Generalization bounds for learning under graph-dependence

Rui-Ray Zhang
rui.zhang@monash.edu
School of Mathematics, Monash University

ML & VL Seminar

The content is based upon

- *McDiarmid-type inequalities for graph-dependent variables and stability bounds* (with Xingwu Liu, Yuyi Wang, Liwei Wang) Spotlight in Advances in Neural Information Processing Systems 32 (NeurIPS 2019).
- *When Janson meets McDiarmid: Bounded difference inequalities under graph-dependence* Statistics & Probability Letters, 2022.
- *Generalization bounds for learning under graph-dependence: A survey* (with Massih-Reza Amini, arXiv:2203.13534).

Background on machine learning

- Given some input x, to choose $f : x \mapsto y$ that performs well on unknown new data.

Background on machine learning

- Given some input x, to choose $f : x \mapsto y$ that performs well on unknown new data.
- A training set S contains n data points $(x_i, y_i) \sim D$ (unknown).
- A loss function measures error between true y and prediction $\widehat{y} = f(x)$

$$\ell : (y, \widehat{y}) \mapsto \ell(y, f(x)),$$

is upper bounded by $M \in \mathbb{R}_+$.

Background on machine learning

- Given some input x, to choose $f : x \mapsto y$ that performs well on unknown new data.
- A training set S contains n data points $(x_i, y_i) \sim D$ (unknown).
- A loss function measures error between true y and prediction $\widehat{y} = f(x)$

$$\ell : (y, \widehat{y}) \mapsto \ell(y, f(x)),$$

  is upper bounded by $M \in \mathbb{R}_+$.
- Expected error: expected loss on new test data $(x, y) \sim D$ (unknown)

$$R(f) = \mathbb{E}[\ell(y, f(x))].$$

- Empirical error: average loss on given training data $(x_i, y_i)_{i=1}^n$

$$\widehat{R}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)).$$

Background on machine learning

- Given some input x, to choose $f : x \mapsto y$ that performs well on unknown new data.
- A training set S contains n data points $(x_i, y_i) \sim D$ (unknown).
- A loss function measures error between true y and prediction $\widehat{y} = f(x)$

$$\ell : (y, \widehat{y}) \mapsto \ell(y, f(x)),$$

  is upper bounded by $M \in \mathbb{R}_+$.
- Expected error: expected loss on new test data $(x, y) \sim D$ (unknown)

$$R(f) = \mathbb{E}[\ell(y, f(x))].$$

- Empirical error: average loss on given training data $(x_i, y_i)_{i=1}^n$

$$\widehat{R}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)).$$

- The goal is to establish generalisation error bounds

$$R(f) \leqslant \widehat{R}(f) + ?$$

Background on machine learning

The ways to establish generalisation error (also called generalization gap) bounds is by

- Measuring of the complexity of the output hypothesis space.
  - VC theory (Vapnik and Chervonenkis 1971), where VC-dimension is used.
  - Rademacher complexity (Bartlett and Mendelson 2002).

Background on machine learning

The ways to establish generalisation error (also called generalization gap) bounds is by

- Measuring of the complexity of the output hypothesis space.
  - VC theory (Vapnik and Chervonenkis 1971), where VC-dimension is used.
  - Rademacher complexity (Bartlett and Mendelson 2002).
- Exploiting properties of the learning algorithm.
  - Algorithmic stability (Bousquet and Elisseeff 2002).
  - PAC-Bayesian bounds (McAllester, 1999).

Background on machine learning

The ways to establish generalisation error (also called generalization gap) bounds is by

- Measuring of the complexity of the output hypothesis space.
  - VC theory (Vapnik and Chervonenkis 1971), where VC-dimension is used.
  - Rademacher complexity (Bartlett and Mendelson 2002).
- Exploiting properties of the learning algorithm.
  - Algorithmic stability (Bousquet and Elisseeff 2002).
  - PAC-Bayesian bounds (McAllester, 1999).
- Considering mutual information between training sample and the output hypothesis of the learning algorithm.
  - Mutual information bound (Russo and Zou 2016).

