LEARNING GRAPHICAL MODELS FOR HYPOTHESIS TESTING

Sujay Sanghavi, Vincent Tan, Alan Willsky

LIDS, MIT

ABSTRACT

We propose a novel procedure for learning tractable graphical models from data samples. The traditional approach is to learn models that are generically good approximations of the underlying distributions. In contrast, we are interested in learning models for a specific purpose: binary hypothesis testing. The distributions corresponding to the hypotheses are not available, instead we are given two labelled sets of training samples.

Our procedure learns two models, one for each hypothesis, which are then used in a likelihood ratio test for classifying a new unlabelled sample. Each model is learnt from *both* sets of training samples. Numerical simulations show that our procedure has a lower probability of classification error, as compared to a procedure that learns each model using only its own training set. The gain is more significant when the problem size is larger and the number of training samples available is smaller.

1. INTRODUCTION

Reduced-order approximate modelling of distributions over high-dimensional spaces is an important problem with many applications. Standard approaches to this problem aim to develop models that are good approximations of the underlying distributions. In this paper we are motivated by the following question: can modelling can be made more effective, and approached differently, if it is known a-priori what application the reduced-order model is to be used for?

The particular application we are interested in is binary hypothesis testing (or equivalently, classification), but where the two candidate distributions are not known a-priori. Instead, we are given two labelled training sets, one from each of the two distributions, which we can use to develop models of the distributions. Once these models are developed, a new unlabelled sample is given and a likelihood ratio test using the two models is used to classify the new sample. The class of lower-order models we are interested in are distributions that are Markov with respect to sparse graphs.

This work is supported by US ARO under contract W911NF-05-1-0207 and by MURI grant W911NF-06-1-0076 through ARO. Vincent Tan is supported by the A*STAR National Science Scholarship. The authors can be reached at {sanghavi,vtan,willsky}@mit.edu

We propose a novel procedure to learn sparse graphical models, with the aim of developing models that are efficient *classifiers* as opposed to efficient *approximators*. In particular, we develop a new performance measure for model selection, and outline a procedure for *jointly* learning the two models, using *both* the training sets.

2. THE PROBLEM

We are interested in the following hypothesis testing problem: $X = (X_1, \ldots, X_n)$ is a length n vector of random variables, each of which can take a finite number of values. X can be generated from one of two hypotheses:

$$H_0 \ : \ X \sim p \qquad H_1 \ : \ X \sim q$$

The distributions p and q, or even any parametric classes they may lie in, are not available a-priori. Instead, we are given labelled training sets T_0 and T_1, consisting of K samples each, that are generated i.i.d. according to p and q respectively. One new unlabelled sample x is given, and the problem is to classify x as coming from one of the two hypotheses H_0 or H_1.

A natural approach to solving this problem is to *(a)* generate empiricals p_e from T_0 and q_e from T_1, and *(b)* do likelihood ratio testing using p_e, q_e. However, when the length n of the vector is large, the number of possible sample vectors is very large: if each X_i is binary, this number is 2^n. Now, unless K is at least of the same order as 2^n, the p_e and q_e will be poor approximations of p and q, and will also be inefficient at classification.

It is thus important to efficiently learn tractable models from insufficient data. Sparse graphical models provide a natural framework for this purpose, as was illustrated most beautifully in the seminal paper by Chow and Liu [1]. In that paper, they consider the problem of approximating *one* unknown distribution from its samples. They describe a procedure for learning the tree model that maximizes the likelihood of the training samples (among the set of all possible tree models), making it a good approximation to the true distribution. Their algorithm is very efficient in time and space usage, and the accuracy of the procedure depends on the accuracy of learning *pairwise* marginal distributions from data (instead of higher-order interactions). There has been a lot of other work (see e.g. [2, 3]) on exact/approximate learning of sparse graphical models. All of these papers have good approximation as their objective. In contrast, in this paper we develop an alternative application-dependent objective.

