REINFORCEMENT LEARNING I

MULTIARMED BANDITS

Sergey Nikolenko

Harbour Space University, Barcelona, Spain
April 11, 2017

MULTIARMED BANDITS

- So far we've either had a set of "correct answers" (supervised learning) or simply nothing (unsupervised learning).
- But is it really how learning works in real life?
- How does a baby learn?

- Hence, *reinforcement learning*.
- An agent interacts with the environment.
- On every step the agent can be in state $s \in S$ and choose an action $a \in A$.
- The environment tells the agent its reward r and the next state $s' \in S$.

- Exploitation vs. exploration: first learn, then apply.
- But when do we switch?
- Always a problem in reinforcement learning.

- Example: tic-tac-toe.
- How does an algorithm learn to play and *win* in tic-tac-toe?
- Example: genetic algorithm. Very slow, does not account for information.

- States are board positions.
- Value function $V(s)$ for every state.
- Reinforcement only at the end: the *credit assignment* problem.

EXAMPLE

- One version — propagate the reward back: if we got from s to s', we update
$$V(s) := V(s) + \alpha\left[V(s') - V(s)\right].$$
- This is called TD-learning (temporal difference learning), works very well in practice; we'll get to it.

- If $|S| = 1$, the agent has a fixed set of actions A and the environment has no memory.
- The multiarmed bandit model.
- No credit assignment, only exploration vs. exploitation.

- Always choose the best option, where *best* is defined with average reward so far:

$$Q_t(a) = \frac{r_1 + r_2 + ... + r_{k_a}}{k_a}.$$

- What's wrong with this algorithm?

- Always choose the best option, where *best* is defined with average reward so far:

$$Q_t(a) = \frac{r_1 + r_2 + ... + r_{k_a}}{k_a}.$$

- What's wrong with this algorithm?
- Easy to miss the optimum if we're unlucky with the initial sample.
- Useful heuristic — *optimism under uncertainty*.
- You need evidence to *reject*, not to accept.

- ϵ-*greedy strategy*: choose the best (as above) action with probability $1-\epsilon$ and random action with probability ϵ.
- Start with large ϵ, then gradually decrease.
- Boltzmann exploration:

$$\pi_t(a) = \frac{e^{Q_t(a)/T}}{\sum_{a'} e^{Q_t(a')/T}},$$

  where ER is the expected reward, T is the *temperature*.
- Temperature usually decreases with time.

LINEAR REWARD-INACTION ALGORITHM

- For the case of binary payoffs (0-1).
- The *linear reward-inaction algorithm* adds linear reward to probability of a_i if it is successful:

$$p_i := p_i + \alpha(1 - p_i),$$

$$p_j := p_j - \alpha p_j, \quad j \neq i,$$

  and nothing changes if unsuccessful.

LINEAR REWARD-INACTION ALGORITHM

- The algorithm converges with probability 1 to a vector with one 1 and the rest 0.
- Does not always converge to the optimal strategy; but by decreasing α we decrease the probability of error.
- *Linear reward-penalty*: same thing, but unsuccessful actions get punished (i.e., all the rest get a reward).

INTERVAL ESTIMATES

- One way to apply the optimism under uncertainty heuristic.
- Store the statistics n and w for every action, compute confidence interval with confidence $1-\alpha$, use the upper bound.
- Example: Bernoulli trials (coin tossing). With probability $.95$ the average lies in the interval

$$\left(\bar{x} - 1.96\frac{s}{\sqrt{n}}, \bar{x} + 1.96\frac{s}{\sqrt{n}}\right),$$

  where 1.96 is taken from Student's t distribution, n is the number of trials, $s = \sqrt{\frac{\sum(x-\bar{x})^2}{n-1}}$.
- A great method if the assumptions hold (which is often unclear).

