AClass: An online algorithm for generative classification

Vikash K. Mansinghka
MIT BCS* and CSAIL†
vkm@mit.edu

Daniel M. Roy
MIT CSAIL†
droy@mit.edu

Ryan Rifkin
Honda Research USA‡
rrifkin@honda-ri.com

Josh Tenenbaum
MIT BCS* and CSAIL†
jbt@mit.edu

Abstract

We present AClass, a simple, online, parallelizable algorithm for supervised multiclass classification. AClass models each class-conditional density as a Chinese restaurant process mixture, and performs approximate inference in this model using a sequential Monte Carlo scheme. AClass combines several strengths of previous approaches to classification that are not typically found in a single algorithm; it supports learning from missing data and yields sensibly regularized nonlinear decision boundaries while remaining computationally efficient. We compare AClass to several standard classification algorithms and show competitive performance.

1 Introduction

Classification is a foundational problem in machine learning, and classic approaches are full of apparent tradeoffs. For example, the ability to handle missing or unlabeled data via simple inductive biases is considered the strong point of generative approaches while discriminative methods often exploit flexible, nonlinear decision boundaries that generalize well while remaining computationally tractable. In this paper, we present the AClass algorithm, a simple but novel approach that combines several strengths of previous classifiers that have not typically been available together in a single algorithm. We also perform preliminary experiments in synthetic and benchmark settings, showing competitive performance.

AClass is an online, parallelizable algorithm for multiclass classification based on approximate, sequential Monte Carlo based inference in a probabilistic generative model. AClass operates by learning a model of the class-conditional densities of the data by assuming they are Chinese restaurant process (CRP) mixtures. Chinese restaurant process mixture models are a standard tool from the nonparametric Bayesian statistics literature, and enable the number of mixture components to be discovered dynamically during posterior inference. Intuitively, each component can be thought of as a mode or prototype associated with its class. The full generative model embeds these class-conditional mixtures in an outer mixture over class labels. As this space permits many modes per class, we can obtain nonlinear decision boundaries, without sacrificing the appealing tractability of popular discriminative approaches. The model can handle both missing labels and missing features, although the approximate inference methods we present requires all labels during training to be observed. The training and testing procedures derived from our Monte Carlo approximation constitute the AClass algorithm.

Our work builds on the approach of Nigam (2001), which used conventional finite mixture models (trained with EM) to represent class-conditional densities. AClass addresses some of the basic inductive and algorithmic limitations of that approach. First, by grounding our algorithm in posterior inference in a bank of CRP mixtures, model complexity is automatically adjusted in a probabilistically coherent, self-consistent way as new datapoints arrive. Second, by approximating posterior inference using sequential Monte Carlo methods (Doucet et al., 2001), we can efficiently and accurately train and test online, without bounding the model complexities we consider. In particular, as new data are encountered, our online inference scheme avoids the computationally expensive re-training (and re-crossvalidation) associated with finite mixture approaches.

* Brain and Cognitive Science Dept, 43 Vassar St., Cambridge, MA 02139

† Computer Science and Artificial Intelligence Lab, 32 Vassar St., Cambridge, MA 02139

‡ Honda Research Institute USA, Inc., 145 Tremont St., Boston, MA 02111

1.1 Related Work

The model underlying AClass is a nonparametric extension of the mixture-of-finite-mixtures approach to supervised and semisupervised classification (see Nigam (2001) for applications to text data and Halberstadt and Glass (1998) for applications to speech recognition). Instead of finite mixture models for each class-conditional density, we use CRP mixture models with conjugate priors (Rasmussen, 2000), which are typically fit to data using Markov chain Monte Carlo (MCMC) methods (Neal, 2000). However, instead of computationally expensive, batch MCMC, we perform inference via the conditional-posterior particle filter for conjugate CRP mixtures (see e.g. (Fearnhead, 2004) and (Sanborn et al., 2006)).

The Forgetron (Dekel et al., 2006) is one recently developed discriminative approach that provides similar functionality to AClass. The Forgetron is a bounded-memory, online, kernelized, binary classification scheme with attractive theoretical properties, but it cannot naturally handle missing data. Widely used discriminative algorithms like regularized least squares (RLS) and logistic regression also provide reasonable performance benchmarks (Rifkin and Lippert, 2006), although they lack some of the desirable features of our approach, especially online training and testing.

In principle, an efficient implementation of AClass should be applicable to large scale classification problems with mixed discrete and continuous features, such as classifying web pages, large image databases or astronomical data. In this paper, we focus on exploring the properties of the basic model algorithm in simple cases.

