Learning graphical models for hypothesis testing and classification[1]

iPAL @ PENNSTATE
Information Processing and Algorithms Laboratory

Umamahesh Srinivas

iPAL Group Meeting

October 22, 2010

[1] Tan et al., IEEE Trans. Signal Processing, Nov. 2010

Outline

1. Background and motivation
2. Graphical models: some preliminaries
3. Generative learning of trees
4. Discriminative learning of trees
    - Discriminative learning of forests
5. Learning thicker graphs via boosting
6. Extension to multi-class problems

Binary hypothesis testing problem

Random vector $\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{X}^n$ generated from either of two hypotheses

$$H_0 : \quad \mathbf{x} \sim p$$
$$H_1 : \quad \mathbf{x} \sim q$$

Given: Training sets $\mathcal{T}_p$ and $\mathcal{T}_q$, K samples each

Goal: Classify new sample as coming from H_0 or H_1

Assumption: Class densities p and q known exactly

Likelihood ratio test (LRT)

$$L(\mathbf{x}) := \frac{p(\mathbf{x})}{q(\mathbf{x})} \underset{H_0}{\overset{H_1}{\gtreqless}} \tau.$$

Binary hypothesis testing problem

Random vector $\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{X}^n$ generated from either of two hypotheses

$$H_0 : \quad \mathbf{x} \sim p$$
$$H_1 : \quad \mathbf{x} \sim q$$

Given: Training sets $\mathcal{T}_p$ and $\mathcal{T}_q$, K samples each

Goal: Classify new sample as coming from H_0 or H_1

Assumption: Class densities p and q known exactly

Likelihood ratio test (LRT)

$$L(\mathbf{x}) := \frac{p(\mathbf{x})}{q(\mathbf{x})} \underset{H_0}{\overset{H_1}{\gtreqless}} \tau.$$

Binary hypothesis testing problem

Random vector $\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{X}^n$ generated from either of two hypotheses

$$H_0 : \quad \mathbf{x} \sim p$$
$$H_1 : \quad \mathbf{x} \sim q$$

Given: Training sets $\mathcal{T}_p$ and $\mathcal{T}_q$, K samples each

Goal: Classify new sample as coming from H_0 or H_1

Assumption: Class densities p and q known exactly

Likelihood ratio test (LRT)

$$L(\mathbf{x}) := \frac{p(\mathbf{x})}{q(\mathbf{x})} \underset{H_0}{\overset{H_1}{\gtreqless}} \tau.$$

What if true densities are not known *a priori*?

- Estimate empiricals p_e and q_e from $\mathcal{T}_p$ and $\mathcal{T}_q$ respectively
- LRT using p_e and q_e

What if true densities are not known *a priori*?

- Estimate empiricals p_e and q_e from $\mathcal{T}_p$ and $\mathcal{T}_q$ respectively
- LRT using p_e and q_e

Problem: For high-dimensional data, need large number of samples to get reasonable empirical estimates.

What if true densities are not known *a priori*?

- Estimate empiricals p_e and q_e from $\mathcal{T}_p$ and $\mathcal{T}_q$ respectively
- LRT using p_e and q_e

Problem: For high-dimensional data, need large number of samples to get reasonable empirical estimates.

Graphical models:

- Efficiently learn tractable models from insufficient data
- Trade-off between consistency and generalization

What if true densities are not known *a priori*?

- Estimate empiricals p_e and q_e from $\mathcal{T}_p$ and $\mathcal{T}_q$ respectively
- LRT using p_e and q_e

Problem: For high-dimensional data, need large number of samples to get reasonable empirical estimates.

Graphical models:

- Efficiently learn tractable models from insufficient data
- Trade-off between consistency and generalization

Generative learning: Learning models to *approximate* distributions.

- Learn $\widehat{p}$ from $\mathcal{T}_p$, and $\widehat{q}$ from $\mathcal{T}_q$.

What if true densities are not known *a priori*?

- Estimate empiricals p_e and q_e from $\mathcal{T}_p$ and $\mathcal{T}_q$ respectively
- LRT using p_e and q_e

Problem: For high-dimensional data, need large number of samples to get reasonable empirical estimates.

