Multi-Armed Bandits

Efficient Algorithms for Learning with Semi-Bandit Feedback

Abdul, Adwait, Anirudh

30^{th} May 2021

Overview

1. Motivation and Introduction
2. FPL with GR Algorithm
3. Regret analysis and Comparison with State of art for FPL with GR
4. CombLinTS Algorithm
5. Regret analysis and Comparison with State of art for CombLinTS
6. CombLinUCB Algorithm
7. Regret analysis and Comparison with State of art for CombLinUCB

Motivation and Introduction

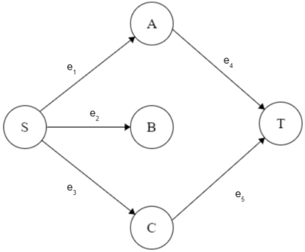
Goals of this section

- Motivate using Online Least Cost Path problem.
- Understanding the Semi-Bandits setting.
- Understanding the Combinatorial problem.
- Understanding Linear Generalization.
- Formally introducing the problem.

Motivation
The Online Least Cost Path problem

- We have a (directed or undirected) graph G with
  - $L =$ number of edge,
  - $w_i =$ cost associate with the edge i.
  - $S =$ start node and,
  - $T =$ target node.
- The goal: Find the cheapest path from S to T w.r.to the cost associated with the path.

Motivation
The Online Least Cost Path problem - An example



- In the above graph $L = 5$.
- A path is denoted as a $5 - D$ vector.
  - Path SAT, $A_1 = \boxed{1 \mid 0 \mid 0 \mid 1 \mid 0}$
  - Path SCT, $A_2 = \boxed{0 \mid 0 \mid 1 \mid 0 \mid 1}$
- Set of valid paths, $\mathcal{A} = \{A_1, A_2\} \subseteq \{0, 1\}^L$

The Online Least Cost Path problem - Cost computation

- Let us say we have decided to go with the path SAT. $\boxed{1}\boxed{0}\boxed{0}\boxed{1}\boxed{0}$
- To compute the cost, we may get one of the following:
  1. Full information setting:
  $$w = \boxed{0.3}\boxed{0.2}\boxed{0.7}\boxed{0.5}\boxed{0.4}$$
  2. Semi-Bandits setting:
  $$w = \boxed{0.3}\boxed{\text{x}}\boxed{\text{x}}\boxed{0.5}\boxed{\text{x}}$$
  3. Full-Bandits setting:
  $$l = \boxed{0.8}$$
- For the Full information and Semi-Bandits setting, the loss is $l = A \cdot w$

Motivation
How is this different from the familiar MAB setting?

- In the MAB setting, we choose $i \epsilon [L]$.
  - i can take one of L values.
- However, here we choose $A \epsilon \mathcal{A} \subseteq \{0, 1\}^L$.
  - A can take one of potentially 2^L values.

- That is, we want to choose one or more arms in each round.
- In the Online Least Cost path problem,
  Edge $\equiv$ Arm
  Path $\equiv$ multiple Arms.
- This is an example of Combinatorial Optimization problem.

Motivation

- An immediate issue with this setting is that applying algorithms such as $EXP - 3$ may not have a good regret bound, since both L and $|\mathcal{A}|$ can be very large in real life settings.
- Can we make it independent of L?
    - Here's a sneak peak...

- In this setting (and many other real life settings), we have access to features of edges, such as,
  1. Length,
  2. Traffic, and
  3. Road Quality.

	Length	Traffic	Road Quality
e_1	10	3	7
e_2	4	6	3
e_3	8	1	4
e_4	6	2	5
e_5	3	7	2

- Hence, we know a (possibly imperfect) generalization matrix Φ.
  In this case,

$$\Phi = \begin{bmatrix} 10 & 3 & 7 \\ 4 & 6 & 3 \\ 8 & 1 & 4 \\ 6 & 2 & 5 \\ 3 & 7 & 2 \end{bmatrix}$$

- We assume that the cost of an edge is a linear combination of its feature values.

$$cost(e) = \theta_1 \; length(e) + \theta_2 \; traffic(e) + \theta_3 \; quality(e) = \vec{\theta} \cdot \phi_e$$

- Now, the task is to estimate the entries in $\vec{\theta}$.
- Notice the advantage:
  The values to be estimated equal the number of features, and not L.