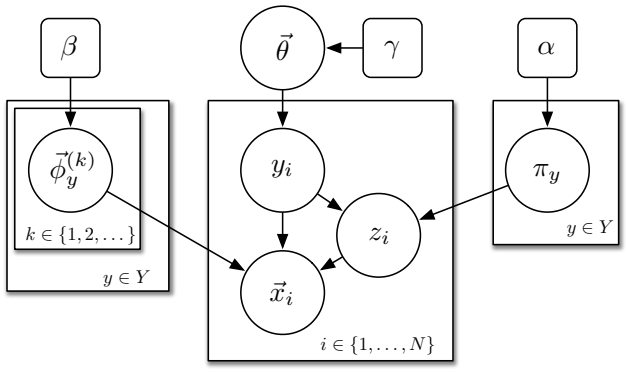

Figure 1: Graphical model for the joint probability distribution over latent parameters, labels and feature vectors in which AClass operates. The training and testing procedures comprising the AClass algorithm are derived from sequential Monte Carlo inference in this model.

2 Model and Inference

We describe our model in terms of a generative process for data sets, comprised of class labels, $y \in Y$, and feature vectors, $\vec{x}$.

1. Let $\vec{\phi}_y^{(k)}$ be the vector of parameters for the kth mixture component in the class-conditional density for class y. For $y \in Y$, $k \in \{1, 2, \dots\}$, sample $\vec{\phi}_y^{(k)}$ from an appropriate conjugate prior (with hyperparameter β).

2. Sample mixing weights over class labels, $\vec{\theta} = \{\theta_y\}$, from a Dirichlet prior with hyperparameter γ.

3. For each datapoint $i \in \{1, \dots, N\}$,

   (a) Sample the class label, y_i, from a multinomial distribution with parameters $\vec{\theta}$.

   (b) Sample the group (mixture component) membership for this datapoint, z_i, according to the CRP for class y_i. Formally, if n_y^g is the number of datapoints with label y assigned to group g, and $n_y = \sum_g n_y^g$ is the total number of datapoints observed with label y so far, we choose each existing group with probability $\frac{n_y^g}{n_y + \alpha}$ and a new group (i.e. create a new mixture component in this class-conditional density) with probability $\frac{\alpha}{n_y + \alpha}$.

   (c) Sample the feature vector $\vec{x}_i$ from the appropriate conjugate likelihood with parameters $\vec{\phi}_{y_i}^{(z_i)}$.

In this paper, we focus on binary features, modeled via the conjugate Beta-Bernoulli model. Continuous features (either independent or correlated, via the Normal-Inverse-Chisquared and Normal-Inverse-Wishart models, respectively) are a straightforward modification.

In essence, the per-class Chinese restaurant process induces a prior on partitions of our training data, ranging from the case where all datapoints are in one group (recovering a naive Bayes classifier or single-mode model for the class conditional density) to the case where each datapoint has its own mode (recovering a kernel density estimate, with the kernel derived from the posterior predictive distribution of the conjugate model for mixture components).

Our model has few free parameters. We set α to the standard value of 1.0, and leave the Beta hyperparameters at reference values of 0.5 throughout.

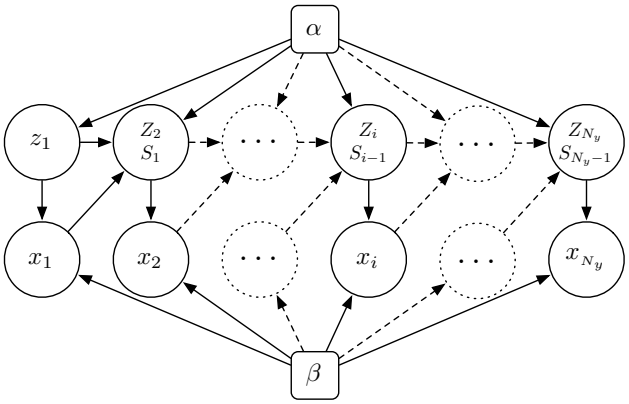

Figure 2: Dynamic model obtained by arbitrarily ordering the datapoints within each class and integrating out class-conditional component parameters. Z_i is the group (mixture component) membership vector for datapoints $1...i$. S_{i-1} is the collection of sufficient statistics for all groups present before datapoint i is processed. Note that the dimensionality of the latent variables Z and S grows as the chain unrolls over datapoints. The AClass inference algorithm can be interpreted as a bank of particle filters for this dynamic Bayesian network, one for each class.