3. LEARNING GRAPHICAL MODELS *SPECIFICALLY* FOR HYPOTHESIS TESTING

In this section we describe our novel model learning procedure, and its underlying motivation. Given models f $\widehat{p}$ and $\widehat{q}$, the new x is classified according to a likelihood test: H_0 is declared if $\widehat{p}(x) > \widehat{q}(x)$, else H_1 is declared. Existing methods for learning graphical models attempt to build good approximations to their underlying distributions. In our setting, this means that $\widehat{p}$ would be generated using only T_0, and $\widehat{q}$ using only T_1. In contrast, our method jointly learns $\widehat{p}$ and $\widehat{q}$, using both T_0 and T_1. Additionally, it is efficient in terms of time and space complexity, and its accuracy depends priamrily on lower-order interactions – i.e. joint distributions of small sets of variables – that can be inferred more accurately from limited training data sets as compared to higher-order interactions.

A natural performance measure to evaluate any choice of $\widehat{p}, \widehat{q}$ is the probability of error. Let e_0 be the conditional error event that H_0 is mistaken to be H_1. Its probability is given by $P(e_0) = \sum_x p(x)\mathbf{1}_{\{\widehat{q}(x)\geq\widehat{p}(x)\}}$. Let e_1 be the the conditional error event that H_1 is mistaken for H_0. Ideally we would like to minimize $P(e_0)$ and $P(e_1)$, but we dont have access to the true distributions p and q. We propose using the empirical distributions p_e and q_e in their place. The resulting expression $P(\widetilde{e}_0) = \sum_x p_e(x)\mathbf{1}_{\{\widehat{q}(x)\geq\widehat{p}(x)\}}$ counts the *fraction of samples in* T_0 *misclassified as coming from* H_1. We attempt to learn $\widehat{p}, \widehat{q}$ so as to have low $P(\widetilde{e}_0)$ and $P(\widetilde{e}_1)$.

Selecting a graphical model involves two inter-related tasks: *(a)* finding the graph structure, and *(b)* finding the parameters. Ideally both tasks should be performed jointly, but that does not seem to be tractable. So we instead concentrate on first finding good candidates for graph structure, and then finding the parameters.

Learning Graph Structure

We now concentrate on learning the graph structures G_0, G_1 of the models $\widehat{p}, \widehat{q}$. Given any joint distribution p of n variables, and any graph G with n nodes, the *projection* p_G of p onto G is the distribution that is the closest to p in KL divergence among all distributions that are a markov on G: $D(p||p_G) \leq D(p_e||p')$ for any other p' that is also markov on G. For the purposes of learning graphs G_0, G_1, we will assume that $\widehat{p}$ is the projection of p_e onto the graph G_0, and similarly $\widehat{q}$ is the projection of q_e onto G_1.

Given models $\widehat{p}, \widehat{q}$, the log-likelihood ratio of a new observation x is $\log(\frac{\widehat{p}(x)}{\widehat{q}(x)})$. In order to have low $P(\widetilde{e}_0)$ and $P(\widetilde{e}_1)$, we would like this log-likelihhod ratio to be positive if $p_e(x) > q_e(x)$, and negative otherwise. We propose choosing G_0 and G_1 so as to maximize

$$\sum_x (p_e(x) - q_e(x)) \log\left(\frac{\widehat{p}(x)}{\widehat{q}(x)}\right)$$

The problem of finding the best pair G_0, G_1 decomposes into two independent problems:

$$\max_{G_0} \sum_x (p_e(x) - q_e(x)) \log \widehat{p}(x) \qquad (1)$$

$$\max_{G_1} \sum_x (q_e(x) - p_e(x)) \log \widehat{q}(x) \qquad (2)$$

where $\widehat{p}, \widehat{q}$ are the projections of p_e, q_e onto G_0, G_1 respectively. The fact that $\widehat{p}$ and $\widehat{q}$ appear in the logarithm means that we can decompose the expressions in (1)-(2) into terms that depend only on lower-order interactions.

