Statistical Techniques in Robotics (16-831, S22) Lecture #13 (Wednesday, March 2)

Thompson Sampling EXP3 EXP4

Lecturer: Kris Kitani *Scribes: Seth Karten, Siva Kailas (Group 1)*

1 Review

In the last lectures, we covered two specific approaches to multi-armed bandit problems. These were the explore-exploit algorithm and UCB algorithm. Both of these approaches pertain to context-free stochastic bandit environments. In this lecture, we will cover another approach for stochastic context-free bandits called Thompsons sampling. We cover an adversarial context-free bandit algorithm called EXP3 and an adversarial contextual bandit algorithm called EXP4.

1.1 Explore-Exploit Algorithm

The explore-exploit algorithm conducts exploration first by pulling each arm M times, and then conducts exploitation by pulling the arm with highest expected mean reward. The algorithm is shown below.

Algorithm 1 Explore-Exploit(M)

for $k = 1, \ldots, K$ **do**
  for $m = 1, \ldots, M$ **do**
    $a = k$
    RECEIVE(r)
    $\hat{\mu}_k = \hat{\mu}_k + \frac{r}{M}$
  end for
end for
for $t = (K * M), \ldots, T$ **do**
  $a^{(t)} = \arg\max_{k'} \hat{\mu}_{k'}$
  RECEIVE$(r^{(t)})$
end for

The proof of the regret bound relies on Hoeffding's inequality, which is shown below.

$$p\left(\left|\sum_{t=1}^{T} \frac{r^{(t)}}{T} - \mu\right| \geq \epsilon\right) \leq 2e^{-2T\epsilon^2}$$

From this, we derived a closed form expression for ϵ given a certain probability δ is desired. This expression is shown below.

$$\epsilon = \sqrt{\frac{\log\left(\frac{2}{\delta}\right)}{2T}}$$

This is used for characterizing the lower confidence bound $\mu - \epsilon < \hat{\mu}$ and the upper confidence bound $\mu + \epsilon > \hat{\mu}$. The regret bound for the explore phase of the algorithm is shown below.

$$R_{\text{explore}} \leq \mathcal{O}(KM)$$

The regret bound for the exploit phase of the algorithm is shown below.

$$R_{\text{exploit}} \leq \mathcal{O}\left(2(T - KM)\sqrt{\frac{1}{M}}\right)$$

The regret bound for the explore-exploit algorithm is shown below in terms of the number of arms (K) and the number of time steps (T) when M is optimal ($M = \left(\frac{T}{K}\right)^{\frac{2}{3}}$).

$$R_{\text{explore-exploit}} = R_{\text{explore}} + R_{\text{exploit}} \leq \tilde{\mathcal{O}}(K^{\frac{1}{3}}T^{\frac{2}{3}})$$

1.2 Upper Confidence Bound Algorithm

The upper confidence bound algorithm is shown below.

Algorithm 2 UCB(δ', T)

for $t = 1, \ldots, T$ **do**
  if $t \leq K$ **then**
    $k = t$
  else
    $k = \arg\max_{k'} \left(\hat{\mu}_{k'} + \sqrt{\frac{\log\left(\frac{2T}{\delta'}\right)}{2T_{k'}^{(t)}}}\right)$
  end if
  `RECEIVE`$(r^{(t)})$
  $T_k^{(t)} = T_k^{(t')} + 1$
  $\hat{\mu}_k = \frac{1}{T_k^{(t)}}\left(\hat{\mu}_k(T_k^{(t)} - 1) + r^{(t)}\right)$
end for

The expression for the upper confidence $\epsilon' = \sqrt{\frac{\log\left(\frac{2T}{\delta'}\right)}{2T_{k'}^{(t)}}}$ is derived from using the Hoeffding's inequality in conjunction with union bound (Boole's inequality). Boole's inequality states that $p(\bigcup_i x_i) \leq \sum_i p(x_i)$. Thus, we can construct the relevant upper confidence bound shown below.

$$p\left(\bigcup_{t=1}^{T}\left\{|\hat{\mu}_k^{(t)} - \mu_k| > \sqrt{\frac{\log\left(\frac{2}{\delta}\right)}{2T_k^{(t)}}}\right\}\right) \leq \sum_{t=1}^{T} p\left(|\hat{\mu}_k^{(t)} - \mu_k| > \sqrt{\frac{\log\left(\frac{2}{\delta}\right)}{2T_k^{(t)}}}\right) \leq T\delta = \delta'$$

This yields the expression for ϵ' (similar to that derived for explore-exploit) $\epsilon' = \sqrt{\frac{\log\left(\frac{2T}{\delta'}\right)}{2T_k^{(t)}}}$. The regret bound for the upper confidence bound algorithm is shown below in terms of the number of arms (K) and the number of time steps (T).

$$R_{\text{UCB}} = \mathcal{O}\left(2\sqrt{\log\left(\frac{2T}{\delta}\right)}\sqrt{KT}\right) = \tilde{\mathcal{O}}(\sqrt{KT})$$

2 Thompson Sampling

Thompson sampling can be broken down into a 2-step high-level strategy as shown below.