- How do we recompute $Q_t(a) = \frac{r_1+\ldots+r_{k_a}}{k_a}$ when new information arrives?
- Easy:

$$Q_{k+1} = \frac{1}{k+1}\sum_{i=1}^{k+1} r_i = \frac{1}{k+1}\left[r_{k+1} + \sum_{i=1}^{k} r_i\right] =$$
$$= \frac{1}{k+1}\left(r_{k+1} + kQ_k\right) = Q_k + \frac{1}{k+1}\left(r_{k+1} - Q_k\right).$$

- This is a special case of a general rule:

$$\text{NewEstimate} := \text{OldEstimate} + \text{StepSize}\left[\text{Target} - \text{OldEstimate}\right].$$

- For the average, the step size is not constant: $\alpha_k(a) = \frac{1}{k_a}$.
- Changing the sequence of steps, we can achieve other effects.

- What if the payoffs change with time?
- We should value recent information highly and outdated information low.
- Example: for an update rule

$$Q_{k+1} = Q_k + \alpha\left[r_{k+1} - Q_k\right]$$

with constant α the weights decay exponentially:

$$Q_k = Q_{k-1} + \alpha\left[r_k - Q_{k-1}\right] = \alpha r_k + (1-\alpha)Q_{k-1} =$$

$$= \alpha r_k + (1-\alpha)\alpha r_{k-1} + (1-\alpha)^2 Q_{k-2} = (1-\alpha)^k Q_0 + \sum_{i=1}^{k} \alpha(1-\alpha)^{k-i} r_i.$$

- This update rule does not necessarily converge, which is good: we want to follow new averages.
- General result – an update rule converges if the sequence of weights satisfies

$$\sum_{k=1}^{\infty} \alpha_k(a) = \infty \quad \text{and} \quad \sum_{k=1}^{\infty} \alpha_k^2(a) < \infty.$$

- E.g., for $\alpha_k(a) = \frac{1}{k_a}$ it does.

- We can simplify the search if we begin with optimistic initial values.
- Start with large $Q_0(a)$, so that any real value is "disappointing".
- But not too large — we need Q_0 to average out with the real r_i.

- The intuition for *reinforcement comparison* is to look for “large” payoffs; what is “large”?
- Let’s compare with average over all arms.
- These methods usually do not have action values Q_k, only preferences $p_t(a)$; probabilities can be obtained, e.g., with softmax:

$$\pi_t(a) = \frac{e^{p_t(a)}}{\sum_{a'} e^{p_t(a')}}.$$

- And on every step we update both preference and average:

$$\bar{r}_{t+1} = \bar{r}_t + \alpha \left(r_t - \bar{r}_t \right),$$
$$p_{t+1}(a) = p_t(a_t) + \beta \left(r_t - \bar{r}_t \right).$$

- Pursuit methods store both expectation estimates and action preferences, and preferences “follow” averages.
- E.g., $\pi_t(a)$ is the probability to choose a at time t; after step t we look for a greedy strategy

$$a^*_{t+1} = \arg\max_a Q_{t+1}(a)$$

and change π towards the greedy strategy:

$$\pi_{t+1}(a^*_{t+1}) = \pi_t(a^*_{t+1}) + \beta \left[1 - \pi_t(a^*_{t+1})\right],$$
$$\pi_{t+1}(a) = \pi_t(a) + \beta \left[0 - \pi_t(a)\right].$$

- Assume finite horizon of h steps.
- We use the Bayesian approach to find the optimal strategy.
- Begin with random parameters $\{p_i\}$, e.g., uniform; compute the mapping from *belief states* (after several rounds) to actions.
- A state is expressed as $\mathcal{S} = \{n_1, w_1, \ldots, n_k, w_k\}$, where each bandit i has been run n_i times with w_i positive (binary) results.

