Learning graphical models for hypothesis testing and classification

Citation: Tan, Vincent Y. F. et al. “Learning Graphical Models for Hypothesis Testing and Classification.” IEEE Transactions on Signal Processing 58.11 (2010): 5481–5495. © Copyright 2010 IEEE

Published Version: http://dx.doi.org/10.1109/TSP.2010.2059019

Publisher: Institute of Electrical and Electronics Engineers (IEEE)

Permanent Link: http://hdl.handle.net/1721.1/73608

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

Learning Graphical Models for Hypothesis Testing and Classification

Vincent Y. F. Tan, *Student Member, IEEE*, Sujay Sanghavi, *Member, IEEE*, John W. Fisher, III, *Member, IEEE*, and Alan S. Willsky, *Fellow, IEEE*

***Abstract*—Sparse graphical models have proven to be a flexible class of multivariate probability models for approximating high-dimensional distributions. In this paper, we propose techniques to exploit this modeling ability for binary classification by discriminatively learning such models from labeled training data, i.e., using both positive and negative samples to optimize for the structures of the two models. We motivate why it is difficult to adapt existing generative methods, and propose an alternative method consisting of two parts. First, we develop a novel method to learn tree-structured graphical models which optimizes an approximation of the log-likelihood ratio. We also formulate a joint objective to learn a nested sequence of optimal forests-structured models. Second, we construct a classifier by using ideas from boosting to learn a set of discriminative trees. The final classifier can interpreted as a likelihood ratio test between two models with a larger set of pairwise features. We use cross-validation to determine the optimal number of edges in the final model. The algorithm presented in this paper also provides a method to identify a subset of the edges that are most salient for discrimination. Experiments show that the proposed procedure outperforms generative methods such as Tree Augmented Naïve Bayes and Chow-Liu as well as their boosted counterparts.**

***Index Terms*—Boosting, classification, graphical models, structure learning, tree distributions.**

I. Introduction

THE formalism of graphical models [3] (also called Markov random fields) involves representing the conditional independence relations of a set of random variables by a graph. This enables the use of efficient graph-based algorithms to perform large-scale statistical inference and learning. Sparse, but loopy, graphical models have proven to be a robust yet flexible class of probabilistic models in signal and image processing [4]. Learning such models from data is an important generic task. However, this task is complicated by the classic tradeoff between consistency and generalization. That is, graphs with too few edges have limited modeling capacity, while those with too many edges overfit the data.

A classic method developed by Chow and Liu [5] shows how to efficiently learn the optimal tree approximation of a multivariate probabilistic model. It was shown that only pairwise probabilistic relationships amongst the set of variables suffice to learn the model. Such relationships may be deduced by using standard estimation techniques given a set of samples. Consistency and convergence rates have also been studied [6], [7]. Several promising techniques have been proposed for learning *thicker* loopy models [8]–[11] (i.e., models containing more edges) for the purpose of *approximating* a distribution given independent and identically distributed (iid) samples drawn from that distribution. However, they are not straightforward to adapt for the purpose of learning models for *binary classification* (or binary hypothesis testing). As an example, for two distributions that are "close" to each other, separately modeling each by a sparse graphical model would likely "blur" the differences between the two. This is because the primary goal of modeling is to faithfully capture the entire behavior of a single distribution, and *not* to emphasize its most salient *differences* from another probability distribution. Our motivation is to retain the generalization power of sparse graphical models, while also developing a procedure that automatically identifies and emphasizes features that help to best discriminate between two distributions.

In this paper, we leverage the modeling flexibility of sparse graphical models for the task of classification: given labeled training data from two unknown distributions, we first describe how to build a pair of tree-structured graphical models to better *discriminate* between the two distributions. In addition, we also utilize boosting [12] to learn a richer (or larger) set of features[1] using the previously mentioned tree-learning algorithm as the weak classifier. This allows us to learn thicker graphical models, which to the best of our knowledge, has not been done before. Learning graphical models for classification has been previously proposed for tree-structured models such as Tree Augmented Naïve Bayes (TAN) [13], [14], and for more complex models using greedy heuristics [15].

We outline the main contributions of this paper in Section I-A and discuss related work in Section I-B. In Section II, we present

Manuscript received February 09, 2010; accepted July 08, 2010. Date of publication July 19, 2010; date of current version October 13, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Cedric Richard. This work was supported in part by the AFOSR through Grant FA9550-08-1-1080, by the MURI funded through an ARO Grant W911NF-06-1-0076, and by MURI through AFOSR Grant FA9550-06-1-0324. The work of V. Tan was supported by A*STAR, Singapore. The work of J. Fisher was partially supported by the Air Force Research Laboratory under Award No. FA8650-07-D-1220. The material in this paper was presented at the SSP Workshop, Madison, WI, August 2007, and at ICASSP, Las Vegas, NV, March 2008.

V. Y. F. Tan and A. S. Willsky are with the Stochastic Systems Group, Laboratory for Information and Decision Systems (LIDS), Massachusetts Institute of Technology (MIT), Cambridge, MA 02139 USA (e-mail: vtan@mit.edu; willsky@mit.edu).

S. Sanghavi is with the Electrical and Computer Engineering Department, University of Texas, Austin, TX 78712 US (e-mail: sanghavi@mail.utexas.edu).

J. W. Fisher III is with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology (MIT), Cambridge, MA 02139 USA (e-mail: fisher@csail.mit.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2010.2059019

[1]We use the generic term *features* to denote the marginal and pairwise class conditional distributions, i.e., $p_i(x_i), q_i(x_i)$ and $p_{i,j}(x_i, x_j), q_{i,j}(x_i, x_j)$.

some mathematical preliminaries. In Section III, we describe discriminative tree learning algorithms specifically tailored for the purpose of classification. This is followed by the presentation of a novel adaptation of boosting [16], [17] to learn a larger set of features in Section IV. In Section V, we present numerical experiments to validate the learning method presented in the paper and also demonstrate how the method can be naturally extended to multiclass classification problems. We conclude in Section VI by discussing the merits of the techniques presented.

A. Summary of Main Contributions

There are three main contributions in this paper. Firstly, it is known that decreasing functions of the J-divergence [a symmetric form of the Kullback-Leibler (KL) divergence] provide upper and lower bounds to the error probability [18]–[20]. Motivated by these bounds, we develop efficient algorithms to maximize a tree-based approximation to the J-divergence. We show that it is straightforward to adapt the generative tree-learning procedure of Chow and Liu [5] to a *discriminative*[2] objective related to the J-divergence over tree models. Secondly, we propose a boosting procedure [12] to learn a richer set of features, thus improving the modeling ability of the distributions $\widehat{p}$ and $\widehat{q}$. Finally, we demonstrate empirically that this family of algorithms lead to accurate classification on a wide range of datasets.

It is generally difficult to adapt existing techniques for learning loopy graphical models directly to the task of classification. This is because direct approaches typically involve first estimating the structure before estimating the parameters. The parameter estimation stage is usually intractable if the estimated structure is loopy. Our main contribution is thus the development of efficient learning algorithms for estimating tree-structured graphical models $\widehat{p}$ and $\widehat{q}$ for classification. We learn $\widehat{p}$ and $\widehat{q}$ which have distinct structures, with each chosen to be simultaneously "close to" one distribution and "far from" another, in a precise sense (Proposition 2). Furthermore, the selection of $\widehat{p}$ and $\widehat{q}$ can be decoupled into two independent max-weight spanning tree (MWST) problems; the cross-dependence on both positively and negatively labeled examples is captured by the edge weights of each MWST. We also show an equivalence between the objective we maximize to the empirical log-likelihood ratio for discrete-valued random variables (Proposition 4). An alternative algorithm, which is closely related to the above, casts the discriminative learning problem as a single MWST optimization problem (Proposition 5). Similar to the above procedure, direct optimization over the pair $(\widehat{p}, \widehat{q})$ leads to two sequences $\{(\widehat{p}^{(k)}, \widehat{q}^{(k)})\}_{1 \leq k \leq n-1}$ of forest-structured distributions of increasing number of edges (pairwise features).

In addition, we develop a systematic approach to learn a richer (or larger) set of features discriminatively using ideas from boosting to learn progressively thicker graphical model classifiers, i.e., models with more edges (Proposition 7). We do this by: (i) Modifying the basic discriminative tree-learning procedure to classify *weighted* training samples. (ii) Using the modification above as a weak classifier to learn multiple pairs of trees. (iii) Combining the resulting trees to learn a larger set of pairwise features.

The optimal number of boosting iterations and hence, the number of trees in the final ensemble models is found by cross-validation (CV) [21]. We note that even though the resulting models are high-dimensional, CV is effective because due to the lower-dimensional modeling requirements of classification as compared to, for example, structure modeling. We show, via experiments, that the method of boosted learning outperforms [5], [13], [14]. In fact, *any* graphical model learning procedure for classification, such as TAN, can be augmented by the boosting procedure presented to learn more *salient* pairwise features and thus to increase modeling capability and subsequent classification accuracy.

B. Related Work

There has been much work on learning sparse, but loopy, graphs purely for modeling purposes (e.g., in the papers [8]–[11] and references therein). A simple form of learning of graphical models for classification is the Naïve Bayes model, which corresponds to the graphs having no edges, a restrictive assumption. A comprehensive study of discriminative versus generative Naïve Bayes was done in Ng *et al.* [22]. Friedman *et al.* [14] and Wang and Wong [13] suggested an improvement to Naïve Bayes using a generative model known as TAN, a specific form of a graphical model geared towards classification. However, the models learned in these papers share the same structure and hence are more restrictive than the proposed discriminative algorithm, which learns trees with possibly distinct structures for each hypothesis.

More recently, Grossman and Domingos [15] improved on TAN by proposing an algorithm for choosing the structures by greedily maximizing the conditional log-likelihood (CLL) with a minimum description length (MDL) penalty while setting parameters by maximum-likelihood and obtained good classification results on benchmark datasets. However, estimating the model parameters via maximum-likelihood is complicated because the learned structures are loopy. Su and Zhang [23] suggested representing variable independencies by conditional probability tables (CPT) instead of the structures of graphical models. Boosting has been used in Rosset and Segal [24] for density estimation and learning Bayesian networks, but the objective was on modeling and not on classification. In Jing *et al.* [25], the authors suggested boosting the parameters of TANs. Our procedure uses boosting to optimize for both the structures and the parameters of the pair of discriminative tree models, thus enabling the learning of thicker structures.

II. Preliminaries and Notation

A. Binary Hypothesis Testing/Binary Classification

In this paper, we restrict ourselves to the *binary hypothesis testing* or *binary classification* problem. In the sequel, we will discuss extensions of the method to the more general M-ary classification problem. We are given a labeled training set $\mathcal{S} := \{(x^{(1)}, y^{(1)}), \ldots, (x^{(L)}, y^{(L)})\}$, where each training pair $(x^{(l)}, y^{(l)}) \in \mathcal{X}^n \times \{+1, -1\}$. Here, $\mathcal{X}$ may be a finite set (e.g., $\mathcal{X} = \{0, \ldots, r-1\}$) or an infinite set (e.g., $\mathcal{X} = \mathbb{R}$). Each $y^{(l)}$,

[2]In this paper, we adopt the term "discriminative" to denote the use of both the positively and negatively labeled training samples to learn the model $\widehat{p}$, the approximate model for the positively labeled samples (and similarly for $\widehat{q}$). This is different from "generative'' learning in which only the positively labeled samples are used to estimate $\widehat{p}$ (and similarly for $\widehat{q}$).

which can only take on one of two values, represents the *class label* of that sample. Each training pair $(x^{(l)}, y^{(l)})$ is drawn independently from some unknown joint distribution P_{XY}. In this paper, we adopt the following simplifying notation: $p(x) := P_{X|Y}(x|y = 1)$ and $q(x) := P_{X|Y}(x|y = -1)$ denote the class conditional distributions.[3] Also, we assume the prior probabilities for the label are uniform, i.e., $P_Y(y = +1) = P_Y(y = -1) = 1/2$. This is not a restrictive assumption and we make it to lighten the notation in the sequel.

