Lecture 12. Intro to Reinforcement Learning

Lecture 12. Intro to Reinforcement Learning

Alex Avdyushenko

Kazakh-British Technical University

November 26, 2022

Good afternoon, dear students. Today we are going to talk about reinforcement learning, it is a slightly separate topic that is not directly related to deep learning, but is included in machine learning of course. It is quite possible to read a semester course on it, but we will limit ourselves to today's one lecture.

Lecture 12. Intro to Reinforcement Learning

Five-minutes block

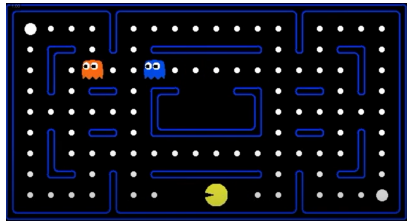
Five-minutes block

- Describe (or draw) the general architecture of the autoencoder
- What is the main advantage of the self-supervised learning method?
- Please, write down GAN loss functions

Ok, and as usual now we have five-minutes block. The last one in our course. So prepare yourself and send right answers to me in five minutes.

- Up to this point, we either restored the function from the training set (X, Y) (supervised learning) or searched for the structure in the set of objects X (unsupervised learning)
- But how does learning work in real world? Usually we do some action and get the result, gradually learning

Great, let's go on and let me remind you, that up to this point, we found the appropriate function f, based on the training set. And it is called supervised learning. Or also we found the inner structure in the set of objects, you know, texts or images, and it is called unsupervised learning, when we haven't answers y.



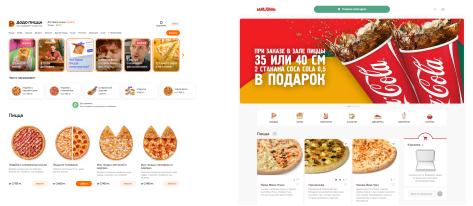
Deep Reinforcement Learning in Pac-man

Ok, so what we have in real world? Usually we do some action and get the result, and so we are learning gradually.

Lecture 12. Intro to Reinforcement Learning

One more motivation

One more motivation
For example, we need to select the main page of a pizzeria website to attract the clients



And another one important example. Suppose you need to choose the most attractive design for the main page of a pizzeria website, but you don't want to lose a lot of customers, time, money on experiments.

Lecture 12. Intro to Reinforcement Learning

What approaches are there?

What approaches are there?

- A/B testing — many users see potentially bad options
- multi-armed bandits — a special case of reinforcement learning

In A/B testing you divide your customers on two groups (A and B), show them different versions of main page and then measure the results, the conversions and make your conclusions. And what is multi-armed bandits?

Probability of winning $\theta = 0.05$

Let's begin from the one-armed bandit. So you pull the handle of a slot machine and you can win with some probability theta.

Lecture 12. Intro to Reinforcement Learning

Multi-Armed Bandits

Multi-Armed Bandits

$\theta_1 = 0.02$ $\theta_2 = 0.01(\text{min})$ $\theta_3 = 0.05$ $\theta_4 = 0.1(\text{max})$

We do not know the true probabilities, but we want to come up with a strategy that maximizes the payoff (reward)

In multi-armed bandits model you have several handles with some winning probabilities theta one, two and so on.

Lecture 12. Intro to Reinforcement Learning

Mathematical statement of the problem

Ok, now let's move on, to mathematical statement of the problem. ...

Mathematical statement of the problem

Given possible actions

$$x_1, \ldots, x_n$$

At the next iteration t, for each action x_i^t performed, we get the answer

$$y_i^t \sim q(y|x_i^t, \Theta),$$

which brings us a **reward**

$$r_i^t = r(y_i^t)$$

There is an optimal action x_{i^*} (sometimes $x_{i_t^*}$)

$$\forall i: \; E(r_{i_t^*}^t) \geq E(r_i^t)$$

Question

How to evaluate different strategies?

Multi-Armed Bandits possible applications

Areas:

- advertising banners
- recommendations (goods, music, movies etc.)
- slot machines in the casino

Approaches:

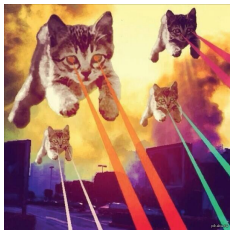
- Thompson sampling
- Upper Confidence Bound (UCB)
- ε-greedy strategy

Note

The main difference from more general reinforcement learning approach is that there is no environment state in multi-armed bandits

Ok, and what is the possible applications of multi-armed bandits? And there are models of implementing and optimizing the multi-armed bandits, but we won't discuss them in detail here.

Do not take into account delayed effects
For example, the effect of clickbait in advertising



SHOCK! Cats want to enslave us

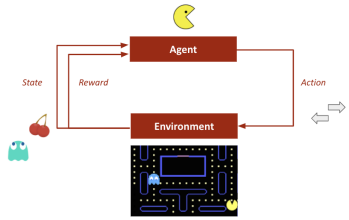
So you may increase your metrics for some short period of time by clickbait, but in the long run of course you will lose clients and money consequently.

