Efficient Learning in Large-Scale Combinatorial Semi-Bandits

Zheng Wen [1] Branislav Kveton [2] Azin Ashkan [3]

[1]Yahoo Labs, Sunnyvale, CA, USA

[2]Adobe Research, San Jose, CA, USA

[3]Technicolor Research, Los Altos, CA, USA

July 7, 2015
ICML 2015, Lille, France

Motivation

- In many optimization problems, one needs to maximize a modular function subject to combinatorial constraints
  - e.g. minimum spanning tree, shortest path, bipartite matching
  - these problems are well-studied (see e.g. [Papadimitriou & Steiglitz (1998)])
- In practice, the optimized modular function is often unknown
  - it needs to be learned while repeatedly solving the problem
- Such problems can be modeled as combinatorial semi-bandits
  - depending on the feedback model (see [Audibert et al. (2014)])
- A lot of literature is dedicated to it
  - e.g. [Gai et al. (2012); Chen et al. (2013); Russo & Van Roy (2014); Kveton et al. (2015); Cesa-Bianchi & Lugosi (2012); Neu & Bartok (2013)]
  - regret bounds have been established for various algorithms

Motivation: Challenges in Large-Scale Problems

- **Issue:** to achieve an $O(\sqrt{n})$ dependence on time n, all the regret bounds are $\Omega(\sqrt{L})$, where L is the number of items
  - L is intractably large in many real-world problems
  - the dependence on L is intrinsic
- **Solution:** exploit available generalization models for the (expected) weights of items
  - e.g. linear regression, logistic regression, neural network
  - such models are available in many practical problems
- **This talk:** combinatorial semi-bandits with linear regression generalization models
  - closely related to existing literature on linear bandits: (see e.g. [Auer (2002); Dani et al. (2008); Abbasi-Yadkori et al. (2011); Agrawal & Goyal (2012)])

Stochastic Combinatorial Semi-Bandits

- A combinatorial semi-bandit is a triple $(E, \mathcal{A}, P)$
  - $E = \{1, 2, \ldots, L\}$ is a set of L items (arms)
  - $\mathcal{A} \subseteq \{A \subseteq E : |A| \leq K\}$ is a family of subsets of E with up to K items
  - P is probability measure over the item weights $w \in \Re^L$
- The agent knows E and $\mathcal{A}$, but does not know P
- At each time $t = 0, 1, \ldots$
  1. the weight w_t is i.i.d. sampled from P
  2. the agent chooses a subset $A_t \in \mathcal{A}$
  3. the agent observes $\{w_t(e) : e \in A_t\}$ (semi-bandit feedback), and receives a reward $\sum_{e \in A_t} w_t(e)$
- Note that A_t affects both observation and reward
  - exploration-exploitation tradeoff

Linear Generalization Model

- Let $\bar{w} = \mathbb{E}_P[w_t] \in \Re^L$
- Linear generalization model across items:
  - the agent knows a generalization matrix $\Phi \in \Re^{L \times d}$ s.t. $\bar{w}$ is "close" to $\mathrm{span}[\Phi]$
  - $d \leq L$ is the dimension of feature vector
  - WLOG we assume $\mathrm{rank}[\Phi] = d$
- **Caveat:** we need $\bar{w} \in \mathrm{span}[\Phi]$ for the analysis, however, this assumption is not required for applying the proposed algorithms in practice

Arm Set Selection

- At each time t, choosing $A_t \in \mathcal{A}$ can be challenging
  - since the combinatorial optimization problem $\max_{A \in \mathcal{A}} \sum_{e \in A} w(e)$ can be NP-hard
- But there might exist computationally efficient approximation/randomized algorithms
- **This talk:** we assume the agent uses a combinatorial optimization algorithm `ORACLE` to choose A_t
  - `ORACLE` can be an approximation/randomized algorithm
  - $\texttt{ORACLE}(E, \mathcal{A}, w)$ is the solution of `ORACLE` to the optimization problem $\max_{A \in \mathcal{A}} \sum_{e \in A} w(e)$

