

Resistive geometry for graph-based transduction EMMDS 2009

Mark Herbster

University College London
Department of Computer Science
Centre for Computational Statistics and Machine Learning

4 July, 2009

with Guy Lever, Massimiliano Pontil, and Sergio Rojas-Galeano

Outline

1. Introduction: transduction with a resistive network
2. Fast prediction on a tree
3. Generalized p -resistive networks

Part I

Transduction with a resistive network

Supervised learning

- ▶ Typical goal
 - Given data (pattern,target) $\mathcal{S} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_\ell, y_\ell)\}$
infer f such that $f(\mathbf{x}_i) \approx y_i$ for future data
$$\mathcal{S}' = \{(\mathbf{x}_{\ell+1}, y_{\ell+1}), (\mathbf{x}_{\ell+2}, y_{\ell+2}), \dots\}.$$
- ▶ Algorithms
 1. Linear Regression
 2. Neural Networks
 3. Decision Trees
 4. Support Vector Machines

Unsupervised learning

▶ Typical goal

- Given data (patterns) $\mathcal{S} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$
Model the data.

- For example (clustering)

Give a partition/function of $f : \mathbf{x} \rightarrow \{1, \dots, k\}$ of \mathcal{S}

Such that $f(\mathbf{x}_i) = f(\mathbf{x}_j) \Rightarrow \mathbf{x}_i \approx \mathbf{x}_j$

$f(\mathbf{x}_i) \neq f(\mathbf{x}_j) \Rightarrow \mathbf{x}_i \not\approx \mathbf{x}_j$

▶ Algorithms

1. Clustering (k-means)
2. Dimensionality Reduction (PCA)

Semi-supervised Learning (Transductive case)

▶ Typical goal

- Given data $\mathcal{S} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_\ell, y_\ell), \mathbf{x}_{\ell+n}, \dots, \mathbf{x}_n\}$
infer f such that $f(\mathbf{x}_i) \approx y_i$ **(No future data)**

▶ Idea

- Combine supervised and unsupervised learning.
- Model data as a graph.

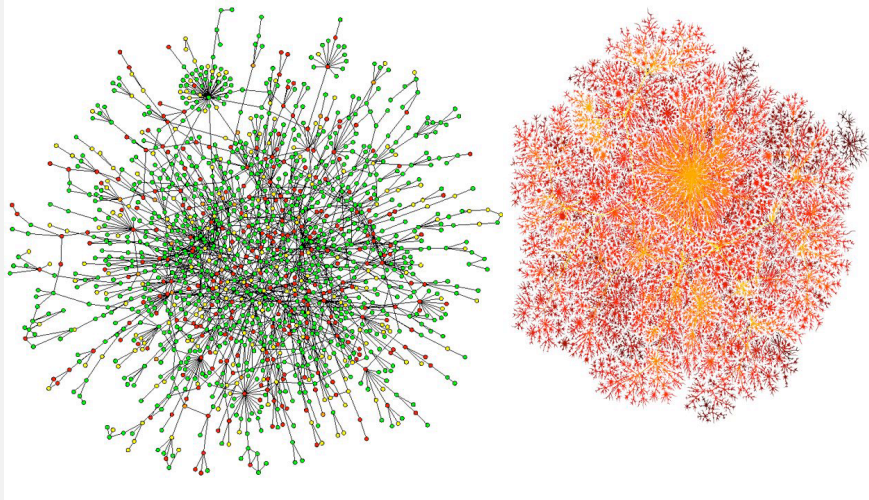
▶ Graph representation of data

- Intrinsic: (protein interaction network, social network)
- Built via a “distance” $d(p, q)$