Suppose, for example, that we are interested in generating tree models, i.e. we require that $\widehat{p}$ and $\widehat{q}$ be markov on trees. In this case the distribution $\widehat{p}(x)$ can be factored into terms over the edges (i,j) and nodes i of G_0

$$\widehat{p}(x) \;=\; \prod_{(i,j)\in G_0} \frac{p_e(x_i,x_j)}{p_e(x_i)p_e(x_j)} \prod_{i\in G_0} p_e(x_i) \qquad (3)$$

It follows that the expression in (1) can be rewritten as $c + \sum_{(i,j)\in G_0} w_{ij}$, where the edge weights are given by

$$w_{ij} \;=\; \sum_{(x_i,x_j)} (p_e(x_i,x_j) - q_e(x_i,x_j)) \log \frac{p_e(x_i,x_j)}{p_e(x_i)p_e(x_j)} \qquad (4)$$

and the constant $c = \sum_i \sum_{x_i} p_e(x_i)$ does not depend on the choice of G_0. Thus, the problem in (1) of finding the best G_0 becomes a max-weight spanning tree problem with the edge weights given as above. This can be efficiently solved using greedy algorithms like Prim or Kruskal. Note also that the value of each edge weight depends only on the joint distribution of (the variables corresponding to) its two endpoints. For reasonable values of sample size K, the empirical joint distribution $p_e(x_i,x_j)$ will be a good approximation of their true joint $p(x_i,x_j)$.

Note also that the space and time complexity of the above tree-finding procedure is the *same* as the complexity of the Chow-Liu procedure.

Learning Model Parameters

For the purposes of learning the graph structure, the models $\widehat{p}, \widehat{q}$ were merely the projections of the respective empiricals p_e, q_e onto the graphs G_0, G_1. However, further reductions in $P(\widetilde{e}_1)$ and $P(\widetilde{e}_2)$ are possible by changing the parameters of $\widehat{p}, \widehat{q}$, so that they remain markov on the same graphs G_0, G_1 learned above but are no longer the projections of p_e, q_e onto these graphs. We now motivate and describe our method to obtain the new parameters.

As a first step, note that for any $a, \lambda \geq 0$, we have that $\mathbf{1}_{\{a \geq 1\}} \leq a^{\lambda}$, which gives us an upper bound on $P(\widetilde{e}_0)$:

$$P(\widetilde{e}_0) \;\leq\; \min_{\lambda \geq 0} \;\sum_x p_e(x) \left(\frac{\widehat{q}(x)}{\widehat{p}(x)} \right)^{\lambda}$$

In the following, we will optimize the parameters of $\widehat{p}, \widehat{q}$ so as to minimize the above upper bound on $P(\widetilde{e}_0)$, and a similar bound on $P(\widetilde{e}_1)$. In doing so, we find it convenient to consider $\widehat{p}$ and $\widehat{q}$ to be members of exponential families: let

$$\widehat{p}(x) \;=\; \exp[\langle \theta_{\widehat{p}}, \phi_p(x) \rangle - \Phi(\theta_{\widehat{p}})] \qquad (5)$$

and similarly for $\widehat{q}$. Here $\phi_p(x)$ is a vector of functions (called "features") of x that characterize the graph structure of G_0. For example, if X is binary and $\widehat{p}$ is Markov on G_0, one choice for $\phi_p(x)$ could be that it contain all functions of the type $\phi_i(x) = x_i$ for variables i, and $\phi_{ij}(x) = x_i x_j$ for all edges $(i,j) \in G_0$. The features characterize the family (in our example the family of all distributions Markov on G_0), and the vector of parameters $\theta_{\widehat{p}}$ specifies the particular member of the family. The function $\Phi(\theta)$, also called the log-partition function, is a normalization constant. In this exponential family notation, we are interested in finding a good choice of $\theta_{\widehat{p}}, \theta_{\widehat{q}}$ given the families ϕ_p, ϕ_q. The following lemma motivates the use of convex optimization to achieve this goal. This connection to convex optimization is the reason we use the formalism of exponential families.