Let's formalize what all we have seen till now.

A Combinatorial Optimization Problem

Can be represented as a triple $(E, \mathcal{A}, w)$ where,

- E = Set of L arms.
- $\mathcal{A}$ = Set of (some or all) subsets A of E with $|A| \leq K$.
- $w : E \rightarrow \mathbb{R}$ is the weight function.

- The total weight of $A \in \mathcal{A}$ is $\sum_{e \in A} w_e$.

Semi-Bandit Problem

The losses associated to only the chosen arms are seen.

Linear Generalization

For $e \epsilon [L], w(e) = \Phi_e \theta$,
where Φ_e is the feature vector of the arm e, and θ indicates the weights of the features.

The Combinatorial semi-bandits

It is an online learning problem where at each step the learning agent chooses a subset of arms A subject to combinatorial constraints, and then observes weights of the selected arms, $\{\mathrm{w}(e) : e \in A\}$, and gets their sum $f(A, \mathrm{w})$ as a payoff.

The Goal

- The learner chooses a subset of arms A^t at round t. The loss suffered at this round $R_t = f(A^*, \mathrm{w_t}) - f(A^t, w)$ (In the Reward setup).
- We want the cumulative Regret to be *good*,
  And the algorithm to be efficient when applied in Large scale settings.

Efficient Algorithm for Learning with Semi-Bandit Feedback (Paper 1)

Learning with Semi-Bandit Feedback

- The paper considers the problem of online Combinatorial Optimization under semi-bandit feedback. (No linear Generalization)
- The paper assumes:
  - Finite Decision Set (potentially very large)
  - Efficient offline combinatorial optimization is possible
  - Elements of the decision set can be described with d-dimensional arrays with at most m non-zero entries.
  - For example a decision vector in which 3 arms out of possible 7 arms are played will look like $\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$

General Protocol for online Combinatorial Optimization

- **Parameters:** set of decision vectors $\mathcal{S} = \{v(1), v(2), \ldots, v(N)\} \subseteq \{0,1\}^d$, number of rounds T;
  For all $t = 1, 2, \ldots, T$, **repeat**
  1. The learner chooses a probability distribution p_t over $\{1, 2, \ldots, N\}$.
  2. The learner draws an action I_t randomly according to p_t. Consequently, the learner plays decision vector $V_t = v(I_t)$.
  3. The environment chooses loss vector ℓ_t.
  4. The learner suffers loss $V_t^\top \ell_t$.
  5. The learner observes some feedback based on ℓ_t and V_t.
- Generic framework to accommodate a number of interesting problem instances such as path planning, ranking and matching problems , finding minimum weight spanning trees and cut sets etc.

Normal Bandit Setting

- Total N arms and we chose only one arm.
- A distribution $\mathsf{p_t}$ over the arms where $p_{t,i} = \mathcal{P}[\mathsf{I_t} = i | \mathcal{F}_{t-1}]$ where $\mathcal{F}_{t-1}$ is the history of the learners observation and choice made upto step $t-1$.
- Most bandit algorithms rely on feeding some loss estimates to black box algorithm like Hedge etc. A common loss estimate used is
  $\hat{\ell}_{t,i} = \frac{\ell_{t,i}}{p_{t,i}} \mathbb{I}[\mathsf{I_t} = \mathsf{i}]$ (An unbiased estimate)
- Almost all existing algorithms use some form of the above loss estimate. But $\mathsf{p_t}$ is not readily available for all algorithms like FPL.
- Following loss estimate is proposed

Geometric Resampling

- In round t the learner draws $\mathrm{I_t} \sim \mathrm{p_t}$
- For n = 1, 2 , ...
  - Let $n \leftarrow n + 1$
  - Draw $I'(n) \sim \mathrm{p_t}$
  - If $I'(n) = I_t$, break
- Let $K_t = n$

- $\hat{\ell_{t,i}} = \ell_{t,i}\mathbb{I}[\mathrm{I}_t = i]K_t$ (Also unbiased)
- The number samples can be unbounded, so we cap the number of samples by M and use $\hat{K}_t = min(K_t, M)$
- Introduces some bias, but if M is chosen appropriately the performance does not hurt much.