[sparse unweighted]: k -nearest neighbors graph

[complete weighted]: Weight edge (p, q) with $w_{pq} = \frac{1}{d(p, q)}$

Graph Examples – 1



Yeast protein network

Internet hosts

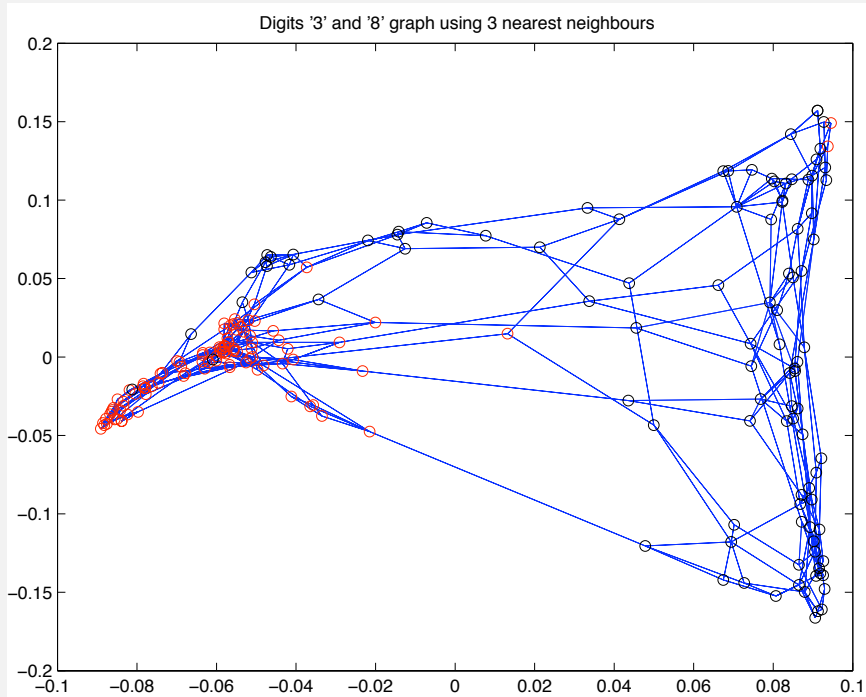
Graph Examples – 2



Web Spam

Twitter Social Network

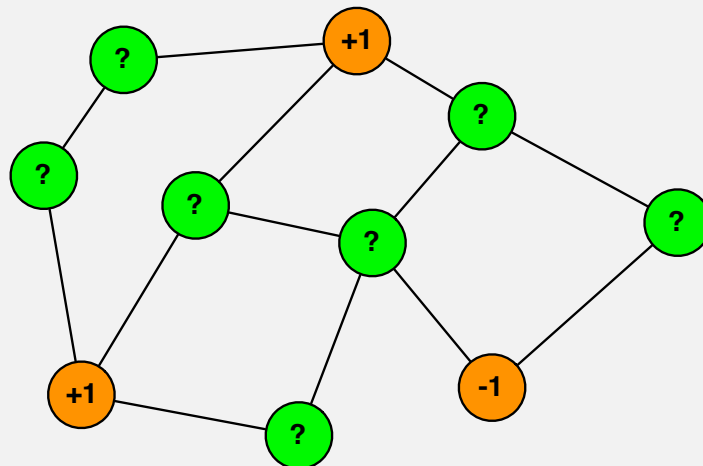
Graph Examples – 3



USPS digits 3 and 8

Some questions

1. How to *label* the unknown vertices?
2. How to efficiently *compute* the labels of the unknown vertices?
3. How to *guarantee* the learning quality of the labelling?
4. How to *build* the graph?



Graph to resistive network

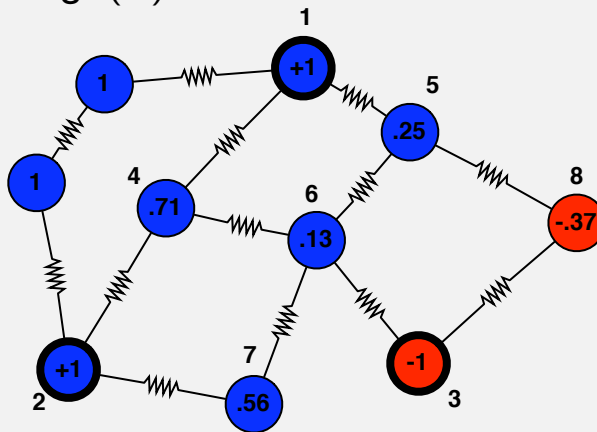
- ▶ Identify graph with a resistive network
- ▶ Labels are voltage constraints.
- ▶ Power of a labeling

$$P(\mathbf{u}) := \|\mathbf{u}\|_{\mathcal{G},2}^2 = \sum_{(i,j) \in E(\mathbf{G})} w_{ij} |u_i - u_j|^2$$