Lemma 3.1 *Let* $A_0(\theta_{\widehat{p}}, \theta_{\widehat{q}}, \lambda) \;=\; \log \left[\sum_x p_e(x) \left(\frac{\widehat{q}(x)}{\widehat{p}(x)} \right)^{\lambda} \right]$. *Then,* A_0 *is convex in* $\theta_{\widehat{p}}$ *for fixed* $\theta_{\widehat{q}}$ *and* λ. *Also,* A_0 *is convex in* λ *for fixed* $\theta_{\widehat{p}}$ *and* $\theta_{\widehat{q}}$.

Note that $\log x$ is an increasing function, and hence minimizing A_0 is equivalent to minimizing $\sum_x p_e(x) \left(\frac{\widehat{q}(x)}{\widehat{p}(x)} \right)^{\lambda}$. However, note that A_0 may not be *jointly* convex in λ and $\theta_{\widehat{p}}$, for fixed $\theta_{\widehat{q}}$. To enable efficient minimization, we now construct an upper bound on A_0 that *is* jointly convex in its parameters. As a first step to this upper bound, we prove the following property for Φ, the log-partition function.

Lemma 3.2 *For any exponential family defined on a discrete model and any* θ, *if* $0 \leq \lambda \leq 1$ *then* $\lambda \Phi(\theta) \leq \Phi(\lambda\theta)$ *and if* $\lambda \geq 1$ *then* $\lambda \Phi(\theta) \geq \Phi(\lambda \theta)$.

Note now that A_0 can be written as

$$\log \left[\; \sum_x p_e(x) [\widehat{q}(x)]^{\lambda} e^{-\lambda \langle \theta_{\widehat{p}}, \phi_p(x) \rangle + \lambda \Phi(\theta_{\widehat{p}})} \; \right]$$

Lemma 3.2 implies that for $\lambda \leq 1$, $A_0(\theta_{\widehat{p}}, \theta_{\widehat{q}}, \lambda) \leq S_0(\lambda\theta_{\widehat{p}}, \lambda)$ where for any a and $\lambda \geq 0$, $S_0(a, \lambda)$ is given by

$$S_0(a, \lambda) = \log \left[\; \sum_x p_e(x) [\widehat{q}(x)]^{\lambda} e^{-\langle a, \phi_p(x) \rangle + \Phi(a)} \; \right]$$

Lemma 3.3 $S_0(a, \lambda)$ *is jointly concave in* (a, λ) *for fixed* $\theta_{\widehat{q}}$.

Thus, S_0 can be efficiently minimized for a given fixed value of $\theta_{\widehat{q}}$. Also, similar to A_0 and S_0 above, define $A_1(\theta_{\widehat{p}}, \theta_{\widehat{q}}, \gamma)$ and $S_1(b, \gamma)$, such that S_1 is jointly convex in b and γ for fixed $\theta_{\widehat{p}}$.

To find the best pair $\theta_{\widehat{p}}, \theta_{\widehat{q}}$, we propose the following alternating procedure using S_0 and S_1: first fix $\theta_{\widehat{q}}$ and find $\min_{(a,\lambda)} S_0(a, \lambda)$. Set $\theta_{\widehat{p}} = \frac{a}{\lambda}$, and find $\min_{(b,\gamma)} S_1(b, \gamma)$. Set $\theta_{\widehat{q}} = \frac{b}{\gamma}$, and find $\min_{(a,\lambda)} S_0(a, \lambda)$, and so on. The values of $\theta_{\widehat{p}}, \theta_{\widehat{q}}$ at convergence of these alternations are the output of the parameter update procedure.

Connection to Error Exponents

Interestingly, the paramter learning procedure above is related to the error exponent of the hypothesis test. This connection serves as additional motivation for the procedure. We now briefly describe the connection.

