Family Discovery

Stephen M. Omohundro
NEC Research Institute
4 Independence Way, Princeton, NJ 088540
om@research.nj.nec.com

Abstract

Many learning situations involve multiple sets of training examples drawn from different but related underlying models. "Family discovery" is the task of discovering a parameterized family of models from this kind of training set. The task naturally arises in density estimation, classification, regression, manifold learning, reinforcement learning, clustering, HMM learning, and other settings. We describe three family discovery algorithms which are based on techniques for manifold learning. We compare these algorithms on a classification task against two alternative approaches and find significant performance improvement.

1 INTRODUCTION

Multi-speaker speech recognition systems are trained on sets of sentences spoken by different speakers. Some current systems train separate models for male and female speakers. During recognition, they attempt to detect the sex of the speaker and thereby select the appropriate recognizer [5]. A powerful extension of this approach would be a system that could learn a family of recognizers parameterized by the speaker's accent. During recognition, the system would first identify the accent and then use the appropriate recognizer for it.

Character recognition is similar. One would like a system which learns a family of recognizers parameterized by the document's font [7]. For recognition, the system

To appear in: Michael Mozer, "Advances in Neural Information Processing Systems 8", 1996, MIT Press, Cambridge, MA.

would first detect a document's font type, and use this to select the appropriate character recognizer.

These are two examples of what we call "family discovery". The training set is partitioned into "episodes". The examples in each episode come from a single model, but different episodes may come from different models from a parameterized family of models. The learning task is to discover the underlying family. If successful, there is the potential for improved performance both in inducing a more correct model for each episode and in better interpreting new test examples.

There are two primary alternatives to family discovery: 1) use separate models for each episode (eg. separate recognizers for each speaker) or 2) try to fit a single model to the data from all episodes. The first approach eliminates the possibility of inductive generalization from one set to another. The second approach ignores the information that the different training sets came from distinct models.

We will describe algorithms for family discovery based on techniques for "manifold learning" [1, 2, 3]. We propose three algorithms and compare them with the two alternatives that ignore family structure. The first algorithm models a family by a single affine subspace of the parameter space. The second algorithm represents a family by smoothly blending a set of local affine models together. The third algorithm directly represents a family by a mapping from the family parameter space into the model parameter space.

2 THE FIVE ALGORITHMS

The family discovery task is fairly independent of both the learning algorithm used for individual models and even the nature of individual learning task. The techniques are applicable to a wide variety of neural networks, statistical models, and machine learning algorithms. The modifications required for each particular learning task or algorithm should be clear.

In each case we assume that there is a model space parameterized by θ (eg. the weights of a neural network). θ-space will typically have a much higher dimension than the family we are seeking. If x is a partial or complete data example, then we assume that the individual learning algorithm can compute $p_\theta(x)$, the probability of example x under model θ. For classification and regression tasks, a test query will specify only the "input" dimensions and the model will respond with a probability distribution over the "output" dimensions. In general, the probability of partial examples is computed by integrating over any missing dimensions. This allows the framework to apply to missing "input" dimensions as well [8].

We assume that the true models are actually drawn from a d-dimensional family parameterized by γ. The training set is assumed to be partitioned into N "episodes". Episode i consists of N_i training examples t_{ij}, $1 \leq j \leq N_i$. Each training example in an episode i is assumed to come from a single underlying model with parameter θ_i^*. A "family disovery" learning algorithm should use this training data to build a model of the underlying model family. The algorithm is tested by presenting it with a set of N_{test} test examples x_k, $1 \leq k \leq N_{test}$ drawn from a single model θ_{test}. These test examples may be complete, partial, or mixed depending on the underlying learning task. A family discovery algorithm should be able to estimate θ_{test} and use it to make predictions about any missing data components (eg. to predict "output" values in a classification or regression task).

We now describe the five algorithms we tested. We describe the algorithms in terms of underlying learning models which attempt to maximize the likelihood of the data. It should be clear how to additionally incorporate regularizers, additional prior information, or "Occam" terms [6] into the procedures.