2.1 Inference

The structure of our sequential Monte Carlo algorithm for inference closely follows our generative process; this conceptual simplicity is an attractive feature of our approach. In the case of fully supervised data, the asserted class labels separate the parameters of the CRP mixture models for each class, so inference reduces to separate problems for each class-conditional CRP. To solve these decoupled problems, we employ a particle filtering approach based on Fearnhead (2004).

The intuition behind a CRP mixture particle filter is that each particle represents a hypothesis about the assignment of the past data points to groups. When a new datapoint is processed by a particle, it is stochastically assigned to a group based on how well it fits with the data previously assigned to that group (via its probability under the group's posterior predictive distribution). Old datapoints are *not* reassigned; instead, the resampling step for the particle filter discourages the propagation of particles which seem to be relatively poor explanations of the data.

To better understand the structure of our inference approach, it is illustrative to view the dynamical graphical model shown in Figure 2. This model can be obtained from our original model by focusing on one class, arbitrarily ordering the datapoints for that class, and integrating out the class-conditional mixture component parameters $\vec{\phi}$ (retaining only their associated per-mixture-component sufficient statistics). Our sequential Monte Carlo algorithm can be directly interpreted as a bank of independent filters for this model, one for each class.

The essence of the transformation from the original model to a dynamic one (where approximate filtering yields approximate posterior inference) is the collection of the z variables, or group memberships, for all datapoints up to i into a single group membership vector Z_i. This transformation directly exploits the self-consistency of the CRP prior on partitions for varying numbers of datapoints to yield well-defined unrollings for an arbitrary number of datapoints. However, since we are using the CRP as a prior on partitions (or component assignment vectors) in a mixture model, we must also similarly transform the sufficient statistics; accordingly, we also track the sufficient statistics for all groups (mixture components) when datapoint i is processed in S_{i-1}. Thus the latent state of this dynamic model tracks a density estimate for the datapoints in terms of a partition of datapoints into mixture components and the parameters associated with the components. Accordingly, S_0 is empty (there are no pre-existing sufficient statistics when processing the first datapoint), and $Z_1 = z_1 = [0]$, since the first datapoint must be in its own group.

To sample from the "motion model" on latent variables, $P(Z_i, S_{i-1}|x_{i-1}, Z_{i-1}, S_{i-2})$, first copy Z_{i-1} into Z_i, then incorporate x_{i-1} into the appropriate part of S (based on its group assignment z_{i-1}), then sample a component assignment for datapoint i from the CRP prior. The "observation model" measures compatibility of x_i with the running density estimate; so, $P(x_i|Z_i, S_{i-1})$ is just the posterior predictive distribution of the group to which i was assigned (obtained by looking up the entry of S_{i-1} for the group z_i).

Particle filtering yields running Monte Carlo estimates of the latent variable in this chain; after processing all datapoints up to i, then, we have samples from $P(z_{1...i}, s_{1...(i-1)}|x_1, \cdots, x_i)$, namely the posterior distribution of interest. AClass is simply a conditional-posterior particle filter for this chain structured model. Since the particle filter can be asked for posterior estimates at any point in time, our algorithm is online. Since the only interactions between particles take place during aggregation steps such as resampling and testing, our algorithm is massively parallelizable with relatively inexpensive communication costs, and is, in principle, suitable for large scale classification tasks.

In the pseudocode shown in Figure 2.1, we use $\vec{x}$ to denote the F features of an observation (possibly with missing values), $y \in Y$ for class labels, α for the CRP hyperparameter, γ for the hyperparameter on the multinomial distribution over class labels, $A.m[l]$ to denote the counts for class l, and

$p.n[g]$ to denote the counts for group g (part of a CRP partition) in particle p. We have N total datapoints. All values are assumed to be initialized to zero unless otherwise specified. Furthermore, we assume some basic utility procedures, such as SAMPLE-DISCRETE, which returns a pseudorandom sample from a vector of probabilities); AVERAGE, which returns a weighted average of its inputs; and RESAMPLE-PARTICLES($\mathcal{F}, \vec{w}$), which resamples the particles in filter $\mathcal{F}$ according to the weights in $\vec{w}$ via the standard resampling algorithm from Sampling-Importance-Resampling[1] (Doucet et al., 2001). This step focuses effort of the sampler on particles that score well locally (i.e. explain the last datapoint well). In practice, all functions manipulating probabilities/weights should be implemented in the log domain to avoid underflow. Note that all vectors are assumed to be initialized to zeros and growable dynamically.