- ▶ Label by minimizing power

$$\bar{\mathbf{u}} = \arg \min_{\mathbf{u} \in \mathbb{R}^n} \{P(\mathbf{u}) : u_{i_1} = y_1, \dots, u_{i_\ell} = y_\ell\}$$

- ▶ Predict with $\text{sign}(\bar{\mathbf{u}})$



Refs: [DS00,ZGL03]

Effective resistance

Graph Laplacian $\mathbf{G} := \mathbf{D} - \mathbf{W}$

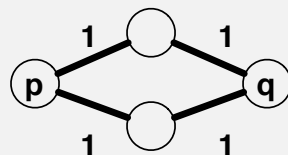
- \mathbf{W} is the weighted adjacency matrix
- $\mathbf{D} := \text{diag}(\sum_{i=1}^n W_{i1}, \dots, \sum_{i=1}^n W_{in})$ (vertex degree matrix)
- \mathbf{G}^+ is the “kernel” (pseudoinverse of \mathbf{G})

Theorem[KR93]

Effective resistance between \mathbf{v}_p and \mathbf{v}_q is

$$r_{\mathbf{G}}(p, q) = \left[\min_{\mathbf{u} \in \mathbb{R}^n} \{P(\mathbf{u}) : u_p = 1, u_q = 0\} \right]^{-1} = \mathbf{G}_{pp}^+ + \mathbf{G}_{qq}^+ - 2\mathbf{G}_{pq}^+$$

Graph is a circuit and edge (i, j) is a resistor $\pi_{ij} := w_{ij}^{-1} = d(i, j)$

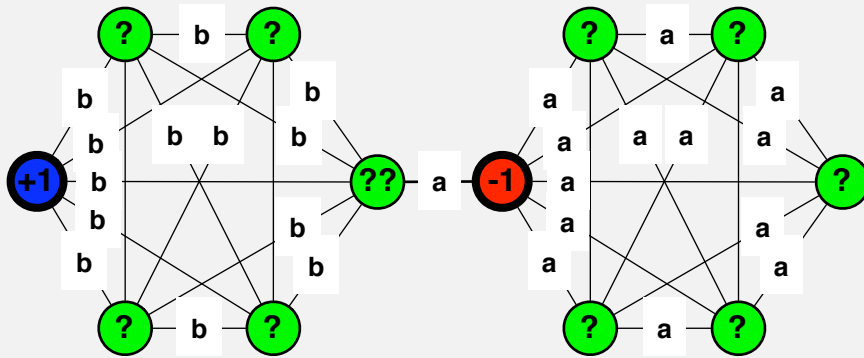


- **Effective resistance is 1** from \mathbf{v}_p to \mathbf{v}_q

The “Intuition”

From “distance” to “resistance”

The base global “distance” $d(p, q)$ is *adapted* according to the empirical distances via the effective resistance $r(p, q)$.