We define the model likelihood for episode i, where $1 \leq i \leq N$ by

$$L_i(\theta) = \prod_{j=1}^{N_i} p_\theta(t_{ij})$$

and the test likelihood L_{test} is defined similarly.

The family discovery algorithms are all based on representing a "family prior" $m(\theta)$ on the parameter space. During recognition, the model parameter θ is chosen to maximize the product of this prior and the data likelihood (this product is proportional to the posterior). The three family discovery algorithms correspond to different ways of representing and learning this prior $m(\theta)$. In each case, it is defined in terms of a projection operator P from the parameter space to itself via:

$$m_{patch}(\theta) = e^{-(\theta - P(\theta))^2}.$$

P should map an arbitrary model onto the closest model in the family being represented. For each algorithm, we will describe how P is represented and learned. In each case the dimension of the underlying family is discovered using techniques described in [1]. A local principle components analysis is performed on the best-fit episode model parameters and the family dimension corresponds to a gap in the principal values.

2.1 Single Model

The first algorithm is one of the simple alternatives meant for comparison with the other algorithms. It ignores the partitioning of the data into separate episodes and fits a single model to all of the data by maximizing the overall likelihood of a model:

$$L(\theta) = \prod_{i=1}^{N} L_i(\theta).$$

2.2 Closest Separate Model

The second algorithm is also meant for comparison with the other algorithms. It fits a separate model for each training episode. During recognition, it selects the training episode model which best fits the test data and uses this for recognition.

2.3 Affine Family

The affine model represents the underlying model family as an affine subspace of the model parameter space. The projection operator P_{affine} projects a parameter vector θ orthogonally onto the affine subspace. This subspace is determined by selecting the top principle vectors in a principle components analysis of the best-fit episode model parameters. A variant of the Expectation Maximization algorithm would iterate refitting the best-fit episode models in the context of the affine family, and the affine family from the episode models.

2.4 Affine Patch Family

The next approach is based on the "manifold learning" procedure described in [1]. The family is represented by a collection of affine patches which are blended together by Gaussian influence functions. The projection mapping P_{patch} is a smooth convex combination of projections onto the affine patches:

$$P_{patch}(\theta) = \sum_{\alpha=1}^{m} I_\alpha(\theta) A_\alpha(\theta)$$

where A_α is the projection operator for an affine patch and

$$I_\alpha(\theta) = \frac{G_\alpha(\theta)}{\sum_\alpha G_\alpha(\theta)}.$$

is a normalized Gaussian blending function. The patches are initialized by peforming k-means clustering of the episode models to choose k patch centers. A local principle components analysis is performed on the episode models which are closest to each center. The Gaussian influence functions and the affine patches are then updated by the Expectation Maximization algorithm [4]. With the affine patches held fixed, the Gaussians G_α are refit to the errors each patch makes in approximating the episode models. Then with the Gaussians held fixed, the affine patches A_α are refit to the epsiode models weighted by the the corresponding Gaussian G_α. As with the affine model, the entire estimation may be repeated with episode model parameters that take into account the family prior.

2.5 Map Mixture Family

The previous approach represented the learned model family in terms of a collection of local patches. This allows topologically non-trivial families to be represented, but does not explicitly parameterize the family. Operations such as optimization searches are more convenient with a parameterization. The final approach therefore attempts to directly learn a parameterization of the model family.

Recall that the model parameters come from θ-space, while the family parameters come from γ-space. We represent a family by an affine mapping G from θ-space to γ-space together with a nonlinear mapping F from γ-space back to θ-space. The corresponding projection operation is

$$P_{map}(\theta) = F(G(\theta)).$$

The affine map $G(\theta)$ defines the family parameter γ on the full θ-space. It is determined by a principle components analyis of the best-fit episode model parameters. Once G is defined, then F is chosen to minimize the difference between $F(G(\theta_i))$ and θ_i for each best-fit episode parameter θ_i.