Generalizing Geometric Sampling to Semi-Bandit Feedback

- In round t again the learner draws $\mathsf{I_t} \sim \mathsf{p_t}$
- For each arm draw M samples (because anyway we are capping by M). We can effectively draw only M samples and use them for all the arms being played. So, draw M additional indices $I'_t(1), I'_t(2) \dots I'_t(M) \sim \mathsf{p_t}$
- Define $K_{t,j} = min\{1 \leq s \leq M : v_j(I'_t(s)) = v_j(I_t)\}$ for all arms.
- Use Loss estimate for each arm as $\hat{\ell}_{t,j} = K_{t,j} V_{t,j} \ell_{t,j}$

- Since $V_{t,j}$ is non-zero only for the arms being played the estimates are well defined.

FPL with Geometric Sampling

-

```
Input: S = {v(1), v(2), ..., v(N)} ⊆ {0,1}^d, η ∈ R+, M ∈ Z+;
Initialization: L̂(1) = ··· = L̂(d) = 0;
for t=1,...,T do
    Draw Z(1), ..., Z(d) independently from distribution Exp(η);
    Choose action I = argmin_{i∈{1,2,...,N}} { v(i)^T (L̂ − Z) };
    K(1) = ··· = K(d) = M;
    k = 0;                                   /* Counter for reoccurred indices */
    for n=1,...,M-1 do                                 /* Geometric Resamplig */
        Draw Z'(1), ..., Z'(d) independently from distribution Exp(η);
        I(n) = argmin_{i∈{1,2,...,N}} { v(i)^T (L̂ − Z') };
        for j=1,...,d do
            if v(I(n))(j) = 1 & K(j) = M then
                K(j) = n;
                k = k + 1;
                if k = ||v(I)||_1 then break;    /* All indices reoccurred */
            end
        end
    end
    for j=1,...,d do  L̂(j) = L̂(j) + K(j)v(I)(j)ℓ(j) ;              /* Update */
end
```

Computational Efficiency under Semi Bandit Feedback

- The expected number of times the algorithm draws an action up to time step T can be upper bounded by dT. This can be shown as

- For each co-ordinate that the original arm had 1, we take sampling until we get 1 in the same co-ordinate again. (Here we do not assume cutoff)
- Let $q_{t,k} = \mathbb{E}[V_{t,k} | \mathcal{F}_{t-1}]$
- At time step t for a given co-ordinate k with 1, the expected number of samples is $\frac{1}{q_{t,k}}$ while the probability of co-ordinate k being 1 is $q_{t,k}$
- So, expected number of samples is $\sum_{k=1}^{d} q_{t,k} \frac{1}{q_{t,k}} = d$. (For one round)
- For T rounds it will be dT.
- The expected running time is comforting.

Regret Bound

- The total expected regret of FPL with Geometric Resampling satisfies $R_n \leq \frac{m(\log d+1)}{\eta} + \eta m d T + \frac{dT}{eM}$ under semi-bandit information.
- For the full information setting if $C_T = \sum_{t=1}^{T} \mathbb{E}[\mathrm{V}_\mathrm{t}^\mathrm{T} \ell_\mathrm{t}]$ then the expected regret of FPL satisfies $R_n \leq \frac{m(\log d+1)}{\eta} + \eta m C_t$ under full information.