Suppose, as before, that we were intersted in declaring a hypothesis (H_0 or H_1) based on a likelihood ratio test using models $\widehat{p}, \widehat{q}$. However, we are now given M new samples, either all generated i.i.d. according to p, or all according to q. Let e_0^M be the (conditional) error event that H_1 is declared when the truth was H_0. It is reasonable to expect that $P(e_0^M)$ will decrease to 0 exponentially in M, and the corresponding *error exponent* is defined as

$$E_0(\widehat{p}, \widehat{q}) \;=\; \lim_{M \to \infty} -\frac{1}{M} \log P(e_0^M)$$

The following lemma gives an expression for $E_0(\widehat{p}, \widehat{q})$.

Lemma 3.4 $E_0(\widehat{p}, \widehat{q}) \;=\; \max_{\lambda \geq 0} \; -\log \left[\sum_x p(x) \left(\frac{\widehat{q}(x)}{\widehat{p}(x)} \right)^{\lambda} \right]$

Note the close similarity between the above expression and that of A_0 in Lemma 3.1. In particular, if we replace p with p_e in the above expression, then the right hand side is exactly $-A_0(\theta_{\widehat{p}}, \theta_{\widehat{q}}, \lambda)$. Thus minimizing A_0 is equivalent to minimizing (an empirical version of) the error exponent.

4. NUMERICAL SIMULATIONS

In this section, we present numerical results to validate the theory and algorithm presented in the preceding section. In our simulations X is a binary vector of length n, and the true distributions p and q are tree-structured distributions. For the selection of p, first a random tree structure G_0^* on n nodes is chosen as follows: $i = 1$ is the root, and $p(x_1)$ is bernoulli with parameter chosen uniformly at random from $[0, 1]$. Each node $2 \leq i \leq n$ chooses a "parent" $j(i)$ uniformly from the set $\{1, \ldots, i-1\}$. Then, for each $i \geq 2$ the conditional distribution $p(x_i|x_{j(i)})$ is chosen to be bernoulli, with the parameters of the bernoulli distribution chosen randomly from $[0, 1]$. p is then given by

$$p(x) \;=\; p(x_1) \prod_{i \geq 2} p(x_i|x_{j(i)})$$

Note that p is markov with respect to the tree G_0^*, which has as its edges $(i, j(i))$ for all $i \geq 2$. This form of specification for p enables easy generation of samples. The distribution q corresponding to hypothesis H_1 is also generated by a similar procedure. For the training sets, K samples are generated from each of p and q. The resulting empirical distributions from the training sets are p_e and q_e.

In our simulations, the class of sparse graphical models we are interested in learning are those that are markov on trees. The graph structures G_0, G_1 of $\widehat{p}, \widehat{q}$ are chosen according to the rules (1)-(2). This means that, as explained before, the graph G_0 for $\widehat{p}$ is the max-weight spanning tree when the edge weights are as given by (4). Similarly the graph G_1 for $\widehat{q}$ will be a max-weight spanning tree with edge weights similar to (4), but with p_e and q_e interchanged.

Figure 1 depicts the first set of simulation results, plotting the probability of classification error for new samples as a function of the number K of training samples for three different problem sizes: $n = 20, 60, 100$. Each figure has three curves: one is the “CL” curve, where classification is done by learning models according to the classical Chow-Liu procedure (i.e. $\widehat{p}$ is learnt only from T_0 and $\widehat{q}$ only from T_1). The next curve is the “LGMHT-proj” curve, wherefor which we set $\widehat{p}, \widehat{q}$ to just be the projections of p_e, q_e onto G_0, G_1 repsectively, without further optimizing the paramters of $\widehat{p}, \widehat{q}$. That is, we assume that $\widehat{p}$ is as given by (3). Note that the complexity of the LGMHT-proj procedure is identical to that of the Chow-Liu procedure.

From Figs 1(b) and 1(c), we observe that the LGMHT-proj algorithm has a lower probability of error than the Chow-Liu algorithm, especially for smaller values of the number K of training samples and larger size n of the problem instance. For instance, for the 100 node example with 40 training samples, the $\Pr(err)$ was reduced from 0.35 to 0.12-3, a threefold improvement. Large problems with small samples sizes are frequently the setting of interest in many practical situations *e.g.* seismic or medical data.