Because we assume conjugate priors on the parameters of the mixture components in each CRP-based class-conditional density estimator, we can integrate out these parameters analytically and only represent the sufficient statistics associated with previous datapoints assigned to a given group. UPDATE-SUFFSTATS and POSTERIOR-PREDICTIVE are standard probability computations in terms of these statistics; missing features are handled simply by ignoring them in these functions. For example, with binary data, the sufficient statistics for each feature are simply counts of observed heads, h_g^p, and observed tails, t_g^p, one for each group in each particle. Then UPDATE-SUFFSTATS increments those counts, and POSTERIOR-PREDICTIVE computes $P(\vec{x}) = \prod_{f=1}^{F} \frac{\beta + h^{x^f} t^{1-x^f}}{2\beta + h + t}$. See e.g. Gelman et al. (2003) for details for other conjugate models.

The memory requirements are worst-case linear, when every datapoint is assigned to its own group in its appropriate class-conditional density. However, this case is exceedingly unlikely in practice - in expectation, the Chinese restaurant process generates $\log \alpha$ groups, and generating as many groups as datapoints would be a signal that no generalization from one datapoint to another is licensed. In practice, we conjecture bounds of a few hundred groups should be adequate for very large scale real-world discrete feature classification tasks, based on (Nigam, 2001).

We can characterize the complexity per iteration of the algorithm in terms of p_i (the number of particles per CRP density estimator), G (the maximum number of groups found in a class-conditional CRP) and the other variables defined above. In particular, TRAIN-CRP and PREDICTIVE-DENSITY both take $O(p_i G D)$ time. This subroutine must be called $O(N)$ times. Overall, then, we get an algorithm whose runtime can be expected to be linear in N (although the extremely unlikely worst-case complexity is quadratic).

[1]Throughout, we resample when the number of effective particles, measured as $1/(\sum \vec{w}_i^2)$, reaches half of the original number of particles in the filter

```
ACLASS-TRAIN(A:aclass, y:class-label, x:observation)
1  TRAIN-CRP(A.F[y], x)

ACLASS-TEST(A:aclass, x:observation)
1  foreach label y in Y
2      prob[y]+= (A.m[y]+γ)/(Sum(A.m)+|Y|·γ)
                 ·PREDICTIVE-DENSITY(A.F[y], x)
3  prob ← NORMALIZE(prob)

TRAIN-CRP(F:filter, x:observation)
1  foreach particle p in F
2      foreach group g in 1, 2, ..., p.numGroups
3          prob[g] ← p.n[g]/(SUM(p.n+α)) · POSTERIOR-PREDICTIVE(p. stats[g], x)
4      prob[g+1] ← α/(SUM(p.n)+α) · POSTERIOR-PREDICTIVE(p. stats[g+1], x)
5      F.w[p] ← SUM(prob)
6      g' ← SAMPLE-DISCRETE(prob)
7      p.n[g'] ← p.n[g'] + 1
8      UPDATE-SUFFICIENT-STATISTICS(p, g', x)
9  RESAMPLE-PARTICLES(F, w)

PREDICTIVE-DENSITY(F:filter, x:observation)
1  foreach particle p in F
2      foreach group g in 1, 2, ..., p.numGroups
3          prob[p][g] ← p.n[g]/(SUM(p.n)+α) · POSTERIOR-PREDICTIVE(p. stats[g], x)
4      prob[p][g+1] ← α/(SUM(p.n)+α) · POSTERIOR-PREDICTIVE(p. stats[g+1], x)
5      prob[p] ← SUM_g(prob[p][g])
6  return AVERAGE(F.w[p], prob[p])
```

Figure 3: Pseudocode for the AClass algorithm. Suitable for use with generic conjugate models for mixture components. Depends on standard statistical procedures for sampling, averaging and predictive density evaluation.

3 Experiments

We performed several preliminary experiments to address the basic performance of our algorithm in synthetic and real-world settings. Throughout, the discriminative benchmarks we compared against were linear logistic regression and nonlinear regularized least squares with a polynomial kernel of degree 2. For the regularized least squares algorithm, we used a one-vs-all scheme and optimized the regularization parameter λ for each classifier individually via efficient leave-one-out cross-validation methods (Rifkin and Lippert, 2006). For logistic regression, we set $\lambda = 10^{-2}$ in all trials. To handle missing data with discriminative methods, we used online mean imputation - that is, we filled in each missing feature value with the running average of the observed values seen before it, starting with 0.5. It is worth noting that this mean imputation strategy can be interpreted as fitting a single mixture component to each class, with independent feature values, and imputing with the mean of the posterior predictive distribution on features.