Performance Metrics

- Let $A^* \leftarrow \texttt{ORACLE}(E, \mathcal{A}, \bar{w})$
  - Recall that $\bar{w} = \mathbb{E}_P[w_t] \in \Re^L$
- Realized regret at time t:

$$R_t = \sum_{e \in A^*} w_t(e) - \sum_{e \in A_t} w_t(e)$$

- **Metric 1:** expected cumulative (pseudo) regret

$$R(n) = \sum_{t=1}^{n} \mathbb{E}[R_t | \bar{w}]$$

- **Metric 2:** Bayes cumulative regret

$$R_{\text{Bayes}}(n) = \mathbb{E}_{\bar{w}}[R(n)]$$

Combinatorial Linear Thompson Sampling

- **Input:** combinatorial structure $(E, \mathcal{A})$, generalization matrix Φ, algorithm parameter $\lambda, \sigma > 0$, oracle `ORACLE`
- **Initialization:** $\Sigma_1 \leftarrow \lambda^2 I$, $\bar{\theta}_1 \leftarrow 0$
- At each time t
  1. sample $\theta_t \sim N\left(\bar{\theta}_t, \Sigma_t\right)$
  2. compute $A_t \leftarrow \texttt{ORACLE}(E, \mathcal{A}, \Phi\theta_t)$
  3. choose A_t, and observe $w_t(e)$, $\forall e \in A_t$
  4. compute $\bar{\theta}_{t+1}$ and Σ_{t+1} based on Kalman filtering

Bayes Regret Bound

Theorem 1: (Bayes Regret Bound on `CombLinTS`)

If (1) $\bar{w} \in \mathrm{span}[\Phi]$, (2) prior and noises are independently drawn from $N(0, \lambda^2 I)$ and $N(0, \sigma^2)$, and (3) $\lambda \geq \sigma$, then under `CombLinTS` with parameter (Φ, λ, σ), we have

$$R_{\mathrm{Bayes}}(n) \leq \tilde{O}\left(K\lambda\sqrt{dn\min\{\log(L), d\}}\right).$$

- (3) is a nonessential technical condition and can be removed

Combinatorial Linear UCB

- **Input:** combinatorial structure $(E, \mathcal{A})$, generalization matrix Φ, algorithm parameter $\lambda, \sigma, c > 0$, oracle `ORACLE`
- **Initialization:** $\Sigma_1 \leftarrow \lambda^2 I$, $\bar{\theta}_1 \leftarrow 0$
- At each time t
  1. compute UCB weight $\hat{w}_t$ as

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

  2. compute $A_t \leftarrow \texttt{ORACLE}(E, \mathcal{A}, \hat{w}_t)$
  3. choose A_t, and observe $w_t(e)$, $\forall e \in A_t$
  4. compute $\bar{\theta}_{t+1}$ and Σ_{t+1} based on Kalman filtering

Regret Bound

Theorem 2: (Regret Bound on `CombLinUCB`)

If (1) $\bar{w} \in \text{span}[\Phi]$, (2) $\text{supp}(P) \subseteq [0,1]^L$, and (3) `ORACLE` exactly solves the offline optimization problem, then under `CombLinUCB` with $\lambda = \sigma = 1$ and appropriate c, we have

$$R(n) \leq \tilde{O}\left(K\lambda d\sqrt{n}\right).$$

- Bound in Theorem 2 is at most $\tilde{O}\left(\sqrt{Kd}\right)$ from tight

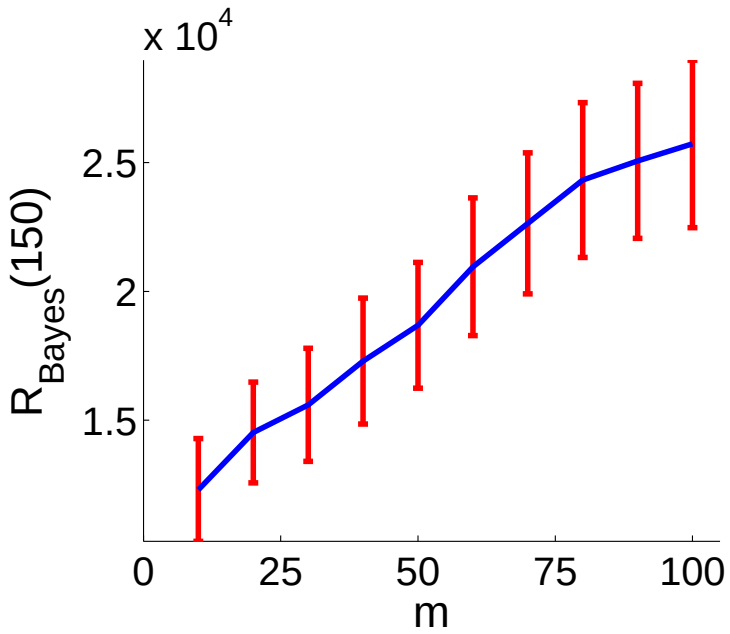
Experiment Results: Overview

- We only evaluate `CombLinTS`
  - in practice TS algorithms usually outperform UCB-like algorithms
- One synthetic example
  - demonstrate the scalability and robustness of `CombLinTS`
- Two examples based on real-world data sets
  - show the value of linear generalization in real-world settings

