SPEAKER RECOGNITION USING LOCAL MODELS

Ryan Rifkin

Honda Research Institute U.S.A.
145 Tremont Street
Boston, MA 02111

ABSTRACT

Many of the problems arising in speech processing are characterized by extremely large training and testing sets, constraining the kinds of models and algorithms that lead to tractable implementations. In particular, we would like the amount of processing associated with each test frame to be sublinear (i.e., logarithmic) in the number of training points. In this paper, we consider smoothed kernel regression models at each test frame, using only those training frames that are close to the desired test frame. The problem is made tractable via the use of approximate nearest neighbors techniques. The resulting system is conceptually simple, easy to implement, and fast, with performance comparable to more sophisticated methods. Preliminary results on a NIST speaker recognition task are presented, demonstrating the feasibility of the method.

1. INTRODUCTION

We consider the problem of text-independent speaker recognition. Traditional appproaches to this problem include the use of Gaussian mixture models (GMM) [1] and neural networks [2]. More recently, discriminative regularization approaches based on SVMs [3], polynomial regression classifiers [4], or extensions thereof [5], have demonstrated very high accuracy. These discriminative approaches are able to outperform generative approaches such as GMM's because they avoid making inappropriate parametric assumptions (such as Gaussianity) about the underlying distribution. However, for the very large datasets involved in speech processing applications (assuming that individual data points correspond to single frames of audio), we hypothesize that simple nonparametric density estimation methods may be able to achieve accuracies comparable to discriminative approaches such as SVMs. Certainly, given sufficient amounts of data, we can estimate densities arbitrarily accurately. The question becomes, how much data is enough?

Nonparametric density methods, implemented naively, would require test-time computation linear in the size of the training set (as each test point is compared against every training point). To alleviate this problem, we suggest the use of *local models* — for each test point, only those training points that are relatively close to the test point are considered.

The basic approach is extremely simple. We consider a straightforward kernel regression classifier, based on the Parzen windows density estimation technique. To achieve computational tractability, we only consider those training points that are closest to a given test point; we find these points rapidly using fast approximate nearest neighbors techniques.

In Section 2, we discuss the smoothed Parzen window model we use for classification. In Section 3, we describe the approximate nearest neighbors techniques that allow us to efficiently deal with massive datasets. In Section 4, we describe a preliminary experiment on the NIST 1998 speaker recognition task. finally, in Section 5, we present several additional comments on the scheme, discuss the weaknesses and advantages of the present work, and suggest directions for future research.

Throughout this paper, we assume that the individual data points are feature vectors derived from frames of audio data, and that the training set consists of ℓ data points in d dimensions.

2. PARZEN WINDOWS AND KERNEL REGRESSION

The Parzen window scheme [6] is a very simple nonparametric density estimation technique. Given the training set $X = \{x_1, \ldots, \mathbf{x}_\ell\}$, the Parzen window scheme produces the following estimated density function:

$$\hat{p}(\mathbf{x}) = \frac{1}{\ell} \sum_{i=1}^{\ell} R(x - x_i),$$

where R is the *smoothing function*

$$R(x - x_i) = \frac{1}{(2\pi)^{\frac{d}{2}} \sigma^d} \exp\left(-\frac{||x - x_i||^2}{2\sigma^{2d}}\right). \quad (1)$$

The choice of the smoothing parameter σ controls the relative contributions of close and far points to the estimated

density at x. As $\sigma \to 0$, $R(x - x_i) \to \delta(x - x_i)$ and $\hat{p}(x) \to \frac{1}{\ell} \sum_{i=1}^{\ell} \delta(x - x_i)$; the estimated density becomes a collection of delta functions located at the training points. On the other hand, as $\sigma \to \infty$, the density tends to zero everywhere.

The Parzen windows technique for density estimation leads directly to the kernel regression technique for binary classification. Assuming that the y values are 1 and -1 (for points inside and outside the target class, respectivley), we compute a classification score using:

$$c(x) = \frac{\sum_{i=1}^{\ell} y_i R(x - x_i)}{\sum_{i=1}^{\ell} R(x - x_i)} \tag{2}$$

Although the Parzen window technique is simple, it is extremely accurate given large datasets. It is not hard to show that if we let $\sigma \to 0$ slowly as $\ell \to \infty$, $\hat{p}(x) \to p(x)$ (the estimated density goes to the true density in the mean-square sense). Additionally, it is possible to derive finite sample-size bounds on the quality of the density estimation, similar in flavor to the generalization error bounds for Support Vector Machines [7].