Given $\mathcal{S}$, we wish to train a model so as to classify, i.e., to assign a label of $+1$ or -1 to a new sample x. This sample is drawn according to the unknown distribution P_X, but its label is unavailable. If we do have access to the true conditional distributions p and q, the optimal test (under both the Neyman-Pearson and Bayesian settings [26, Ch. 11]) is known to be the log-likelihood ratio test given by

$$\log \varphi(x) \underset{\hat{y}=-1}{\overset{\hat{y}=+1}{\gtrless}} \eta \tag{1}$$

where the *likelihood ratio* $\varphi : \mathcal{X}^n \to \mathbb{R}^+$ is the ratio of the class-conditional distributions p and q, i.e.

$$\varphi(x) := \frac{p(x)}{q(x)}. \tag{2}$$

In (1), $\eta \in \mathbb{R}$ is the *threshold* of the test. In the absence of fully specified p and q, we will instead develop efficient algorithms for constructing approximations $\widehat{p}$ and $\widehat{q}$ from the set of samples $\mathcal{S}$ such that the following statistic [for approximating $\varphi(x)$] is as discriminative as possible.

$$\log \widehat{\varphi}(x) \underset{\hat{y}=-1}{\overset{\hat{y}=+1}{\gtrless}} \eta \tag{3}$$

where $\widehat{\varphi} : \mathcal{X}^n \to \mathbb{R}^+$ is an approximation of the likelihood ratio, defined as

$$\widehat{\varphi}(x) := \frac{\widehat{p}(x)}{\widehat{q}(x)}. \tag{4}$$

In (4), $\widehat{p}$ and $\widehat{q}$ are multivariate distributions (or graphical models) estimated *jointly* from both the positively and negatively labeled samples in the training set $\mathcal{S}$. We use the empirical distribution formed from samples in $\mathcal{S}$ to estimate $\widehat{p}$ and $\widehat{q}$.

B. Undirected Graphical Models

Undirected graphical models [3] can be viewed as generalizations of Markov chains to arbitrary undirected graphs. A graphical model over a random vector of n variables $x = (x_1, \ldots, x_n) \in \mathcal{X}^n$ specifies the factorization properties of the joint distribution of x. We say that the distribution $p(x)$ is *Markov* with respect to an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a vertex (or node) set $\mathcal{V} = \{1, \ldots, n\}$ and an edge set $\mathcal{E} \subset \binom{\mathcal{V}}{2}$ (where $\binom{\mathcal{V}}{2}$ represents the set of all unordered pairs of nodes) if the *local Markov property* holds, i.e.

$$p(x_i|x_{\mathcal{V}\setminus i}) = p(x_i|x_{\mathcal{N}(i)}), \quad \forall\, i \in \mathcal{V} \tag{5}$$

where the set of neighbors of node i is denoted as $\mathcal{N}(i) := \{j \in \mathcal{V} : (i, j) \in \mathcal{E}\}$ and $x_{\mathcal{A}} = \{x_i : i \in \mathcal{A}\}$ for any set $\mathcal{A} \subset \mathcal{V}$. Eqn. (5) states that the conditional distribution of variable $x_i \in \mathcal{X}$ on all the other variables is only dependent on the values its neighbors $x_{\mathcal{N}(i)}$ take on.

In this paper, we consider two important families of graphical models: the sets of *trees* and the set of k-edge *forests*, which we denote as $\mathcal{T}$ and $\mathcal{T}^{(k)}$ respectively.[4] A tree-structured probability distribution is one that is Markov on a (connected) tree-an undirected, acyclic graph with exactly $n - 1$ edges. A k-edge forest-structured distribution is one whose graph may not be connected (i.e., it contains $k < n - 1$ edges). Any tree- or forest-structured distribution $\widehat{p}$, Markov on $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, admits the following factorization property [3]:

$$\widehat{p}(x) = \prod_{i \in \mathcal{V}} \widehat{p}_i(x_i) \prod_{(i,j) \in \mathcal{E}} \frac{\widehat{p}_{i,j}(x_i, x_j)}{\widehat{p}_i(x_i)\widehat{p}_j(x_j)} \tag{6}$$

where $\widehat{p}_i(x_i)$ is the marginal of the random variable x_i and $\widehat{p}_{i,j}(x_i, x_j)$ is the pairwise marginal of the pair (x_i, x_j). Given some (non-tree) distribution p, and a tree or forest with fixed edge set $\mathcal{E}$, the *projection* $\widehat{p}$ of p onto this tree is given by

$$\widehat{p}(x) := \prod_{i \in \mathcal{V}} p_i(x_i) \prod_{(i,j) \in \mathcal{E}} \frac{p_{i,j}(x_i, x_j)}{p_i(x_i)p_j(x_j)}. \tag{7}$$

This implies that the marginals on $\mathcal{V}$ and pairwise marginals on $\mathcal{E}$ of the projection $\widehat{p}$ are the same as those of p. Finally, given a distribution p, we define the set of distributions that are the projection of p onto *some* tree $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as

$$\mathcal{T}_p := \left\{ \widehat{p}(x) = \prod_{i \in \mathcal{V}} p_i(x_i) \prod_{(i,j) \in \mathcal{E}} \frac{p_{i,j}(x_i, x_j)}{p_i(x_i)p_j(x_j)} : \mathcal{G} = (\mathcal{V}, \mathcal{E}) \text{ is a tree} \right\}. \tag{8}$$

To distinguish between forests and trees, we use the notation $\mathcal{E}^{(k)}$ to denote the edge set of a k-edge forest distribution and simply $\mathcal{E}$ [instead of $\mathcal{E}^{(n-1)}$] to denote a (connected) tree (with $n - 1$ edges).

C. The Chow-Liu Algorithm for Learning Tree Distributions

The Chow-Liu algorithm [5] provides a *generative* method to approximate a full joint distribution with one that is tree-structured. Recall that the KL-divergence [27] is given as $D(p_0 \| p_1) := \mathbb{E}_{p_0} \log(p_0/p_1)$ and is a natural measure of the separation between two probability distributions p_0 and p_1. Given any multivariate distribution p, the Chow-Liu algorithm considers the following optimization problem:

$$\min_{\widehat{p} \in \mathcal{T}} D(p \,\|\, \widehat{p}) \tag{9}$$

[3]Therefore if $\mathcal{X}$ is finite, p and q are probability mass functions. If $\mathcal{X} = \mathbb{R}$, then p and q are probability densities functions (wrt the Lebesgue measure).

[4]We will frequently abuse notation and say that $\mathcal{T}$ (and $\mathcal{T}^{(k)}$) are sets of tree (and forest) graphs as well as sets of tree-structured (and forest-structured) graphical models, which are probability distributions. The usage will be clear from the context.

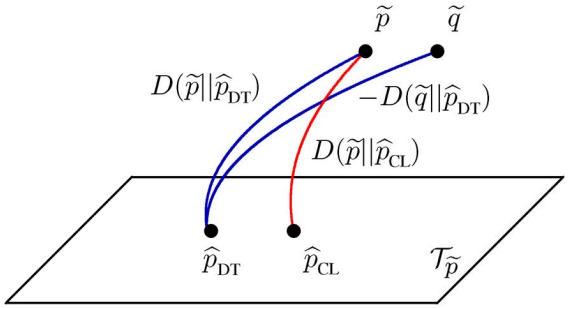


Fig. 1. Illustration of Proposition 2. As defined in (8), $\mathcal{T}_{\widetilde{p}}$ is the subset of tree distributions that are marginally consistent with $\widetilde{p}$, the empirical distribution of the positively labeled samples. $\widetilde{p}$ and $\widetilde{q}$ are not trees, thus $\widetilde{p}, \widetilde{q} \notin \mathcal{T}_{\widetilde{p}}$. The generatively learned distribution (via Chow-Liu) $\widehat{p}_{\rm CL}$, is the projection of $\widehat{p}$ onto $\mathcal{T}_{\widetilde{p}}$ as given by the optimization problem in (9). The discriminatively learned distribution $\widehat{p}_{\rm DT}$, is the solution of (20a) which is "further" (in the KL-divergence sense) from $\widetilde{q}$ (because of the $-D(\widetilde{q}||\widehat{p})$ term).

where recall that $\mathcal{T}$, understood to be over the same alphabet as p, is the set of tree-structured distributions. Thus, we seek to find a tree approximation for an arbitrary joint distribution p which is closest to p in the KL-divergence sense. See Fig. 1. Exploiting the fact that $\widehat{p}$ decomposes into its marginal and pairwise factors as in (6), Chow and Liu showed that the above optimization reduces to a MWST problem where the edge weights are given by the mutual information between pairs of variables. That is, the optimization problem in (9) reduces to

$$\max_{\mathcal{E}:\mathcal{G}=(\mathcal{V},\mathcal{E}) \text{ is a tree}} \sum_{(i,j)\in\mathcal{E}} I(x_i;x_j) \tag{10}$$

where $I(x_i;x_j) := D(p_{i,j}\|p_ip_j)$ is the *mutual information* between random variables x_i and x_j [26, Ch. 1] under the p model. It is useful to note that partial knowledge of p, specifically only the *marginal* and *pairwise* statistics [i.e., $p_i(x_i)$ and $p_{i,j}(x_i,x_j)$], is all that is required to implement Chow-Liu fitting. In the absence of exact statistics, these are estimated from the training data.

It is worth emphasizing that for Chow-Liu fitting (and also for discriminative trees in Section III), without loss of generality, we only consider learning *undirected* tree-structured graphical models (in contrast to directed ones in as [14]). This is because in the case of trees, a distribution that is Markov on an undirected graph can be converted to an equivalent distribution that is Markov on a directed graph (or Bayesian network) [3] by selecting an arbitrary node and directing all edges away from it. Similarly, directed trees can also be easily converted to undirected ones. Note also that there is no assumption on the true distributions p and q. They can be either characterized by either directed or undirected models.

D. The J-Divergence

The *J-divergence* between two probability distributions p and q is defined as [27]

$$J(p,q) := D(p\,\|\,q) + D(q\,\|\,p) \tag{11}$$

and is a fundamental measure of the separability of (or distance between) distributions. It has the property that $J = 0$ if and only if $p = q$ almost everywhere. In contrast to KL-divergence, J is symmetric in its arguments. However, it is still not a metric as it does not satisfy the triangle inequality. Nevertheless, the following useful upper and lower bounds on the probability of error [18]–[20] can be obtained from the J-divergence between two distributions:

$$\frac{1}{4}\exp(-J) \le \Pr(\text{err}) \le \frac{1}{2}\left(\frac{J}{4}\right)^{-1/4}. \tag{12}$$

Thus, maximizing J minimizes both upper and lower bounds on the Pr(err). Motivated by the fact that increasing the J-divergence decreases the upper and lower bounds in (12), we find $\widehat{\varphi}(x)$ in (4) by choosing graphical models $\widehat{p}$ and $\widehat{q}$ which maximize an approximation to the J-divergence.

III. Discriminative Learning of Trees and Forests

In this section, we propose efficient discriminative algorithms for learning two tree models by optimizing a surrogate statistic for J-divergence. We show that this is equivalent to optimizing the empirical log-likelihood ratio. We then discuss how to optimize the objective by using MWST-based algorithms. Before doing so, we define the following constraint on the parameters of the learned models.

Definition 1: The approximating distributions $\widehat{p}$ and $\widehat{q}$ are said to be *marginally consistent* with respect to the distributions p and q if their pairwise marginals on their respective edge sets $\mathcal{E}_{\widehat{p}}$ and $\mathcal{E}_{\widehat{q}}$ are equal, i.e., for the model $\widehat{p}$, we have

$$\widehat{p}_{i,j}(x_i,x_j) = p_{i,j}(x_i,x_j), \quad \forall (i,j) \in \mathcal{E}_{\widehat{p}}. \tag{13}$$

It follows from (13) that $\widehat{p}_i(x_i) = p_i(x_i)$ for all nodes $i \in \mathcal{V}$.

We will subsequently see that if $\widehat{p}$ and $\widehat{q}$ are marginally consistent, the optimization for the optimal structures of $\widehat{p}$ and $\widehat{q}$ is tractable. Now, one naïve choice of $\widehat{p}$ and $\widehat{q}$ to approximate the log-likelihood ratio is to construct generative tree or forest models of p and q from the samples, i.e., learn $\widehat{p} \in \mathcal{T}$ (or $\widehat{p} \in \mathcal{T}^{(k)}$) from the positively labeled samples and $\widehat{q}$ from the negatively labeled samples using the Chow-Liu method detailed in Section II-C. The set of generative models under consideration can be from the set of trees $\mathcal{T}$ or the set of k-edge forests $\mathcal{T}^{(k)}$. Kruskal's MWST algorithm [28] can be employed in either case. If we do have access to the true distributions, then this process is simply fitting lower-order tree (or forest) approximations to p and q. However, the true distributions p and q are usually not available. Motivated by Hoeffding and Wolfowitz [18] (who provide guarantees when optimizing the likelihood ratio test), and keeping in mind the final objective which is classification, we design $\widehat{p}$ and $\widehat{q}$ in a *discriminative* fashion to obtain $\widehat{\varphi}(x)$, defined in (4).