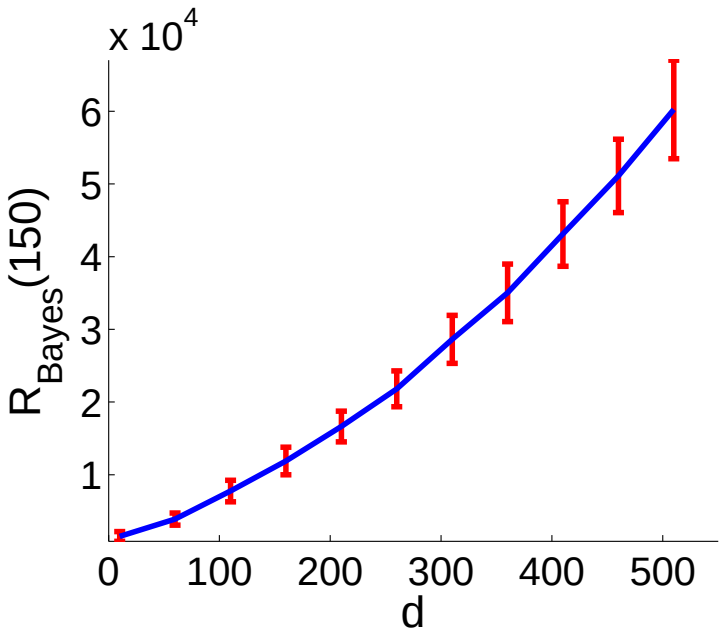
Longest Path

- Learn to find the longest path in an $(m+1) \times (m+1)$ grid
  - E: edges in the grid
  - $\mathcal{A}$: all paths from the upper-left corner to bottom-right corner
- In each round of experiment
  - $\Phi \in \Re^{L \times d}$ is randomly sampled
  - sample $\theta^* \sim N(0, \lambda_{\text{true}}^2 I)$ and set $\bar{w} = \Phi\theta^*$
  - noises are i.i.d. sampled from $N(0, \sigma_{\text{true}}^2)$
  - run `CombLinTS` with parameter (λ, σ)
- Average the results of 200 rounds
- Default case: $\lambda_{\text{true}} = \lambda = 10$, $\sigma_{\text{true}} = \sigma = 1$, $m = 30$, $d = 200$
  - $L = 1860$ and $|\mathcal{A}| = 1.18 \times 10^{17}$
  - vary one and only one parameter in experiments