A more intuitive way to motivate this is to note that discriminative regularization methods, such as Support Vector Machines, Regularized Least-Squares Classification [7, 8], or the Relevance Vector Machine [9], which give the best currently known performance on many datasets, fit models of the following form:[1]

$$f(x) = \sum_{i=1}^{\ell} c_i R(x - x_i).$$

As the training size grows, we can show that for both SVM and RLSC, it is appropriate to constrain the model so that all the $c_i \to 0$ simultaneously; we may therefore intuit (this is of course not a proof) that for large datasets there may not be much difference between fitting the c_i explicitly and simply setting $c_i = \frac{1}{\ell}$ for all i. The Parzen window approach has the advantage that it requires no training whatsoever, as opposed to the solution of a linear system for mean-square error training or RLSC, or a quadratic programming problem for SVM.

Unfortunately, compared to the SVM or the RVM, the Parzen window scheme is at a crippling computational disadvantage. Both the SVM and the RVM exhibit *sparsity*: as a byproduct of the way the classifiers are formulated, nearly all of the c_i are zero (or in the case of the RVM, sufficiently close to zero that they can be thresholded away). At test time, we can compute $f(x)$ for such a point by computing the distances only to those points with non-zero c_i. By contrast, to compute the output of the Parzen window scheme, we must compute the distances to *all* ℓ points. This linear dependance of testing time on the size of the training set makes the classical Parzen window scheme unacceptable for large datasets.[2]

[1] In many formulations, the models optionally include a "bias" term b ($f(x) = \sum c_i R(x - x_i) + b$), but this is not relevant to the present discussion.

3. APPROXIMATE NEAREST NEIGHBORS TECHNIQUES

Suppose that, given a test point x, the points in X are ordered in increasing distance from x:

$$||x - x_1||^2 \leq ||x - x_2||^2 \leq \ldots \leq ||x - x_\ell||^2.$$

Noting that the smoothing function R yields an inverse exponential relation between the contribution of x_i to $\hat{p}(x)$ and the $||x - x_i||^2$, we consider the idea of truncating the representation of $\hat{p}(x)$ to $k \ll \ell$ terms:

$$\hat{p}(x) = \sum_{i=1}^{k} R(x - x_i).$$

Using the direct approach, finding the k closest points involves finding the distance to all ℓ points as a subproblem. A large body of work has been devoted to finding the k nearest neighbors more rapidly. The basic idea behind these methods is to create a data structure which recursively partitions the input space, and then to search this data structure as an alternative to searching the entire training set. Friedman, Bentley and Finkel [10] showed that kd-trees could provide expected case performance of $O(\ell)$ space and $O(\log \ell)$ time to find a nearest neighbor for *fixed* dimensionality d; extensions and improvements to this approach can be found in [11, 12, 13]. Although these methods require $O(\ell)$ storage space and $O(\log \ell)$ query time *for fixed dimensionality* d, there is an exponential dependance on d in time, space, or both, in both theory and practice [14].

We can do better if we are willing to relax our requirements and consider *approximate nearest neighbors* formulations — a kth $(1 + \epsilon)$-approximate nearest neighbor for a given query point is a point that is no further from the query point than $(1 + \epsilon)$ times the distance to the true kth nearest neighbor. Relatively recent results [15, 16] have introduced algorithms with no exponential dependance on the dimensionality. In this work, we use instead the approach developed by Arya et al. [17], for two main reasons. The first is that although the worst-case query time for their method is exponential in the dimensionality (the query time to find the first k ϵ-approximate nearest neighbors is $O((d\lceil 1 + 6\frac{d}{\epsilon}\rceil^d + kd)\log \ell)$), in practice this method performs quite well even in several dozens of dimensions. The second is that there is a publicly available toolkit [18] which implements the algorithm.

[2] For small datasets, the computational time might be acceptable, but we would expect the regularization methods to be substantially more accurate.

4. EXPERIMENTAL WORK

We consider a subset of the 1998 NIST speaker recognition one-speaker detection task [19]. Specifically, we trained using the male, two-session-enrollment training data, and tested on the 30 second, "same phone number" segments. Each sound file is segmented into 25ms frames at a frame rate of 100 frames/second. For each frame, 15 cepstral and delta-cepstral features are extracted. For each file, only the frames with the largest first component (energy) were kept. The first cepstral (and deltra-cepstral) coefficient was removed, resulting in a 28 dimensional feature vector.