A. The Tree-Approximate J-Divergence Objective

We now formally define the approximation to the J-divergence, defined in (11).

Definition 2: The *tree-approximate J-divergence* of two tree-structured distributions $\widehat{p}$ and $\widehat{q}$ with respect to two arbitrary distributions p and q is defined as

$$\widehat{J}(\widehat{p},\widehat{q};p,q) := \int_{\Omega} (p(x)-q(x)) \log \left[\frac{\widehat{p}(x)}{\widehat{q}(x)}\right] dx \tag{14}$$

for distributions that are mutually absolutely continuous[5] and

$$\widehat{J}(\widehat{p},\widehat{q};p,q) := \sum_{x\in\mathcal{X}^n} (p(x)-q(x)) \log \left[\frac{\widehat{p}(x)}{\widehat{q}(x)}\right] \tag{15}$$

for discrete distributions.

Observe that the difference between J and $\widehat{J}$ is the replacement of the true distributions p and q by the approximate distributions $\widehat{p}$ and $\widehat{q}$ in the logarithm. As we see in Proposition 4, maximizing the tree-approximate J-divergence over $\widehat{p}$ and $\widehat{q}$ is equivalent to maximizing the *empirical log-likelihood ratio* if the random variables are discrete. Note however, that the objective in (14) does not necessarily share the properties of the true J-divergence in (12). The relationship between (14) and the J-divergence requires further theoretical analysis but this is beyond the scope of the paper. We demonstrate empirically that the maximization of the tree-approximate J-divergence results in good discriminative performance in Section V.

There are several other reasons for maximizing the tree-approximate J-divergence. First, trees have proven to be a rich class of distributions for modeling high-dimensional data [29]. Second, as we demonstrate in the sequel, we are able to develop efficient algorithms for learning marginally consistent $\widehat{p}$ and $\widehat{q}$. We now state a useful property of the tree-approximate J-divergence assuming $\widehat{p}$ and $\widehat{q}$ are trees.

Proposition 1: (Decomposition of the Tree-Approximate J-Divergence): Assume that: (i) the pairwise marginals $p_{i,j}$ and $q_{i,j}$ in (14) are mutually absolutely continuous; and (ii) $\widehat{p}$ and $\widehat{q}$ are tree distributions with edge sets $\mathcal{E}_{\widehat{p}}$ and $\mathcal{E}_{\widehat{q}}$ respectively and are also marginally consistent with p and q. Then the tree-approximate J-divergence can be expressed as a sum of marginal J divergences and weights

$$\widehat{J}(\widehat{p},\widehat{q};p,q) = \sum_{i\in\mathcal{V}} J(p_i,q_i) + \sum_{(i,j)\in\mathcal{E}_{\widehat{p}}\cup\mathcal{E}_{\widehat{q}}} w_{ij}. \tag{16}$$

The multivalued edge weights w_{ij} are given by

$$w_{ij} := \begin{cases} I_p(x_i;x_j) - I_q(x_i;x_j) & \\ \quad + D(q_{i,j}\|p_{i,j}) - D(q_iq_j\|p_ip_j) & (i,j)\in\mathcal{E}_{\widehat{p}}\setminus\mathcal{E}_{\widehat{q}} \\ I_q(x_i;x_j) - I_p(x_i;x_j) & \\ \quad + D(p_{i,j}\|q_{i,j}) - D(p_ip_j\|q_iq_j) & (i,j)\in\mathcal{E}_{\widehat{q}}\setminus\mathcal{E}_{\widehat{p}} \\ J(p_{i,j},q_{i,j}) - J(p_ip_j,q_iq_j) & (i,j)\in\mathcal{E}_{\widehat{p}}\cap\mathcal{E}_{\widehat{q}} \end{cases} \tag{17}$$

where $I_p(x_i;x_j)$ and $I_q(x_i;x_j)$ denote the mutual information quantities between random variables x_i and x_j under the p and q probability models, respectively.

[5]Two distributions p and q (for $p\neq q$) are *mutually absolutely continuous* if the corresponding *measures* ν_p and ν_q are absolutely continuous with respect to each other. The integral in (14) is understood to be over the domain in which the measures are equivalent $\Omega\subset\mathcal{X}^n$.

Proof: Since $\widehat{p}$ is a tree-structured distribution, it admits the factorization as in (6) with the node and pairwise marginals given by p (by marginal consistency). The distribution $\widehat{q}$ has a similar factorization. These factorizations can be substituted into (14) or (15) and the KL-divergences can then be expanded. Finally, by using the identities

$$\sum_{x\in\mathcal{X}^n} p(x)\log\left[\frac{p_i(x_i)}{q_i(x_i)}\right] = D(p_i\|q_i) \tag{18a}$$

$$\sum_{x\in\mathcal{X}^n} p(x)\log\left[\frac{p_{i,j}(x_i,x_j)}{q_{i,j}(x_i,x_j)}\right] = D(p_{i,j}\|q_{i,j}) \tag{18b}$$

and marginal consistency of $\widehat{p}$ and $\widehat{q}$, we can group terms together and obtain the result. ∎

Denote the empirical distributions of the positive and negatively labeled samples as $\widetilde{p}$ and $\widetilde{q}$ respectively. Given the definition of $\widehat{J}$ in (14), the optimization problem for finding approximate distributions $\widehat{p}$ and $\widehat{q}$ is formally formulated as

$$(\widehat{p},\widehat{q}) = \operatorname*{arg\,max}_{\widehat{p}\in\mathcal{T}_{\widetilde{p}},\widehat{q}\in\mathcal{T}_{\widetilde{q}}} \widehat{J}(\widehat{p},\widehat{q};\widetilde{p},\widetilde{q}) \tag{19}$$

where $\mathcal{T}_{\widetilde{p}}$ is the set of tree-structured distributions which are marginally consistent with $\widetilde{p}$. We will see that this optimization reduces to two tractable MWST problems. Furthermore, as in the Chow-Liu solution to the generative problem, only marginal and pairwise statistics need to be computed from the training set in order to estimate the information quantities in (17). In the sequel, we describe how to estimate these statistics and also how to devise efficient MWST algorithms to optimize (19) over the set of trees.

B. Learning Spanning Trees

In this section, we describe an efficient algorithm for learning two trees that optimize the tree-approximate J-divergence defined in (14). We assume that we have no access to the true distributions p and q. However, if the distributions are discrete, we can compute the empirical distributions $\widetilde{p}$ and $\widetilde{q}$ from the positively labeled and negatively labeled samples respectively. If the distributions are continuous and belong to a parametric family such as Gaussians, we can estimate the statistics such as means and covariances from the samples using maximum-likelihood fitting. For the purpose of optimizing (19), we only require the marginal and pairwise empirical statistics, i.e., the quantities $\widetilde{p}_i(x_i)$, $\widetilde{q}_i(x_i)$, $\widetilde{p}_{i,j}(x_i,x_j)$, and $\widetilde{q}_{i,j}(x_i,x_j)$. Estimating these pairwise quantities from the samples is substantially cheaper than computing the full empirical distribution or all the joint statistics. To optimize (19), we note that this objective can be rewritten as two independent optimization problems.

Proposition 2 (Decoupling of Objective Into Two MWSTs): The optimization in (19) decouples into

$$\widehat{p} = \operatorname*{arg\,min}_{p\in\mathcal{T}_{\widetilde{p}}} D(\widetilde{p}\,\|\,p) - D(\widetilde{q}\,\|\,p) \tag{20a}$$

$$\widehat{q} = \operatorname*{arg\,min}_{q\in\mathcal{T}_{\widetilde{q}}} D(\widetilde{q}\|\,q) - D(\widetilde{p}\,\|\,q). \tag{20b}$$

Proof: The equivalence of (19) and (20) can be shown by using the definition of the tree-approximate J-divergence and noting that $\sum_x \widetilde{p}(x)\log\widehat{p}(x) + H(\widetilde{p}) = -D(\widetilde{p}||\widehat{p})$. ∎

We have the following intuitive interpretation: the problem in (20a) is, in a precise sense, finding the distribution $\widehat{p}$ that is simultaneously "close to" the empirical distribution $\widetilde{p}$ and "far from" $\widetilde{q}$, while the reverse is true for $\widehat{q}$. See Fig. 1 for an illustration of the proposition. Note that all distances are measured using the KL-divergence. Each one of these problems can be solved by a MWST procedure with the appropriate edge weights given in the following proposition.

Proposition 3 (Edge Weights for Discriminative Trees): Assume that $\widehat{p}$ and $\widehat{q}$ are marginally consistent with $\widetilde{p}$ and $\widetilde{q}$ respectively as defined in (13). Then, for the selection of the edge set of $\widehat{p}$ in (20a), we can apply a MWST procedure with the *weights* on each pair of nodes $(i,j) \in \binom{\mathcal{V}}{2}$ are given by

$$\psi_{i,j}^{(+)} := \mathbb{E}_{\widetilde{p}_{i,j}}\left[\log\frac{\widetilde{p}_{i,j}}{\widetilde{p}_i\widetilde{p}_j}\right] - \mathbb{E}_{\widetilde{q}_{i,j}}\left[\log\frac{\widetilde{p}_{i,j}}{\widetilde{p}_i\widetilde{p}_j}\right]. \tag{21}$$

Proof: The proof can be found in Appendix A. ∎

From (21), we observe that *only* the marginal and pairwise statistics are needed in order to compute the edge weights. Subsequently, the MWST is used to obtain $\mathcal{E}_{\widehat{p}}$. Then, given this optimal tree structure, the model $\widehat{p}$ is the projection of $\widetilde{p}$ onto $\mathcal{E}_{\widehat{p}}$. A similar procedure yields $\widehat{q}$, with edge weights $\psi_{i,j}^{(-)}$ given by an expression similar to (21), but with $\widetilde{p}$ and $\widetilde{q}$ interchanged. The algorithm is summarized in Algorithm 1.

Algorithm 1 Discriminative Trees (DT)

Given: Training set $\mathcal{S}$.

1: Using the samples in $\mathcal{S}$, estimate the pairwise statistics $\widetilde{p}_{i,j}(x_i,x_j)$ and $\widetilde{q}_{i,j}(x_i,x_j)$ for all edges (i,j) using, for example, maximum-likelihood estimation.

2: Compute edge weights $\{\psi_{i,j}^{(+)}\}$ and $\{\psi_{i,j}^{(-)}\}$, using (21), for all edges (i,j).

3: Given the edge weights, find the optimal tree structures using a MWST algorithm such as Kruskal's [28], i.e., $\mathcal{E}_{\widehat{p}} = \mathrm{MWST}(\{\psi_{i,j}^{(+)}\})$, and $\mathcal{E}_{\widehat{q}} = \mathrm{MWST}(\{\psi_{i,j}^{(-)}\})$.

4: Set $\widehat{p}$ to be the projection of $\widetilde{p}$ onto $\mathcal{E}_{\widehat{p}}$ and $\widehat{q}$ to be the projection of $\widetilde{q}$ onto $\mathcal{E}_{\widehat{q}}$.

5: **return** Approximate distributions $\widehat{p}(x)$ and $\widehat{q}(x)$ to be used in a likelihood ratio test $h(x) = \mathrm{sgn}[\log(\widehat{p}(x)/\widehat{q}(x))]$ to assign a binary label to a test sample x.

This discriminative tree (DT) learning procedure produces at most $n-1$ edges (pairwise features) in each tree model $\widehat{p}$ and $\widehat{q}$ (some of the edge weights $\psi_{i,j}^{(+)}$ in (21) may turn out to be negative so the algorithm may terminate early). The tree models $\widehat{p}$ and $\widehat{q}$ will then be used to construct $\widehat{\varphi}$, which is used in the likelihood ratio test (3). Section V-B compares the classification performance of this method with other tree-based methods such as Chow-Liu as well as TAN [13], [14]. Finally, we remark that the proposed procedure has exactly the same complexity as learning a TAN network.

C. Connection to the Log-Likelihood Ratio

We now state a simple and intuitively-appealing result that relates the optimization of the tree-approximate J-divergence to the likelihood ratio test in (1).