- $V^*(\mathcal{S})$ — expected remaining payoff.
- Recursion: if $\sum_{i=1}^{k} n_i = h$, $V^*(\mathcal{S}) = 0$ since there's no time left.
- If we know V^* for all states when t time slots are left, we can recompute for $t+1$:

$$\begin{aligned} &V^*(n_1, w_1, \ldots, n_k, w_k) = \\ &\qquad = \max_i \left(\rho_i(1 + V^*(\ldots, n_i + 1, w_i + 1, \ldots)) + \right. \\ &\qquad\qquad \left. (1 - \rho_i) V^*(\ldots, n_i + 1, w_i, \ldots)\right), \end{aligned}$$

where ρ_i is the posterior probability of action i to be rewarded (if p_i had uniform priors then Laplace rule applies: $\rho_i = \frac{w_i+1}{n_i+2}$).

- Let's look at multiarmed bandits in a general probabilistic form.
- Binary case for simplicity: reward either 1 or 0.
- Suppose that at time t we have state $\theta_t = (\theta_{1t}, \dots, \theta_{Kt})$ for K arms, and we want to maximize the total expected number of successes.
- Reward function $R_i(\theta_t, \theta_{t+1})$ – reward for choosing action i (a_i) that changes state θ_t to θ_{t+1}.
- Transition probability $p\left(\theta_{t+1} \mid \theta_t, a_i\right)$.
- And we want to traing a strategy $\pi(\theta_t)$ that says which arm to pull.

- Then the value function in the most general form until horizon T is

$$
\begin{aligned}
V_T(\pi, \theta_0) = \mathrm{E}\left[R_{\pi(\theta_0)}(\theta_0, \theta_1) + V_{T-1}(\pi, \theta_1)\right] = \\
= \int p\left(\theta_1 \mid \theta_0, \pi(\theta_0)\right)\left[R_{\pi(\theta_0)}(\theta_0, \theta_1) + V_{T-1}(\pi, \theta_1)\right] d\theta_1.
\end{aligned}
$$

- If we know everything, and T is small, we can use dynamic programming.
- But it's usually very expensive.

- For large/unbounded T let's consider

$$R = R(0) + \gamma R(1) + \gamma^2 R(2) + \dots, \quad 0 < \gamma < 1.$$

- Gittins' theorem (1979): the search for an optimal strategy

$$\pi(\theta_t) = \arg\max_{\pi} V(\pi, \theta_t = (\theta_{1t}, \dots, \theta_{Kt}))$$

  can be factorized and reduced to

$$\pi(\theta_t) = \arg\max_{i} g(\theta_{it}).$$

- $g(\theta_{it})$ is called the *Gittins index*; the gold standard, but also hard to compute (there are approximations).

- Other possibility – let's compute the priority for every arm i in order to bound regret immediately.
- [Auer, 2002]: UCB1 strategy. Accounts for the uncertainty "left" in an action, aims to bound regret.
- If we've have n experiments, including n_i experiments with action i and average reward $\hat{\mu}_i$, the UCB1 algorithm assigns it priority

$$\mathrm{Priority}_i = \hat{\mu}_i + \sqrt{\frac{2\log n}{n_i}}.$$

Then we simply choose the action with highest priority.

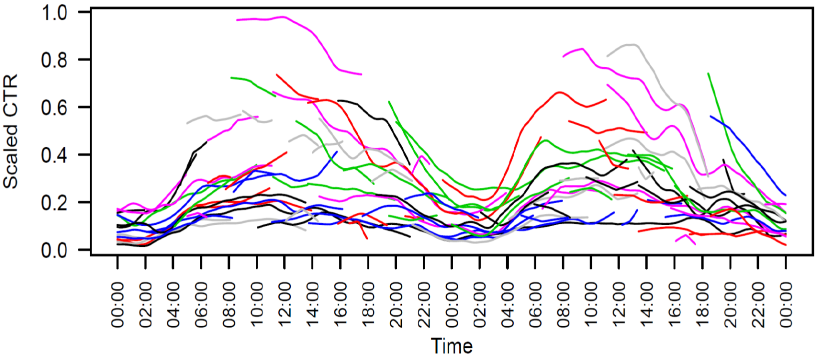
- Theorem: suboptimal actions will be selected $O(\log n)$ times, and regret is bounded by $O(\log n)$.
- There is a matching lower bound but constants are important too.
- UCB1 is a good strategy, but there are even better variations (with better constants).