Lecture 12. Intro to Reinforcement Learning

Reinforcement Learning

Let's go on to the Reinforcement Learning model.

Reinforcement Learning
Examples



- robot control
- (video) games
- security management

Lecture 12. Intro to Reinforcement Learning

Agent Model in a Changing Environment

Agent Model in a Changing Environment

Definitions:

- state $s \in S$
- agent action $a \in A$
- reward $r \in R$
- state transition dynamics $P(s_{t+1}|s_t, a_t, \ldots, s_{t-i}, a_{t-i}, \ldots, s_0, a_0)$
- win function

$$r_t = r(s_t, a_t, \ldots, s_0, a_0)$$

- total reward $R = \sum_t r_t$
- agent policy $\pi(a|s)$

Task:

$\pi(a|s) : \mathbb{E}_\pi[R] \to \max$

So, more formally you have states, actions and rewards. And your task is to find optimal policy to maximize the total reward.

Lecture 12. Intro to Reinforcement Learning

Cross-entropy method. Algorithm

Cross-entropy method. Algorithm

Trajectory — $[s_0, a_0, s_1, a_1, s_2, \ldots, a_{T-1}, s_T]$

Initialize strategy model $\pi(a|s)$

Repeat:

- play N sessions
- choose the best K of them and take their trajectories
- adjust $\pi(a|s)$ so that s is able to maximize the probabilities of actions from the best trajectories

The first simple algorithm to find appropriate policy is called cross-entropy method. Here you have some initialization, maybe random, as usual, then you play N sessions, choose best of them by your reward and update your strategy.

Lecture 12. Intro to Reinforcement Learning

There is a problem..

There is a problem..

that there are a lot of states:

So the problem is, that we can not even store in memory the whole table of states.

Lecture 12. Intro to Reinforcement Learning

Approximate cross-entropy methods

Possible solutions:

- split the state space into sections and treat them as states
- get probabilities from $\pi_\theta(a|s)$ machine learning model: linear model, neural network, random forest
- often these probabilities need to be further specified later

So people invented Approximate cross-entropy methods.

Lecture 12. Intro to Reinforcement Learning

Disadvantages of the cross-entropy method

Disadvantages of the cross-entropy method

- it is unstable for small samples
- in the case of a non-deterministic environment, it chooses lucky cases (random played in favor of the agent)
- it is focusing on behavior in simple states
- it ignores a lot of information
- there are tasks in which the end never comes (stock market game)

... So we need to correct these shortcomings.

Lecture 12. Intro to Reinforcement Learning

Markov Decision Process

Markov Decision Process

Definitions:

- state $s \in S$
- agent action $a \in A$
- reward $r \in R$
- transition dynamics $P(s_{t+1}|s_t, a_t, \ldots, s_{t-i}, a_{t-i}, \ldots, s_0, a_0) = P(s_{t+1}|s_t, a_t)$ **(Markov's assumption)**
- win function

$$r_t = r(s_t, a_t)$$

(Markov's assumption)

- total reward $R = \sum_t r_t$
- agent policy

$$\pi(a|s)$$

Task:

$$\pi(a|s) : \mathbb{E}_\pi[R] \to \max$$

Here we have Markov's assumption, which is actually not so strong, as we could expect. For example it is true in chess and Starcraft also. Only your current state determines everything, and not the history of the game.

Lecture 12. Intro to Reinforcement Learning

TD-training

TD-training
temporal difference

We arbitrarily initialize the function $V(s)$ and the strategy π
Then we repeat

- initialize s
- for each agent step
  - select a by strategy π
  - do action a, get result r and next state s'
  - update function $V(s)$ by formula

$$V(s) = V(s) + \alpha(r + \gamma V(s') - V(s))$$

  - go to the next step by assigning $s := s'$

The amazing fact is, that temporal difference training works.

Lecture 12. Intro to Reinforcement Learning

└DQN (Deep Q-Learning Network)

Ok, and the last reinforcement learning model in our lecture is the Deep Q-Learning Network.

DQN (Deep Q-Learning Network)

Environment is Atari game emulator, each frame $210 \times 160 pix$, $128 col$

Pong | Breakout | Space Invaders | Seaquest | Beam Rider

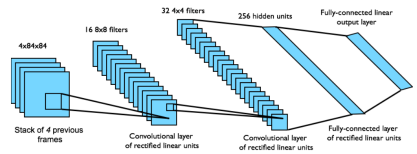
s states: 4 consecutive frames compressed to 84×84
Actions **a**: from 4 to 18, depending on the game
r rewards: current in-game SCORE
Value function $Q(s, a; w)$:
ConvNN with input s and $|A|$ outputs

V.Mnih et al. (DeepMind). Playing Atari with deep reinforcement learning. 2013

Lecture 12. Intro to Reinforcement Learning

Network architecture for the value function

Network architecture for the value function



Here your see actual CNN structure for Atari game — quite simple, isn't it? And I think you know what happened next :)

Temporary page!

LaTeX was unable to guess the total number of pages correctly. A
was some unprocessed data that should have been added to the
this extra page has been added to receive it.
If you rerun the document (without altering it) this surplus page
away, because LaTeX now knows how many pages to expect for t
document.