- For $\eta = \sqrt{\frac{\log d+1}{dT}}$ and $M \geq \frac{\sqrt{dT}}{eM(\sqrt{\log d+1})}$, $R_n = \mathcal{O}(m\sqrt{dT \log d})$
- For $\eta = \sqrt{\frac{(\log d+1)}{mT}}$ $R_n = \mathcal{O}(m^{\frac{3}{2}}\sqrt{T(\log d + 1)})$.

Comparison with State of Art

- Full information setting $\rightarrow$ Optimal Regret $\rightarrow \mathcal{O}(m\sqrt{T\log d})$
  FPL-with-GR achives $\mathcal{O}(m^{\frac{3}{2}}\sqrt{T(\log d + 1)})$ off by a factor of $\sqrt{m}$
- Semi-bandit setting $\rightarrow$ Optimal Regret $\mathcal{O}(\sqrt{mdT})$.
  FPL-with-GR achives $\mathcal{O}(m\sqrt{dT\log d})$ off by a factor of $\sqrt{m\log d}$

Conclusion and Future work

- The paper has introduced the first general efficient algorithm for online combinatorial optimization under semi-bandit feedback.
- It remains as an open problem whether the gaps we have shown can be closed for FPL-style algorithms.
- The most important open problem is the development efficient online linear optimization with full bandit feedback.

Efficient Learning in Large Scale Combinatorial Semi-Bandits (Paper 2)

Large Scale Combinatorial Semi-Bandits

- This paper looks at efficient algorithms for stochastic combinatorial semi-bandits with **linear generalization**
- We look at algorithms that have regret bound **INDEPENDENT OF L** (hence large scale)

Setup Used for Combinatorial optimization

Combinatorial setup

$$(E, \mathcal{A}, P)$$

- $E = \{1, \ldots, L\} \equiv$ arms (the ground set)
- $\mathcal{A} \subseteq \{A \subseteq E : |A| \leq K\} \equiv$ allowed combinations of arms
- $P \equiv$ probability distribution over weights $\mathrm{w} \in \mathbb{R}^L$ on E. ($\bar{\mathrm{w}} = \mathbb{E}[\mathrm{w}]$)
- After the arms A are pulled, we observe the individual return of each arm $\{\mathrm{w}(e) : e \in A\}$
- $f(A, \mathrm{w}) \equiv$ loss on pulling $A \in \mathcal{A}$

We assume linear generalization(possibly imperfect) and that the agent knows the generalization matrix $\Phi \in \mathbb{R}^{L \times d}$.If $\bar{\mathrm{w}} \in span[\Phi]$ we call it as coherent learning case otherwise agnostic. WLOG rank$[\Phi] = d$.

Performance Metrics

- $R_t = f(A^*, \mathrm{w_t}) - f(A^t, w)$ where $A^* = ORACLE(E, \mathcal{A}, \bar{w})$
- For fixed $\bar{\mathrm{w}}$,
  $R(T) = \sum_{t=1}^{T} \mathbb{E}[R_t | \bar{\mathrm{w}}] \equiv$ **EXPECTED CUMULATIVE REGRET** where expectation is over random weights and possible randomization in the algorithm.
- for randomly generated $\bar{\mathrm{w}}$,
  $R_{bayes}(T) = \sum_{t=1}^{T} \mathbb{E}[R_t] \equiv$ **BAYES CUMULATIVE REGRET** where expectation is over random weights, possible randomization in the algorithm and also over $\bar{\mathrm{w}}$.
- $\theta^* = argmin_{\theta} || \bar{\mathrm{w}} - \Phi\theta ||$. Since $rank[\Phi] = d$, θ^* is uniquely defined. Also in coherent learning case $\bar{\mathrm{w}} = \Phi\theta^*$.