- Suppose you want to test a set of changes in a web site's interface or some such.
- Often done with A/B testing:
    - choose experimental group (separate for every change);
    - choose control group that remains unchanged;
    - collect statistics and estimate whether improvement (if any) is significant.

EXAMPLE: BANDITS FOR A/B TESTING

- Main problem: how much statistics is sufficient?
- Bandits can help; it's exactly the same problem setting:
  - showing a variation corresponds to an action;
  - user actions represent the environment;
  - there is no need to ever stop, "sufficient sample size" is determined automatically.
- Bandits are a great way to A/B test.

- Example:



1.0
0.8
0.6
0.4
0.2
0.0
Scaled CTR
00:00
02:00
04:00
06:00
08:00
10:00
12:00
14:00
16:00
18:00
20:00
22:00
00:00
02:00
04:00
06:00
08:00
10:00
12:00
14:00
16:00
18:00
20:00
22:00
00:00
Time

- We want, at time t, to redistribute page views $(x_1, x_2, \ldots, x_K)$ in order to optimize CTR.
- The simplest case: two time moments, $t = 0$ and $t = 1$, choice of two objects:
  - object P has CTR p_0 at time moment $t = 0$ and p_1 at time moment $t = 1$, but we are not sure what, there is a distribution;
  - object Q is known exactly, q_0 and q_1.
- We need to find x, share of views for P at time moment $t = 0$; we have N_0 views to distribute at $t = 0$ and N_1 to $t = 1$.

- Suppose we've had c clicks after choosing x; c is a random value.
- We observe c and on the second step the optimal solution is clear: we give all N_1 clicks to P iff

$$\hat{p}_1(x, c) = \mathrm{E}\left[p_1 \mid x, c\right] > q_1.$$

- I.e., we need to optimize x w.r.t. the total expected number of clicks, before we have this new information on p_1 that we will have at time $t = 1$.

- Expected number of clicks:

$$
\begin{aligned}
&N_0 x \hat{p}_0 + N_0(1-x)q_0 + N_1 \mathrm{E}_c\left[\max\{\hat{p}_1(x,c), q_1\}\right] = \\
&= N_0 q_0 + N_1 q_1 + N_0 x(\hat{p}_0 - q_0) + N_1 \mathrm{E}_c\left[\max\{\hat{p}_1(x,c) - q_1, 0\}\right].
\end{aligned}
$$

- The second term is the profit for exporing P:

$$
\mathrm{Gain}(x, q_0, q_1) = N_0 x(\hat{p}_0 - q_0) + N_1 \mathrm{E}_c\left[\max\{\hat{p}_1(x,c) - q_1, 0\}\right],
$$

this is the function we optimize w.r.t. x.

EXAMPLE: CLICKS ON A NEWS SITE

- If we approximate $\hat{p}_1(x, c)$ by a normal distribution (central limit theorem):

$$\begin{aligned}
\mathrm{Gain}(x, q_0, q_1) &= N_0 x(\hat{p}_0 - q_0) + N_1 \left[\sigma_1(x)\Phi\left(\frac{q_1 - \hat{p}_1}{\sigma_1(x)}\right) + \right. \\
&\quad \left. + \left(1 - \Phi\left(\frac{q_1 - \hat{p}_1}{\sigma_1(x)}\right)\right)(\hat{p}_1 - q_1)\right], \\
p_1 &\sim \mathrm{Beta}(a, b) \text{ (prior)}, \\
\hat{p}_1 &= \mathrm{E}_c\left[\hat{p}_1(x, c)\right] = \frac{a}{a+b}, \\
\sigma_1^2(x) &= \mathrm{Var}\left[\hat{p}_1(x, c)\right] = \frac{xN_0}{a+b+xN_0} \frac{ab}{(a+b)^2(1+a+b)}.
\end{aligned}$$

- For $K > 2$ the problem is much harder.
- What changes for several time slots?