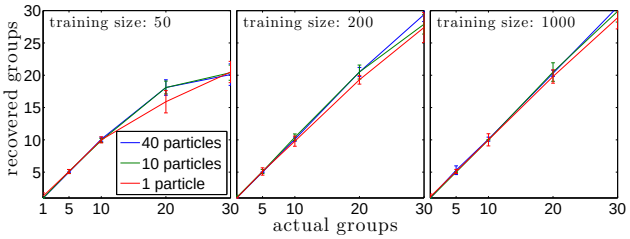

Figure 4: Accuracy of sequential Monte Carlo for Chinese restaurant process mixture models on synthetic data. Each plot shows the number of recovered groups (mixture components) versus the number of groups in the true model from which synthetic data was sampled. Each plot shows accuracy for a different number of training points. Each line shows results for inference with a different number of particles. All results are averages over 10 training sets.

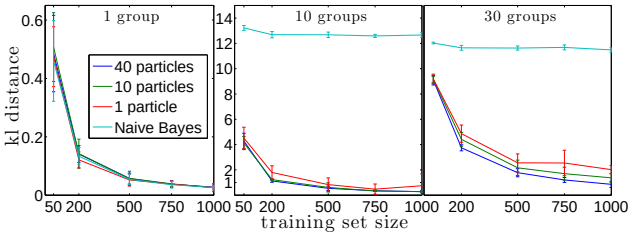

Figure 5: Effectiveness of sequential Monte Carlo inference in Chinese restaurant process mixture models at producing accurate density estimators. Each plot shows a Monte Carlo estimate of the KL divergence between the learned density model and ground truth versus the number of training data points. Each plot shows accuracy for a different number of groups (mixture components) in the true density. Each line shows results for inference with a different number of particles (and a one-component model, equivalent to the density estimation done as part of a naive Bayes classifier, as a baseline). All results are averages over 10 training/testing sets. Note the scale difference in the 1 group case.

Our core approach to classification is grounded in class-conditional density estimation. Accordingly, we first report the performance of our density estimator (the TRAIN-CRP and PREDICTIVE-DENSITY subroutines) on synthetic data. We generated synthetic data from mixture models over 50-dimensional binary feature vectors with varying numbers of components (from 1 to 30), to test our particle filter inference approximation (and model robustness to the setting of α, which was fixed to 1 throughout). The mixing weights (over the components) were fixed to be uniform, yielding groupings a priori disfavored under the Chinese restaurant process and therefore a conservative test of particle filter accuracy. For each component in each mixture, we generated parameters for each binary feature (corresponding to a single coin weight) from a Beta$(0.5, 0.5)$ distribution, yielding moderately noisy clusters. We then measured accuracy at component discovery and density estimation under our model.

Figure 4 summarizes our results with respect to discovery of the right complexity mixture. We found that the particle filter approximation discovered complexities accurately, though for very small training sizes, we underestimate the number of groups if the true number is large enough that some may not have been observed in the training sample.

We also computed a Monte Carlo estimate of the KL divergence between the density learned by our particle filters and the true mixture density. For comparison, we show the KL divergence between truth and a one-mode model (which, if used for class-conditional density estimation and classification, would yield a naive Bayesian classifier). When the true density has multiple modes, the naive Bayesian model performs substantially worse than a CRP mixture, though they agree when the true distribution is unimodal. Otherwise, we find all particle filters yield accurate density estimators with small numbers of samples, though the 40 particle version is roughly twice as accurate as the 1-particle version when the true density has many groups. This is unsurprising, as the 1-particle filter will be sensitive to random errors in the assignment of early datapoints, while similarly erroneous particles die in resampling in the 40-particle filter.

Our second experiment explored the issue of whether these density estimators can be used as the basis of competitive generative classifiers. We investigated several synthetic classification problems, and report here the results on one of them: a four-class problem, where the class-conditional density for each class was a mixture model (as in experiment 1, with uniform mixing weights and Beta(0.5, 0.5) priors on the Bernoulli parameter for each dimension). The mixtures for the four classes had 4, 10, 5 and 20 modes respectively, representing a wide range of complexities requiring cross-validation over a broad range under traditional approaches. We report average 0-1 loss on a held-out test set of 500 datapoints for a range of training set sizes, as well as performance in a missing data setting, where varying fractions of feature values were missing at random during training *and* testing. At all three levels of missing data, AClass outperformed the other algorithms. Naive Bayes was the least effective. With no missing data, the 1-particle AClass performs essentially as well as AClass, since classification is easier than density estimation (at least under 0-1 loss, so predictive uncertainties are unimportant). However, with 50% data missing, the 1-particle AClass suffers slightly, as small errors in class-conditional density estimation have a greater impact on test accuracy.