Proposition 4 (Empirical Log-Likelihood Ratio): For discrete distributions, optimizing the tree-approximate J-divergence in (19) is equivalent to maximizing the *empirical log-likelihood ratio* of the training samples, i.e.

$$(\widehat{p},\widehat{q}) = \underset{\widehat{p}\in\mathcal{T}_{\widehat{p}},\widehat{q}\in\mathcal{T}_{\widehat{q}}}{\arg\max} \sum_{l=1}^{L} y^{(l)} \log\left[\frac{\widehat{p}(x^{(l)})}{\widehat{q}(x^{(l)})}\right]. \tag{22}$$

Proof: Partition the training set $\mathcal{S}$ into positively labeled samples $\mathcal{S}^+ := \{x^{(l)} : y^{(l)} = +1\}$ and negatively labeled samples $\mathcal{S}^- := \{x^{(l)} : y^{(l)} = -1\}$ and split the sum in (22) corresponding to these two parts accordingly. Then the sums (over the sets $\mathcal{S}^+$ and $\mathcal{S}^-$) are equal to (20a) and (20b), respectively. Finally use Proposition 2 to conclude that the optimizer of the empirical log-likelihood ratio is the same as the optimizer of the tree-approximate J-divergence. ∎

This equivalent objective function has a very intuitive meaning. Once $\widehat{p}$ and $\widehat{q}$ have been learned, we would like $\log\widehat{\varphi}(x^{(l)}) := \log[\widehat{p}(x^{(l)})/\widehat{q}(x^{(l)})]$ to be positive (and as large as possible) for all samples with label $y^{(l)} = +1$, and negative (with large magnitude) for those with label $y^{(l)} = -1$. The objective function in (22) precisely achieves this purpose.

It is important to note that (19) involves maximizing the tree-approximate J-divergence. This does not mean that we are directly minimizing the probability of error. In fact, we would not expect convergence to the true distributions p and q when the number of samples tends to infinity if we optimize the discriminative criterion (20).[6] However, since we are explicitly optimizing the log-likelihood ratio in (22), we would expect that if one has a limited number of training samples, we will learn distributions $\widehat{p}$ and $\widehat{q}$ that are better at discrimination than generative models in the likelihood ratio test (3). This can be seen in the objective function in (20a) which is a *blend* of two terms. In the first term $D(\widetilde{p}||p)$, we favor a model $\widehat{p}$ that minimizes the KL-divergence to its empirical distribution $\widetilde{p}$. In the second term $D(\widetilde{p}||q)$, we favor the maximization of the empirical *type-II error exponent* $D(\widetilde{q}||p)$ for testing p against the alternative distribution q (the Chernoff-Stein Lemma [26, Ch. 12]).

D. Learning Optimal Forests

In this subsection, we mention how the objective in (19), can be jointly maximized over pairs of *forest* distributions $\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$. Both $\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$ are Markov on forests with at most $k \leq n-1$ edges. This formulation is important since if we are given a fixed budget of only k edges per distribution, we would like to maximize the *joint* objective over *both* pairs of

[6]However, if the true distributions are tree-structured, minimizing the KL-divergence over the set of trees as in (9) is a maximum-likelihood procedure. It consistently recovers the structure of the true distribution p exponentially fast in n [6], [7].

distributions instead of decomposing the objective into two independent problems as in (20). This formulation also provides us with a natural way to incorporate costs for the selection of edges.

We use that notation $\mathcal{T}_{\widetilde{p}}^{(k)}$ to denote the set of probability distributions that are Markov on forests with *at most* k edges and have the same node and edge marginals as $\widetilde{p}$, i.e., marginally consistent with the empirical distribution $\widetilde{p}$. We now reformulate (19) as a joint optimization over the class of forests with at most k edges given empiricals $\widetilde{p}$ and $\widetilde{q}$

$$\left(\widehat{p}^{(k)}, \widehat{q}^{(k)}\right) = \underset{\widehat{p}\in\mathcal{T}_{\widetilde{p}}^{(k)}, \widehat{q}\in\mathcal{T}_{\widetilde{q}}^{(k)}}{\arg\max} \widehat{J}(\widehat{p}, \widehat{q}; \widetilde{p}, \widetilde{q}). \tag{23}$$

For each k, the resulting distributions $\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$ are optimal with respect to the tree-approximate J-divergence and the final pair of distributions $\widehat{p}^{(n-1)}$ and $\widehat{q}^{(n-1)}$ corresponds exactly to $\widehat{p}$ and $\widehat{q}$, the outputs of the DT algorithm as detailed in Algorithm 1. However, we emphasize that $\widehat{p}^{(k)}, \widehat{q}^{(k)}$ (for $k < n-1$) will, in general, be different from the outputs of the DT algorithm (with at most k edges chosen for each model) because (23) is a *joint* objective over forests. Furthermore, each forest has *at most* k edges but could have fewer depending on the sign of the weights in (17). The number of edges in each forest may also be different. We now show that the objective in (23) can be optimized easily with a slight modification of the basic Kruskal's MWST algorithm [28].

We note the close similarity between the discriminative objective in (16) and the Chow-Liu optimization for a single spanning tree in (10). In the former, the edge weights are given by w_{ij} in (17) and in the latter, the edge weights are the mutual information quantities $I(x_i; x_j)$. Note that the two objective functions are *additive*. With this observation, it is clear that we can equivalently choose to maximize the second term in (16), i.e., $W := \sum_{(i,j)\in\mathcal{E}_{\widehat{p}}\cup\mathcal{E}_{\widehat{q}}} w_{ij}$, over the set of trees, where each w_{ij} is a function of the empirical pairwise statistics $\widetilde{p}_{i,j}(x_i, x_j)$ and $\widetilde{q}_{i,j}(x_i, x_j)$ (and corresponding information-theoretic measures) that can be estimated from the training data. To maximize the sum W, we use the same MWST algorithm with edge weights given by w_{ij}. In this case, we must consider the maximum of the three possible values for w_{ij}. Whichever is the maximum (or if all three are negative) indicates one of four possible actions:

1) Place an edge between i and j for $\widehat{p}$ and *not* $\widehat{q}$ (corresponding to $(i,j) \in \mathcal{E}_{\widehat{p}} \setminus \mathcal{E}_{\widehat{q}}$).
2) Place an edge between i and j for $\widehat{q}$ and *not* $\widehat{p}$ (corresponding to $(i,j) \in \mathcal{E}_{\widehat{q}} \setminus \mathcal{E}_{\widehat{p}}$).
3) Place an edge between i and j for *both* $\widehat{p}$ and $\widehat{q}$ (corresponding to $(i,j) \in \mathcal{E}_{\widehat{p}} \cap \mathcal{E}_{\widehat{q}}$).
4) Do not place an edge between i and j for $\widehat{p}$ or $\widehat{q}$ if all three values of w_{ij} in (17) are negative.

Proposition 5 (Optimality of Kruskal's Algorithm for Learning Forests): For the optimization problem in (23), the k-step Kruskal's MWST algorithm, considering the maximum over the three possible values of w_{ij} in (17) and the four actions above, results in optimal forest-structured distributions $\widehat{p}^{(k)}(x)$ and $\widehat{q}^{(k)}(x)$ with edge sets $\mathcal{E}_{\widehat{p}^{(k)}}$ and $\mathcal{E}_{\widehat{q}^{(k)}}$.

Proof: This follows from the additivity of the objective in (16) and the optimality of Kruskal's MWST algorithm [28] for each step k. See [30, Sec. 23.1] for the details. ∎

The k-step Kruskal's MWST algorithm is the usual Kruskal's algorithm terminated after at most $k \leq n-1$ edges have been added. The edge sets are nested and we state this formally as a corollary of Proposition 5.

Corollary 6 (Nesting of Estimated Edge Sets): The edge sets $\mathcal{E}_{\widehat{p}^{(k)}}$ obtained from the maximization in (23) are nested, i.e., $\mathcal{E}_{\widehat{p}^{(k-1)}} \subseteq \mathcal{E}_{\widehat{p}^{(k)}}$ for all $k \leq n-1$ and similarly for $\mathcal{E}_{\widehat{q}^{(k)}}$. This appealing property ensures that *one* single run of Kruskal's MWST algorithm recovers *all* $n-1$ pairs of substructures $\{(\widehat{p}^{(k)}, \widehat{q}^{(k)})\}_{1\leq k\leq n-1}$. Thus, this procedure is computationally efficient.

E. Assigning Costs to the Selection of Edges

In many applications, it is common to associate the selection of more features with higher costs. We now demonstrate that it is easy to incorporate this consideration into our optimization program in (23).

Suppose we have a set of costs $\mathcal{C} := \{c_{ij} \geq 0 : (i,j) \in \binom{\mathcal{V}}{2}\}$, where each element c_{ij} is the cost of selecting edge (i,j). For example, in the absence of any prior information, we may regard each of these costs c_{ij} as being equal to a constant $c \geq 0$. We would like to maximize optimize $\widehat{J}$, given in (23), over the two models $\widehat{p}$ and $\widehat{q}$ taking the costs of selection of edges into consideration. From Proposition 1, the new objective function can now be expressed as

$$\widehat{J}_{\mathcal{C}}(\widehat{p}, \widehat{q}; p, q) = \sum_{i\in\mathcal{V}} J(p_i, q_i) + \sum_{(i,j)\in\mathcal{E}_{\widehat{p}}\cup\mathcal{E}_{\widehat{q}}} \overline{w}_{ij} \tag{24}$$

where the cost-modified edge weights are defined as $\overline{w}_{ij} := w_{ij} - c_{ij}$. Thus, the costs c_{ij} appear only in the new edge weights $\overline{w}_{ij}$. We can perform the same greedy selection procedure with the new edge weights $\overline{w}_{ij}$ to obtain the "cost-adjusted" edge sets $\overline{\mathcal{E}}_{\widehat{p}^{(k)}}$ and $\overline{\mathcal{E}}_{\widehat{q}^{(k)}}$. Interestingly, this also gives a natural stopping criterion. Indeed, whenever all the remaining $\overline{w}_{ij}$ are negative the algorithm should terminate as the overall cost will not improve.

IV. Learning a Larger Set of Features via Boosting

We have described efficient algorithms to learn tree distributions discriminatively by maximizing the empirical log-likelihood ratio in (22) (or the tree-approximate J-divergence). However, learning a larger set of features (more than $n-1$ edges per model) would enable better classification in general if we are *also* able to prevent overfitting. In light of the previous section, the first natural idea for learning thicker graphical models (i.e., graphical models with more edges) is to attempt to optimize an expression like (19), but over a set of thicker graphical models, e.g., the set of graphical models with bounded treewidth. However, this approach is complicated because the graph selection problem was simplified for trees as it was possible to determine *a-priori*, using (8), the projection of the empirical distribution onto the learned structure. Such a projection also holds for the construction of junction trees, but maximum-likelihood structure learning is known to be NP-hard [31]. For graphs that are

not junction trees, computing the projection parameters *a priori* is, in general, intractable. Furthermore, the techniques proposed in [8]–[11] used to learn such graphs are tightly coupled to the generative task of approximating $\widetilde{p}$, and even for these it is not straightforward to learn parameters given the loopy structure.

A. Discrete-Adaboost and Real-Adaboost: A Review

In this paper, we get around the aforementioned problem by using a novel method based on boosting [12] to acquire a larger set of features. *Boosting* is a sequential learning technique designed for classification. Given a set of "weak" classifiers (or "base learners"), boosting provides a way to iteratively select and combine these into a "strong" (or "ensemble") classifier, one which has a much lower probability of error on the training samples. The set of weak classifiers is chosen as follows: at iteration 0, each training sample is given uniform weights $w_0^{(l)} = 1/L$. In each iteration t, a weak classifier $h_t : \mathcal{X}^n \rightarrow \{-1, +1\}$, a map from the feature space to one of two labels, is chosen to minimize the *weighted* training error (i.e., the total weight of all misclassified training samples). Then, the sample weights are updated, with weight shifted to misclassified samples. After T iterations, the boosting procedure outputs $\mathrm{sgn}[\sum_t \alpha_t h_t(x)]$, a weighted average of the weak classifiers, as its strong classifier and the *sign* function $\mathrm{sgn}(a) = 1$ if $a \geq 0$ and -1 otherwise. The coefficients α_t's are chosen to minimize the weighted training error [12]. This procedure is known in the literature as *Discrete-AdaBoost*.

Real-AdaBoost [16], [17] is a variant of the above algorithm for the case when it is possible to obtain real-valued *confidences* from the weak classifiers, i.e., if $h_t : \mathcal{X}^n \rightarrow \mathbb{R}$ [with more positive $h_t(x)$ signifying higher bias for positively labeled samples].[7] It has been observed empirically that Real-AdaBoost often performs better than its discrete counterpart [16], [17]. We found this behavior in our experiments also as will be reported in Section V-D. The strong classifier resulting from the Real-AdaBoost procedure is

$$H_T(x) = \mathrm{sgn}\left[\sum_{t=1}^{T} \alpha_t h_t(x)\right] \tag{25}$$

where the set of coefficients are given by $\{\alpha_t \geq 0\}_{t=1}^T$.