Background on machine learning

The ways to establish generalisation error (also called generalization gap) bounds is by

- Measuring of the complexity of the output hypothesis space.
  - VC theory (Vapnik and Chervonenkis 1971), where VC-dimension is used.
  - Rademacher complexity (Bartlett and Mendelson 2002).
- Exploiting properties of the learning algorithm.
  - Algorithmic stability (Bousquet and Elisseeff 2002).
  - PAC-Bayesian bounds (McAllester, 1999).
- Considering mutual information between training sample and the output hypothesis of the learning algorithm.
  - Mutual information bound (Russo and Zou 2016).

Most of them assume that samples are i.i.d., which is not the case in many settings.

Concentration inequalities

Let $\boldsymbol{X}$ be a vector of i.i.d. random variables.

Concentration inequalities

Let $\boldsymbol{X}$ be a vector of i.i.d. random variables. Concentration inequalities bounding the probability of deviation of a function from its expectation

$$\mathbb{P}(g(\boldsymbol{X}) - \mathbb{E}[g(\boldsymbol{X})] \geqslant t).$$

Concentration inequalities

Let $\boldsymbol{X}$ be a vector of i.i.d. random variables. Concentration inequalities bounding the probability of deviation of a function from its expectation

$$\mathbb{P}(g(\boldsymbol{X}) - \mathbb{E}[g(\boldsymbol{X})] \geqslant t).$$

They are basic tools to establish generalization theory, in which

$$g(\boldsymbol{x}) = \mathbb{E}[\ell(y, f(x))] - \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f(x_i))$$

is the difference of expected error and empirical error.

Bounded difference inequality

c-Lipschitz

Given $\boldsymbol{c} = (c_1, \ldots, c_n) \in \mathbb{R}_+^n$, a function g is $\boldsymbol{c}$-Lipschitz if

$$\left| g(x_1, \ldots, x_i, \ldots, x_n) - g(x_1, \ldots, x_i', \ldots, x_n) \right| \leq c_i.$$

Bounded difference inequality

c-Lipschitz

Given $\boldsymbol{c} = (c_1, \dots, c_n) \in \mathbb{R}_+^n$, a function g is $\boldsymbol{c}$-Lipschitz if

$$\left| g(x_1, \dots, x_i, \dots, x_n) - g(x_1, \dots, x_i', \dots, x_n) \right| \leqslant c_i.$$

Bounded difference inequality (McDiarmid 1989)

If we have that

1. g is $\boldsymbol{c}$-Lipschitz,
2. $\boldsymbol{X} = (X_1, \dots, X_n)$ is a vector of independent random variables,

Bounded difference inequality

c-Lipschitz

Given $\boldsymbol{c} = (c_1, \ldots, c_n) \in \mathbb{R}_+^n$, a function g is $\boldsymbol{c}$-Lipschitz if

$$\left| g(x_1, \ldots, x_i, \ldots, x_n) - g(x_1, \ldots, x_i', \ldots, x_n) \right| \leqslant c_i.$$

Bounded difference inequality (McDiarmid 1989)

If we have that

1. g is $\boldsymbol{c}$-Lipschitz,
2. $\boldsymbol{X} = (X_1, \ldots, X_n)$ is a vector of independent random variables,

then for $t > 0$,

$$\mathbb{P}(g(\boldsymbol{X}) - \mathbb{E}[g(\boldsymbol{X})] \geqslant t) \leqslant \exp\left(-\frac{2t^2}{\|\boldsymbol{c}\|_2^2} \right).$$

This is also called Azuma-Hoeffding inequality.

Bounded difference inequality

c-Lipschitz

Given $\boldsymbol{c} = (c_1, \ldots, c_n) \in \mathbb{R}_+^n$, a function g is $\boldsymbol{c}$-Lipschitz if

$$\left| g(x_1, \ldots, \mathrm{x_i}, \ldots, x_n) - g(x_1, \ldots, \mathrm{x_i'}, \ldots, x_n) \right| \leqslant \mathrm{c_i}.$$

Bounded difference inequality (McDiarmid 1989)

If we have that

1. g is $\boldsymbol{c}$-Lipschitz,
2. $\boldsymbol{X} = (X_1, \ldots, X_n)$ is a vector of independent random variables,

then for $t > 0$,

$$\mathbb{P}\left(g(\boldsymbol{X}) - \mathbb{E}[g(\boldsymbol{X})] \geqslant t\right) \leqslant \exp\left(-\frac{2t^2}{\|\boldsymbol{c}\|_2^2}\right).$$

This is also called Azuma-Hoeffding inequality.