1. Maintain a running estimate of the prior distribution hyperparameter by observing the rewards
2. Select the best arm by sampling from the estimated posterior distribution

To formally characterize the situation, let $r \sim p(r|a, \theta)$. That is, assume that each reward is drawn from a paramterized distribution (likelihood function). If we know the true parameters of the likelihood function, then the best action or arm to select can be represented as shown below.

$$a = \arg\max_k \mathbb{E}_{p(r|a_k, \theta_k^*)}[r|a_k, \theta_k^*]$$

Here, θ_k^* is the true parameter. The main issue is that we do not know the true parameters. As a result, we attempt to estimate the true parameter by computing the following posterior distribution shown below.

$$p(\theta|h^{(t)} = p(\theta|a^{(1)}, r^{(1)}, \ldots, a^{(t)}, r^{(t)})$$

This posterior distribution represents the distribution over the parameter value given a history of actions and rewards. From this, we can obtain the best estimate for θ^*, denoted $\hat{\theta}$, as shown below.

$$\hat{\theta} = \arg\max_\theta p(\theta|h^{(t)})$$

The posterior distribution $p(\theta|h^{(t)}$ can be represented using Bayes' theorem and further simplified using the Markov assumption as shown below.

$$p(\theta|h^{(t)}) = \frac{p(h^{(t)}|\theta)p(\theta)}{p(h^{(t)})} \propto \prod_t p(r^{(t)}|a^{(t)}, \theta)p(\theta)$$

Note that the above representation has an equivalent recursive representation shown below.

$$p(\theta|h^{(t)}) \propto p(r^{(t)}|a^{(t)}, \theta)p(\theta|h^{(t-1)})$$

With this, we can now compute the best estimate for θ^*, denoted $\hat{\theta}$, incrementally as shown below.

$$\hat{\theta} = \arg\max_\theta p(r^{(t)}|a_k^{(t)}, \theta_k)p(\theta_k|h_k^{(t-1)})$$

2.1 Conjugate Priors

A conjugate distribution is defined as a distribution in which the posterior $p(\theta|x)$ and the prior $p(\theta)$ are the same type of distribution. From this, the prior is denoted as the conjugate prior of the likelihood function. To demonstrate this (and use later), let us define the Bernoulli and Beta distributions respectively shown below.

$$p(r|\theta) = \theta^r(1-\theta)^{1-r}$$

$$p(\theta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1}$$

Here, the Gamma function is $\Gamma(n) = (n-1)!$. We can show that the Beta distribution is the conjugate prior of the Bernoulli distribution. This is shown below.

$$p(\theta|r) \propto p(r|\theta)p(\theta)$$

$$p(\theta|r) \propto (\theta^r(1-\theta)^{1-r})\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1}$$

$$p(\theta|r) \propto \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{(r+\alpha)-1}(1-\theta)^{(1-r+\beta)-1}$$

$$p(\theta|r) \propto \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{(r+\alpha)-1}(1-\theta)^{(1-r+\beta)-1}$$

$$p(\theta|r) \propto \theta^{\alpha'-1}(1-\theta)^{\beta'-1}$$

From above, we see that the posterior distribution is also a Beta distribution with parameters $\alpha' = (r+\alpha)$ and $\beta' = (1-r+\beta)$. We use this fact to construct the Beta-Bernoulli bandit by parameterizing the likelihood, prior, and posterior using conjugate distributions. This will allow for efficient updates.as a Bernoulli distribution and the prior (and posterior) as a Beta distribution.