B. Learning a Larger Set of Pairwise Features by Combining Discriminative Trees and Boosting

In the language of Real-AdaBoost, the *tree-based classifiers* or the *forests-based classifiers* presented in Section III may be regarded as *weak classifiers* to be combined to form a stronger classifier. More specifically, each weak classifier $h_t : \mathcal{X}^n \rightarrow \mathbb{R}$ is given by the log-likelihood ratio $h_t(x) = \log(\widehat{\varphi}_t(x)) = \log[\widehat{p}_t(x)/\widehat{q}_t(x)]$, where $\widehat{p}_t$ and $\widehat{q}_t$ are the tree-structured graphical model classifiers learned at the tth boosting iteration. Running T boosting iterations, now allows us to learn a larger set of features and to obtain a better approximation of the likelihood ratio $\widehat{\varphi}(x)$ in (4). This is because the strong ensemble classifier H_T can be written as

$$\begin{aligned} H_T(x) &= \mathrm{sgn}\left[\sum_{t=1}^{T} \alpha_t \log\left(\frac{\widehat{p}_t(x)}{\widehat{q}_t(x)}\right)\right] && \text{(26a)} \\ &= \mathrm{sgn}\left[\log\left(\frac{\prod_{t=1}^{T} \widehat{p}_t(x)^{\alpha_t}}{\prod_{t=1}^{T} \widehat{q}_t(x)^{\alpha_t}}\right)\right] && \text{(26b)} \\ &= \mathrm{sgn}\left[\log\left(\frac{\widehat{p}^*(x)}{\widehat{q}^*(x)}\right)\right]. && \text{(26c)} \end{aligned}$$

In (26c), $\widehat{p}^*(x)$, an unnormalized distribution, is of the form

$$\widehat{p}^*(x) := \prod_{t=1}^{T} \widehat{p}_t(x)^{\alpha_t}. \tag{27}$$

Define $Z_p(\alpha) = Z_p(\alpha_1, \ldots, \alpha_T) = \sum_x \widehat{p}^*(x)$ to be the normalizing constant for $\widehat{p}^*$ in (27). Hence the distribution (or graphical model) $\widehat{p}^*(x)/Z_p(\alpha)$ sums/integrates to unity.

Proposition 7 (Markovianity of Normalized Distributions): The normalized distribution $\widehat{p}^*(x)/Z_p(\alpha)$ is Markov on a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}_{\widehat{p}^*})$ with edge set

$$\mathcal{E}_{\widehat{p}^*} = \bigcup_{t=1}^{T} \mathcal{E}_{\widehat{p}_t}. \tag{28}$$

The same relation in (28) holds for $\widehat{q}^*(x)/Z_q(\alpha)$.

Proof: (sketch): This follows by writing each $\widehat{p}_t$ as a member of an exponential family, combining $\widehat{p}_t$'s to give $\widehat{p}^*$ as in (27) and finally applying the Hammersley-Clifford Theorem [32]. See Appendix B for the details. ■

Because we are entirely concerned with accurate classification, and the value of the ratio $\widehat{\varphi}^*(x) = \widehat{p}^*(x)/\widehat{q}^*(x)$ in (26c), we do not need to normalize our models $\widehat{p}^*$ and $\widehat{q}^*$. By leaving the models unnormalized, we retain the many appealing theoretical guarantees [12] afforded by the boosting procedure, such as the exponential decay in the training error. Furthermore, we are able to interpret the resulting *normalized* models[8] as being Markov on particular loopy graphs (whose edge sets are given in Proposition 7), which contain a larger set of features as compared to simple tree models.

Note that after T boosting iterations, we have a maximum of $(n-1)T$ pairwise features in each model as each boosting iteration produces at most $n-1$ pairwise features. To learn these features, we now need to learn tree models to minimize the *weighted* training error, as opposed to unweighted error as in Section III. This can be achieved by replacing the empirical distributions $\widetilde{p}$, $\widetilde{q}$ with the weighted empirical distributions $\widetilde{p}_w$, $\widetilde{q}_w$ and the weights are updated based on whether each sample $x^{(l)}$ is classified correctly. The resulting tree models will thus be projections of the weighted empirical distributions onto the corresponding learned tree structures. The method for learning a larger set of features from component tree models is summarized in Algorithm 2. Note that Algorithm 2 is essentially

[7]For instance, if the weak classifier is chosen to be the logistic regression classifier, then the confidences are the probabilistic outputs $p(y|x)$.

[8]We emphasize that the unnormalized models $\widehat{p}^*$ and $\widehat{q}^*$ are not probability distributions and thus cannot be interpreted as *graphical models*. However, the discriminative tree models learned in Section III are indeed normalized and hence are graphical models.

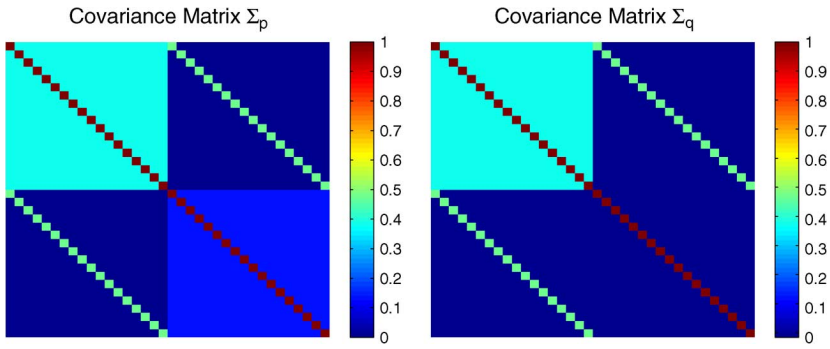


Fig. 2. The class covariance matrices Σ_p and Σ_q as described in Section V-A. The only discriminative information arises from the lower-right block.

a restatement of Real-Adaboost but with the weak classifiers learned using Discriminative Trees (Algorithm 1).

Algorithm 2 Boosted Graphical Model Classifiers (BGMC)

Given: Training data $\mathcal{S}$. Number of boosting iterations T.

1: Initialize the weights to be uniform, i.e., set $w_0^{(l)} = 1/L$ for all $1 \leq l \leq L$.

2: **for** $t = 1 : T$ **do**

3: Find discriminative trees $\widehat{p}_t, \widehat{q}_t$ using Algorithm 1, but with the *weighted* empirical distributions $\widetilde{p}_w, \widetilde{q}_w$.

4: The weak classifier $h_t : \mathcal{X}^n \rightarrow \mathbb{R}$ is given by $h_t(x) = \log[\widehat{p}_t(x)/\widehat{q}_t(x)]$.

5: Perform a convex line search to find the optimal value of the coefficients α_t

$$\alpha_t = \arg\min_{\beta \geq 0} \sum_{l=1}^{L} w_t^{(l)} \exp\left[-\beta y^{(l)} h_t\left(x^{(l)}\right)\right].$$

6: Update and normalize the weights:

$$w_{t+1}^{(l)} = \frac{w_t^{(l)}}{\zeta_t} \exp\left[-\alpha_t y^{(l)} h_t\left(x^{(l)}\right)\right], \quad \forall l = 1, \ldots, L$$

where $\zeta_t := \sum_{l=1}^{L} w_t^{(l)} \exp[-\alpha_t y^{(l)} h_t(x^{(l)})]$ is the normalization constant to ensure that the weights sum to unity after the update.

7: **end for**

8: **return** Coefficients $\{\alpha_t\}_{t=1}^T$ and models $\{\widehat{p}_t, \widehat{q}_t\}_{t=1}^T$. The final classifier is given in (26).

V. Numerical Experiments

This section is devoted to an extensive set of numerical experiments that illustrate the classification accuracy of discriminative trees and forests, as well as thicker graphical models. It is subdivided into the following subsections.

1) First, in Section V-A, we present an illustrate example to show that our discriminative tree/forest learning procedure as detailed in Sections III-B and D results in effective tree-based classifiers.
2) Second, in Section V-B we compare our discriminative trees procedure to other tree-based classifiers using real datasets. We also extend our ideas naturally to multiclass classification problems.
3) Finally, in Section V-D, we demonstrate empirically on a range of datasets that our method to learn thicker models outperforms standard classification techniques.

A. Discriminative Trees (DT): An Illustrative Example

We now construct two Gaussian graphical models p and q such that the real statistics are not trees and the maximum-likelihood trees (learned from Chow-Liu) are exactly the *same*, but the discriminative trees procedure gives distributions that are *different*. Let p and q be the probability density functions of two zero-mean n-variate (n even) Gaussian random vectors with class-conditional covariance matrices Σ_p and Σ_q, respectively, i.e., $p(x) = \mathcal{N}(x; 0, \Sigma_p) \propto \exp(-x^T \Sigma_p^{-1} x/2)$, where

$$\Sigma_p := \begin{bmatrix} \Sigma_{\mathrm{C}} & 0 \\ 0 & \Sigma_{\mathrm{A}} \end{bmatrix} + \Sigma_{\mathrm{N}}, \quad \Sigma_q := \begin{bmatrix} \Sigma_{\mathrm{C}} & 0 \\ 0 & \Sigma_{\mathrm{B}} \end{bmatrix} + \Sigma_{\mathrm{N}} \quad (29)$$

and the noise matrix is given as

$$\Sigma_{\mathrm{N}} := \begin{bmatrix} I_{n/2} & \rho I_{n/2} \\ \rho I_{n/2} & I_{n/2} \end{bmatrix}. \quad (30)$$

In (29), Σ_{C}, Σ_{A} and Σ_{B} are carefully selected $n/2 \times n/2$ positive definite matrices.

Note, from the construction, that the *only discriminative* information comes from the lower block terms in the class conditional covariance matrices as these are the only terms that differ between the two models. We set ρ to be the highest correlation coefficient of any off-diagonal element in Σ_p or Σ_q. This ensures that those edges are the first $n/2$ chosen in any Chow-Liu tree. These edges connect discriminative variables to non-discriminative variables. Next we design Σ_{C}, Σ_{A}, $\Sigma_{\mathrm{B}} \succ 0$ such that all of the correlation coefficient terms in the (common) upper block Σ_{C} are higher than any in Σ_{A} or Σ_{B}. This results in generative trees learned under Chow-Liu which provide no discriminative information. The additive noise term will not affect off-diagonal terms in either Σ_{A} or Σ_{B}. The two matrices Σ_p and Σ_q are shown in Fig. 2.

We now apply two structure learning methods (Chow-Liu [5] and the discriminative forest-learning method in Section III-D) to learn models $\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$ sequentially. For this toy example,

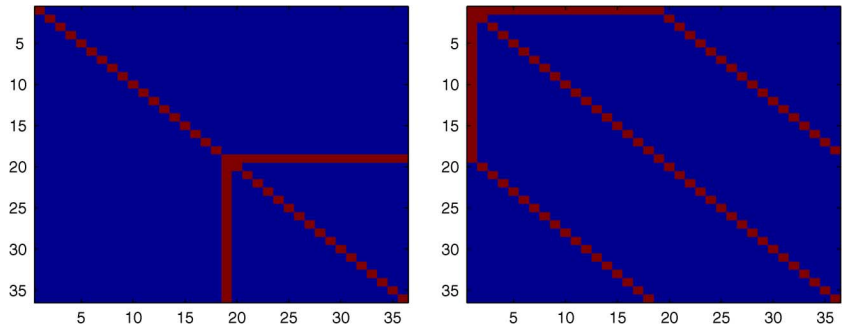


Fig. 3. Structures of $\widehat{p}^{(k)}$ at iteration $k = n-1$. The figures show the adjacency matrices of the graphs, where the edges selected at iteration $n-1$ are highlighted in red. In the left plot, we show the discriminative model, which extracts the edges corresponding to the discriminative block (lower-right corner) of the class conditional covariance matrix. In the right plot, we show the generative model, which does not extract the discriminative edges.