ALGORITHMS

KALMAN FILTERING for updating parameters

Algorithm 1

Input: $\bar{\theta}_t$, Σ_t, σ, and feature-observation pairs $\{(\phi_e, \mathrm{w}_t(e)) : e \in A^t\}$
Initialize $\bar{\theta}_{t+1} \leftarrow \bar{\theta}_t$ and $\Sigma_{t+1} \leftarrow \Sigma_t$
for $k = 1, \ldots, |A^t|$ **do**
Update $\bar{\theta}_{t+1}$ and Σ_{t+1} as follows, where a_k^t is the kth element in A^t

$$\bar{\theta}_{t+1} \leftarrow \left[I - \frac{\Sigma_{t+1}\phi_{a_k^t}\phi_{a_k^t}^T}{\phi_{a_k^t}^T \Sigma_{t+1}\phi_{a_k^t} + \sigma^2} \right] \bar{\theta}_{t+1} + \left[\frac{\Sigma_{t+1}\phi_{a_k^t}}{\phi_{a_k^t}^T \Sigma_{t+1}\phi_{a_k^t} + \sigma^2} \right] \mathrm{w}_t\left(a_k^t\right) \quad \text{and} \quad \Sigma_{t+1} \leftarrow \Sigma_{t+1} - \frac{\Sigma_{t+1}\phi_{a_k^t}\phi_{a_k^t}^T \Sigma_{t+1}}{\phi_{a_k^t}^T \Sigma_{t+1}\phi_{a_k^t} + \sigma^2},$$

end for
Output: $\bar{\theta}_{t+1}$ and Σ_{t+1}

Combinatorial Linear Thompson Sampling

CombLinTS

Input: Combinatorial structure $(E, \mathcal{A})$, generalization matrix $\Phi \in \mathbb{R}^{L \times d}$, algorithm parameters $\lambda, \sigma > 0$, oracle `ORACLE`

Initialize $\Sigma_1 \leftarrow \lambda^2 I \in \mathbb{R}^{d \times d}$ and $\bar{\theta}_1 = 0 \in \mathbb{R}^d$

for all $t = 1, 2, \ldots, n$ **do**

  Sample $\theta_t \sim N\left(\bar{\theta}_t, \Sigma_t\right)$

  Compute $A^t \leftarrow \texttt{ORACLE}(E, \mathcal{A}, \Phi\theta_t)$

  Choose set A^t, and observe $\mathrm{w}_t(e), \forall e \in A^t$

  Compute $\bar{\theta}_{t+1}$ and Σ_{t+1} based on Algorithm 1

end for

CombLinTS: Key Points

- λ $\rightarrow$inverse regularization parameter.
- smaller λ makes the covariance matrix $\sum_t$ closer to 0.
- smaller $\lambda \Rightarrow$ narrower prior$\Rightarrow$ insufficient exploration $\Rightarrow$ degraded performance of CombLinTS.
- σ controls the decrease rate of covariance matrix Σ_t.
- Large σ will lead to slow learning and a smaller σ will make the algorithm quickly converge to some sub-optimal coefficient vector.

CombLinTS: Regret bound

Regret Bound

- If $\bar{w} = \Phi\theta^*$, the prior on θ^* is $\mathcal{N}(0, \lambda^2 I)$, the noises $(w_t(e) - \bar{w}(e))$ are i.i.d sampled from $\mathcal{N}(0, \sigma^2)$ then CombLinTS guarantees: $R_{Bayes}(T) = \tilde{O}(K\lambda\sqrt{dTmin\{\ln(L), d\}})$

- The conditions ensure it is a coherent gaussian case.
- The $\tilde{O}$ notation hides the logarithmic factors.
- The regret bound is a minimum of two bounds. The first bound is L-dependent as $O(\sqrt{\ln(L)})$. The second bound is L-independent but is $\tilde{O}(d)$ instead of $\tilde{O}(\sqrt{d})$.

