Graphs in Machine Learning

Michal Valko

*Inria Lille - Nord Europe, France*

Partially based on material by: Tomáš Kocák
Last Lecture

- Examples of applications of online SSL
- Analysis of online SSL
- SSL Learnability
- When does graph-based SSL provably help?
- Scaling harmonic functions to millions of samples
Previous Lab Session

- 16. 11. 2015 by Daniele Calandriello
- Content
  - Semi-supervised learning
  - Graph quantization
  - Online face recognizer
- Short written report
- Questions to piazza
- **Deadline: 30. 11. 2015**
- [http://researchers.lille.inria.fr/~calandri/teaching.html](http://researchers.lille.inria.fr/~calandri/teaching.html)
This Lecture

- Online decision-making on graphs
- Graph bandits
- Smoothness of rewards (preferences) on a given graph
- Observability graphs
- Exploiting side information
Final Class projects

- detailed description on the class website
- preferred option: you come up with the topic
- theory/implementation/review or a combination
- one or two people per project (exceptionally three)
- grade 60%: report + short presentation of the team
- deadlines
  - 23. 11. 2015 - strongly recommended DL for taking projects
  - 30. 11. 2015 - hard DL for taking projects
  - 06. 01. 2015 - submission of the project report
  - 11. 01. 2016 (or later) - project presentation
- list of suggested topics on piazza
Online Decision Making on Graphs
Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some structural property of the setting
  - goal: maximize the sum of the outcomes
  - application: recommender systems

- First application: Exploiting smoothness
  - fixed graph
  - iid outcomes
  - neighboring nodes have similar outcomes
Online Decision Making on Graphs

Movie recommendation: (in each time step)
- Recommend movies to a single user.
- Good prediction after a few steps ($T \ll N$).

Goal:
- Maximize overall reward (sum of ratings).

Assumptions:
- Unknown reward function $f : V(G) \rightarrow \mathbb{R}$.
- Function $f$ is smooth on a graph.
- Neighboring movies $\Rightarrow$ similar preferences.
- Similar preferences $\not\Rightarrow$ neighboring movies.
Let’s be lazy: Ignore the structure!

This is an multi-armed bandit problem!

The performance depends on the number of movies $(N$ arms$)$.

Worst case regret (to the best fixed strategy) $R_T = O\left(\sqrt{NT}\right)$

What is $N$ for imdb.com? 3,538,545 http://www.imdb.com/stats
Let’s be lazy: Ignore the structure!

Another problem of the typical bandits strategies for recommendation?

If there is no information shared, we need to try all of the options!

UCB/MOSS and likely TS start with pulling each of the arms once

This is a problem both algorithmically and theoretically . . . .

Watch all the movies and then I tell you which one you like . . . .

What do we need for movie recommendation?

An algorithm useful in the case $T \ll N$!

Exploiting the structure is a must!
Recap: Smooth graph functions

- \( f = (f_1, \ldots, f_N)^T \): Vector of function values.
- Let \( L = Q\Lambda Q^T \) be the eigendecomposition of the Laplacian.
  - Diagonal matrix \( \Lambda \) whose diagonal entries are eigenvalues of \( L \).
  - Columns of \( Q \) are eigenvectors of \( L \).
  - Columns of \( Q \) form a basis.
- \( \alpha^* \): Unique vector such that \( Q\alpha^* = f \)  
  \[ \text{Note: } Q^Tf = \alpha^* \]

\[
S_G(f) = f^T L f = f^T Q \Lambda Q^T f = \alpha^{*T} \Lambda \alpha^* = \|\alpha^*\|^2_{\Lambda} = \sum_{i=1}^{N} \lambda_i (\alpha^*_i)^2
\]

**Smoothness and regularization:** Small value of

(a) \( S_G(f) \)  
(b) \( \Lambda \) norm of \( \alpha^* \)  
(c) \( \alpha^*_i \) for large \( \lambda_i \)
Smooth graph functions: Flixster eigenvectors

