k} \) when taking action \( a^j_t \) in state \( \Theta^j \). \( \gamma \) is a discount factor applied to future rewards.

After separate training for perception and control individually, an end-to-end fine-tuning is conducted for the combined network (perception + control) using weighted task \( (L_q) \) and perception \( (L_p) \) losses. The control network is updated using only \( L_q \), while the perception network is updated using the weighted loss:

\[
L = \beta L_p + (1 - \beta) L^{BN}_q,
\]

where \( L^{BN}_q \) is a pseudo-loss which reflects the loss of \( q \) in the bottleneck (BN); \( \beta \in [0, 1] \) is a balancing weight. From the backpropagation algorithm [11], we can infer that \( \delta_L = \beta \delta_{L_p} + (1 - \beta) \delta_{L^{BN}_q} \), where \( \delta_L \) is the gradients resulted by \( L \); \( \delta_{L_p} \) and \( \delta_{L^{BN}_q} \) are the gradients resulting respectively from \( \delta_{L_p} \) and \( \delta_{L^{BN}_q} \) (equivalent to that resulting from \( L_q \) in the perception module).

\[3. \text{Experiments and Results} \]

We evaluated the feasibility of the proposed approach using the metrics of Euclidean distance error \( d \) (between the end-effector and target) and average accumulated reward \( \bar{R} \) (a bigger accumulated reward means a faster and closer reaching to a target) in 400 simulated trials. For comparison, we evaluated three networks: \textbf{Initial}, \textbf{Fine-tuned} and \textbf{CR}. \textbf{Initial} is a combined network without end-to-end fine-tuning, which is labelled as EE2 (comprising FT75 and CR) in [5]. FT75 and CR are the selected perception and control modules which have the best performance individually. \textbf{Fine-tuned} is obtained by fine-tuning \textbf{Initial} using the proposed approach. \textbf{CR} works as a baseline indicating performance upper-limit.

In fine-tuning, \( \beta = 0.8 \), we used a learning rate between 0.01 and 0.001, a mini-batch size of 64 and 256 for task and perception losses respectively, and an exploration possibility of 0.1 for K-GPS. These parameters were empirically selected. To make sure that the perception module remembers the skills for both simulated and real scenarios, the 1418 real samples were also used to obtain \( \delta_{L_p} \). Similar to FT75, 75% samples in a mini-batch were from real scenarios, i.e., at each weight updating step, 192 extra real samples were used in addition to the 64 simulated samples in the mini-batch for \( \delta_{L_q} \).

![Figure 3. The box-plots of distance errors of different networks, with median values displayed. The crosses represent outliers.](image)

Table 1. Planar Reaching Performance

<table>
<thead>
<tr>
<th>Nets</th>
<th>( d_{\text{med}} ) [cm]</th>
<th>( d_{\text{Q3}} ) [cm]</th>
<th>( \bar{R} ) \mid []</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial</td>
<td>4.598</td>
<td>1.929</td>
<td>6.150</td>
</tr>
<tr>
<td>Fine-tuned</td>
<td>3.568</td>
<td>1.497</td>
<td>4.813</td>
</tr>
<tr>
<td>CR</td>
<td>3.449</td>
<td>1.447</td>
<td>4.330</td>
</tr>
</tbody>
</table>

Results are summarized in Fig. 3 and Table 1. \( d_{\text{med}} \) and \( d_{\text{Q3}} \) are the median and third quartile of \( d \). The error distance in pixels in the 84 \times 84 input image is also listed. We can see that \textbf{Fine-tuned} achieved a much better performance (22.4% smaller \( d_{\text{med}} \) and 96.2% bigger \( \bar{R} \)) than \textbf{Initial}. The fine-tuned performance is even very close to that of the control module (CR) which controls the arm using ground-truth \( \Theta \) as sensing inputs. We also did the same evaluations in 20 real-world trials on Baxter, and achieved similar results.

The experimental results show the feasibility of the proposed fine-tuning approach to improve hand-eye coordination in modular deep visuo-motor policies. The adaptation to real scenarios was effectively kept by presenting real samples for perception loss in the end-to-end fine-tuning.

\[\text{References}\]