- If all $c_i = c$, then for any fixed $\delta \in (0,1)$, with probability at least $1 - \delta$, we have

$$f - \mathbb{E}[f] \leqslant \|\boldsymbol{c}\|_2 \sqrt{\frac{1}{2} \log\left(\frac{1}{\delta}\right)} = c\sqrt{\frac{n}{2} \log\left(\frac{1}{\delta}\right)}.$$

Dependent random variables

- Mixing coefficients: α, β, ϕ-mixing, etc.
  - quantitatively measure the dependencies, and widely used in probability, statistics, e.g.,

$$\alpha(s) = \sup\left\{\left|\mathbb{P}(A \cap B) - \mathbb{P}(A)\mathbb{P}(B)\right| : A \in \sigma\left(\{X_i\}_{-\infty}^{t}\right), B \in \sigma\left(\{X_i\}_{t+s}^{\infty}\right)\right\}$$

Dependent random variables

- Mixing coefficients: α, β, ϕ-mixing, etc.
  - quantitatively measure the dependencies, and widely used in probability, statistics, e.g.,

$$\alpha(s) = \sup\left\{\left|\mathbb{P}(A \cap B) - \mathbb{P}(A)\mathbb{P}(B)\right| : A \in \sigma\left(\{X_i\}_{-\infty}^{t}\right), B \in \sigma\left(\{X_i\}_{t+s}^{\infty}\right)\right\}$$

- Dependency graphs: combinatorial, relate to independent sets, degrees, etc.

Dependent random variables

- Mixing coefficients: α, β, ϕ-mixing, etc.
  - quantitatively measure the dependencies, and widely used in probability, statistics, e.g.,

$$\alpha(s) = \sup \left\{ \left| \mathbb{P}(A \cap B) - \mathbb{P}(A)\mathbb{P}(B) \right| : A \in \sigma\left(\{X_i\}_{-\infty}^{t}\right), B \in \sigma\left(\{X_i\}_{t+s}^{\infty}\right) \right\}$$

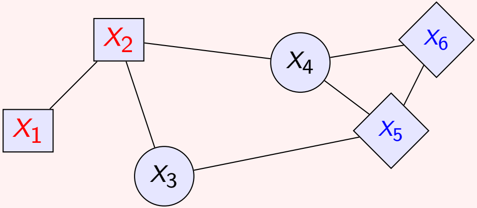
- Dependency graphs: combinatorial, relate to independent sets, degrees, etc.
- Copula, graphical models (random field, Bayesian network, etc.), time series, etc.

Dependency Graphs

Definition

Graph G is a dependency graph for random variables $\boldsymbol{X} = (X_1, \ldots, X_n)$ if

- *Vertex set $V(G) = [n] = \{1, \ldots, n\}$.*

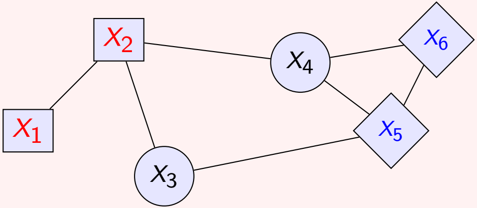


Dependency Graphs

Definition

Graph G is a dependency graph for random variables $\boldsymbol{X} = (X_1, \dots, X_n)$ if

- *Vertex set $V(G) = [n] = \{1, \dots, n\}$.*



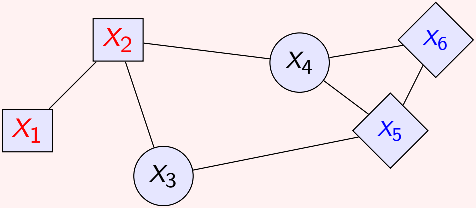
- *If disjoint subsets $I, J \subset [n]$ are non-adjacent in G, then $\{X_i\}_{i \in I}$ and $\{X_j\}_{j \in J}$ are independent.*
  - *In the above example, $\{X_1, X_2\}$ and $\{X_5, X_6\}$ are independent.*