Eigenvectors from the Flixster data corresponding to the smallest few eigenvalues of the graph Laplacian projected onto the first principal component of data. Colors indicate the values.
Online Learning Setting - Bandit Problem

Learning setting for a bandit algorithm $\pi$

- In each time $t$ step choose a node $\pi(t)$.
- the $\pi(t)$-th row $x_{\pi(t)}$ of the matrix $Q$ corresponds to the arm $\pi(t)$.
- Obtain noisy reward $r_t = x^T_{\pi(t)} \alpha^* + \varepsilon_t$.
  - Note: $x^T_{\pi(t)} \alpha^* = f_{\pi(t)}$
  - $\varepsilon_t$ is $R$-sub-Gaussian noise. $\forall \xi \in \mathbb{R}, \mathbb{E}[e^{\xi \varepsilon_t}] \leq \exp (\xi^2 R^2 / 2)$

- Minimize cumulative regret

$$R_T = T \max_a (x_a^T \alpha^*) - \sum_{t=1}^{T} x_{\pi(t)}^T \alpha^*.$$
Online Decision Making on Graphs: Smoothness

- Linear bandit algorithms
  - **LinUCB**
    - Regret bound $\approx D\sqrt{T \ln T}$
  - **LinearTS**
    - Regret bound $\approx D\sqrt{T \ln N}$

  **Note:** $D$ is ambient dimension, in our case $N$, length of $x_i$. Number of actions, e.g., all possible movies $\rightarrow$ **HUGE**!

- Spectral bandit algorithms
  - **SpectralUCB**
    - Regret bound $\approx d\sqrt{T \ln T}$
    - Operations per step: $D^2N$
  - **SpectralTS**
    - Regret bound $\approx d\sqrt{T \ln N}$
    - Operations per step: $D^2 + DN$

  **Note:** $d$ is effective dimension, usually much smaller than $D$. 
Effective dimension

- **Effective dimension**: Largest $d$ such that

\[(d - 1)\lambda_d \leq \frac{T}{\log(1 + T/\lambda)}.

- Function of time horizon and graph properties
- $\lambda_i$: $i$-th smallest eigenvalue of $\Lambda$.
- $\lambda$: Regularization parameter of the algorithm.

**Properties:**
- $d$ is small when the coefficients $\lambda_i$ grow rapidly above time.
- $d$ is related to the number of “non-negligible” dimensions.
- Usually $d$ is much smaller than $D$ in real world graphs.
- Can be computed beforehand.
Effective dimension vs. Ambient dimension

Barabasi–Albert graph $N=500$

Flixster graph: $N=4546$

\[ d \ll D \]

Note: In our setting $T < N = D$. 

UCB-style algorithms: Estimate

![Graph showing expected reward estimates for UCB-style algorithms. The graph plots expected reward against different trials, with error bars indicating the variability in the estimates.]
UCB-style algorithms: Sample
UCB-style algorithms: Estimate...
Given a vector of weights $\alpha$, we define its $\Lambda$ norm as

$$\|\alpha\|_{\Lambda} = \sqrt{\sum_{k=1}^{N} \lambda_k \alpha_k^2} = \sqrt{\alpha^T \Lambda \alpha},$$

and fit the ratings $r_v$ with a (regularized) least-squares estimate

$$\hat{\alpha}_t = \arg\min_{\alpha} \left( \sum_{v=1}^{t} \left[ \langle x_v, \alpha \rangle - r_v \right]^2 + \|\alpha\|_{\Lambda}^2 \right).$$

$\|\alpha\|_{\Lambda}$ is a penalty for non-smooth combinations of eigenvectors.
**SpectralUCB**