The final curve in Figure 1 is “LGMHT-altmax”, where we optimize the paramters using the procedure described in the last section. It is seen that performance of the alternating minimization procedure is actually worse than the performance of LGMHT-proj, which just used the paramters from p_e. Interestingly however, its performance is starkly different when applied to the training samples.

Recall that the alternating minimization procedure was developed with the aim of reducing $P(\widetilde{e}_0)$ and $P(\widetilde{e}_1)$, the fraction of misclassified samples *in the training data*, with the hope that the new parameters would generalize to better performance for new samples. Figure 1 indicates that this generalization may not hold. Figure 2 plots the average of $P(\widetilde{e}_0)$ and $P(\widetilde{e}_1)$ as a function of K for different sizes n. We see that the alternating parameter learning procedure *does* lead to lower $P(\widetilde{e}_0)$ and $P(\widetilde{e}_1)$, the purpose for which it was designed.

5. CONCLUSION AND DISCUSSION

In this paper we motivate and develop a novel procedure for learning tractable graphical models specifically for the purpose of binary hypothesis testing. The novelty arises from the fact that each model is learnt using samples drawn from *both* candidate distributions. Numerical simulations show that learning each model from both sets of samples leads to a lower clasification error as compared to learning each model from only one set of samples. This reduction in error probability is especially evident for larger problem sizes and fewer training samples, which is often the scenario of interest. Additionally, an alternating paramter-learning procedure, which has higher complexity, is shown to reduce the number of misclassified training samples.

We believe this work raises many interesting questions. Primary among them is the odd dichotomy in the performance of the parameter learning procedure: it leads to higher probability of error for new samples, but lower for the training samples. This suggests that it is somehow “overfitted” to the training data. A modification of this method to prevent overfitting would be of interest. Another interesting direction is the problem where there are multiple hypotheses.

6. REFERENCES

[1] C. K. Chow and C. N. Liu, “Approximating discrete probability distributions with dependence trees.,” *IEEE Transactions on Information Theory*, vol. 14, no. 3, pp. 462–7, May 1968.

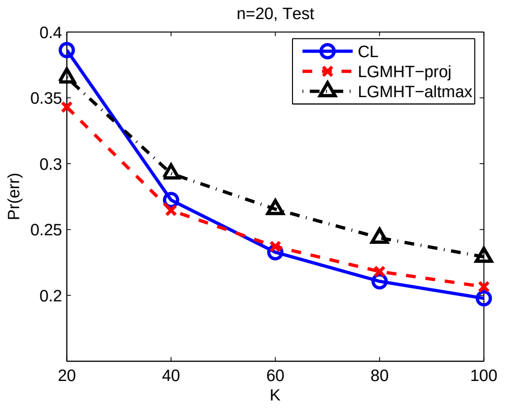
[2] F. R. Bach and M. I. Jordan, “Thin Junction Trees,” in *Advances in Neural Information Processing Systems*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds., Cambridge, MA, 2002, vol. 14, pp. 569–76, MIT Press.

[3] D. Karger and N. Srebro, “Learning Markov networks: maximum bounded tree-width graphs,” in *Symposium on Discrete Algorithms*, 2001, pp. 392–401.

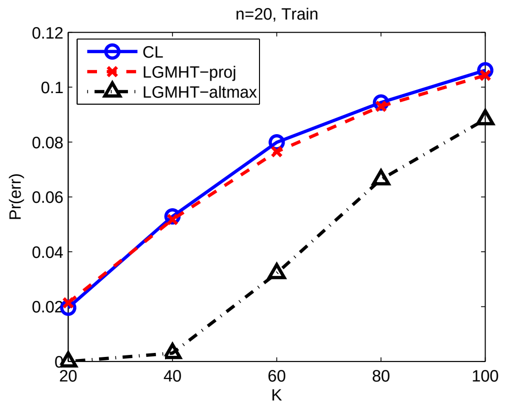
[4] M. X. Goemans and R. Ravi, “The Constrained Minimum Spanning Tree Problem,” in *Fifth Scandinavian Workshop on Algorithm Theory, LNCS 1097*, Reykjavik, Iceland, 1996, pp. 66–75.