Scalability of CombLinTS



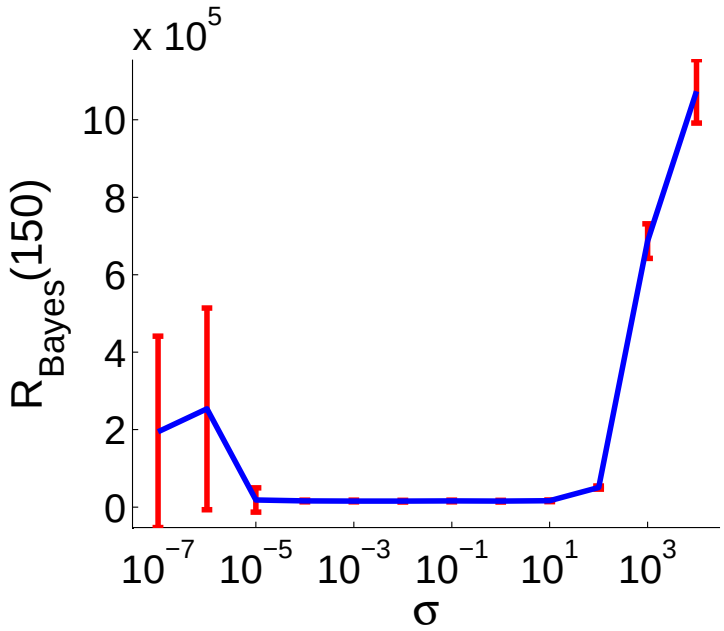
(a) R_{Bayes} vs. m



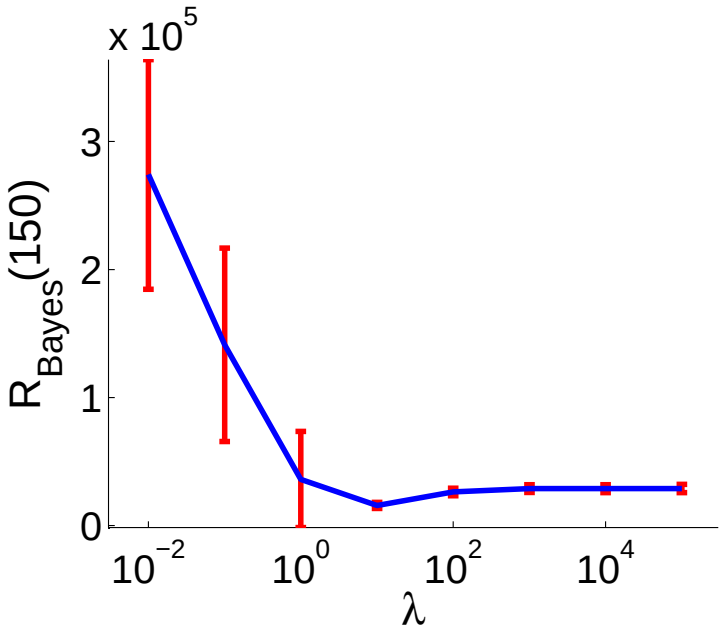
(b) R_{Bayes} vs. d

- R_{Bayes} roughly increases linearly with m or d

Robustness of CombLinTS



(c) R_{Bayes} vs. σ



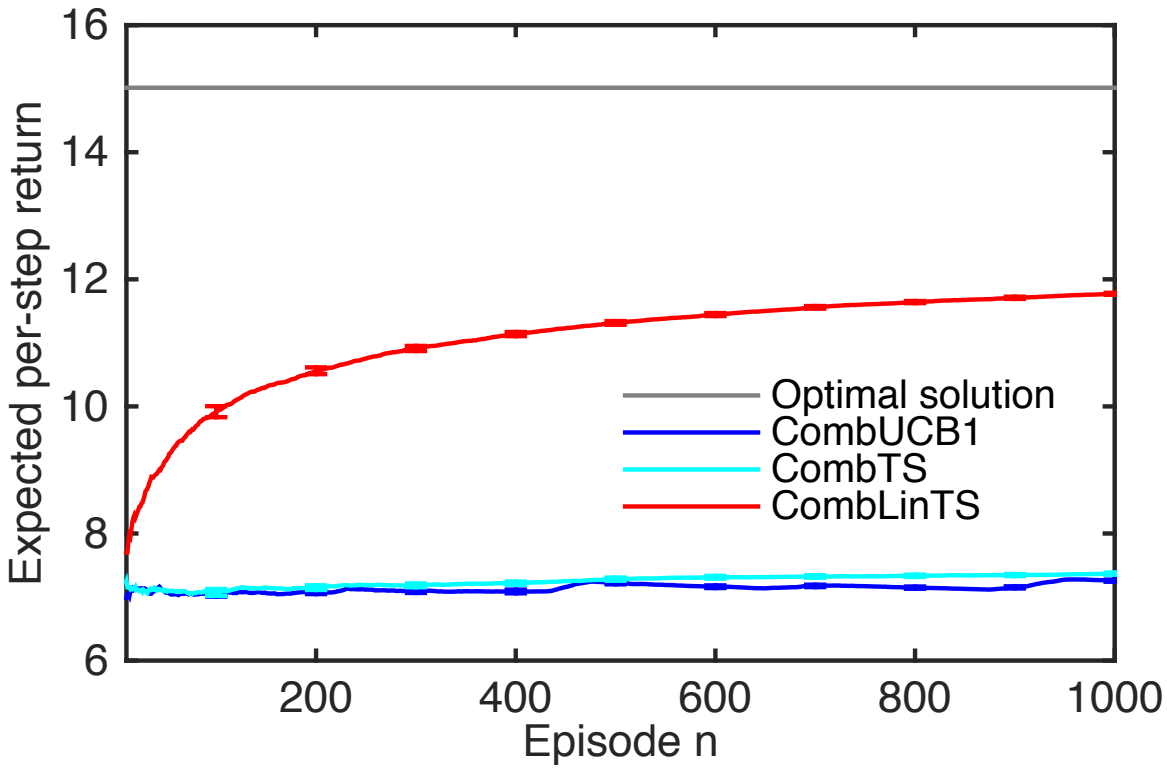
(d) R_{Bayes} vs. λ

- CombLinTS performs well for a wide range of σ and λ

Online Advertising

- **Goal:** learn to identify 100 people that are most likely to accept an ad offer, subject to the targeting constraint that exactly 50 of them are females
- E: 33k representative users from Adult dataset [Asuncion & Newman (2007)]
- $\bar{w}(e) = 0.15 \times \mathbf{1}\{\text{income} \geq 50k\} + 0.05 \times \mathbf{1}\{\text{income} < 50k\}$
- Φ encodes age, gender, per-week working hours, length of education
- Compare `CombLinTS` with `CombUCB1`, `CombTS`, and the optimal offline solution