Dependency Graphs

Definition

Graph G is a dependency graph for random variables $\boldsymbol{X} = (X_1, \dots, X_n)$ if

- *Vertex set $V(G) = [n] = \{1, \dots, n\}$.*



- *If disjoint subsets $I, J \subset [n]$ are non-adjacent in G, then $\{X_i\}_{i \in I}$ and $\{X_j\}_{j \in J}$ are independent.*
  - *In the above example, $\{X_1, X_2\}$ and $\{X_5, X_6\}$ are independent.*

- The dependency graph for a set of random variables is not necessarily unique.

Idea: to utilise independence among variables

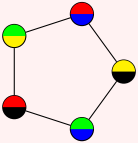
Given a graph G with n vertices, a fractional vertex covering $\{(I_j, w_j)\}_j$ of G satisfies

1. each $I_j \subseteq [n]$ is an independent set (no two vertices are adjacent),

Idea: to utilise independence among variables

Given a graph G with n vertices, a fractional vertex covering $\{(I_j, w_j)\}_j$ of G satisfies

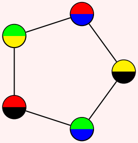
1. each $I_j \subseteq [n]$ is an independent set (no two vertices are adjacent),
2. $\sum_{j:v \in I_j} w_j = 1$ for each vertex.



Idea: to utilise independence among variables

Given a graph G with n vertices, a fractional vertex covering $\{(I_j, w_j)\}_j$ of G satisfies

1. each $I_j \subseteq [n]$ is an independent set (no two vertices are adjacent),
2. $\sum_{j: v \in I_j} w_j = 1$ for each vertex.



A function g is *decomposable* $\boldsymbol{c}$*-Lipschitz* with respect to graph G if there exist $(c_i)_{i \in I_j}$-Lipschitz functions $\{g_j\}_j$ such that

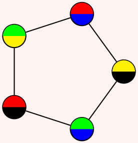
$$g(\boldsymbol{x}) = \sum_j w_j g_j(\mathrm{x}_{I_j}),$$

for all $\mathrm{x} = (x_1, \ldots, x_n)$, and for all fractional vertex covers $\{(I_j, w_j)\}_j$ of G.

Idea: to utilise independence among variables

Given a graph G with n vertices, a fractional vertex covering $\{(I_j, w_j)\}_j$ of G satisfies

1. each $I_j \subseteq [n]$ is an independent set (no two vertices are adjacent),
2. $\sum_{j : v \in I_j} w_j = 1$ for each vertex.



A function g is *decomposable* $\boldsymbol{c}$*-Lipschitz* with respect to graph G if there exist $(c_i)_{i \in I_j}$-Lipschitz functions $\{g_j\}_j$ such that

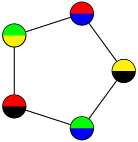
$$g(\boldsymbol{x}) = \sum_j w_j g_j(\mathrm{x}_{I_j}),$$

for all $\mathrm{x} = (x_1, \ldots, x_n)$, and for all fractional vertex covers $\{(I_j, w_j)\}_j$ of G.

- Summation is decomposable $\boldsymbol{c}$-Lipschitz.

Given a graph G with n vertices, a fractional vertex covering $\{(I_j, w_j)\}_j$ of G satisfies

1. each $I_j \subseteq [n]$ is an independent set (no two vertices are adjacent),
2. $\sum_{j: v \in I_j} w_j = 1$ for each vertex.



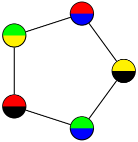
Theorem (Usunier et al. NIPS05; Z 2022; Z. and Amini 2022+)

If we have that

1. g *is decomposable* $\boldsymbol{c}$*-Lipschitz,*
2. $\boldsymbol{X} = (X_1, \dots, X_n)$ *is* G*-dependent,*

Given a graph G with n vertices, a fractional vertex covering $\{(I_j, w_j)\}_j$ of G satisfies

1. each $I_j \subseteq [n]$ is an independent set (no two vertices are adjacent),
2. $\sum_{j: v \in I_j} w_j = 1$ for each vertex.