2.2 Beta-Bernoulli Bandit

To construct the Beta-Bernoulli bandit, we first parameterize the likelihood as a Bernoulli distribution as shown below.

$$p(r|a_k, \theta_k) = \theta_k^r(1-\theta_k)^{1-r}$$

Next, we parameterize the prior as a Beta distribution as shown below.

$$p(\theta_k) = \frac{\Gamma(\alpha_k+\beta_k)}{\Gamma(\alpha_k)\Gamma(\beta_k)}\theta_k^{\alpha_k-1}(1-\theta_k)^{\beta_k-1}$$

Since the posterior is simply a Beta distribution with parameters $\alpha' = (r+\alpha)$ and $\beta' = (1-r+\beta)$, it is easy to compute. This can be shown inductively. For timestep 1, we have the following update shown below for the posterior for a single arm.

$$p(\theta|r) \propto p(r|\theta)p(\theta)$$

$$p(\theta|r) \propto (\theta^r(1-\theta)^{1-r})\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1}$$

$$p(\theta|r) \propto \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{(r+\alpha)-1}(1-\theta)^{(1-r+\beta)-1}$$

$$p(\theta|r) \propto \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{(r+\alpha)-1}(1-\theta)^{(1-r+\beta)-1}$$

$$p(\theta|r) \propto \theta^{\alpha'-1}(1-\theta)^{\beta'-1}$$

Evidently, the posterior update is simply $\alpha' = (r+\alpha)$ and $\beta' = (1-r+\beta)$. Similarly, the update at timestep t is shown below.

$$p(\theta|r) \propto p(r|\theta)p(\theta|h^{t-1})$$

Note that inductively $p(\theta|h^{t-1})$ follows a Beta distribution. The remainder of the derivation is shown below.

$$p(\theta|r) \propto (\theta^r(1-\theta)^{1-r})\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1}$$

$$p(\theta|r) \propto \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{(r+\alpha)-1}(1-\theta)^{(1-r+\beta)-1}$$

$$p(\theta|r) \propto \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{(r+\alpha)-1}(1-\theta)^{(1-r+\beta)-1}$$

$$p(\theta|r) \propto \theta^{\alpha'-1}(1-\theta)^{\beta'-1}$$

Thus, the update at any given timestep is simply $\alpha' = (r+\alpha)$ and $\beta' = (1-r+\beta)$. We can now present the algorithm for the generic Thompson sampling as shown below.

Algorithm 3 Thompsons Sampling

for $t = 1, \ldots, T$ **do**
  $\theta_k \sim p(\theta|h_k)\,\forall k$
  $a_{\hat{k}}^{(t)} = \arg\max_k \mathbb{E}_{p(r|a_k,\theta_k)}[r|a_k,\theta_k]$
  `RECEIVE`$(r^{(t)})$
  $p(\theta_{\hat{k}}|h_{\hat{k}}) \propto p(r^{(t)}|a_{\hat{k}}^{(t)},\theta_{\hat{k}})p(\theta_{\hat{k}}|h_{\hat{k}})$
end for

The algorithm for the specific variant of Thompson sampling which uses a Bernoulli likelihood function, Beta prior distribution (and posterior distribution), and the derived posterior update is shown below.

Algorithm 4 Bern-Beta Thompsons Sampling

for $t = 1, \ldots, T$ **do**
  $\theta_k \sim p(\theta;\alpha_k,\beta_k)\,\forall k$
  $a_{\hat{k}}^{(t)} = \arg\max_k \mathbb{E}_{p(r|a_k,\theta_k)}[r|a_k,\theta_k]$
  `RECEIVE`$(r^{(t)})$
  $\alpha_{\hat{k}} = \alpha_{\hat{k}} + r^{(t)}$
  $\beta_{\hat{k}} = \beta_{\hat{k}} + 1 - r^{(t)}$
end for

2.3 Empirical Performance and Regret

Thompson sampling has better empirical performance when compared to the UCB algorithm, especially after a large number of time steps have passed. The regret of Thompson sampling is shown below.

$$BR(T) = \mathcal{O}(\sqrt{KT\log(T)}$$

3 Bandit: EXP3

The main idea of Exponential-Weight Update algorithm for Exploration and Exploitation (EXP3) is addressing context-free bandits in an adversarial environment.

Algorithm 5 EXP3($\gamma \in [0, 1]$)

for $t = 1, \ldots, T$ **do**
  $p^t = \frac{w^t}{\sum_k w_k^t}$
  $k \sim \texttt{Multinomial}(p^t)$
  $a^t = a_k$
  $\texttt{RECEIVE}(r^t \in [0, 1])$
  $w_k^{t+1} = w_k^t \exp\{\gamma \frac{r^t}{p_k^t}\}$
end for

3.1 The Unbiased Estimator

To understand the ratio in the update term $\frac{r^t}{p_k^t}$, we need to understand the unbiased estimator.