- But this is simply estimating a static situation; how do we follow trends? (online recommender systems)
- Dynamic Gamma–Poisson (DGP) model: fix a (short) period of time t and count shows and clicks over time t.
- Suppose that over time t we have shown an item n_t times and got total reward r_t (e.g., total number of clicks $r_t \leq n_t$).
- Then we know at time t a sequence $n_1, r_1, n_2, r_2, \dots, n_t, r_t$, and want to predict p_{t+1} (CTR at time moment $t+1$).

- Probabilistic assumptions of the DGP model:
  1. $(r_t \mid n_t, p_t) \sim \text{Poisson}(n_t, p_t)$ (for given n_t and p_t the probability r_t follows a Poisson distribution).
  2. $p_t = \epsilon_t p_{t-1}$, where $\epsilon_t \sim \text{Gamma}(\mu = 1, \sigma = \eta)$ (average share of successes p_t does not change too fast, it is multiplied by a random value ϵ_t which has gamma distrubition with mean 1).
  3. Model parameters are parameters of $p_1 \sim \text{Gamma}(\mu = \mu_0, \sigma = \sigma_0)$ and η that shows how “smooth” p_t can change.
  4. Accordingly, the problem is to estimate the parameters of the posterior distribution

$$(p_{t+1} \mid n_1, r_1, n_2, r_2, \dots, n_t, r_t) \sim \text{Gamma}(\mu = ?, \sigma = ?).$$

- And Bayesian updates can be computed analytically.
- Suppose that on the previous step $t-1$ we have obtained estimates μ_t, σ_t for model parameters:

$$(p_t \mid n_1, r_1, n_2, r_2, \dots, n_{t-1}, r_{t-1}) \sim \mathrm{Gamma}(\mu = \mu_t, \sigma = \sigma_t),$$

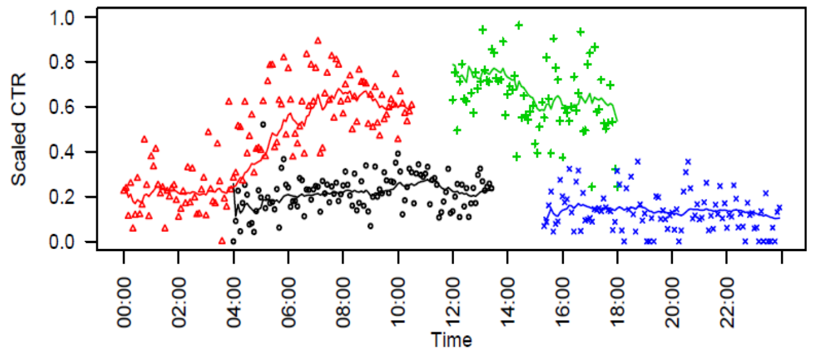
  and then got a new data point (n_t, r_t).
- Then, denoting $\gamma_t = \frac{\mu_t}{\sigma_t^2}$ (efficient sample size), we first refine the estimates μ_t, σ_t:

$$\begin{aligned} \gamma_{t|t} &= \gamma_t + n_t, \\ \mu_{t|t} &= \frac{\mu_t \gamma_t + r_t}{\gamma_{t|t}}, \\ \sigma^2_{t|t} &= \frac{\mu_{t|t}}{\gamma_{t|t}}. \end{aligned}$$

- And then generate a new prediction for $(p_{t+1} \mid n_1, r_1, \dots, n_t, r_t)$:

$$\begin{aligned}
\mu_{t+1} &= \mu_{t|t}, \\
\sigma^2_{t+1} &= \sigma^2_{t|t} + \eta \left(\mu^2_{t|t} + \sigma^2_{t|t} \right).
\end{aligned}$$

EXAMPLE



Scaled CTR
1.0
0.8
0.6
0.4
0.2
0.0
00:00
02:00
04:00
06:00
08:00
10:00
12:00
14:00
16:00
18:00
20:00
22:00
Time

Thank you for your attention!