Our third experiment assessed the performance of our approach on a standard classification task based on a subset of the 20 Newsgroups data set collected by

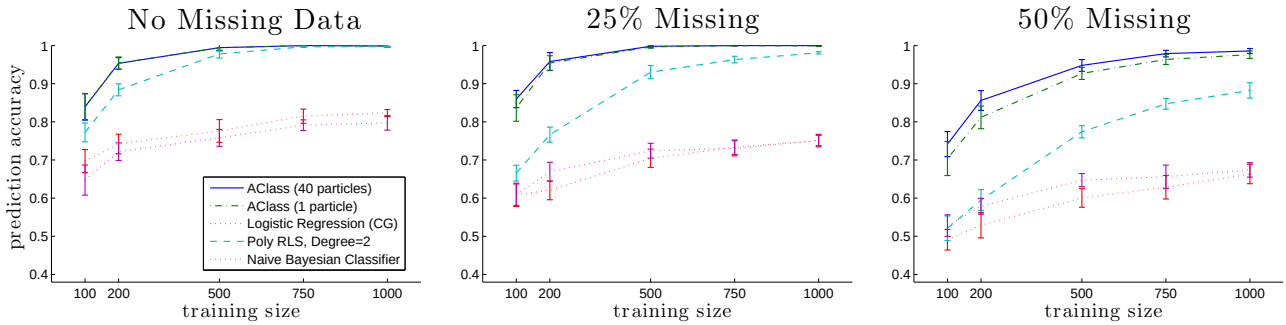

Figure 6: Classification performance (via average 0-1 loss) versus training set size on a 100 feature synthetic benchmark with 4 classes. The class-conditional densities had 4 modes, 5 modes, 10 modes and 20 modes respectively. Each plot shows performance when a different fraction of feature values are missing during training and testing. Each line shows classification performance for a different classification algorithm. All results are averages over 10 training/test sets.

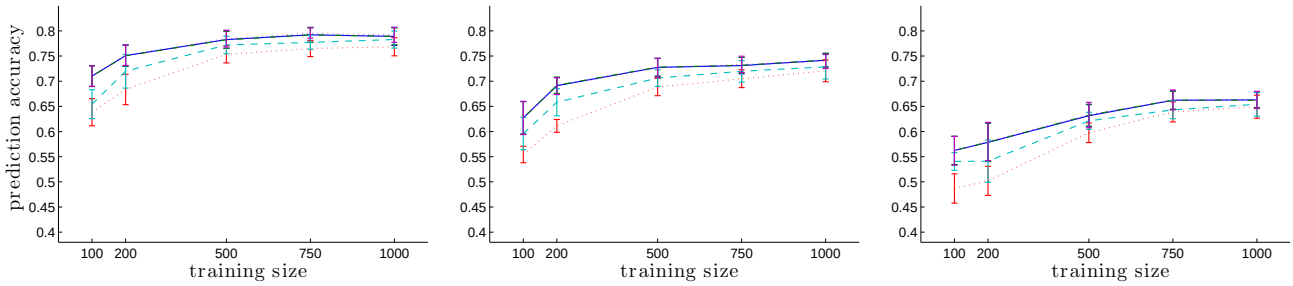

Figure 7: Classification performance, as above, on a 4-class problem on a 20-Newsgroup dataset with 100 binary features. Note the effectiveness of the naive Bayes classifier on this task, and the recovery of that performance by AClass, due to sensible regularization.