Combinatorial Linear UCB

CombLinUCB

Input: Combinatorial structure $(E, \mathcal{A})$, generalization matrix $\Phi \in \mathbb{R}^{L \times d}$, algorithm parameters $\lambda, \sigma, c > 0$, oracle `ORACLE`

Initialize $\Sigma_1 \leftarrow \lambda^2 I \in \mathbb{R}^{d \times d}$ and $\bar{\theta}_1 = 0 \in \mathbb{R}^d$

for all $t = 1, 2, \ldots, n$ **do**

Define the UCB weight vector $\hat{\mathrm{w}}_t$ as

$$\hat{\mathrm{w}}_t(e) = \langle \phi_e, \bar{\theta}_t \rangle + c\sqrt{\phi_e^T \Sigma_t \phi_e} \quad \forall e \in E$$

Compute $A^t \leftarrow \texttt{ORACLE}(E, \mathcal{A}, \hat{\mathrm{w}}_t)$

Choose set A^t, and observe $\mathrm{w}_t(e), \forall e \in A^t$

Compute $\bar{\theta}_{t+1}$ and Σ_{t+1} based on Algorithm 1

end for

CombLinUCB: Key Points

- $\lambda \rightarrow$inverse regularization parameter.
- σ controls the decrease rate of covariance matrix Σ_t..
- The constant c controls the degree of optimism.
- Small $c \Rightarrow$ algorithm might converge to some sub-optimal coefficient vector due to insufficient exploration.
- Large $c \Rightarrow$ excessive exploration and slow learning.

CombLinUCB: Regret Bound

Assuming P is a subset of $[0,1]^L$, the stochastic item weights are statistically independent under P. Then for the coherent learning case i.e. $\bar{\mathrm{w}} = \Phi\theta$ we have: For any λ , σ and $\delta \in (0,1)$ and any c satisfying $c \geq \frac{1}{\sigma}\sqrt{d\ln(1+\frac{Tk\lambda^2}{d\sigma^2}) + 2\ln(\frac{1}{\delta})} + \frac{||\theta^*||_2}{\lambda}$ we have:

$$R(T) \leq 2cK\lambda\sqrt{\frac{dT\ln(1+\frac{Tk\lambda^2}{d\sigma^2})}{\ln(1+\frac{\lambda^2}{\sigma^2})}} + TK\delta$$

Specifically if we chose $\lambda = \sigma = 1$, $\delta = \frac{1}{nK}$ and c is the lower bound on its above condition then:

Regret Bound

$$R(T) = \tilde{O}(Kd\sqrt{T})$$

Comparison with state of art

Standard Result

Standard results have been set for the no generalisation case.

No generalization $\Rightarrow$ lower bound of $\Omega(\sqrt{LKT})$

$\Rightarrow \Phi = \mathrm{I} \Rightarrow L = d \Rightarrow$ No generalization lower bound = $\Omega(\sqrt{KdT})$

CombLinTS

$\tilde{O}(\sqrt{Kmin\{\ln(L), d\}})$ larger than the $\Omega(\sqrt{KdT})$

CombLinUCB

$\tilde{O}(\sqrt{Kd})$ larger than the $\Omega(\sqrt{KdT})$

Note

The $\Omega(\sqrt{d})$ and $\Omega(\sqrt{K})$ factors are due to linear generalization. Full tightness analysis has been left to future work.

Conclusion and Future Work

- This paper has introduced two learning algorithms CombLinTS and CombLinUCB for stochastic combinatorial semi bandits with linear genrelariation.
- The main contribution has been that the paper has successfully introduced L independent bounds which is highly useful for real life problems e.g Online advertisements(millions of users and products).
- This paper has left it open to derive bounds for the agnostic learning case.
- Another open problem is how to extend the results to combinatorial semi bandits with non-linear generalization.

TAKEAWAYS

What have we seen so far

What have we seen so far

- **Geometric Resampling:**
  - Don't need p_t explicitly;
  - Computationally efficient;
  - Fits into any semi bandit algorithm

What have we seen so far

- **Geometric Resampling:**
  - Don't need p_t explicitly;
  - Computationally efficient;
  - Fits into any semi bandit algorithm
- **Linear Generalization:**
  - Makes regret bound L independent

THANK YOU