Pruning Neural Networks with Lottery Tickets in a MDP Approach

Andrey de Aguiar Salvi

Automated Planning 2019/02

From Previous Presentation...

We propose to modify the Lottery Tickets Hypothesis (a model compression method) with a Markov Decision Process with Q-Learning.

From Previous Presentation...

The Lottery Tickets Hypothesis.

Algorithm 1 Lottery Tickets Hypothesis
Source: adapted from (Frankle et al. 2019)

Require: weight matrix W, mask W', pruning rate p

$W_0 \leftarrow W$

$train(W, X, Y, n\_epochs)$

while $remaining\_weights > total\_weights \times p$ **do**

  $indexes \leftarrow find\_smallest\_values(|W|, p)$

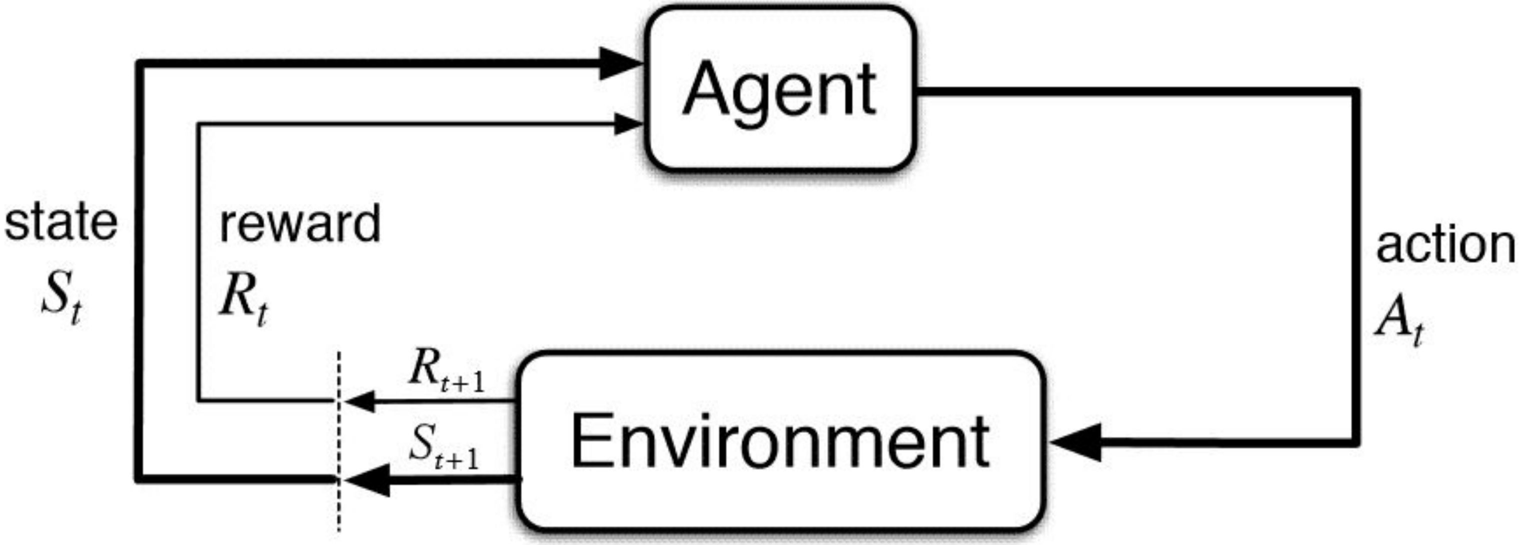
  $W'[indexes] \leftarrow 0$

  $W \leftarrow W_0$

  $train(W, W', X, Y, n\_epochs)$

end while

Markov Decision Process



Source: Sutton, R. S., Barto, A. G., & Bach, F. (1998). Reinforcement learning: An introduction

Q-Learning

Q-learning

loop

$a \leftarrow \mathsf{Select}_a\{Q(s,a)\}$

apply action a

observe resulting reward $r(s,a)$ and next state s'

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r(s,a) + \max_{a'}\{Q(s',a')\} - Q(s,a)] \qquad (i)$$

$s \leftarrow s'$

until termination condition

Source: Ghallab, M., Nau, D., & Traverso, P. (2016). Automated planning and acting

PUCRS
Pontifícia Universidade Católica
do Rio Grande do Sul

GPIN

The Proposed Method

Algorithm 3 Our proposed method

Require: weight matrix W, mask W', pruning rate p

$W_0 \leftarrow W$

$train(W, X, Y, n\_epochs)$

while $remaining\_weights > total\_weights \times p$ **do**

    $s \leftarrow W'$

    $indexes \leftarrow \pi(s)$

    $W'[indexes] \leftarrow 0$

    $W \leftarrow W_0$

    $train(W, W', X, Y, n\_epochs)$

end while

Some Details

- Creating the Q-Table:
  - First train LeNet 300-100 by 30 epochs
  - The agent will perform a decreasing epsilon-search
    - Total of steps: 1000
    - Number of iterations: 15
    - epsilon decreation [1.0; ...; 0.1]
    - Reward: $r_a(s, s') \leftarrow \frac{-(1 - accuracy) * remaining\_weights}{total\_weights}$
    - Q-Table update: or

$$Q(s,a) \leftarrow Q(s,a) + \alpha \cdot \left[r_a(s,s') + \max_{a'} \{Q(s',a')\} - Q(s,a) \right]$$

$$Q(s,a) \leftarrow (1-\alpha) \cdot Q(s,a) + \alpha \left(r_a(s,s') + \gamma \cdot \max_{a} Q(s',a) \right)$$

Some Details

- Pruning the model:
  - Create another LeNet 300-100 from scratch
  - Iteratively:
    - Train 9 epochs
    - Prune (action which maximizes the Quality based on the state)

Results

Table 1: Train and validation accuracy's

Q-update	α	γ	Train_Acc	Valid_Acc
EQ_3	0.5	-	89.68%	89.31%
EQ_3	0.6	-	90.70%	90.40%
EQ_3	0.7	-	91.09%	90.86%
EQ_3	0.8	-	8.85%	0.09%
EQ_3	0.9	-	90.59%	89.82%
EQ_3	**1.0**	**-**	**91.21%**	**90.86%**
EQ_6	0.9	0.4	9.10%	0.09%
EQ_6	0.9	0.5	88.98%	88.69%
EQ_6	**0.9**	**0.6**	**90.60%**	**89.94%**
EQ_6	0.9	0.7	89.83%	89.22%
EQ_6	0.9	0.8	90.09%	89.39%
EQ_6	0.9	0.9	90.16%	89.79%
EQ_6	0.9	1.0	9.10%	0.09%
LTH	**-**	**-**	**96.86%**	**95.92%**

Pontifícia Universidade Católica do Rio Grande do Sul

Results

Table 2: Test accuracy of the best models

Class	Model 1	Model 2	Model 3	Instances
0	97.44%	97.55%	98.57%	980
1	97.26%	97.18%	98.23%	1135
2	91.08%	88.56%	95.54%	1032
3	88.81%	87.62%	96.33%	1010
4	92.46%	94.80%	95.92%	982
5	83.85%	81.83%	94.05%	892
6	94.67%	94.05%	96.97%	958
7	91.34%	90.66%	96.39%	1028
8	89.73%	86.34%	94.76%	974
9	88%	86.91%	94.84%	1009
Overall	91.6%	90.7%	96.2%	10000

Pontifícia Universidade Católica do Rio Grande do Sul

Thank you!

Andrey de Aguiar Salvi

andrey.salvi@edu.pucrs.br / andreysalvi@gmail.com