1: **Input:**
2: \( N, T, \{\Lambda_L, Q\}, \lambda, \delta, R, C \)
3: **Run:**
4: \( \Lambda \leftarrow \Lambda_L + \lambda I \)
5: \( d \leftarrow \max\{d : (d - 1)\lambda_d \leq T / \ln(1 + T / \lambda)\} \)
6: **for** \( t = 1 \) **to** \( T \) **do**
7: Update the basis coefficients \( \hat{\alpha} \):
8: \( X_t \leftarrow [x_{\pi(1)}, \ldots, x_{\pi(t-1)}]^T \)
9: \( r \leftarrow [r_1, \ldots, r_{t-1}]^T \)
10: \( V_t \leftarrow X_tX_t^T + \Lambda \)
11: \( \hat{\alpha}_t \leftarrow V_t^{-1}X_tr \)
12: \( c_t \leftarrow 2R\sqrt{d \ln(1 + t / \lambda) + 2 \ln(1 / \delta)} + C \)
13: \( \pi(t) \leftarrow \arg \max_a \left( x_a^T \hat{\alpha} + c_t \|x_a\|_{V_t^{-1}} \right) \)
14: Observe the reward \( r_t \)
15: **end for**
SpectralUCB: Synthetic experiment

Barabasi–Albert $N=250$, basis size=3, effective $d=1$

- SpectralEliminator
- SpectralUCB
- LinUCB

Cumulative regret vs. time $T$
SpectralUCB: Movie data experiments

Movielens: Cumulative regret for randomly sampled users. $T = 100$

Flixster: Cumulative regret for randomly sampled users. $T = 100$
SpectralUCB: Regret Bound

- $d$: Effective dimension.
- $\lambda$: Minimal eigenvalue of $\Lambda = \Lambda_L + \lambda I$.
- $C$: Smoothness upper bound, $\|\alpha^*\|_\Lambda \leq C$.
- $x_i^T \alpha^* \in [-1, 1]$ for all $i$.

The cumulative regret $R_T$ of SpectralUCB is with probability $1 - \delta$ bounded as

\[
R_T \leq \left( 8R \sqrt{d \ln \frac{\lambda + T}{\lambda}} + 2\ln \frac{1}{\delta} + 4C + 4 \right) \sqrt{dT \ln \frac{\lambda + T}{\lambda}}.
\]

\[
R_T \approx d \sqrt{T \ln T}
\]
SpectralUCB: Regret Bound

- Derivation of the confidence ellipsoid for $\hat{\alpha}$ with probability $1 - \delta$.
  - Using analysis of OFUL (Abbasi-Yadkori et al., 2011)

$$\left| x^T(\hat{\alpha} - \alpha^*) \right| \leq \|x\|_{V_t^{-1}} \left( R \sqrt{2 \ln \left( \frac{|V_t|^{1/2}}{\delta |\Lambda|^{1/2}} \right)} + C \right)$$

- Regret in one time step: $r_t = x^*_T \alpha^* - x^T_{\pi(t)} \alpha^* \leq 2c_t \left\| x_{\pi(t)} \right\|_{V_t^{-1}}$
- Cumulative regret:

$$R_T = \sum_{t=1}^{T} r_t \leq \sqrt{T \sum_{t=1}^{T} r_t^2} \leq 2(CT + 1) \sqrt{2T \ln \frac{|V_T|}{|\Lambda|}}$$

- Upperbound for $\ln\left( \frac{|V_t|}{|\Lambda|} \right)$

$$\ln \frac{|V_t|}{|\Lambda|} \leq \ln \frac{|V_T|}{|\Lambda|} \leq 2d \ln \left( \frac{\lambda + T}{\lambda} \right)$$
SpectralUCB: Regret Bound

Sylvester’s determinant theorem:

\[ |A + xx^T| = |A||I + A^{-1}xx^T| = |A|(1 + x^TA^{-1}x) \]

Goal:

- Upperbound determinant \( |A + xx^T| \) for \( \|x\|_2 \leq 1 \)
- Upperbound \( x^TA^{-1}x \)

\[ x^TA^{-1}x = x^TQ\Lambda^{-1}Q^Tx = y^T\Lambda^{-1}y = \sum_{i=1}^{N} \lambda_i^{-1}y_i^2 \]

- \( \|y\|_2 \leq 1 \).
- \( y \) is a canonical vector.
- \( x = Qy \) is an eigenvector of \( A \).
**SpectralUCB: Regret Bound**