Theorem (Usunier et al. NIPS05; Z 2022; Z. and Amini 2022+)

If we have that

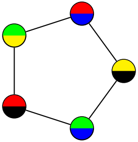
1. *g is decomposable $\boldsymbol{c}$-Lipschitz,*
2. *$\boldsymbol{X} = (X_1, \ldots, X_n)$ is G-dependent,*

then for $t > 0$,

$$\mathbb{P}(g(\boldsymbol{X}) - \mathbb{E}[g(\boldsymbol{X})] \geqslant t) \leqslant \exp\left(-\frac{2t^2}{\chi^\star(G)\|\boldsymbol{c}\|_2^2}\right),$$

Given a graph G with n vertices, a fractional vertex covering $\{(I_j, w_j)\}_j$ of G satisfies

1. each $I_j \subseteq [n]$ is an independent set (no two vertices are adjacent),
2. $\sum_{j: v \in I_j} w_j = 1$ for each vertex.



Theorem (Usunier et al. NIPS05; Z 2022; Z. and Amini 2022+)

If we have that

1. *g is decomposable $\boldsymbol{c}$-Lipschitz,*
2. *$\boldsymbol{X} = (X_1, \ldots, X_n)$ is G-dependent,*

then for $t > 0$,

$$\mathbb{P}(g(\boldsymbol{X}) - \mathbb{E}[g(\boldsymbol{X})] \geqslant t) \leqslant \exp\left(-\frac{2t^2}{\chi^\star(G)\|\boldsymbol{c}\|_2^2}\right),$$

where $\chi^\star(G) = \sum_j w_j \leqslant \Delta(G) + 1$.

- In the above example, $\chi^\star(G) = 5/2$.

Forest-dependent random variables

Theorem (Z. et al. NeurIPS19, Z. 2022)

If we have that

1. *g is $\boldsymbol{c}$-Lipschitz,*
2. *F is a dependency graph for $\boldsymbol{X}$, where $F = \{T_i\}_{i=1}^k$ is a forest,*

Forest-dependent random variables

Theorem (Z. et al. NeurIPS19, Z. 2022)

If we have that

1. *g is $\boldsymbol{c}$-Lipschitz,*
2. *F is a dependency graph for $\boldsymbol{X}$, where $F = \{T_i\}_{i=1}^k$ is a forest,*

then for $t > 0$,

$$\mathbb{P}(g(\boldsymbol{X}) - \mathbb{E}[g(\boldsymbol{X})] \geqslant t) \leqslant \exp\left(-\frac{2t^2}{\sum_{i=1}^k c_{\min,i}^2 + \sum_{\{i,j\} \in E(F)} (c_i + c_j)^2}\right),$$

where $c_{\min,i} = \min\{c_j : j \in V(T_i)\}$ is the minimum entry of $\boldsymbol{c}$ in each tree T_i.

Forest-dependent random variables

Theorem (Z. et al. NeurIPS19, Z. 2022)

If we have that

1. *g is $\boldsymbol{c}$-Lipschitz,*
2. *F is a dependency graph for $\boldsymbol{X}$, where $F = \{T_i\}_{i=1}^k$ is a forest,*

then for $t > 0$,

$$\mathbb{P}(g(\boldsymbol{X}) - \mathbb{E}[g(\boldsymbol{X})] \geqslant t) \leqslant \exp\left(-\frac{2t^2}{\sum_{i=1}^k c_{\min,i}^2 + \sum_{\{i,j\}\in E(F)}(c_i + c_j)^2}\right),$$

where $c_{\min,i} = \min\{c_j : j \in V(T_i)\}$ is the minimum entry of $\boldsymbol{c}$ in each tree T_i.