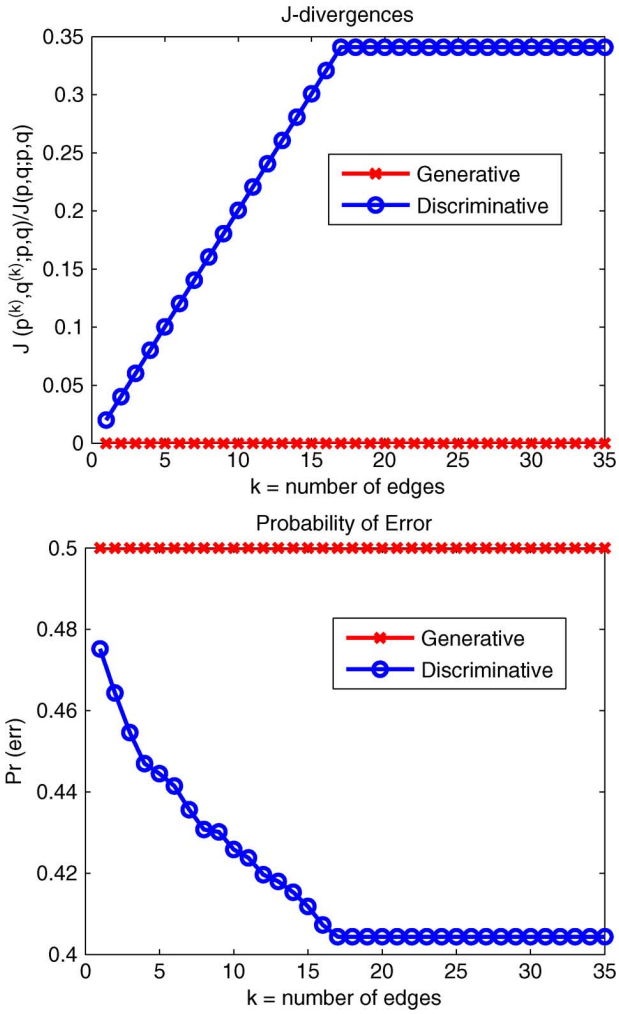


Fig. 4. Tree-approximate J-divergence and Pr(err). Note the monotonic increase of the tree-approximate J-divergence for the discriminative model. The generative model provides no discrimination as evidenced by the zero divergence and $\Pr(\text{err}) = 1/2$.

we assume that we have the true distributions. The learned structures are shown in Fig. 3. Note that, by construction, the discriminative algorithm terminates after $n/2$ steps since no more discriminative information can be gleaned without the addition of an edge that results in a loop. The generative structure is very different from the discriminative one. In fact, both the $\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$ structures are exactly the same for each k. This is further validated from Fig. 4, where we plot the tree-approximate J-divergence between $\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$ (relative to p and q) and the probability of error Pr(err) as a function of k. The Pr(err) is approximated using 10 000 test samples generated from the original distributions p and q. We see that the generative method provides no discrimination in this case, evidenced by the fact that the J-divergence is identically 0 and the Pr(err) is exactly 1/2. As expected, the J-divergence of the discriminative models increases monotonically and the Pr(err) decreases monotonically. Thus, this example clearly illustrates the differences between the generative [5] and discriminative learning algorithms. Clearly, it is advantageous to optimize the discriminative objective (23) if the purpose, namely binary classification, is known *a-priori*.

B. Comparison of DT to Other Tree-Based Classifiers

We now compare various tree-based graphical model classifiers, namely our proposed DT learning algorithm, Chow-Liu and finally TAN [14]. We perform the experiment on a quantized version of the MNIST handwritten digits dataset.[9] The results are averaged over 50 randomly partitioned training (80% of available data) and test sets (20%). The probability of error Pr(err) as a function of the number of training examples L is plotted in Fig. 5. We observe that in general our DT algorithm performs the best, especially in the absence of a large number of training examples. This makes good intuitive sense: With a limited number of training samples, a discriminative learning method, which captures the *salient differences* between the classes, should generalize better than a generative learning method, which models the distributions of the individual classes. Also, the computational complexities of DT and TAN are exactly the same.

C. Extension to Multiclass Problems

Next, we consider extending the sequential forest learning algorithm described in Section III-D to handle multiclass problems.[10] In multiclass problems, there are $M \geq 2$ classes, i.e., the class label Y described in Section II-A can take on more than 2 values. For example, we would like to determine which

[9]Each pixel with a non-zero value is quantized to 1.

[10]The DT algorithm can also be extended to multiclass problems in the same way.

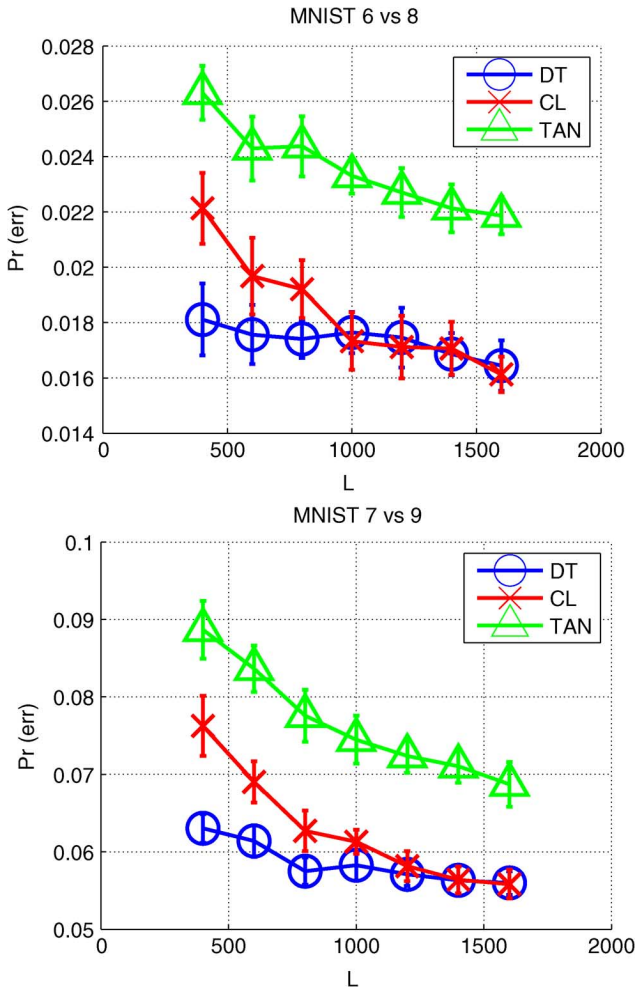


Fig. 5. Pr(err) between DT, Chow-Liu and TAN using a pair of trees. Error bars denote 1 standard deviation from the mean. If the total number of training samples L is small, then typically DT performs much better than Chow-Liu and TAN.

digit in the set $\mathcal{I} := \{0, 1, \ldots, 9\}$ a particular noisy image contains. For this experiment, we again use images from the MNIST database, which consists of $M = 10$ classes corresponding to the digits in the set $\mathcal{I}$. Since each of the $L = 60,000$ images in the database is of size 28 by 28, the dimensionality of the data is $n = 28 \times 28 = 784$. There is a separate test set containing 10 000 images, which we use to estimate the error probability. We preprocessed each image by concatenating the columns. We modeled each of the M classes by a multivariate Gaussian with length-n mean vector μ_i and positive definite covariance matrix Σ_i. To handle this multiclass classification problem, we used the well-known *one-versus-all* strategy described in Rifkin and Klautau [33] to classify the test images. We define $\widehat{p}_{i|j}^{(k)}(x)$ and $\widehat{p}_{j|i}^{(k)}(x)$ to be the learned forest distributions with at most k edges for the binary classification problem for digits i (positive class) and j (negative class), respectively. For each k, we also define the family of functions $f_{ij}^{(k)} : \mathcal{X}^n \to \mathbb{R}$ as

$$f_{ij}^{(k)}(x) := \log \left[\frac{\widehat{p}_{i|j}^{(k)}(x)}{\widehat{p}_{j|i}^{(k)}(x)} \right], \quad i, j \in \mathcal{I}. \tag{31}$$

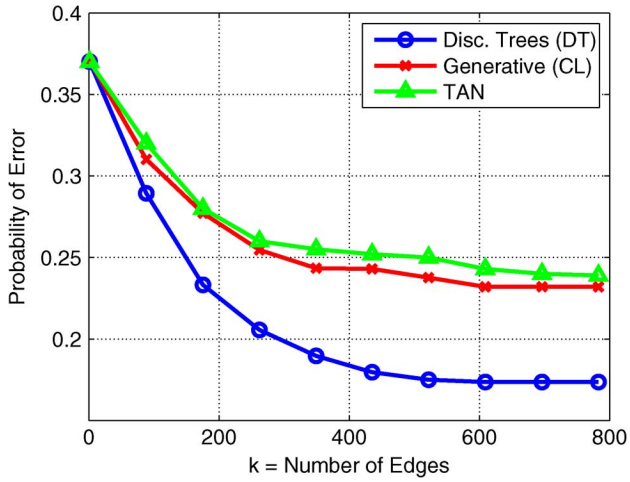


Fig. 6. Pr(err)'s for the MNIST Digits dataset for the multiclass problem with $M = 10$ classes (hypotheses). The horizontal axis is k, the number of edges added to each model $\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$. Note that the discriminative method outperforms the generative (Chow-Liu) method and TAN.

Thus, $\operatorname{sgn} f_{ij}^{(k)} : \mathcal{X}^n \to \{-1, +1\}$ is the classifier or decision function (for which both forests have no more than k edges) that discriminates between digits i and j. Note that $f_{ij}^{(k)}(x) = -f_{ji}^{(k)}(x)$. These distributions correspond to the $\widehat{p}^{(k)}$ and $\widehat{q}^{(k)}$ for the binary classification problem. The decision for the multiclass problem is then given by the composite decision function [33] $g^{(k)} : \mathcal{X}^d \to \mathcal{I}$, defined as

$$g^{(k)}(x) := \arg\max_{i \in \mathcal{I}} \sum_{j=0}^{M-1} f_{ij}^{(k)}(x). \tag{32}$$

The results of the experiment are shown in Fig. 6. We see that the discriminative method to learn the sequence of forests results in a lower Pr(err) (estimated using the test set) than the generative method for this dataset and TAN. This experiment again highlights the advantages of our proposed discriminative learning method detailed in Section III as compared to Chow-Liu trees [5] or TAN [14].

D. Comparison of Boosted Graphical Model Classifiers to Other Classifiers

In this section, we show empirically that our boosting procedure results in models that are better at classifying various datasets as compared to boosted versions of tree-based classifiers. Henceforth, we term our method, described in Section IV (and in detail in Algorithm 2) as Boosted Graphical Model Classifiers (BGMC).

In Fig. 7, we show the evolution of the training and test errors for discriminating between the digits 7 and 9 in the MNIST dataset as a function of T, the number of boosting iterations. We set the number of training samples $L = 500$. We compare the performance of four different methods: Chow-Liu learning with either Discrete-AdaBoost or Real-AdaBoost and Discriminative Trees with either Discrete-AdaBoost or Real-AdaBoost. We observe that the test error for Discriminative Trees+Real-AdaBoost, which was the method (BGMC) proposed in Section IV,

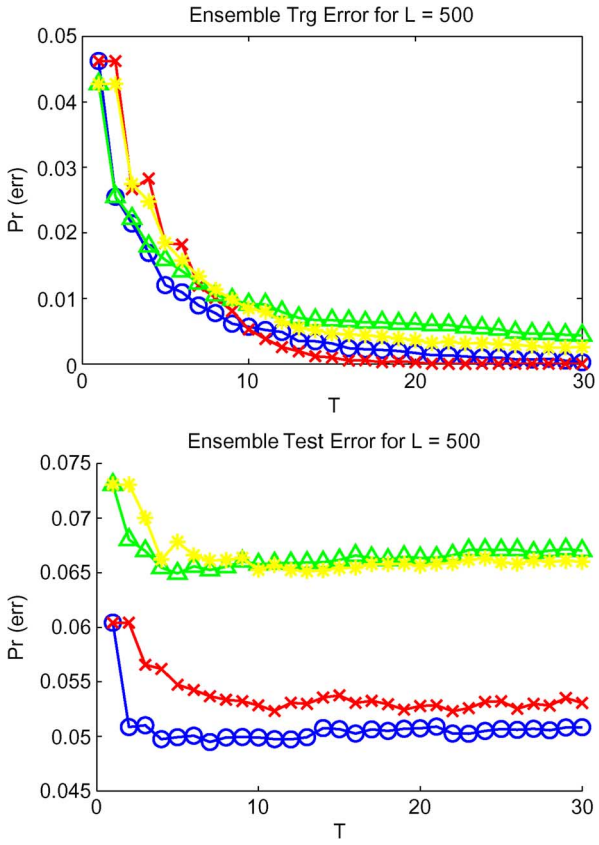


Fig. 7. Discrimination between the digits 7 and 9 in the MNIST dataset. T is the number of boosting iterations. Yellow $\Diamond$: (Chow-Liu + Discrete-AdaBoost), Green $\triangle$: (Chow-Liu + Real-AdaBoost), Red $\times$: Discriminative Trees + Discrete-AdaBoost, Blue $\circ$: Discriminative Trees + Real-AdaBoost (the proposed algorithm, BGMC). BGMC demonstrates lower training and test errors on this dataset. The training error decreases monotonically as expected. CV can be used to find the optimal number of boosting iterations to avoid overfitting. Observe from (b) that boosting (and in particular BGMC) is fairly robust to overfitting because even if T increases, the test error (also called generalization error) does not increase drastically.

is the minimum. Also, after a small number of boosting iterations, the test error does not decrease any further. Cross-validation (CV) [21] may thus be used to determine the optimal number of boosting iterations. We now compare **BGMC** to a variety of other classifiers:

1) **BCL**: A boosted version of the Chow-Liu algorithm [5] where a pair of trees is learned generatively, one for each class using the basic Chow-Liu algorithm. Note that only the positively (resp., negatively) labeled samples are used to estimate $\widehat{p}$ (resp. $\widehat{q}$). Subsequently, the trees are combined using the method detailed in Section IV.
2) **BTAN**: A boosted version of TAN [14]. Recall that TAN is such that two trees with the *same* structure are learned.
3) **SVM**: Support Vector Machines [34] using the quadratic kernel $K_2(x^{(a)}, x^{(b)}) = (1 + \langle x^{(a)}, x^{(b)} \rangle)^2$, with the slack parameter $C > 0$ found by CV.[11] We obtained the SVM code from [35].