We then used the ANN toolkit [18] to construct a data structure containing all the training data. For each test frame, we extracted the 500 closest training frames. Given the ith neighbor of a frame, we let $m(i)$ denote the speaker that training frame was taken from. A single frame was scored against a specific model m using the following simplified versions of Equations 1 and 2; these simplifications make no difference given that we are actually interested in classification rather than regression, as they do not affect the relative ordering of scores:

$$s(x) = \frac{\sum_{m(i)=m} R(x, x_i)}{\sum_i R(x, x_i)},$$

where

$$R(x, x_i) = \exp\left(\frac{||x - x_i||^2}{\sigma}\right).$$

Given an utterance, the score for a model for the entire utterance is simply the sum of the scores for the individual frames.

There were 250 speakers in the training set, and 2 one-minute training utterances for each speaker. We experimented with keeping 500, 1000 and 2,000 frames per utterance, resulting in training sets of size 500,000, 1,000,000 and 2,000,000 data points, respectively. For each test utterance, 500 frames were kept. Informal experiments indicated that $\sigma = .5$ was a good choice for all model sizes. The results are shown in figure 1. We see that using 1,000,000 or 2,000,000 points results in improved accuracy over using 500,000 points; the equal error rate is better for 2,000,000 points than for 1,000,000, although the difference is not large.

The approximate nearest neighbors framework is insensitive to the data set size ℓ; empirically, extracting k nearest neighbors seems to take time $O(k\ell d + \log \ell)$. As a consequence, the total test time for the three different data set sizes is essentially identical; the "training time", which is simply the time required to build the ANN structure, is negligible compared to the testing time. On the other hand, we do pay linearly for the number of testing frames kept, and the number of nearest neighbors extracted per frame.

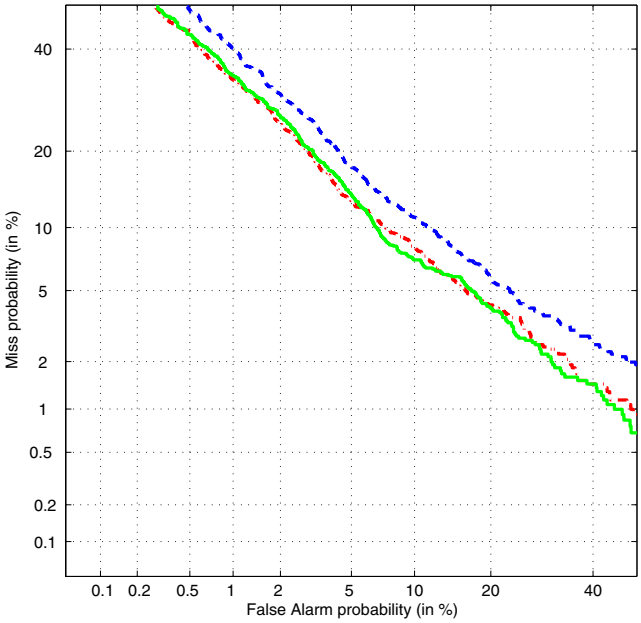

Fig. 1. Results on the NIST 1998 same phone number, male, 30 second classification task. The (blue) dashed line corresponds to a model trained with 500,000 training points, the (red) dotted line to a model trained with 1,000,000 points, and the (green) solid line to a model trained with 2,000,000 points.

5. DISCUSSION AND CONCLUSIONS

We have introduced and demonstrated the feasibility of the use of local models for speaker recognition. Although the initial results are not as good as those of [5] on the same data, the experiments are very preliminary, and further tuning (of either the features or the parameters of the algorithm) may yield improvements in performance.

An important issue is to what extent a data set can be considered "large". One operational definition is to describe a dataset as large if it is infeasible to spend time linear in the size of the training set in order to classify new points; it is relatively easy to determine whether or not this is the case. Another important characteristic of large datasets is that we expect the performance of discriminative classification and classification via nonparametric density estimation to be essentially the same. Because we cannot directly train most discriminative methods (such as SVMs) on datasets of half a million or more points, this condition is much more difficult to check. However, if we believe (for example by comparison to [5], after other conflating features such as differing feature sets have been accounted for) that our accuracy is not high enough, we could attempt to build discriminative, *local* models at each test point, using only that point's near neighbors. This idea was discussed and explored in [20, 21];

because the authors did not make use of approximate nearest neighbors techniques, the method was viewed as a new learning paradigm, rather than an efficiency consideration. This approach would be substantially more computationally intensive then the current methods, but could possibly still be tractable, and will be the subject of future study.