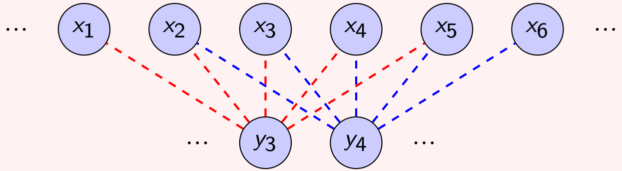
- General graphs are handled via tree-partitions (transforming a graph to a forest by merging vertices).

Applications

Stability bound for learning m-dependent data

Example

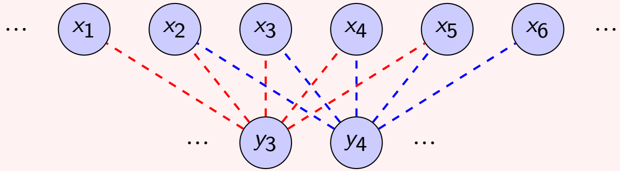
- y_i: observation at location i, e.g., house price
- x_i: random variable modelling influential factors at location i



Stability bound for learning m-dependent data

Example

- y_i: observation at location i, e.g., house price
- x_i: random variable modelling influential factors at location i

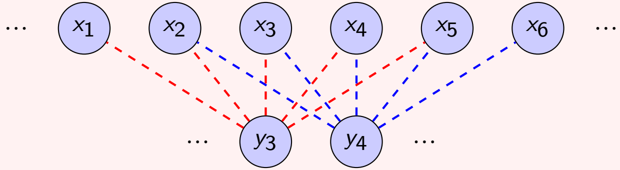


- Given training data: $\mathrm{S} = \{\ldots, ((x_1, x_2, x_3, x_4, x_5), y_3), ((x_2, x_3, x_4, x_5, x_6), y_4), \ldots\}$.
- To find $f : (x_{i-2}, x_{i-1}, x_i, x_{i+1}, x_{i+2}) \mapsto y_i$.

Stability bound for learning m-dependent data

Example

- y_i: observation at location i, e.g., house price
- x_i: random variable modelling influential factors at location i



- Given training data: $S = \{\ldots, ((x_1, x_2, x_3, x_4, x_5), y_3), ((x_2, x_3, x_4, x_5, x_6), y_4), \ldots\}$.
- To find $f : (x_{i-2}, x_{i-1}, x_i, x_{i+1}, x_{i+2}) \mapsto y_i$.

Definition (Hoeffding and Robbins 1948)

A sequence of random variables $(X_i)_{i=1}^n$ is m-dependent for some $m \geqslant 1$ if $(X_j)_{j=1}^i$ and $(X_j)_{j=i+m+1}^n$ are independent for all $i > 0$.

Stability bound for learning m-dependent data

Given a sample S, a learning algorithm $\mathscr{A} : \mathsf{S} \mapsto f_{\mathsf{S}}^{\mathscr{A}}$ outputs $f_{\mathsf{S}}^{\mathscr{A}}$.

Uniform stability (Bousquet and Elisseeff 2002)

A learning algorithm $\mathscr{A}$ is β_n-uniformly stable if

$$\max_{i \in [n]} \left| \ell(y, f_{\mathsf{S}}^{\mathscr{A}}(x)) - \ell(y, f_{\mathsf{S}^{\setminus i}}^{\mathscr{A}}(x)) \right| \leqslant \beta_n,$$

where $\mathsf{S}^{\setminus i}$ is by deleting i-th data point from S.

Stability bound for learning m-dependent data

Given a sample S, a learning algorithm $\mathscr{A} : \mathrm{S} \mapsto f_{\mathrm{S}}^{\mathscr{A}}$ outputs $f_{\mathrm{S}}^{\mathscr{A}}$.

Uniform stability (Bousquet and Elisseeff 2002)

A learning algorithm $\mathscr{A}$ is β_n-uniformly stable if

$$\max_{i \in [n]} \left| \ell(y, f_{\mathrm{S}}^{\mathscr{A}}(x)) - \ell(y, f_{\mathrm{S}^{\setminus i}}^{\mathscr{A}}(x)) \right| \leqslant \beta_n,$$

where $\mathrm{S}^{\setminus i}$ is by deleting i-th data point from S.