Two m – cliques connected by an edge with distances $a < b$

- ▶ What is the label at “??”
- ▶ Effective resistance between vertices $r_{m\text{-clique}} = \frac{2d}{m}$
- ▶ If $\frac{2b}{m} < a$ then label “+1”
- ▶ Conclusion: labelings respect cluster structure

Trees

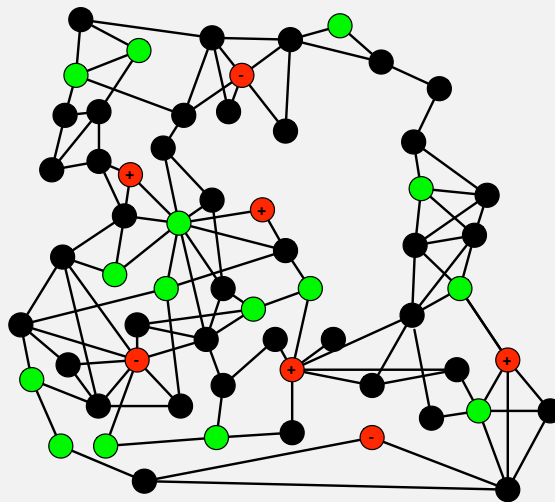
Part II

Fast prediction on a tree

Overview

- ▶ Aim: Speed up Laplacian-based semi-supervised learning
- ▶ Model: **large** graph with **small** label and test set
- ▶ Method:
 1. Approximate graph with a tree
 2. Compute Lap. kernel quickly via resistive network analogy
- ▶ Experiments:
 1. Approximate with MST and SPT trees (ensembles)
 2. Predict with kernel perceptron
 3. Web graph data 400,000 vertices 10 million+ edges

Setup



Legend:

<i>l</i>	:	# labeled points
<i>p</i>	:	# test points
<i>u</i>	:	# unlabeled points

- Predict only a small subset of points
- Expectation: $l + p \ll u$

Graph connectivity

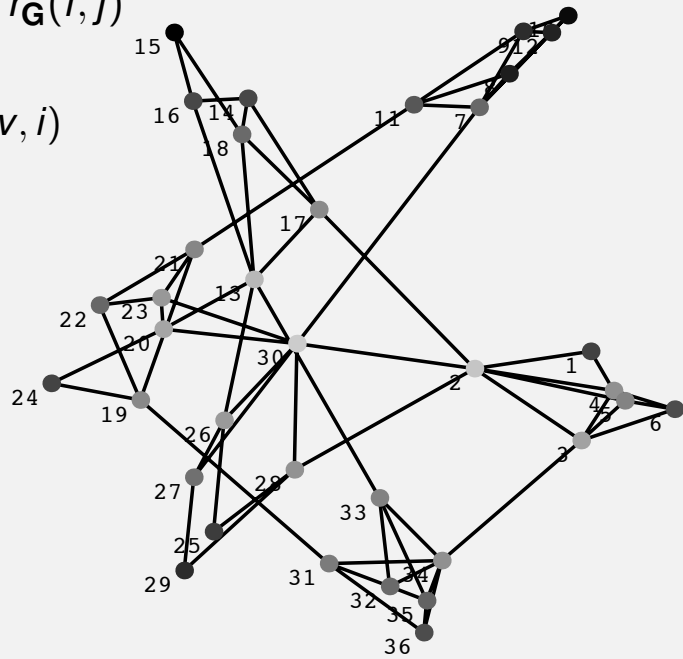
Inverse Connectivities

▶ Graph : $R_{\text{tot}} := \sum_{i>j} r_{\mathbf{G}}(i, j)$

▶ Vertex :

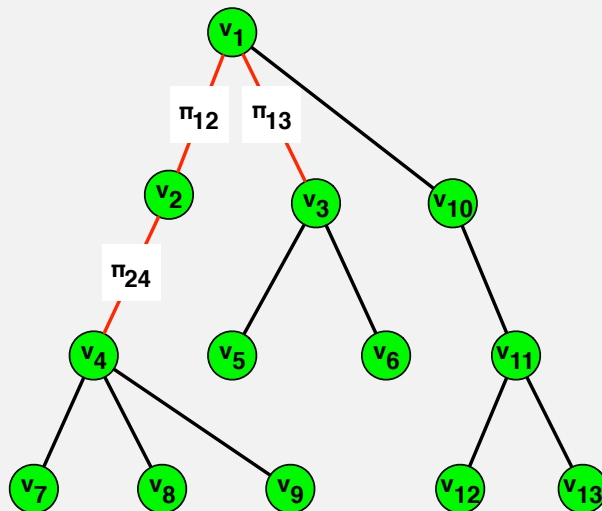
1. $R(v) := \sum_{i=1}^n r_{\mathbf{G}}(v, i)$