The initial version of the system is implemented in batch mode (all features are extracted, then all neighbors are computed, then all models are scored), but in terms of the amount of computation required, the system as described could be implemented to run in real time. The only training required by the system is the construction of the data structure to support the approximate nearest neighbor queries, (requiring $O(d\ell \log \ell)$ time), and this data structure can be *updated* in time $O(d \log \ell)$. This raises the intriguing possibility that this approach could be used for *online* speaker recognition, where the number and identity of speakers is not known a priori.

It is clear that as data sets grow very large, we will require methods that allow us to classify test points with an amount of computation sublinear in the size of the training set. This paper presents initial explorations of one such approach. Further work is required to determine whether the accuracy of this method can be brought in line with other state-of-the-art methods, while maintaining the computational advantages and simplicity of the current system.

6. REFERENCES

[1] Douglas A. Reynolds, "Automatic speaker recognition using gaussian mixture models," *The Lincoln Laboratory Journal*, vol. 8, no. 2, pp. 173–192, 1975.

[2] Kevin R. Farrell, Richard J. Mammone, and Khaled T. Assaleh, "Speaker recognition using neural networks and conventional classifiers," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 1, pp. 194–205, 1994.

[3] Nathan Smith, Mark Gales, and Mahesan Niranjan, "Data-dependent kernels in svm classification of speech patterns," Tech. Rep. CUED/F-INFENG/TR.387, Cambridge University Engineering Department, 2001.

[4] William M. Campbell and Khaled T. Assaleh, "Polynomial classifer techniques for speaker verification," in *Proceedings of the Internation Conference on Acoustics, Speech, and Signal Processing*, 1999, pp. 321–324.

[5] William M. Campbell, "Generalized linear discriminant sequence kernels for speaker recognition," in *Proceedings of the Internation Conference on Acoustics, Speech, and Signal Processing*, 2002.

[6] E. Parzen, "On estimtaion of a probability density functoin and mode," *Annals of Mathematical Statistics*, vol. 33, pp. 1065–1076, 1962.

[7] Vladimir N. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, 1998.

[8] Ryan M. Rifkin, *Everything Old Is New Again: A Fresh Look at Historical Approaches to Machine Learning*, Ph.D. thesis, Massachusetts Institute of Technology, 2002.

[9] Michael E. Tipping, "Sparse bayesian learning and the relevance vector machine," *Journal of Machine Learning Research*, vol. 1, pp. 211–244, 2001.

[10] Jerome H. Friedman, Jon Louis Bentley, and Raphael A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Transactions on Mathematical Software*, vol. 3, no. 3, pp. 209–226, 1977.

[11] A. C. Yao and F. F. Yao, "A general approach to d-dimensional geometric queries," in *Proceedings of the 17th Annual ACM Symposium on the Theory of Computing*, 1985, pp. 163–168.

[12] K. L. Clarkson, "An algorithm for approximate closest-point queries," *SIAM Journal on Computing*, vol. 17, no. 4, pp. 830–847, 1988.

[13] S. Meiser, "Point location in arrangements of hyperplanes," *Informatoin and Computation*, vol. 106, no. 2, pp. 286–303, 1993.

[14] R. L. Sproull, "Refinements to nearest-neighbor searching," *Algorithmica*, vol. 6, pp. 579–589, 1991.

[15] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing*, 1998, pp. 604–613.

[16] Eyal Kushilevitz, Rafael Ostrovsky, and Yuval Rabani, "Efficient search for approximate nearest neighbors in high dimensional spaces," in *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing*, 1998, pp. 614–623.

[17] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu, "An optimal algorithm for approximate nearest neighbor searching in fixed dimensions," *Journal of the ACM*, vol. 45, pp. 891–923, 1998.

[18] David M. Mount, "The ANN library," `http://www.cs.umd.edu/ mount/ANN/`.

[19] George R. Doddington, Mark A. Przybocki, Alvin F. Martin, and Douglas A. Reynolds, "The NIST speaker recogntion evaluation — overview, methodology, systems, results, perspective," *Speech Communication*, vol. 31, no. 2-3, pp. 225–254, 2000.

[20] Leon Bottou and Vladimir Vapnik, "Local learning algorithms," *Neural Computation*, vol. 4, pp. 888–900, 1992.

[21] V. Vapnik and L. Bottou, "Local algorithms for pattern recognition and dependency estimation," *Neural Computation*, vol. 5, pp. 893–909, 1993.