Online Advertising

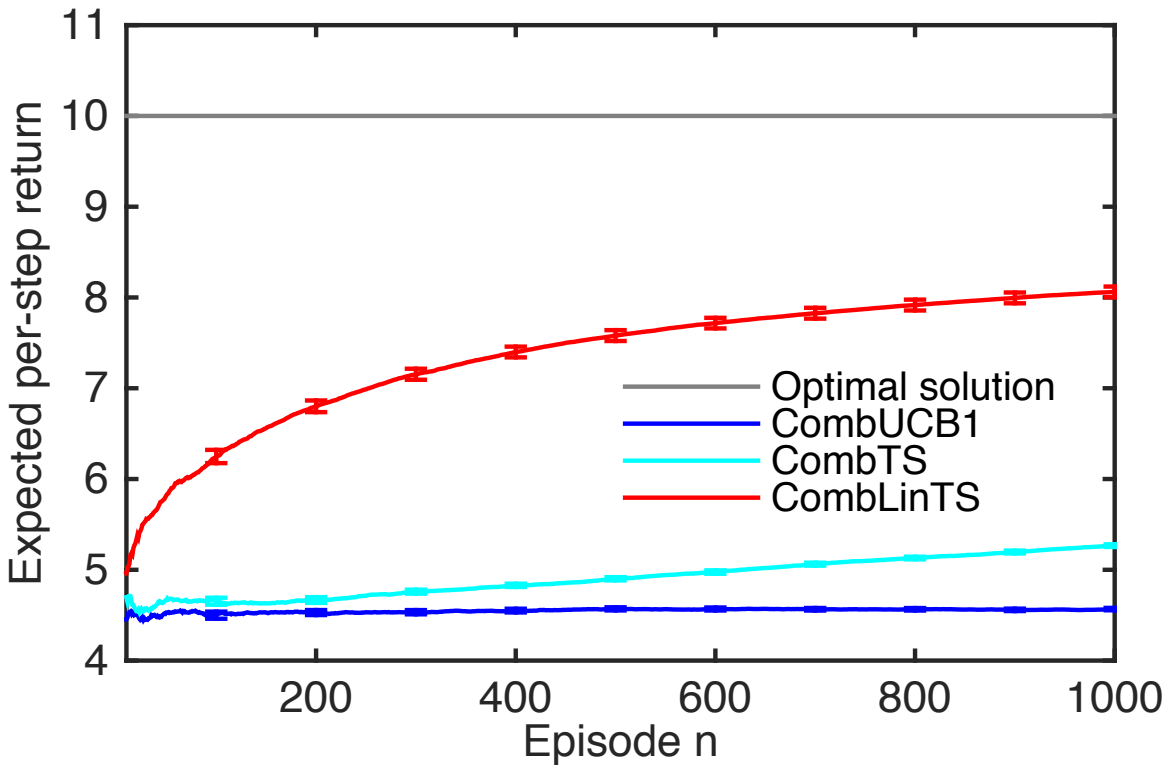


- `CombLinTS` learns very fast
- `CombUCB1` and `CombTS` perform poorly

Artist Recommendation

- **Goal:** learn to recommend 10 music artists to an average user
- E: about 6k artists from Last.fm dataset [Cantador et al. (2011)] that was
  1. listened by at least two users
  2. assigned at least one tag from the 20 most popular tags
- $\bar{w}(e)$ is estimated by Naïve Bayes classifiers
- $\phi_e \in [0,1]^{20}$
  - $\phi_{e,j}$ is the fraction of users who assigned tag j to artist e
- Compare `CombLinTS` with `CombUCB1`, `CombTS`, and the optimal offline solution

Artist Recommendation



- `CombLinTS` learns very fast
- `CombUCB1` and `CombTS` perform poorly

Conclusion

- **Contributions:**
  1. proposed `CombLinTS` and `CombLinUCB`
  2. derived L-independent regret bound
  3. evaluated `CombLinTS` on a variety of problems
- **Future work:**
  1. analyses for the case when $\bar{w} \notin \text{span}[\Phi]$
  2. algorithms & analyses for combinatorial semi-bandits with nonlinear generalization
     - e.g. logistic regression, neural network