2. $\mathbf{G}_{vv}^+ = \frac{R(v)}{n} - \frac{R_{\text{tot}}}{n^2}$



Grey-scale \mathbf{G}_{vv}^+ : $\mathbf{G}_{30,30}^+ = .21$ (min), $\mathbf{G}_{15,15}^+ = .94$ (max)

Effective resistance on a tree



Effective resistance from v_3 to v_4

$$r_{\mathbf{T}}(3, 4) = \pi_{13} + \pi_{12} + \pi_{24}$$

▶ “Resistors in series” : sum along unique path

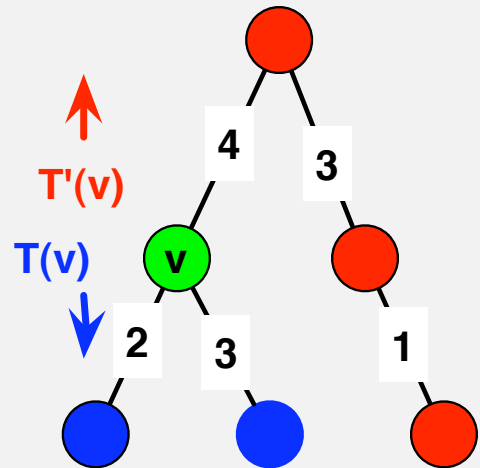
Sketch: computing $m \times m$ block of the kernel \mathbf{G}^+ (tree)

Computing the diagonal in $O(n)$ time

1. Compute $R(v)$, $v = 1, \dots, n$.
2. $R_{\text{tot}} = \frac{1}{2} \sum_{v=1}^n R(v)$
3. $\mathbf{G}_{vv}^+ = \frac{R(v)}{n} - \frac{R_{\text{tot}}}{n^2}$

Computing an $R(v)$

1. Define $T(v) = \sum_{x \in \text{descendants}(v)} r_{\mathbf{G}}(v, x)$
2. Define $T'(v) = \sum_{x \notin \text{descendants}(v)} r_{\mathbf{G}}(v, x)$
3. $R(v) = T(v) + T'(v)$



Computing $m \times m$ block of the kernel \mathbf{G}^+ (tree)

Computing the off-diagonal

Recall: $\mathbf{G}_{ij}^+ = \frac{1}{2}(\mathbf{G}_{ii}^+ + \mathbf{G}_{jj}^+ - r_{\mathbf{G}}(i, j))$

1. Single: \mathbf{G}_{ij}^+ in $O(S)$ (S is diameter)
2. Amortized Block ($m \times m$): $O(m^2 + mS)$

Amortized algorithm for $m \times m$ block

$O(m^2 + mS)$ time

1. **Input:** $\{v_1, \dots, v_m\} \subseteq V$
2. **Initialization:** $\text{visited}(\text{all}) = \emptyset$
3. **for** $i = 1, \dots, m$ **do**
4. $p = -1$; $c = v_i$; $r_{\mathbf{T}}(c, c) = 0$
5. **Repeat**
6. **for** $w \in \text{visited}(c) \cap \overline{\{p\} \cup \downarrow^*(p)}$ **do**
7. $r_{\mathbf{T}}(v_i, w) = r_{\mathbf{T}}(w, v_i) = r_{\mathbf{T}}(v_i, c) + r_{\mathbf{T}}(c, w)$
8. **end**
9. $\text{visited}(c) = \text{visited}(c) \cup v_i$
10. $p = c$; $c = \uparrow(c)$
11. $r_{\mathbf{T}}(v_i, c) = r_{\mathbf{T}}(c, v_i) = r_{\mathbf{T}}(v_i, p) + \pi_{p,c}$
12. **until** (" p is the root")
13. **end**