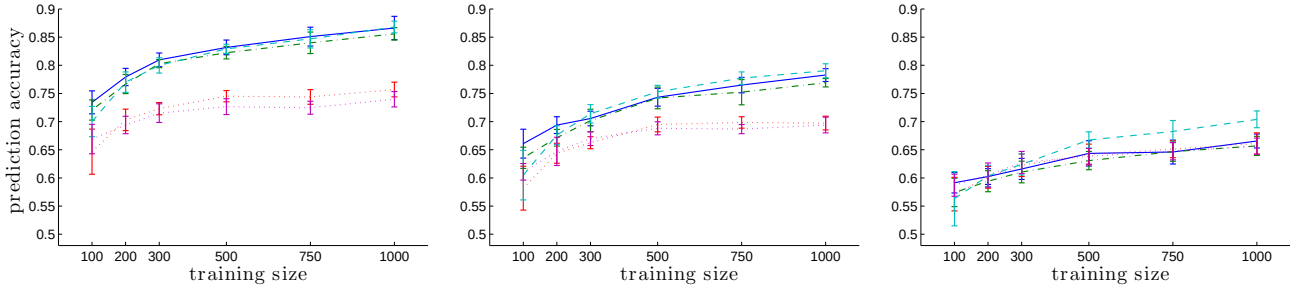

Figure 8: Classification performance, as in Figure 6, on a 2-class, odd versus even MNIST digit classification task. Feature vectors were obtained by binarizing coefficients with respect to the first 50 principal components of MNIST. Note AClass' competitive performance with polynomial regularized last squares when 0% and 25% of feature values are missing, and the crossover to naive Bayes' lower performance when 50% of feature values are missing.

Roweis (2006). This dataset consists of text documents belonging to one of four broad categories (the `comp.*`, `rec.*`, `sci.*`, and `talk.*` usenet hierarchies). For each document, there are 100 binary features indicating the presence or absence of each of 100 common non-stop words. Figure 7 summarizes these results. As above, we report average 0-1 loss on a held-out test set of 500 for all algorithms. Qualitative properties of the performance comparison are consistent with the synthetic tests above, with performance degrading gracefully (and remaining above the discriminative approaches) as features are made missing at random. It is worth noting that in this data set, AClass appears to be finding a single mode per class, likely due to the inherent feature sparsity and lack of hyperparameter inference, recovering the performance of naive Bayes.

For our fourth experiment, we assess performance on a binarized variant of MNIST, collapsed into an odd-vs-even digits classification task. Following standard preprocessing procedures, we first extracted the top 50 principal components from 5000 randomly chosen examples, each originally 784-dimensional vectors with discrete values in the range [0,255]. We then binarized the components (by taking their sign); this binary data was the input to all algorithms. Our results are in Figure 8. When all data is observed, AClass performs competitively with polynomial RLS and significantly outperforms naive Bayes and logistic regression, suggesting the helpfulness of a properly regularized nonlinear decision boundary. However, once half the feature values are missing, AClass and naive Bayes perform almost identically, likely because AClass no longer has sufficient evidence to sustain multimodal class conditional densities. Without preprocessing via PCA, on fully observed data, AClass performs as naive Bayes, and is beaten by polynomial RLS.

4 Discussion

Our model and inference algorithm are both conceptually attractive, as they each address a significant limitation of previous work – namely, principled choice of the complexity of the class-conditional density estimators and support for tractable, online inference. It is the combination that enables us to construct AClass, an algorithm that yields good computational and generative performance while still being grounded in a simple procedure and intuitive inductive bias.

This model/algorithm pair is also interesting from the standpoint of the broader generative/discriminative debate in prediction. By 'generative model', we mean a model for datapoints and labels which includes a model for the class-conditional densities, where prediction (at test time) is done by Bayes' rule. By 'discriminative model', we mean a model for only the probability of a label given a datapoint[2].

Typically, discriminative models admit a larger space of possible decision boundaries than generative ones, and are often nonparametric (e.g. SVMs, RLS). Therefore discriminative models often perform better asymptotically, but can overfit on very small data sets. Generative models, on the other hand, can perform poorly in the large data limit if they don't allow arbitrarily complex class-conditional densities via nonparametric density estimators; this is the case with e.g. finite mixture models and certainly with naive Bayes. Furthermore, if the inference approximations used to train a generative model don't accurately incorporate Bayesian model averaging, as is the case with standard approaches like EM-trained finite mixture models, prediction performance may be quite poor for very small data sets due to overfitting. This leaves little room for generative methods to perform well, supporting the conventional wisdom that discriminative approaches are more appropriate for classification problems.

This conventional wisdom can be refined based on our algorithm and experiments. Generative methods such as AClass can in fact exploit the power of nonparametrics via appropriate class-conditional modeling, as well as perform competitively on a range of data set sizes via tractable approximate inference. However, they can still suffer relative to discriminative methods in the presence of large numbers of noisy or irrelevant features (as in raw MNIST, without PCA preprocessing), by modeling aspects of the data that do not vary across classes. One possible remedy would be to construct generative models where only part of the features of each datapoint (e.g. a subset of features, or an additive component for all features) is modeled conditioned on the class label, while the rest is modeled using a single set of parameters shared across all classes. This would in some sense be a probabilistic elaboration on PCA-based preprocessing. Hyperparameter learning, ideally involving coupling across classes to support ignoring universally noisy or irrelevant features, is one very simple means of addressing this. Our polynomial RLS implementation, for example, used leave-one-out cross-validation to adjust kernel bandwidth, a step in this direction.