For boosting, the optimal number of boosting iterations T^*, was also found by CV. For the set of experiments we performed, we found that T^* is typically small ($\approx$3–4); hence the resulting models remain sparse (Proposition 7).

1) Synthetic Dataset: We generated a dataset by assuming that p and q are Markov on $n = 10 \times 10$ binary grid models with different randomly chosen parameters. We generated $L = 1200$ samples to learn boosted discriminative trees. The purpose of this experiment was to compare the number of edges added to the models and the (known) number of edges in the original grid models. The original grid models each have $2 \times 9^2 = 162$ edges and the learned models have at most $(n-1)T^* = 99 \times 3 = 297$ edges since the CV procedure results in an optimal boosting iteration count of $T^* = 3$. However, some of the edges in $\widehat{p}_1, \widehat{p}_2, \widehat{p}_3$ (respectively, $\widehat{q}_1, \widehat{q}_2, \widehat{q}_3$) coincide and this results in $|\cup_{t=1}^{T^*} \mathcal{E}_{\widehat{p}_t}| = 180$ (respectively, $|\cup_{t=1}^{T^*} \mathcal{E}_{\widehat{q}_t}| = 187$). Thus, there are 180 and 187 *distinct* edges in the $\widehat{p}^*$ and $\widehat{q}^*$ models respectively. From the top left plot in Fig. 8, we see that CV is effective for the purpose of finding a good balance between optimizing modeling ability and preventing overfitting.

2) Real-World Datasets: We also obtained five different datasets from the UCI Machine Learning Repository [36] as well as the previously mentioned MNIST database. For datasets with continuous variables, the data values were quantized so that each variable only takes on a finite number of values. For datasets without separate training and test sets, we estimated the test error by averaging over 100 randomly partitioned training-test sets from the available data. The Pr(err) as a function of the number of training examples L is plotted in Fig. 8 for a variety of datasets. We observe that, apart from the Pendigits dataset, BGMC performs better than the other two (boosted) graphical model classifiers. Also, it compares well with SVM. In particular, for the synthetic, three MNIST, Optdigits and Chess datasets, the advantage of BGMC over the other tree-based methods is evident.

VI. Discussion and Conclusion

In this paper, we proposed a discriminative objective for the specific purpose of learning two tree-structured graphical models for classification. We observe that Discriminative Trees outperforms existing tree-based graphical model classifiers like TANs, especially in the absence of a large number of training examples. This is true for several reasons. First, our discriminative tree learning procedure is designed to optimize an approximation to the expectation of the log-likelihood ratio (22), while TAN is a generative procedure. Thus, if the intended purpose is known (e.g., in [37] the task was prediction), we can learn graphical models differently and often, more effectively for the task at hand. Second, we allowed the learned structures of the two models to be distinct, and each model is dependent on data with *both* labels. It is worth noting that the proposed discriminative tree learning procedure does not incur any computational overhead compared to existing tree-based methods.

We showed that the discriminative tree learning procedure can be adapted to the weighted case, and is thus amenable to use the models resulting from this procedure as weak classifiers for boosting to learn thicker models, which have better modeling ability. This is what allows us to circumvent the intractable

[11] We used 20% of the training samples to determine the best value of C.

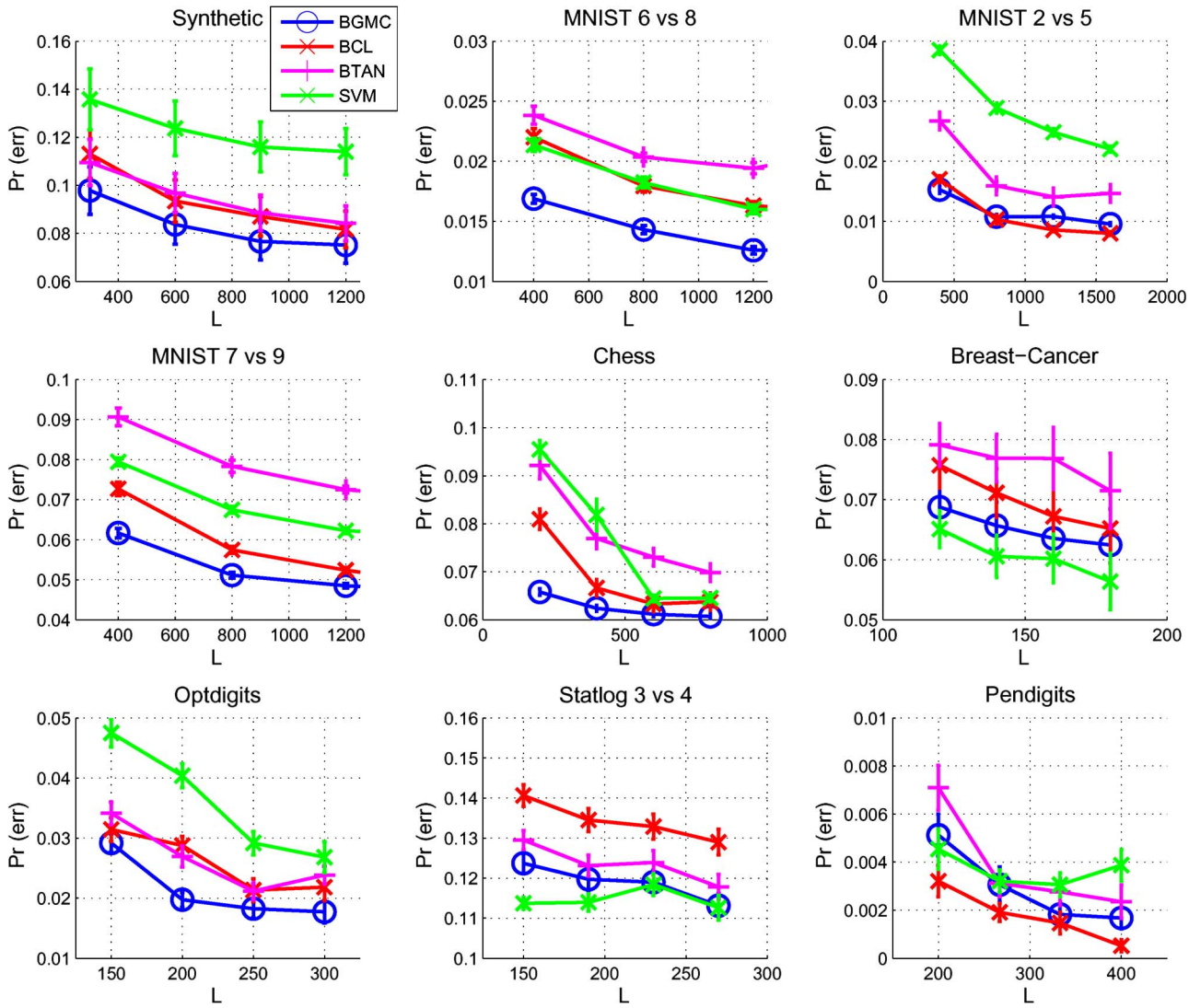


Fig. 8. Pr(err) against L, the number of training samples, for various datasets using Boosted Graphical Model Classifiers (BGMC, blue $\circ$), Boosted Chow-Liu (BCL, red $\times$), Boosted TAN (BTAN, magenta $+$) and SVM with quadratic kernel (green $\times$). In all cases, the performance of BGMC is superior to Boosted TAN.

problem of having to find the maximum-likelihood parameters of loopy graphical models.

In addition to learning two graphical models specifically for the purpose of discrimination, the proposed method also provides a principled approach to learn which pairwise features (or edges) are the most salient for classification (akin to the methods described in [38]). Our method for sequentially learning optimal forests serves precisely this purpose and also provides a natural way to incorporate costs of adding edges. Furthermore, to learn more edges than in a tree, we used boosting in a novel way to learn more complex models for the purpose of classification. Indeed, at the end of T boosting iterations, we can precisely characterize the set of edges for the normalized versions of the boosted models (Proposition 7). We can use these pairwise features, together with the marginal features, as inputs to *any* standard classification algorithm. Finally, our empirical results on a variety of synthetic and real datasets adequately demonstrate that the forests, trees and thicker models learned serve as good classifiers.

Appendix A
Proof of Proposition 3

Proof: We use $\stackrel{c}{=}$ to denote equality up to a constant. Also, to shorten notation, let $p(x_i) = p_i(x_i)$. Now, we can simplify the objective in the optimization problem in (20a), namely $D(\widetilde{p}||\widehat{p}) - D(\widetilde{q}||\widehat{p})$

$$\stackrel{c}{=} \sum_{x} (\widetilde{q}(x) - \widetilde{p}(x)) \log \left[\prod_{i \in \mathcal{V}} \widehat{p}(x_i) \prod_{(i,j) \in \mathcal{E}} \frac{\widehat{p}(x_i, x_j)}{\widehat{p}(x_i)\widehat{p}(x_j)} \right] \quad (33)$$

$$\stackrel{c}{=} \sum_{x} (\widetilde{q}(x) - \widetilde{p}(x)) \sum_{(i,j) \in \mathcal{E}} \log \left[\frac{\widetilde{p}(x_i, x_j)}{\widetilde{p}(x_i)\widetilde{p}(x_j)} \right] \quad (34)$$

$$= \sum_{(i,j) \in \mathcal{E}} \sum_{x_i, x_j} (\widetilde{q}(x_i, x_j) - \widetilde{p}(x_i, x_j)) \log \left[\frac{\widetilde{p}(x_i, x_j)}{\widetilde{p}(x_i)\widetilde{p}(x_j)} \right], \quad (35)$$

where (33) follows from the fact that $\widehat{p}$ is a tree [and hence factorizes as (6)] and (34) follows from marginal consistency and the fact that we are optimizing only over the edge set of $\widehat{p}$ and

thus the marginals can be dropped from the optimization. The final equality in (35), derived using (18a) and (18b), shows that we need to optimize over all tree structures with edge weights given by the expression in (21). ■

APPENDIX B
PROOF OF PROPOSITION 7

Proof: This result holds even when the $\widehat{p}_t$ are not trees, and the proof is straightforward. In general, a (everywhere nonzero) distribution p is Markov [3] with respect to some edge set $\mathcal{E}$ if and only if

$$\log p(x) \stackrel{c}{=} \sum_{(i,j)\in\mathcal{E}} \theta_{ij}\phi_{ij}(x_i,x_j) + \sum_{i\in\mathcal{V}} \theta_i\phi_i(x_i) \tag{36}$$

for some constants $\theta = \{\theta_{ij} : (i,j) \in \mathcal{E}\} \cup \{\theta_i : i \in \mathcal{V}\}$ and sufficient statistics $\phi = \{\phi_{ij} : (i,j) \in \mathcal{E}\} \cup \{\phi_i : i \in \mathcal{V}\}$. This means that each tree model $\widehat{p}_t$ can be written as

$$\log \widehat{p}_t(x) \stackrel{c}{=} \sum_{(i,j)\in\mathcal{E}_{\widehat{p}_t}} \theta^t_{ij}\phi^t_{ij}(x_i,x_j) + \sum_{i\in\mathcal{V}} \theta^t_i\phi^t_i(x_i). \tag{37}$$