3.1.1 Case 1: Single-arm stochastic bandit

In the single-arm stochastic bandit case, the selected action from the probability distribution is always 1. Then we can sample the reward from the true distribution. We compute the empirical mean reward in the following manner:

First define an "Estimator",

$$c_1^t(a^t) = 1[a^t = 1] * r^t.$$

c_1 represents the first arm. We can take the expected value of the estimator to find the estimated reward,

$$\mathbb{E}_{p(a)}[c_1^t(a^t)] = p(a^t = 1) * 1[a^t = 1] * r^t = r^t,$$

since there is only one arm, the expected value is just the reward. The expected value of the estimator over T rounds is,

$$\mathbb{E}_{p(a)}[\frac{1}{T}\sum_{t=1}^{T} c_t^1(a^t)] = \frac{1}{T}\sum_{t=1}^{T} r^t,$$

which is the empirical estimate of the true mean, μ_t.

3.1.2 Case 2: Two-arm stochastic bandit

In the two-arm stochastic bandit case, the selected action from the probability distribution is 50/50. We still sample the reward from the true distribution.

We can find the empirical mean reward again. Naively we can start with the same estimators. For arm 1,

$$c_1^t(a^t) = 1[a^t = 1] * r^t.$$

For arm 2,

$$c_2^t(a^t) = 1[a^t = 2] * r^t.$$

We can find the expected value of the estimator of the first arm (one step),

$$\mathbb{E}_{p(a)}[c_1^t(a^t)] = p(a^t = 1) * c_1^t(1) + p(a^t = 2) * c_1^t(2)$$

$$\mathbb{E}_{p(a)}[c_1^t(a^t)] = p(a^t = 1)1[a^t = 1]r^t + p(a^t = 2)1[a^t = 2]r^t$$

$$\mathbb{E}_{p(a)}[c_1^t(a^t)] = 0.5 * 1 * r^t + 0.5 * 0 * r^t = 0.5 * r^t.$$

The expected value of the estimator over round T is,

$$\mathbb{E}_{p(a)}[\frac{1}{T}\sum_{t-1}^{T} c_1^t(a^t)] = \frac{1}{T}\sum_{t-1}^{T} 0.5r^t.$$

Since the reward is scaled back by 0.5, we need to implement importance sampling (inverse probability weighting) as a solution.

3.2 Importance Sampling

We can fix the previously seen scaling issue by dividing by the inverse probability, p_1, to get an unbiased estimator,

$$c_1^t(a^t) = 1[a^t = 1] * r^t * \frac{1}{p_1}.$$

Then the expected value of the estimator will be,

$$\mathbb{E}_{p(a)}[c_1^t(a^t)] = p(a^t = 1) * c_1^t(1) + p(a^t = 2) * c_1^t(2)$$

$$\mathbb{E}_{p(a)}[c_1^t(a^t)] = p(a^t = 1)1[a^t = 1]r^t\frac{1}{p_1} + p(a^t = 2)1[a^t = 2]r^t\frac{1}{p_1}$$

$$\mathbb{E}_{p(a)}[c_1^t(a^t)] = 0.5 * 1 * r^t\frac{1}{0.5} + 0.5 * 0 * r^t\frac{1}{0.5} = r^t.$$

Thus we have showed in the two arm case that the expected value is the reward value. In general for k arms,

$$c_k^t(a^t) = 1[a^t = k] * r^t * \frac{1}{p_k} \ \forall k$$

Then in the general case, the expected value of the estimator is,

$$\mathbb{E}[c_a^t(a^t)] = \sum_{k=1}^{K} p_k * 1[a^t = a] * r^t * \frac{1}{p_k}$$

$$\mathbb{E}[c_a^t(a^t)] = p(a^t = a) * r^t * \frac{1}{p_a} = r^t.$$

Let us look at the update equation again to determine the relationship between the reward r and the probability p,

$$w_k^{t+1} = w_k^t \exp\{\gamma * \frac{r^t}{p_k^t}\}.$$

A huge reward and a tiny probability will result in a large exponent, which results in exploitation. A small reward and tiny probability will result in a small exponent, which results in exploration. A 0 reward and moderate probability will result in a zero exponent, which will not change the weights.

3.3 High-level Strategy and Regret

Additionally, we can convert the partially observed loss/gain function to a fully observed loss/gain function using the unbiased estimate of the loss/gain to analyze as a prediction with one expert advice problem.