Another approach, supported by the surprisingly good performance of polynomial RLS with online mean imputation, would be to build hybrid classifiers out of one generative model and one discriminative model.

[2]Training procedures for discriminative models might well be explicitly Bayesian or have Bayesian interpretations (as with regularized least squares and logistic regression), but this simply means they are *probabilistic*, not generative.

The trained generative model would provide priors on missing labels and features, which could be used in fitting and testing a discriminative model. One simple, heuristic way to realise this combination, using existing generative and discriminative algorithms, would be to train a generative model, fill in missing features (or even labels, during training) according to the mean of its posterior predictive distribution, and use this filled-in data to train and test a discriminative algorithm.

The most salient avenue for future work on AClass is in empirical evaluation. Developing a parallel implementation and doing an in-depth classification benchmark study, including other discriminative methods such as the Forgetron and other kernel machines, will be key; these were omitted due to the preliminary nature of our present experiments. We also note that constant-memory applications (such as classification on embedded devices) could enforce a hard bound by constraining the Chinese restaurant process to produce no more than some constant number (say G^+) of groups for each class. This is *not* equivalent to fixing the number of groups in advance; each partition with a number of groups less than or equal to G^+ remains a distinct hypothesis in the space, is considered during the course of inference, and has finite probability.

Furthermore, our inference algorithm can be elaborated in many ways to improve its performance and increase its scope. For example, the transformation from the AClass model to the chain in Figure 2 motivates a general strategy for sequentializing Bayesian inference (over datapoints and over numbers of variables). This strategy permits more sophisticated particle filters, including ones that use proposal distributions obtained via exactly or approximately summing out over future values of the latent states, as well as filters that alternate between particle filtering updates and Gibbs sampling of past assignments. We are currently developing this general case in a separate paper, as well as working on their application to the AClass model to reduce the sensitivity of the one-particle approximation to mistakes in the beginning of a data set.

Finally, it should be possible to develop approximations to inference in the AClass model suitable for situations with missing labels as well as missing features. In particular, one could attempt to estimate missing class labels by a nested particle filtering scheme, where an outer particle filter assigns labels to unlabeled datapoints, and each particle in this filter contains an instance of the AClass algorithm. Even in data sets where AClass falls back on a single mode per class, such a strategy remains online, and therefore a potentially appealing alternative to EM.

5 Conclusion

We have presented AClass, a simple, online, parallelizable learning algorithm for supervised multiclass classification in the presence of missing data. Our algorithm operates by sequential Monte Carlo inference in a mixture of Chinese restaurant process mixtures. This generative model encodes a simple inductive bias, namely prototype discovery for each class, and our algorithm performs well in preliminary experiments.

Classification algorithms grounded in generative probabilistic models can support principled treatment of missing feature values and inference over unlabeled training examples, both of which are important in real-world applications. However, these advantages must be achieved while retaining algorithmic tractability and good generalization performance. We hope that AClass (and the generative model in which it operates) represents a useful step in this direction.

Acknowledgments

The authors would like to acknowledge Keith Bonawitz and Charles Kemp for helpful discussions. Tom Minka generously provided logistic regression code. This work was partially supported by NTT Communication Science Laboratories and Eli Lilly and Company.

References

O. Dekel, S. Shalev-Shwartz, and Y. Singer. The forgetron: A kernel-based perceptron on a fixed budget. In *NIPS 18*. 2006.

A. Doucet, N. De Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. 2001.

P. Fearnhead. Particle filters for mixture models with an unknown number of components. *Statistics and Computing*, 14:11–21, 2004.

A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis, Second Edition*. Chapman & Hall, 2003.

A. Halberstadt and J. Glass. Heterogeneous measurements and multiple classifiers for speech recognition. In *Proceedings of ICSLP*, 1998.

R. M. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9:249–265, 2000.

K. Nigam. Using Unlabeled Data to Improve Text Classification. Technical Report CMU-CS-01-126, Computer Science Department, Carnegie-Mellon University, 2001.

C. Rasmussen. The infinite gaussian mixture model. In *NIPS*, 2000.

R. Rifkin and R. Lippert. What you should know about matrix decompositions and regularized least squares. Unpublished manuscript, 2006.

S. Roweis. `http://www.cs.toronto.edu/roweis/data.html`, 2006.

A. N. Sanborn, T. L. Griffiths, and D. J. Navarro. A more rational model of categorization. In *COGSCI*, 2006.