We have

$$R(f_{\mathrm{S}}^{\mathscr{A}}) \leqslant \widehat{R}(f_{\mathrm{S}}^{\mathscr{A}}) + 2\beta_{n,2m}(2m+1) + (4n\beta_n + M)\sqrt{\frac{2m}{n} \log\left(\frac{1}{\delta}\right)},$$

which introduces a factor $4m$ comparing with the independent case (Bousquet and Elisseeff 2002)

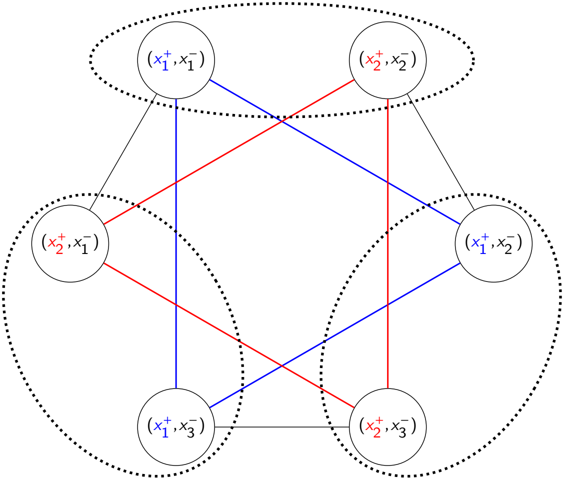
$$R(f_{\mathrm{S}}^{\mathscr{A}}) \leqslant \widehat{R}(f_{\mathrm{S}}^{\mathscr{A}}) + 2\beta_n + (4n\beta_n + M)\sqrt{\frac{1}{2n} \log\left(\frac{1}{\delta}\right)}.$$

Bipartite ranking

- Training set: $T = (x_i, y_i)_{i=1}^m$ with $y_i \in \{-1, +1\}$.
- The goal: to find a scoring function h that gives higher scores to instances of the positive class than the ones of the negative class.

Bipartite ranking

- Training set: $T = (x_i, y_i)_{i=1}^{m}$ with $y_i \in \{-1, +1\}$.
- The goal: to find a scoring function h that gives higher scores to instances of the positive class than the ones of the negative class.
- For $(x, y), (x', y')$ with $y \neq y'$, we consider unordered pairs of examples (x, x').



Bipartite ranking

- Let $\mathrm{S} = \left\{(x, x') \in T \times T : y \neq y'\right\}$ be the set of unordered pairs of examples from different classes in T.

Bipartite ranking

- Let $\mathsf{S} = \{(x, x') \in T \times T : y \neq y'\}$ be the set of unordered pairs of examples from different classes in T.
- The empirical loss of a scoring function h over T can be written as a summation over the pairs of instances of different classes:

$$\widehat{R}(h) = \frac{1}{|\mathsf{S}|} \sum_{(x,x') \in \mathsf{S}} \mathbb{1}_{\{z_{x,x'}(h(x) - h(x')) \leqslant 0\}},$$

where $z_{x,x'} = 2\mathbb{1}_{\{y - y' > 0\}} - 1$.

  - If $y = 1$ and $y' = -1$, then $z_{x,x'}(h(x) - h(x')) = h(x) - h(x')$.

Bipartite ranking

An approach based on fractional Rademacher complexity gives the following.

Corollary (Z. and Amini 2022+)

Let T be a training set composed of m_+ positive instances and m_- negative ones. Then for any scoring functions in $\{h : (x, x') \mapsto \langle \mathrm{w}, \phi(x) - \phi(x') \rangle ; \|\mathrm{w}\| \leqslant B\}$, where ϕ is a feature mapping with bounded norm, such that $\forall (x, x'), \|\phi(x) - \phi(x')\| \leqslant \Gamma$, and for any $\delta \in (0, 1)$ with probability at least $1 - \delta$, we have

$$R(f) \leqslant \widehat{R}(f) + \frac{4B\Gamma}{\sqrt{m}} + 3\sqrt{\frac{1}{2m} \log\left(\frac{2}{\delta}\right)},$$

where $m = \min(m_-, m_+)$.

Thanks for your attention!