Any form of trainable nonlinear mapping could be used for F (eg. backprop neural networks or radial basis function networks). We represent F as a mixture of experts [4] where each expert is an affine mapping and the mixture coefficients are Gaussians. The mapping is trained by the EM algorithm.

3 ALGORITHM COMPARISON

Given the motivating speech and character recognition tasks, it is natural to test these algorithms on a classification task. We consider a two-class task with unit-variance normal class-conditional distributions on a 5-dimensional feature space. The means of the class distributions are parameterized by a nonlinear two-parameter family:

$$\begin{aligned} m_1 &= (\gamma_1 + \tfrac{1}{2}\cos\phi)\hat{e_1} + (\gamma_2 + \tfrac{1}{2}\sin\phi)\hat{e_2} \\ m_2 &= (\gamma_1 - \tfrac{1}{2}\cos\phi)\hat{e_1} + (\gamma_2 - \tfrac{1}{2}\sin\phi)\hat{e_2}. \end{aligned}$$

where $0 \leq \gamma_1, \gamma_2 \leq 10$ and $\phi = (\gamma_1 + \gamma_2)/3$. The class means are kept a distance of 1 apart, ensuring significant class overlap over the whole family. The angle ϕ varies with the parameters so that the correct classification boundary changes orientation over the family. This choice of parameters introduces sufficient non-linearity in the task to distinguish the non-linear algorithms from the linear one.

Figure 1 shows the comparative performance of the 5 algorithms. The x-axis is the total number of training examples. Each set of examples consisted of approximately $N = \sqrt{x}$ epixodes of approximately $N_i = \sqrt{x}$ examples each. The classifier parameters for an episode were drawn uniformly from the classifier family. The episode training examples were then sampled from the chosen classifier according to the classifier's distribution. Each of the 5 algorithms was then trained on these examples. The number of patches in the surface patch algorithm and the number of affine

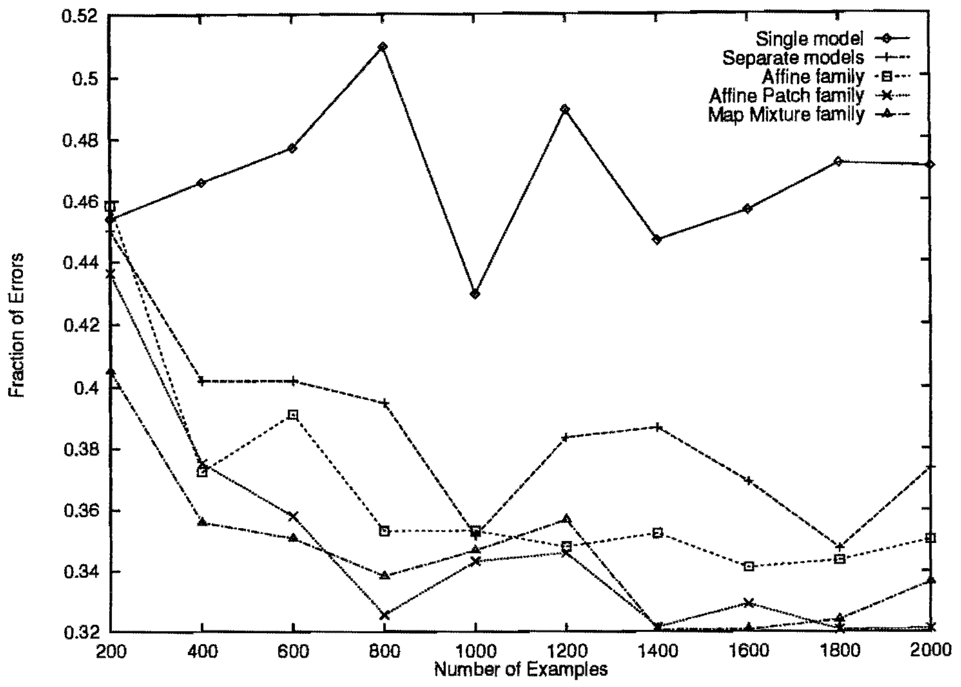

Figure 1: A comparison of the 5 family discovery algorithms on the classification task.

components in the surface map algorithm were both taken to be the square-root of the number of training episodes.