**Corollary**: Determinant $|V_T|$ of $V_T = \Lambda + \sum_{t=1}^{T} x_t x_t^\top$ is maximized when all $x_t$ are aligned with axes.

$$|V_T| \leq \max_{t_i=T} \prod (\lambda_i + t_i)$$

$$\ln \frac{|V_T|}{|\Lambda|} \leq \max_{t_i=T} \sum \ln \left(1 + \frac{t_i}{\lambda_i}\right)$$

$$\ln \frac{|V_T|}{|\Lambda|} \leq \sum_{i=1}^{d} \ln \left(1 + \frac{T}{\lambda}\right) + \sum_{i=d+1}^{N} \ln \left(1 + \frac{t_i}{\lambda_{d+1}}\right)$$

$$\leq d \ln \left(1 + \frac{T}{\lambda}\right) + \frac{T}{\lambda_{d+1}}$$

$$\leq 2d \ln \left(1 + \frac{T}{\lambda}\right)$$
SpectralUCB: Improving the running time

- **Reduced basis:** We only need first few eigenvectors.
- **Getting \( J \) eigenvectors:** \( \mathcal{O}(Jm \log m) \) time for \( m \) edges
- Computationally less expensive, comparable performance.

![Graph showing cumulative regret vs. time and computational time for different \( J \) values](image-url)
SpectralUCB: How to make it even faster?

- UCB-style algorithms need to (re)-compute UCBs every $t$
- Can be a problem for large set of arms $\rightarrow D^2N \rightarrow N^3$
- Optimistic (UCB) approach vs. Thompson Sampling
  - Play the arm maximizing probability of being the best
    - Sample $\tilde{\alpha}$ from the distribution $\mathcal{N}(\hat{\alpha}, v^2V^{-1})$
    - Play arm which maximizes $x^T\tilde{\alpha}$ and observe reward
  - Compute posterior distribution according to reward received
- Only requires $D^2 + DN \rightarrow N^2$ per step update
Thomson Sampling: Estimate
Thomson Sampling: Sample
Thomson Sampling: Estimate
Thomson Sampling: Sample
Thomson Sampling: Estimate
SpectralTS for Graphs

1: **Input:**
2: \( N, T, \{ \Lambda_L, Q \}, \lambda, \delta, R, C \)
3: **Initialization:**
4: \( \nu = R \sqrt{6d \log((\lambda + T)/\delta \lambda)} + C \)
5: \( \hat{\alpha} = 0_N \)
6: \( f = 0_N \)
7: \( V = \Lambda_L + \lambda I_N \)
8: **Run:**
9: for \( t = 1 \) to \( T \) do
10: Sample \( \tilde{\alpha} \sim \mathcal{N}(\hat{\alpha}, \nu^2 V^{-1}) \)
11: \( \pi(t) \leftarrow \arg\max_a x_a^T \tilde{\alpha} \)
12: Observe a noisy reward \( r(t) = x_{\pi(t)}^T \alpha^* + \varepsilon_t \)
13: \( f \leftarrow f + x_{\pi(t)} r(t) \)
14: Update \( V \leftarrow V + x_{\pi(t)} x_{\pi(t)}^T \)
15: Update \( \hat{\alpha} \leftarrow V^{-1} f \)
16: end for
SpectralTS: Regret bound

- $d$: Effective dimension.
- $\lambda$: Minimal eigenvalue of $\Lambda = \Lambda_L + \lambda I$.
- $C$: Smoothness upper bound, $\|\alpha^*\|_\Lambda \leq C$.
- $x_i^T \alpha^* \in [-1, 1]$ for all $i$.