Graphical models:

- Efficiently learn tractable models from insufficient data
- Trade-off between consistency and generalization

Generative learning: Learning models to *approximate* distributions.

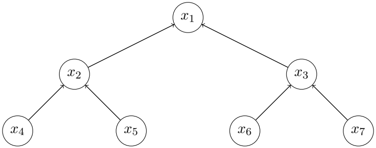
- Learn $\widehat{p}$ from $\mathcal{T}_p$, and $\widehat{q}$ from $\mathcal{T}_q$.

Discriminative learning: Learning models for *binary classification*.

- Learn $\widehat{p}$ from $\mathcal{T}_p$ and $\mathcal{T}_q$; likewise $\widehat{q}$.

Graphical models: Preliminaries

- (Undirected) Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ defined by a set of nodes $\mathcal{V} = \{1, \ldots, n\}$, and a set of edges $\mathcal{E} \subset \binom{\mathcal{V}}{2}$.
- Graphical model: Random vector defined on a graph such that each node represents one (or more) random variables, and edges reveal conditional dependencies.
- Graph structure defines factorization of joint probability distribution.



$f(\mathbf{x}) = f(x_1)f(x_2|x_1)f(x_3|x_1)f(x_4|x_2)f(x_5|x_2)f(x_6|x_3)f(x_7|x_3).$

- Local Markov property:

$$p(x_i|x_{\mathcal{V}\setminus i}) = p(x_i|x_{\mathcal{N}(i)}),\ \forall\, i \in \mathcal{V}.$$

Such a $p(\mathbf{x})$ is Markov *w.r.t.* $\mathcal{G}$.

Trees and forests

- Tree: Undirected acyclic graph with exactly $(n-1)$ edges.
- Forest: Contains $k < (n-1)$ edges $\rightarrow$ not connected.

Trees and forests

- Tree: Undirected acyclic graph with exactly $(n-1)$ edges.
- Forest: Contains $k < (n-1)$ edges $\rightarrow$ not connected.
- Factorization property:

$$\widehat{p}(\mathbf{x}) = \prod_{i \in \mathcal{V}} \widehat{p}(x_i) \prod_{(i,j) \in \mathcal{E}} \frac{\widehat{p}_{i,j}(x_i, x_j)}{\widehat{p}(x_i)\widehat{p}(x_j)}.$$

Trees and forests

- Tree: Undirected acyclic graph with exactly $(n-1)$ edges.
- Forest: Contains $k < (n-1)$ edges $\rightarrow$ not connected.
- Factorization property:

$$\widehat{p}(\mathbf{x}) = \prod_{i \in \mathcal{V}} \widehat{p}(x_i) \prod_{(i,j) \in \mathcal{E}} \frac{\widehat{p}_{i,j}(x_i, x_j)}{\widehat{p}(x_i)\widehat{p}(x_j)}.$$

- Notational convention: p represents probability distribution, $\widehat{p}$ represents a graphical approximation (tree- or forest-structured).

Trees and forests

- Tree: Undirected acyclic graph with exactly $(n-1)$ edges.
- Forest: Contains $k < (n-1)$ edges $\rightarrow$ not connected.
- Factorization property:

$$\widehat{p}(\mathbf{x}) = \prod_{i \in \mathcal{V}} \widehat{p}(x_i) \prod_{(i,j) \in \mathcal{E}} \frac{\widehat{p}_{i,j}(x_i, x_j)}{\widehat{p}(x_i)\widehat{p}(x_j)}.$$

- Notational convention: p represents probability distribution, $\widehat{p}$ represents a graphical approximation (tree- or forest-structured).
- Projection $\widehat{p}$ of p onto a tree (or forest):

$$\widehat{p}(\mathbf{x}) := \prod_{i \in \mathcal{V}} p(x_i) \prod_{(i,j) \in \mathcal{E}} \frac{p_{i,j}(x_i, x_j)}{p(x_i)p(x_j)}.$$

Generative learning of trees[2]

- Optimal tree approximation of a distribution

$$\text{Given } p, \text{ find } \widehat{p} = \arg\min_{\widehat{p} \in \mathcal{T}} D(p||\widehat{p}).$$

$$\left(D(p||\widehat{p}) := \int p(\mathbf{x}) \log \left(\frac{p(\mathbf{x})}{\widehat{p}(\mathbf{x})} \right) d\mathbf{x}. \right)$$