[5] Bertsekas D. P., *Nonlinear Programming*, Athena Scientific, 2nd edition, 1999.

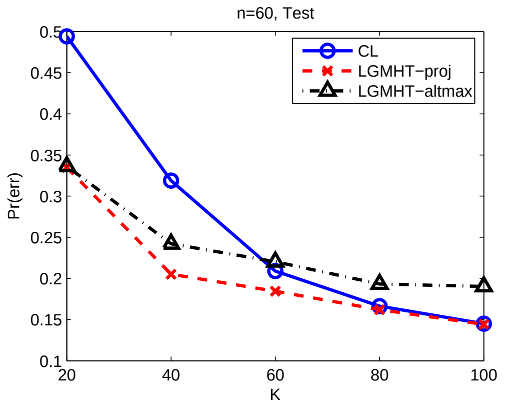
[6] J. K. Johnson, V. Chandrasekaran, and A. S. Willsky, “Learning Markov structure by maximum entropy relaxation,” in *11th Inter. Conf. on AI and Stat. (AISTATS ’07)*, 2007, pp. 392–401.



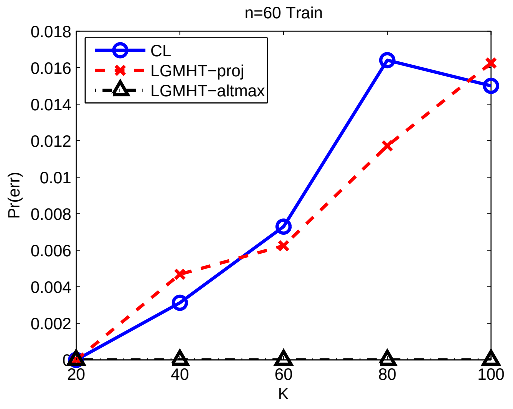
(a) $\Pr(err)$ for $n = 20$



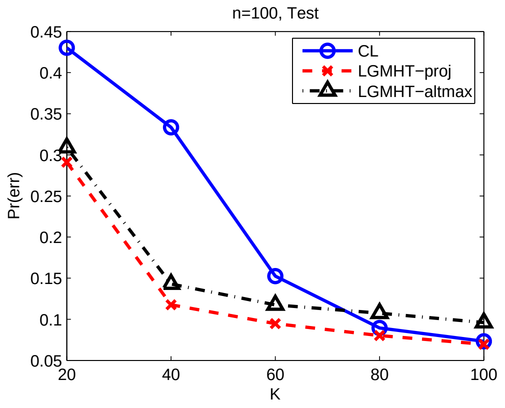
(a) $\Pr(err)$ for $n = 20$



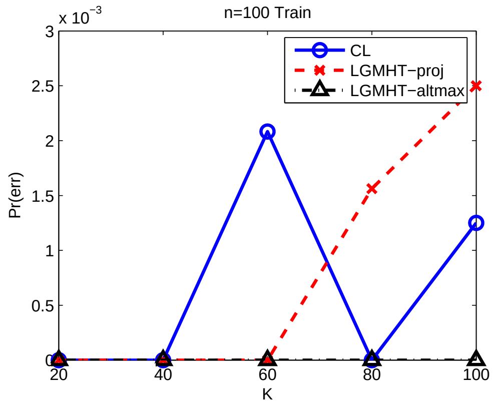
(b) $\Pr(err)$ for $n = 60$



(b) $\Pr(err)$ for $n = 60$



(c) $\Pr(err)$ for $n = 100$



(c) $\Pr(err)$ for $n = 100$

Fig. 1. Classification of *new* samples. Probability of error is fraction of misclassified samples from a set of 1000 samples, 500 of each distribution

Fig. 2. Classification of training samples. Probability of error is the average of $P(\widetilde{e}_0)$ and $P(\widetilde{e}_1)$.