We note that the regret for EXP3 is

$$R \leq O(\sqrt{TK \log K},$$

where T is the time horizon, K is the number of arms, and

$$\gamma = \sqrt{\frac{\log K}{TK}}.$$

Thus EXP3 is a No Regret algorithm.

4 Contextualized Bandits: EXP4

The main idea of Exponential-Weight Update algorithm for Exploration and Exploitation with Experts (EXP4) is addressing contextual bandits in an adversarial environment.

Algorithm 6 EXP4($\gamma \in (0, 1], T$)

$w^1 \leftarrow 1 \in \mathbb{R}^N$
for $t = 1, \ldots, T$ **do**
  `RECEIVE(`$X^t \in \mathbb{R}^{N \times K}$`)`
  $q^t = \frac{w^t}{\|w^t\|} X^t \in \Delta^K$
  $k^t \sim$ `Multinomial`(q^t)
  `Receive`(r^t)
  $\hat{r}^t = \frac{r^t}{q_k^t} \mathbb{I}[k = k^t] \in \mathbb{I}^K$
  $g^t = X^t * \hat{r}^t \in \mathbb{R}^N$
  $w_n^{t+1} = w_n^t \exp\{\gamma * g_n^t\}\ \forall n$
end for

We refer to the following variable definitions for the `EXP4` algorithm.

- $[K]$ is the set of arms
- $x_n \in \Delta^K$ is the set of advice of the n-th expert, which is a distribution over K arms
- $X \in \mathbb{R}^{N \times K}$ is the matrix of expert advice from N experts
- $w \in \mathbb{R}^{1 \times N}$ is the vector of unnormalized weights over experts
- $\mathbb{I}[k] \in \mathbb{I}^K$ is the indicator vector, k is one or zero otherwise
- r^t is the reward for pulling arm k at time t
- $\hat{r}^t \in \mathbb{R}^K$ is the fully observed reward vector at time t

4.1 EXP3 vs. EXP4

In `EXP4`, we parameterize the distribution by

$$q^t = \frac{w^t}{\|w^t\|} * X^t,$$

while in `EXP3`, we parameterize the distribution by

$$p^t = \frac{w^t}{\|w^t\|}.$$

`EXP4` uses the matrix of expert advice from N experts to determine the probability over actions and then draw the action from the distribution as the key difference.

This is reflected in the g_n^t terms in `EXP4` versus the reward over inverse probability ratio, $\frac{r^t}{p_k^t}$, in `EXP3`.

4.2 Regret

The regret of EXP4 is,

$$R_{\texttt{EXP4}} \leq \sqrt{KT \log N},$$

where K is the number of arms, T is the time, and N is the number of experts.

Appendix

In the lecture, there were some people that were confused about Bayesian probability and Markov properties. In the appendix, we will go over some of these properties. Given,

$$P(x, y) = P(x|y)P(y),$$

and

$$P(x, y) = P(y|x)P(x),$$

it follows that,

$$P(x|y)P(y) = P(y|x)P(x).$$

Thus, we have Bayes's rule,

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}.$$

Intuitively, this can be thought of as,

$$P(x|y) = \frac{\texttt{likelihood} * \texttt{prior}}{\texttt{evidence}}.$$

The $P(y)$, our evidence, may be treated as a constant, so we have,

$$P(x|y) \propto P(y|x)P(x),$$

where we can estimate which hypothesis is best given the data, i.e., a Max a posteriori (MAP) estimation. In certain scenarios, we do not have access to $P(x)$ but we may assume it has a uniform distribution,

$$P(x|y) \propto P(y|x),$$

which is a maximum likelihood estimate.

Now, let us understand conditional independence. Let X, Y, Z be events. We say that X and Y are conditionally independent given Z if and only if $P(C) > 0$ and,

$$P(X|Y, Z) = P(X|Z).$$

This property is equivalent to saying,

$$P(X, Y|Z) = P(X|Z)P(Y|Z).$$

From our Markov assumption, we know that a node is independent of its non-descendants given its parents. So, given $X_1, \ldots, X_n$ are topologically sorted (that is, if X_i is a parent of X_j, then $i < j$), then with the chain rule, the full joint distribution is,

$$P(X_1, \ldots, X_n) = \Pi_{i=1}^{n} P(X_i|X_1, \ldots, X_{i-1}),$$

which due to conditional independence, can be written as,

$$P(X_1, \ldots, X_n) = \Pi_{i=1}^{n} P(X_i|\texttt{Parents}(X_i)).$$

Another property of note is that a node is conditionally independent of all other nodes in the network gives its Markov blanket, which consists of its parents, children, and children's parents.