[2] Chow and Liu, IEEE Trans. Inf. Theory 1968

Generative learning of trees[2]

- Optimal tree approximation of a distribution

$$\text{Given } p, \text{ find } \widehat{p} = \arg\min_{\widehat{p}\in\mathcal{T}} D(p||\widehat{p}).$$

$$\left(D(p||\widehat{p}) := \int p(\mathbf{x}) \log\left(\frac{p(\mathbf{x})}{\widehat{p}(\mathbf{x})}\right) d\mathbf{x}. \right)$$

- Equivalent max-weight spanning tree (MWST) problem:

$$\max_{\mathcal{E}:\mathcal{G}=(\mathcal{V},\mathcal{E}) \text{ is a tree}} \sum_{(i,j)\in\mathcal{E}} I(x_i;x_j).$$

[2] Chow and Liu, IEEE Trans. Inf. Theory 1968

Generative learning of trees[2]

- Optimal tree approximation of a distribution

$$\text{Given } p, \text{ find } \widehat{p} = \arg\min_{\widehat{p}\in\mathcal{T}} D(p||\widehat{p}).$$

$$\left(D(p||\widehat{p}) := \int p(\mathbf{x}) \log\left(\frac{p(\mathbf{x})}{\widehat{p}(\mathbf{x})}\right) d\mathbf{x}. \right)$$

- Equivalent max-weight spanning tree (MWST) problem:

$$\max_{\mathcal{E}:\mathcal{G}=(\mathcal{V},\mathcal{E}) \text{ is a tree}} \sum_{(i,j)\in\mathcal{E}} I(x_i; x_j).$$

- Need only marginal and pairwise statistics
- Kruskal MWST algorithm.

[2] Chow and Liu, IEEE Trans. Inf. Theory 1968

J-divergence

Given distributions p and q,

$$J(p,q) := D(p||q) + D(q||p) = \int_{\Omega \subset \mathcal{X}^n} (p(\mathbf{x}) - q(\mathbf{x})) \log\left(\frac{p(\mathbf{x})}{q(\mathbf{x})}\right) d\mathbf{x}.$$

$$\frac{1}{4}\exp(-J) \leq \ \Pr(\text{err}) \leq \frac{1}{2}\left(\frac{J}{4}\right)^{-\frac{1}{4}}.$$

- Maximize J to minimize upper bound on Pr(err).

Tree-approximate J-divergence of $\widehat{p}, \widehat{q}$ *w.r.t* p, q:

$$\widehat{J}(\widehat{p}, \widehat{q}; p, q) := \int_{\Omega \subset \mathcal{X}^n} (p(\mathbf{x}) - q(\mathbf{x})) \log\left(\frac{\widehat{p}(\mathbf{x})}{\widehat{q}(\mathbf{x})}\right) d\mathbf{x}.$$

Marginal consistency of $\widehat{p}$ *w.r.t.* p:

$$\widehat{p}_{(i,j)}(x_i, x_j) = p_{(i,j)}(x_i, x_j),\ \forall\ (i,j) \in \mathcal{E}_{\widehat{p}}.$$

Marginal consistency of $\widehat{p}$ *w.r.t.* p:

$$\widehat{p}_{(i,j)}(x_i, x_j) = p_{(i,j)}(x_i, x_j), \ \forall \ (i,j) \in \mathcal{E}_{\widehat{p}}.$$

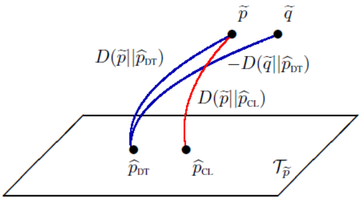
Benefits of tree-approx. J-divergence:

- Maximizing tree-approx. J-divergence gives good discriminative performance (shown experimentally).
- Marginal consistency leads to tractable optimization for $\widehat{p}$ and $\widehat{q}$.
- Trees provide rich class of distributions to model high-dimensional data.