The cumulative regret $R_T$ of SpectralTS is with probability $1 - \delta$ bounded as

$$R_T \leq \frac{11g}{p} \sqrt{\frac{4 + 4\lambda}{\lambda} dT \log \frac{\lambda + T}{\lambda}} + \frac{1}{T} + \frac{g}{p} \left( \frac{11}{\sqrt{\lambda}} + 2 \right) \sqrt{2T \log \frac{2}{\delta}},$$

where $p = 1/(4e\sqrt{\pi})$ and

$$g = \sqrt{4 \log TN} \left( R \sqrt{6d \log \left( \frac{\lambda + T}{\delta \lambda} \right)} + C \right) + R \sqrt{2d \log \left( \frac{(\lambda + T)T^2}{\delta \lambda} \right)} + C.$$

$$R_T \approx d \sqrt{T \log N}$$
SpectralTS: Analysis sketch

Divide arms into two groups

- $\Delta_i = x_*^T \alpha - x_i^T \alpha \leq g \|x_i\|_{V_t^{-1}}$ arm $i$ is unsaturated
- $\Delta_i = x_*^T \alpha - x_i^T \alpha > g \|x_i\|_{V_t^{-1}}$ arm $i$ is saturated

Saturated arm

- Small standard deviation $\rightarrow$ accurate regret estimate.
- **High regret** on playing the arm $\rightarrow$ **Low probability** of picking

Unsaturated arm

- **Low regret** bounded by a factor of standard deviation
- **High probability** of picking
SpectralTS: Analysis sketch

- Confidence ellipsoid for estimate $\hat{\mu}$ of $\mu$ (with probability $1 - \delta/T^2$)
  - Using analysis of OFUL algorithm (Abbasi-Yadkori et al., 2011)

$$|x_i^T \hat{\alpha} - x_i^T \alpha| \leq \left( R \sqrt{2d \log \left( \frac{(\lambda + T)T^2}{\delta \lambda} \right)} + C \right) \|x_i\|_{V_t^{-1}} = \ell \|x_i\|_{V_t^{-1}}$$

- The key result coming from spectral properties of $V_t$.

$$\log \frac{|V_t|}{|\Lambda|} \leq 2d \log \left( 1 + \frac{T}{\lambda} \right)$$

- Concentration of sample $\tilde{\alpha}$ around mean $\hat{\alpha}$ (with probability $1 - 1/T^2$)
  - Using concentration inequality for Gaussian random variable.

$$|x_i^T \tilde{\alpha} - x_i^T \hat{\alpha}| \leq \left( R \sqrt{6d \log \left( \frac{\lambda + T}{\delta \lambda} \right)} + C \right) \|x_i\|_{V_t^{-1}} \sqrt{4 \log(TN)} = \nu \|x_i\|_{V_t^{-1}} \sqrt{4 \log(TN)}$$
**SpectralTS: Analysis sketch**

Define \( \text{regret}'(t) = \text{regret}(t) \cdot 1 \{ |\mathbf{x}_i^\top \hat{\alpha}(t) - \mathbf{x}_i^\top \alpha| \leq \ell \| \mathbf{x}_i \| \nu^{-1} \} \)

\[
\text{regret}'(t) \leq \frac{11g}{p} \| \mathbf{x}_{a(t)} \| \nu^{-1} + \frac{1}{T^2}
\]

**Super-martingale** (i.e. \( \mathbb{E}[Y_t - Y_{t-1} | \mathcal{F}_{t-1}] \leq 0 \))

\[
X_t = \text{regret}'(t) - \frac{11g}{p} \| \mathbf{x}_{a(t)} \| \nu^{-1} - \frac{1}{T^2}
\]

\[
Y_t = \sum_{w=1}^{t} X_w.
\]

\((Y_t; t = 0, \ldots, T)\) is a **super-martingale** process w.r.t. history \( \mathcal{F}_t \).

**Azuma-Hoeffding inequality for super-martingales**, w.p. \( 1 - \delta/2 \):

\[
\sum_{t=1}^{T} \text{regret}'(t) \leq \frac{11g}{p} \sum_{t=1}^{T} \| \mathbf{x}_{a(t)} \| \nu^{-1} + \frac{1}{T} + \frac{g}{p} \left( \frac{11}{\sqrt{\lambda}} + 2 \right) \sqrt{2 T \ln \frac{2}{\delta}}
\]
Spectral Bandits: Synthetic experiment

Barabasi–Albert $N=250$, basis size=3, effective $d=1$

![Graph showing cumulative regret over time for different algorithms: SpectralTS, LinearTS, SpectralUCB, LinUCB. The x-axis represents time $T$, and the y-axis represents cumulative regret. The graph compares the performance of these algorithms.]
Spectral Bandits: Real world experiment

MovieLens dataset of 6k users who rated one million movies.