Let $\mathcal{E} := \bigcup_{t=1}^{T} \mathcal{E}_{\widehat{p}_t}$ be the union of edge sets after T boosting iterations. Then $\log \widehat{p}_*(x)$ is equal (up to constants) to

$$\begin{aligned}
&\sum_t \alpha_t \left(\sum_{(i,j)\in\mathcal{E}_{\widehat{p}_t}} \theta^t_{ij}\phi^t_{ij}(x_i,x_j) + \sum_{i\in\mathcal{V}} \theta^t_i\phi^t_i(x_i) \right) \\
&\quad = \sum_{(i,j)\in\mathcal{E}} \left(\sum_t \alpha_t \theta^t_{ij}\phi^t_{ij}(x_i,x_j) \right) \\
&\qquad + \sum_{i\in\mathcal{V}} \left(\sum_t \alpha_t \theta^t_i\phi^t_i(x_i) \right),
\end{aligned} \tag{38}$$

where in we interpret the right hand side of the last equality as $\theta^t_{ij} = 0$ if and only if $(i,j) \notin \mathcal{E}_{\widehat{p}_t}$. This is seen to be of the same form as (36)—to see this, define the functions $\xi_{ij}(x_i,x_j) := \sum_t \alpha_t\theta^t_{ij}\phi^t_{ij}(x_i,x_j)$, and $\xi_i(x_i) := \sum_t \alpha_t\theta^t_i\phi^t_i(x_i)$, so that $\log \widehat{p}_*(x) \stackrel{c}{=} \sum_{(i,j)\in\mathcal{E}} \xi_{ij}(x_i,x_j) + \sum_{i\in\mathcal{V}} \xi_i(x_i)$. By the Hammersley-Clifford Theorem [32], we have proven the desired Markov property. ■

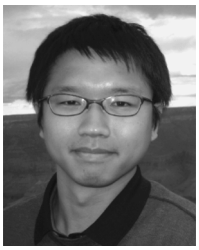
ACKNOWLEDGMENT

The authors would like to acknowledge Prof. M. Collins (CSAIL, MIT) for many helpful discussions on boosting. The authors also wish to express their gratitude to the anonymous reviewers, whose comments helped to improve the clarity of the exposition.

REFERENCES

[1] S. Sanghavi, V. Y. F. Tan, and A. S. Willsky, "Learning graphical models for hypothesis testing," in *Proc. 14th IEEE Statist. Signal Process. Workshop*, Aug. 2007, pp. 69–73.
[2] V. Y. F. Tan, J. W. Fisher, and A. S. Willsky, "Learning max-weight discriminative forests," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2008, pp. 1877–1880.
[3] S. Lauritzen, *Graphical Models*. Oxford, U.K.: Oxford Univ. Press, 1996.
[4] A. S. Willsky, "Multiresolution Markov models for signal and image processing," *Proc. IEEE*, vol. 90, no. 8, pp. 1396–1458, Aug. 2002.
[5] C. K. Chow and C. N. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE Trans. Inf. Theory*, vol. 14, no. 3, pp. 462–467, May 1968.
[6] V. Y. F. Tan, A. Anandkumar, L. Tong, and A. S. Willsky, "A large-deviation analysis for the maximum likelihood learning of tree structures," in *Proc. IEEE Int. Symp. Inf. Theory*, Seoul, Korea, Jul. 2009, pp. 1140–1144.
[7] V. Y. F. Tan, A. Anandkumar, and A. S. Willsky, "Learning gaussian tree models: Analysis of error exponents and extremal structures," *IEEE Trans. Signal Process.*, vol. 58, no. 5, pp. 2701–2714, May 2010.
[8] P. Abbeel, D. Koller, and A. Y. Ng, "Learning factor graphs in polynomial time and sample complexity," *J. Mach. Learn. Res.*, vol. 7, pp. 1743–1788, Dec. 2006.
[9] N. Meinshausen and P. Bühlmann, "High-dimensional graphs and variable selection with the Lasso," *Ann. Statist.*, vol. 34, no. 3, pp. 1436–1462, 2006.
[10] M. J. Wainwright, P. Ravikumar, and J. Lafferty, "High-dimensional graphical model selection using ℓ_1-regularized logistic regression," in *Proc. Neural Inf. Process. Syst.*, 2006.
[11] S. Lee, V. Ganapathi, and D. Koller, "Efficient structure learning of Markov networks using l_1-regularization," in *Proc. Neural Inf. Process. Syst.*, 2006.
[12] R. E. Schapire, "A brief introduction to boosting," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 1999.
[13] C. C. Wang and C. Wong, "Classification of discrete data with feature space transformation," *IEEE Trans. Autom. Control*, vol. AC-24, no. 3, pp. 434–437, Jun. 1979.
[14] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Mach. Learn.*, vol. 29, pp. 131–163, 1997.
[15] D. Grossman and P. Domingos, "Learning Bayesian network classifiers by maximizing conditional likelihood," in *Proc. Int. Conf. Mach. Learn.*, 2004.
[16] J. Friedman, T. Hastie, and R. Tibshirani, Additive Logistic Regression: A Statistical View of Boosting Dep. Statistics, Stanford Univ. Tech. Rep., Stanford, CA, 1998, Tech. Rep..
[17] R. E. Schapire and Y. Singer, "Improved boosting using confidence-rated predictions," *Mach. Learn.*, vol. 37, no. 3, pp. 297–336, 1999.
[18] W. Hoeffding and J. Wolfowitz, "Distinguishability of sets of distributions," *Ann. Math. Statist.*, vol. 29, no. 3, pp. 700–718, 1958.
[19] T. Kailath, "The divergence and bhattacharyya distance measures in signal selection," *IEEE Trans. Commun. Technol.*, vol. 15, no. 1, pp. 52–60, 1967.
[20] M. Basseville, "Distance measures for signal processing and pattern recognition," *Signal Process.*, vol. 18, no. 4, pp. 349–369, 1989.
[21] D. M. Allen, "The relationship between variable selection and data augmentation and a method for prediction," *Technometr.*, vol. 16, no. 1, pp. 125–127, Feb. 1974.
[22] A. Ng and M. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and Naïve Bayes," in *Proc. Neural Inf. Process. Syst.*, 2002.
[23] J. Su and H. Zhang, "Full Bayesian network classifiers," in *Proc. Int. Conf. Mach. Learn.*, 2006, pp. 897–904.
[24] S. Rosset and E. Segal, "Boosting density estimation," in *Proc. Neural Inf. Process. Syst.*, 2002, pp. 641–648.
[25] Y. Jing, V. Pavlovi, and J. M. Rehg, "Boosted Bayesian network classifiers," *Mach. Learn.*, vol. 73, no. 2, pp. 155–184, 2008.
[26] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. New York: Wiley-Intersci., 2006.
[27] S. Kullback, *Information Theory and Statistics*. New York: Wiley, 1959.
[28] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proc. Amer. Math. Soc.*, vol. 7, no. 1, pp. 48–50, 1956.
[29] F. Bach and M. I. Jordan, "Beyond independent components: Trees and clusters," *J. Mach. Learn. Res.*, vol. 4, pp. 1205–1233, 2003.
[30] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. New York: McGraw-Hill Sci./Eng./Math, 2003.
[31] D. Karger and N. Srebro, "Learning Markov networks: Maximum bounded tree-width graphs," in *Symp. Discr. Algorithms (SODA)*, 2001, pp. 392–401.
[32] J. M. Hammersley and M. S. Clifford, Markov fields on finite graphs and lattices 1970.
[33] R. Rifkin and A. Klautau, "In defense of one-vs-all classification," *J. Mach. Learn. Res.*, vol. 5, pp. 101–141, Nov. 2004.
[34] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer, 1999.

[35] S. Canu, Y. Grandvalet, V. Guigue, and A. Rakotomamonjy, "SVM and Kernel Methods Matlab Toolbox," in *Perception Systèmes et Information, INSA de Rouen*, Rouen, France, 2005.
[36] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz, *UCI Repository of Machine Learning Databases*. Irvine, CA: Univ. Calif., 1998.
[37] M. J. Wainwright, "Estimating the "Wrong" graphical model: Benefits in the computation-limited setting," *J. Mach. Learn. Res.*, vol. 7, no. 1829–1859, Dec. 2006.
[38] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, 2003.



Vincent Y. F. Tan (S'07) received the B.A. and M.Eng. degrees in electrical engineering from Sidney Sussex College, Cambridge University, Cambridge, U.K., in 2005.

He is currently pursuing the Ph.D. degree in electrical engineering and computer science in the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge. He was also a research intern with Microsoft Research in 2008 and 2009. His research interests include statistical signal processing, machine learning and information theory.

Mr. Tan received the Public Service Commission Scholarship in 2001 and the National Science Scholarship from the Agency for Science Technology and Research (A*STAR) in 2006. In 2005, he received the Charles Lamb Prize, a Cambridge University Engineering Department prize awarded annually to the candidate who demonstrates the greatest proficiency in electrical engineering.

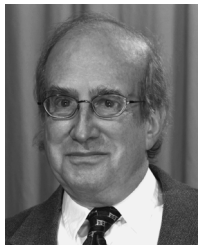
Sujay Sanghavi (M'06) received the M.S. degree in electrical and computer engineering (ECE) in 2002, the M.S. degree in mathematics in 2005, and the Ph.D. degree in ECE in 2006, all from the University of Illinois at Urbana-Champaign.

In 2009, he joined the ECE Department, University of Texas, Austin, where he is currently an Assistant Professor. From 2006 to 2008, he was a Postdoctoral Associate with LIDS, MIT, and from 2008 to 2009, was with Purdue University, West Lafayette, IN, as an Assistant Professor of ECE. His research interests span communication and social networks, and statistical learning and signal processing.

Dr. Sanghavi received the NSF CAREER award in 2010.

John W. Fisher, III (M'01) received the Ph.D. degree in electrical and computer engineering from the University of Florida, Gainesville, in 1997.

He is currently a Principal Research Scientist with the Computer Science and Artificial Intelligence Laboratory and affiliated with the Laboratory for Information and Decision Systems, both at the Massachusetts Institute of Technology (MIT), Cambridge. Prior to joining MIT, he was affiliated with the Electronic Communications Laboratory , University of Florida, from 1987 to 1997, during which time he conducted research in the areas of ultrawideband radar for ground and foliage penetration applications, radar signal processing, and automatic target recognition algorithms. His current area of research focus includes information theoretic approaches to signal processing, multimodal data fusion, machine learning, and computer vision.



Alan S. Willsky (S'70–M'73–SM'82–F'86) received the S.B. degree in 1969 and the Ph.D. degree in 1973 from the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology (MIT), Cambridge.

He joined the Massachusetts Institute of Technology (MIT), Cambridge, in 1973. He is the Edwin Sibley Webster Professor of Electrical Engineering and Director of the Laboratory for Information and Decision Systems. He was a founder of Alphatech, Inc. and Chief Scientific Consultant, a role in which he continues at BAE Systems Advanced Information Technologies. His research interests are in the development and application of advanced methods of estimation, machine learning, and statistical signal and image processing. He is coauthor of the text *Signals and Systems* (Englewood Cliffs, NJ: Prentice-Hall, 1996).

Dr. Willsky served on the US Air Force Scientific Advisory Board from 1998 to 2002. He has received a number of awards including the 1975 American Automatic Control Council Donald P. Eckman Award, the 1979 ASCE Alfred Noble Prize, the 1980 IEEE Browder J. Thompson Memorial Award, the IEEE Control Systems Society Distinguished Member Award in 1988, the 2004 IEEE Donald G. Fink Prize Paper Award, Doctorat Honoris Causa from Université de Rennes in 2005, and the 2010 Technical Achievement Award from the IEEE Signal Processing Society. In 2010, he was elected to the National Academy of Engineering. He and his students, have also received a variety of Best Paper Awards at various conferences and for papers in journals, including the 2001 IEEE Conference on Computer Vision and Pattern Recognition, the 2003 Spring Meeting of the American Geophysical Union, the 2004 Neural Information Processing Symposium, Fusion 2005, and the 2008 award from the journal *Signal Processing* for the Outstanding Paper in the year 2007. He has delivered numerous keynote addresses.