Discriminative learning of trees

$\widetilde{p}$ and $\widetilde{q}$: empirical distributions from $\mathcal{T}_p$ and $\mathcal{T}_q$ respectively.

$$(\widehat{p}, \widehat{q}) = \arg \max_{\widehat{p} \in \mathcal{T}_{\widetilde{p}}, \widehat{q} \in \mathcal{T}_{\widetilde{q}}} \widehat{J}(\widehat{p}, \widehat{q}; \widetilde{p}, \widetilde{q}).$$



Decoupling into two independent MWST problems:

$$\begin{aligned} \widehat{p} &= \arg\min_{p \in \mathcal{T}_{\widetilde{p}}} D(\widetilde{p} \| p) - D(\widetilde{q} \| p) \\ \widehat{q} &= \arg\min_{q \in \mathcal{T}_{\widetilde{q}}} D(\widetilde{q} \| q) - D(\widetilde{p} \| q). \end{aligned}$$

Edge weights:

$$\psi_{i,j}^{p} := \mathbb{E}_{\widetilde{p}_{i,j}}\left[\log\frac{\widetilde{p}_{i,j}}{\widetilde{p}_i\widetilde{p}_j}\right] - \mathbb{E}_{\widetilde{q}_{i,j}}\left[\log\frac{\widetilde{p}_{i,j}}{\widetilde{p}_i\widetilde{p}_j}\right]$$
$$\psi_{i,j}^{q} := \mathbb{E}_{\widetilde{q}_{i,j}}\left[\log\frac{\widetilde{q}_{i,j}}{\widetilde{q}_i\widetilde{q}_j}\right] - \mathbb{E}_{\widetilde{p}_{i,j}}\left[\log\frac{\widetilde{q}_{i,j}}{\widetilde{q}_i\widetilde{q}_j}\right].$$

Algorithm 1 Discriminative trees (DT)

Given: Training sets $\mathcal{T}_p$ and $\mathcal{T}_q$.

1: Estimate pairwise statistics $\widetilde{p}_{i,j}(x_i, x_j)$, $\widetilde{q}_{i,j}(x_i, x_j)$ for all edges (i, j).
2: Compute edge weights $\psi_{i,j}^{p}$ and $\psi_{i,j}^{q}$ for all edges (i, j).
3: Find $\mathcal{E}_{\widehat{p}} =$ MWST$(\psi_{i,j}^{p})$ and $\mathcal{E}_{\widehat{q}} =$ MWST$(\psi_{i,j}^{q})$.
4: Get $\widehat{p}$ by projection of $\widetilde{p}$ onto $\mathcal{E}_{\widehat{p}}$; likewise $\widehat{q}$.
5: LRT using $\widehat{p}$ and $\widehat{q}$.

Discriminative learning of forests

$\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$: Markov on forests with at most $k \leq (n-1)$ edges.

Discriminative learning of forests

$\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$: Markov on forests with at most $k \leq (n-1)$ edges.

Maximize *joint objective* over both pairs of distributions:

$$(\widehat{p}^{(k)}, \widehat{q}^{(k)}) = \arg \max_{\widehat{p} \in \mathcal{T}_{\widetilde{p}^{(k)}}, \widehat{q} \in \mathcal{T}_{\widetilde{q}^{(k)}}} \widehat{J}(\widehat{p}, \widehat{q}; \widetilde{p}, \widetilde{q})$$

Discriminative learning of forests

$\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$: Markov on forests with at most $k \leq (n-1)$ edges.

Maximize *joint objective* over both pairs of distributions:

$$(\widehat{p}^{(k)}, \widehat{q}^{(k)}) = \arg \max_{\widehat{p} \in \mathcal{T}_{\widetilde{p}^{(k)}}, \widehat{q} \in \mathcal{T}_{\widetilde{q}^{(k)}}} \widehat{J}(\widehat{p}, \widehat{q}; \widetilde{p}, \widetilde{q})$$

Useful property: $\widehat{J}(\widehat{p}, \widehat{q}; p, q) = \sum_{i \in \mathcal{V}} J(p_i, q_i) + \sum_{(i,j) \in \mathcal{E}_{\widehat{p}} \cup \mathcal{E}_{\widehat{q}}} w_{ij},$

where w_{ij} can be expressed in terms of mutual information terms and KL-divergences involving marginal and pairwise statistics.

Discriminative learning of forests

$\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$: Markov on forests with at most $k \leq (n-1)$ edges.