Movielens data $N=2019$, average of 10 users, $T=200$, $d = 5$
Spectral Bandits Summary

- **Spectral bandit setting** *(smooth graph functions).*

- **SpectralUCB**
  - Regret bound $R_T = \tilde{O}\left(d \sqrt{T \ln T}\right)$

- **SpectralTS**
  - Regret bound $R_T = \tilde{O}\left(d \sqrt{T \ln N}\right)$
  - Computationally more efficient.

- **SpectralEliminator**
  - Regret bound $R_T = \tilde{O}\left(\sqrt{dT \ln T}\right)$
  - Better upper, empirically does not seem to work well (yet)

- Bounds scale with **effective dimension** $d \ll D$. 
SpectralEliminator: Pseudocode

Input:
\[ N : \text{the number of nodes}, \ T : \text{the number of pulls} \]
\[ \{\Lambda_L, Q\} : \text{spectral basis of } L \]
\[ \lambda : \text{regularization parameter} \]
\[ \beta, \{t_j\} : \text{parameters of the elimination and phases} \]
\[ A_1 \leftarrow \{x_1, \ldots, x_K\} \]

for \( j = 1 \) to \( J \) do

\[ V_{t_j} \leftarrow \gamma\Lambda_L + \lambda I \]

for \( t = t_j \) to \( \min(t_{j+1} - 1, T) \) do

Play \( x_t \in A_j \) with the largest width to observe \( r_t \):

\[ x_t \leftarrow \arg\max_{x \in A_j} ||x||_{V_t^{-1}} \]

\[ V_{t+1} \leftarrow V_t + x_t x_t^T \]

end for

Eliminate the arms that are not promising:

\[ \hat{\alpha}_t \leftarrow V_t^{-1} [x_{t_j}, \ldots, x_t] [r_{t_j}, \ldots, r_t]^T \]

\[ A_{j+1} \leftarrow \left\{ x \in A_j, \langle \hat{\alpha}_t, x \rangle + ||x||_{V_t^{-1}} \beta \geq \max_{x \in A_j} \left[ \langle \hat{\alpha}_t, x \rangle - ||x||_{V_t^{-1}} \beta \right] \right\} \]

end for
SpectralEliminator: Analysis

SpectralEliminator

- Divide time into sets \( t_1 = 1 \leq t_2 \leq \ldots \) to introduce independence for Azuma-Hoeffding inequality and observe
  \[
  R_T \leq \sum_{j=0}^{J} (t_{j+1} - t_j) \left[ \langle x^* - x_t, \hat{\alpha}_j \rangle + (\|x^*\|_{V_j^{-1}} + \|x_t\|_{V_j^{-1}})\beta \right]
  \]
- Bound \( \langle x^* - x_t, \hat{\alpha}_j \rangle \) for each phase
- No bad arms: \( \langle x^* - x_t, \hat{\alpha}_j \rangle \leq (\|x^*\|_{V_j^{-1}} + \|x_t\|_{V_j^{-1}})\beta \)
- By algorithm: \( \|x\|^2_{V_j^{-1}} \leq \frac{1}{t_j - t_{j-1}} \sum_{s=t_{j-1}+1}^{t_j} \|x_s\|^2_{V_{s-1}} \)
- \( \sum_{s=t_{j-1}+1}^{t_j} \min \left( 1, \|x_s\|^2_{V_{s-1}} \right) \leq \log \frac{|V_j|}{|\Lambda|} \)
Spectral Bandits: Is it possible to do better?

Is $d$ a good quantity that embodies the difficulty?

For any $d$, we construct a graph that for any reasonable algorithm, the regret is at least $\Omega(\sqrt{dT})$.

How? By reduction to $d$-arm bandits problem.

Lower bound!