- "Initialization" + amortized algorithm: $O(n + m^2 + mS)$

Approximate the graph

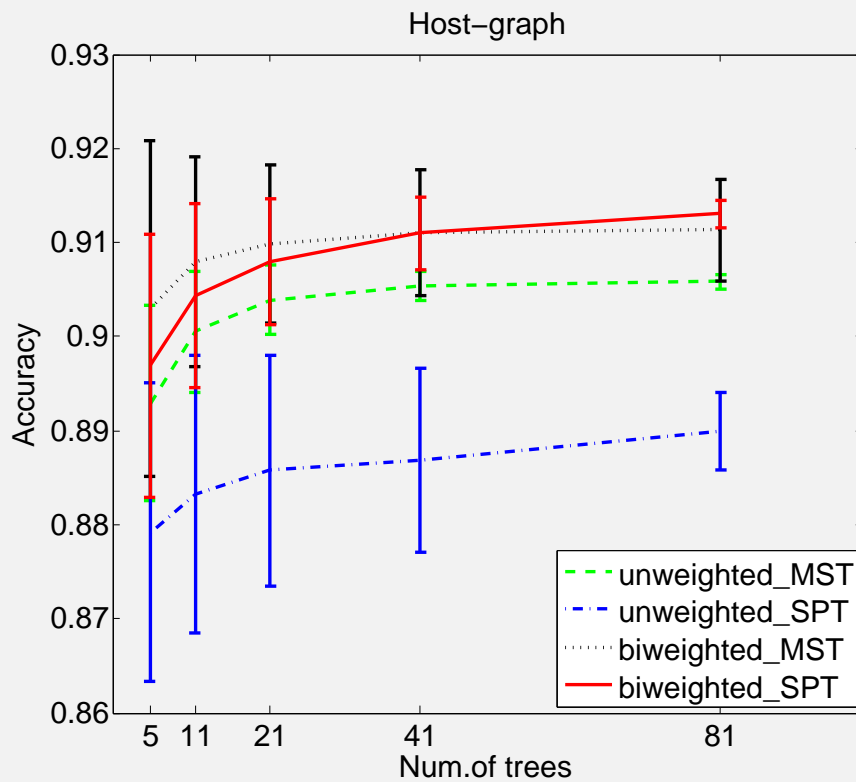
- What if the graph is not a tree?
- Approximate with minimum or shortest paths spanning tree
- Use ensembles of trees

Results

Method	Agg. (81/21)	AUC	Single	AUC
Host-graph (9,072 vertices, 464,959 edges; aggregates: 81)				
MST	0.907	0.950	0.857±0.022	0.841±0.045
SPT	0.889	0.952	0.850±0.026	0.804±0.063
MST (bidir)	0.912	0.944	0.878±0.033	0.851±0.100
SPT (bidir)	0.913	0.960	0.873±0.028	0.846±0.065
Abernathy <i>et al.</i>	0.896	0.952
Tang <i>et al.</i>	0.906	0.951
Filoché <i>et al.</i>	0.889	0.927
Benczúr <i>et al.</i>	0.829	0.877
Web-graph (400,000 vertices, 10,455,544 edges; aggregates: 21)				
MST (bidir)	0.991	1.000	0.976±0.011	0.993±0.005
SPT (bidir)	0.994	0.999	0.985±0.002	0.992±0.003
Witschel <i>et al.</i>	0.995	0.998
Filoché <i>et al.</i>	0.973	0.991
Benczúr <i>et al.</i>	0.942	0.973
Tang <i>et al.</i>	0.296	0.989

- ▶ Classifier: kernel perceptron
- ▶ “bidir”: double-weight mutually linked edges

Aggregate performances



Accuracy versus number of trees

p -Resistance

Part III

Generalized p -resistive networks

p -Resistive Networks

- ▶ Inspired by the p -norm perceptron [GLS97]
- ▶ Redefine power as

$$P_p(\mathbf{u}) := \|\mathbf{u}\|_{\mathcal{G},p}^p = \sum_{(i,j) \in E(\mathbf{G})} w_{ij} |u_i - u_j|^p$$