Maximize *joint objective* over both pairs of distributions:

$$(\widehat{p}^{(k)}, \widehat{q}^{(k)}) = \arg \max_{\widehat{p} \in \mathcal{T}_{\widetilde{p}^{(k)}}, \widehat{q} \in \mathcal{T}_{\widetilde{q}^{(k)}}} \widehat{J}(\widehat{p}, \widehat{q}; \widetilde{p}, \widetilde{q})$$

Useful property: $\widehat{J}(\widehat{p}, \widehat{q}; p, q) = \sum_{i \in \mathcal{V}} J(p_i, q_i) + \sum_{(i,j) \in \mathcal{E}_{\widehat{p}} \cup \mathcal{E}_{\widehat{q}}} w_{ij}$,

where w_{ij} can be expressed in terms of mutual information terms and KL-divergences involving marginal and pairwise statistics.

- Additivity of cost function and optimality of k-step Kruskal MWST algorithm for each $k \Rightarrow k$-step Kruskal MWST leads to optimal forest-structured distribution.

Discriminative learning of forests

$\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$: Markov on forests with at most $k \leq (n-1)$ edges.

Maximize *joint objective* over both pairs of distributions:

$$(\widehat{p}^{(k)}, \widehat{q}^{(k)}) = \arg \max_{\widehat{p} \in \mathcal{T}_{\widetilde{p}^{(k)}}, \widehat{q} \in \mathcal{T}_{\widetilde{q}^{(k)}}} \widehat{J}(\widehat{p}, \widehat{q}; \widetilde{p}, \widetilde{q})$$

Useful property: $\widehat{J}(\widehat{p}, \widehat{q}; p, q) = \sum_{i \in \mathcal{V}} J(p_i, q_i) + \sum_{(i,j) \in \mathcal{E}_{\widehat{p}} \cup \mathcal{E}_{\widehat{q}}} w_{ij},$

where w_{ij} can be expressed in terms of mutual information terms and KL-divergences involving marginal and pairwise statistics.

- Additivity of cost function and optimality of k-step Kruskal MWST algorithm for each $k \Rightarrow k$-step Kruskal MWST leads to optimal forest-structured distribution.
- Estimated edges sets are nested, i.e., $\mathcal{T}_{\widetilde{p}^{(k-1)}} \subseteq \mathcal{T}_{\widetilde{p}^{(k)}}, \ \forall \ k \leq n-1$.

Discriminative learning of forests

$\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$: Markov on forests with at most $k \leq (n-1)$ edges.

Maximize *joint objective* over both pairs of distributions:

$$(\widehat{p}^{(k)}, \widehat{q}^{(k)}) = \arg \max_{\widehat{p} \in \mathcal{T}_{\widetilde{p}^{(k)}}, \widehat{q} \in \mathcal{T}_{\widetilde{q}^{(k)}}} \widehat{J}(\widehat{p}, \widehat{q}; \widetilde{p}, \widetilde{q})$$

Useful property: $\widehat{J}(\widehat{p}, \widehat{q}; p, q) = \sum_{i \in \mathcal{V}} J(p_i, q_i) + \sum_{(i,j) \in \mathcal{E}_{\widehat{p}} \cup \mathcal{E}_{\widehat{q}}} w_{ij},$

where w_{ij} can be expressed in terms of mutual information terms and KL-divergences involving marginal and pairwise statistics.

- Additivity of cost function and optimality of k-step Kruskal MWST algorithm for each $k \Rightarrow k$-step Kruskal MWST leads to optimal forest-structured distribution.
- Estimated edges sets are nested, i.e., $\mathcal{T}_{\widetilde{p}^{(k-1)}} \subseteq \mathcal{T}_{\widetilde{p}^{(k)}}, \ \forall \ k \leq n-1$.
- Single run of Kruskal MWST recovers all $(n-1)$ pairs of edge substructures!

Learning thicker graphs via boosting

- Trees learn $(n-1)$ edges $\rightarrow$ sparse representation.
- Desirable to learn more graph edges for better classification, if we can also avoid overfitting.
- Learning of junction trees known to be NP-hard.
- Learning general graph structures is intractable.