The y-axis shows the percentage correct for each algorithm on an independent test set. Each test set consisted of 50 episodes of 50 examples each. The algorithms were presented with unlabelled data and their classification predictions were then compared with the correct classification label.

The results show significant improvement through the use of family discovery for this classification task. The single model approach performed significantly worse than any of the other approaches, especially for larger numbers of episodes (where the family discovery becomes possible). The separate model approach improves with the number of episodes, but is nearly always bested by the approaches which take explicit account of the underlying parameterized family. Because of the nonlinearity in this task, the simple affine model performs more poorly than the two nonlinear methods. It is simple to implement, however, and may well be the method of choice when the parameters aren't so nonlinear. From this data, there is not a clear winner between the surface patch and surface map approaches.

4 TRAINING SET DISCOVERY

Throughout this paper, we have described the tasks as if a teacher explicitly tells the learner which training examples consitute each training set. An agent interacting with the world may not get this explicit information. For example, a speech recognition system may not be explicitly told when it is conversing with a new speaker. Similarly, a character recognition system would probably not have explicit information about font changes. The learner can sometimes use the data itself to detect these changes, however. In many situations there is a strong prior that successive events are likely to have come from a single model and only occasionally does the model change. In this setting, the boundaries between the training sets may naturally be found using the EM algorithm. An initial partitioning of the training examples into episodes is made. A family is fit to these episodes. The partition is then redone according to similarity of the induced family parameters. A similar approach may be used in cases where the model parameters are known to vary slowly with time rather than to exhibit rare discontinuous jumps.

5 CONCLUSION

We have described three general algorithms for family discovery and compared their performance with two alternatives on a classification task. For this task, the use of family discovery led to significantly improved classification performance. The approach can be applied to a wide variety of tasks and types of underlying models. It is also clear that there are many variants of the algorithms we propose. We have not yet applied these algorithms to "accent-based" speech recognition or "font-based" character recognition but our experiments suggest that it may be fruitful to do so.

References

[1] Christoph Bregler and Stephen M. Omohundro. Surface learning with applications to lipreading. In Jack D. Cowan, Gerry Tesauro, and Josh Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 43–50, San Francisco, CA, 1994. Morgan Kaufmann Publishers.

[2] Christoph Bregler and Stephen M. Omohundro. Nonlinear image interpolation using manifold learning. In Gerry Tesauro, David S. Touretzky, and Tod K. Leen, editors, *Advances in Neural Information Processing Systems 7*, Cambridge, MA, 1995. MIT Press.

[3] Christoph Bregler and Stephen M. Omohundro. Nonlinear manifold learning for visual speech recognition. In W. Eric L. Grimson, editor, *Proceedings of the Fifth International Conference on Computer Vision*, June 1995.

[4] Michael I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.

[5] Rich Lippmann. Speech technology and naive users, lecture at Emerging Opportunities Workshop, Key West, Florida, 1995.

[6] David J. C. MacKay. Probable networks and plausible predictions - a review of practical bayesian methods for supervised neural networks. *Network*, to appear(to appear), 1995.

[7] Michael Revow, Christopher K. I. Williams, and Geoffrey Hinton. Using generative models for handwritten digit recognition. Technical report, University of Toronto, 1994.

[8] Howard Scott Roy. *Sharp, Reliable Predictions Using Supervised Mixture Models.* PhD thesis, Stanford University, 1995.

To appear in: Michael Mozer, "Advances in Neural Information Processing Systems 8", 1996, MIT Press, Cambridge, MA.