- ▶ Analogues of usual “electric network” theory now follow ...
1. Kirchoff’s laws
 2. Ohm’s law
 3. Conservation of energy principle
 4. Rayleigh’s monotonicity principle
 5. Black box principle (2-port)
 6. The “rules” of resistors in parallel and in series,

p -resistance

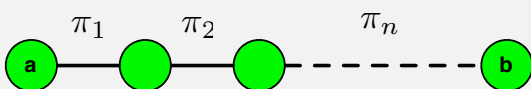
Definition

The (effective) p -resistance between any two vertices a and b is

$$r_p(a, b) = \left[\min_{\mathbf{u} \in \mathbb{R}^n} \{P_p(\mathbf{u}) : u_a = 1, u_b = 0\} \right]^{-1}.$$

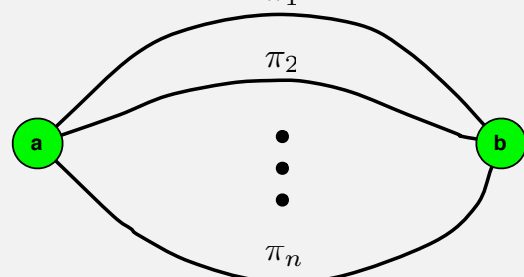
Resistors in series

$$r_p(a, b) = \left(\sum_{i=1}^n \pi_i^{\frac{1}{p-1}} \right)^{p-1}$$



Resistors in parallel

$$r_p(a, b) = \left(\sum_{i=1}^n \frac{1}{\pi_i} \right)^{-1}$$



Note if $p = 1$ then $r_{\mathcal{G},1}(a, b) = \frac{1}{\text{“mincut between } a \text{ and } b\text{”}}$

Online Learning Model

- ▶ Aim: learn a function $\mathbf{u} : V \rightarrow \{-1, +1\}$ corresponding to a labeling of a graph $\mathcal{G} = (V, E)$ and $V = \{1, \dots, n\}$.
- ▶ Learning proceeds in trials
for $t = 1, \dots, \ell$ **do**
 1. Nature selects $v_t \in V$
 2. Learner predicts $\hat{y}_t \in \{-1, +1\}$
 3. Nature selects $y_t \in \{-1, +1\}$
 4. If $\hat{y}_t \neq y_t$ then mistakes = mistakes + 1
- ▶ Learner's goal: *minimize* mistakes
- ▶ mistakes $\leq f(\text{complexity}(\mathbf{u}), \text{structure}(\mathcal{G}))$

Power interpolation algorithm

- ▶ Choose parameter $1 < p \leq 2$
- ▶ Given a sequence of online trials

$$\{(v_{i_1}, y_1), (v_{i_2}, y_2), \dots, (v_{i_\ell}, y_\ell)\}$$

- ▶ Hypothesis vector is interpolant

$$\bar{\mathbf{u}}_t = \arg \min_{\mathbf{u} \in \mathbb{R}^n} \{P_p(\mathbf{u}) : u_{i_1} = y_1, \dots, u_{i_t} = y_t\}$$