Learning thicker graphs via boosting

- Trees learn $(n-1)$ edges $\rightarrow$ sparse representation.
- Desirable to learn more graph edges for better classification, if we can also avoid overfitting.
- Learning of junction trees known to be NP-hard.
- Learning general graph structures is intractable.
- Use boosting to learn more than $(n-1)$ edges per model.

Boosted graphical model classification

- DT classifier used as a *weak learner*
- Training sets $\mathcal{T}_p$ and $\mathcal{T}_q$ remain unchanged

Boosted graphical model classification

- DT classifier used as a *weak learner*
- Training sets $\mathcal{T}_p$ and $\mathcal{T}_q$ remain unchanged
- In t-th iteration, learn trees $\widehat{p}_t$ and $\widehat{q}_t$, and classify using:

$$h_t(\mathbf{x}) := \log\left(\frac{\widehat{p}_t(\mathbf{x})}{\widehat{q}_t(\mathbf{x})}\right).$$

- Learn trees by minimizing weighted training error: use $(\widetilde{p}_w, \widetilde{q}_w)$ instead of $(\widetilde{p}, \widetilde{q})$.

Boosted graphical model classification

- DT classifier used as a *weak learner*
- Training sets $\mathcal{T}_p$ and $\mathcal{T}_q$ remain unchanged
- In t-th iteration, learn trees $\widehat{p}_t$ and $\widehat{q}_t$, and classify using:

$$h_t(\mathbf{x}) := \log\left(\frac{\widehat{p}_t(\mathbf{x})}{\widehat{q}_t(\mathbf{x})}\right).$$

- Learn trees by minimizing weighted training error: use $(\widetilde{p}_w, \widetilde{q}_w)$ instead of $(\widetilde{p}, \widetilde{q})$.
- Final boosted classifier:

$$\begin{aligned} H_T(\mathbf{x}) &= \operatorname{sgn}\left[\sum_{t=1}^{T} \alpha_t \log\left(\frac{\widehat{p}_t(\mathbf{x})}{\widehat{q}_t(\mathbf{x})}\right)\right] \\ &= \operatorname{sgn}\left[\log\left(\frac{\widehat{p}^*(\mathbf{x})}{\widehat{q}^*(\mathbf{x})}\right)\right], \end{aligned}$$

where $\widehat{p}^*(\mathbf{x}) := \prod_{t=1}^{T} \widehat{p}_t(\mathbf{x})^{\alpha_t}$.

Some comments

- Boosting learns at most $(n-1)$ edges per iteration $\Rightarrow$ maximum of $(n-1)T$ edges (pairwise features)
- With suitable normalization, $\widehat{p}^*(\mathbf{x})/Z_p(\alpha)$ is a probability distribution.
- $\widehat{p}^*(\mathbf{x})/Z_p(\alpha)$ is Markov on a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}_{\widehat{p}^*})$ with edge set

$$\mathcal{E}_{\widehat{p}^*} = \bigcup_{t=1}^{T} \mathcal{E}_{\widehat{p}_t}.$$

- How to avoid overfitting?

  Use cross-validation to determine optimum number of iterations T^*.

Extension to multi-class problems

- Set of classes $\mathcal{I}$: one-versus-all strategy.
- $\widehat{p}_{i|j}^{(k)}(\mathbf{x})$ and $\widehat{p}_{j|i}^{(k)}(\mathbf{x})$ - learned forests for the binary classification problem Class i versus Class j.

$$f_{ij}^{(k)}(\mathbf{x}) := \log \left[\frac{\widehat{p}_{i|j}^{(k)}(\mathbf{x})}{\widehat{p}_{j|i}^{(k)}(\mathbf{x})} \right], \ i, j \in \mathcal{I}.$$

- Multi-class decision function:

$$g^{(k)}(\mathbf{x}) := \arg\max_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} f_{ij}^{(k)}(\mathbf{x}).$$

Conclusion

- Discriminative learning optimizes an approximation to the expectation of log-likelihood ratio
- Superior performance in classification applications compared to generative approaches.
- Learned tree models can have *different* edge structures $\rightarrow$ removes the restriction of Tree Augmented Naive (TAN) Bayes framework .
- No additional computational overhead compared to existing tree-based methods.
- Amenable to boosting $\rightarrow$ weak learners on weighted empiricals.
- Learning thicker graphical models in a principled manner.