- ▶ Predict

$$\hat{y}_t = \text{sign}(\bar{u}_{i_t})$$

- ▶ As $p \rightarrow 1$ predictions correspond to label-consistent mincut

Mistake Bound

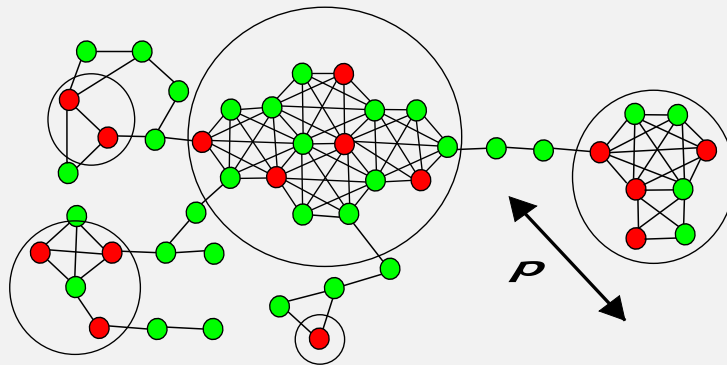
Theorem

Given $\rho \in (1, 2]$ then

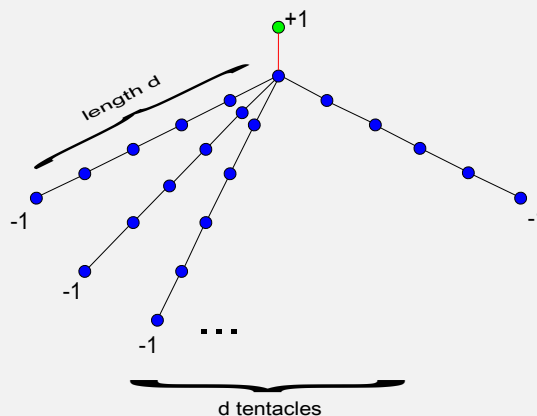
$$M \leq \mathcal{N}_\rho + \frac{[\rho \times P_\rho(\mathbf{u})]^{\frac{2}{\rho}}}{\rho - 1}$$

for all $0 < \rho$, and for all consistent $\mathbf{u} \in \mathbb{R}^n$.

- ▶ Relative to any resistive cover w.r.t. r_ρ
- ▶ Both number of covering sets \mathcal{N}_ρ and resistance ρ



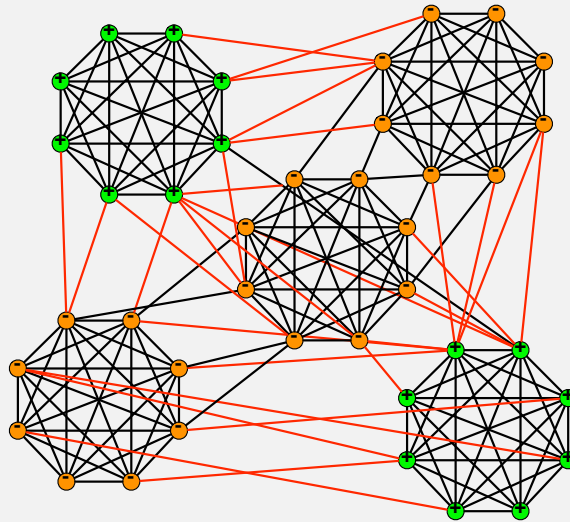
Example: Octopus



A d -tentacle octopus graph with length d tentacles ($n = d^2 + 1$)

- ▶ Cover: $N = 1$, Power: $P_\rho(\mathbf{u}) = 1 \times 2^\rho$
- ▶ Connectivity: $k = 1$, Diameter: $\Delta_1 = 2d$
- ▶ When $\rho = 1 + \frac{1}{\log(2d)-1}$ then $\mathcal{M}_A \leq 1 + 4e^2(\log(2d) - 1)$
- ▶ When $\rho = 2$ then $\sqrt{n} \leq \mathcal{M}_A \leq O(\sqrt{n})$

Example: Prototypical Clusters



$c \times m$ -cliques with ℓ cut edges

- ▶ Cover: $N = c$, Power: $P_p(\mathbf{u}) = \ell \times 2^p$
- ▶ Connectivity: $k = m - 1$, Wide Diameter: $\Delta_{m-1} = 2$
- ▶ When $p = 2$ then $M \leq c + \frac{8\ell}{m-1}$

Conclusions

- ▶ Graph labelings predicted via power minimization
- ▶ Base distances were transformed to effective resistances
- ▶ Eff. resistance demonstrated to respect cluster structure
- ▶ Fast computation of tree kernels via effective resistance
- ▶ A p -resistive network theory may be given
- ▶ Selection of p enables logarithmic mistake bounds

Thanks

