USING AGGREGATION TO IMPROVE THE PERFORMANCE OF MIXTURE GAUSSIAN ACOUSTIC MODELS[1]

Timothy J. Hazen and Andrew K. Halberstadt

Spoken Language Systems Group
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139 USA

ABSTRACT

This paper investigates the use of aggregation as a means of improving the performance and robustness of mixture Gaussian models. This technique produces models that are more accurate and more robust to different test sets than traditional cross-validation using a development set. A theoretical justification for this technique is presented along with experimental results in phonetic classification, phonetic recognition, and word recognition tasks on the TIMIT and Resource Management corpora. In speech classification and recognition tasks error rate reductions of up to 12% were observed using this technique. A method for utilizing tree-structured density functions for the purpose of pruning the aggregated models is also presented.

1. INTRODUCTION

Mixture Gaussian models are typically trained using a procedure that combines supervised and unsupervised training. Supervision is provided through the class labels of the training data which are known during training. In contrast, the weights which determine how much each data point contributes to the training of the mean and covariance of each individual mixture component are not known when training begins, but rather are determined during the training process in an unsupervised manner. The algorithms used to determine these weights, such as the K–means clustering algorithm and the Expectation-Maximization (EM) algorithm, do not guarantee a globally optimal solution. These algorithms often converge to a locally optimal solution, where the exact local optimum that will be reached is highly dependent on the initialization of the unknown weights at the beginning of the training process. As a specific example of this phenomenon, word recognition accuracies on the Resource Management task were obtained from 24 trials of training mixture Gaussian models by randomly-initialized K–means clustering followed by EM iterations. The average word error rate was 4.55%, with a maximum, minimum and standard deviation of 4.83%, 4.33%, and 0.127, respectively. The error rate reduction achieved in traversing from the worst trial to the best trial is 10.4%. In light of this variation, one may ask: What is a good strategy for using the results on development data to choose which models to use on an independent test set?

[1]This material is based upon work supported by NSF under Grant No. IRI-9618731 and by DARPA under Contract N66001-96-C-8526, monitored through Naval Command, Control, and Ocean Surveillance Center.

In the past, many have simply chosen the model training trial that performs the best on the development data, i.e., cross-validation. One problem with this strategy is that noise on the development data contributes a random component to the performance. As a result, better performance on the development set may not indicate models which are better matched to the true underlying distribution of the data. Instead, it may only indicate that the models are superficially better matched to the idiosyncrasies of the development set. As an example of this, TIMIT phonetic recognition accuracies were calculated on development and test sets for 24 independently trained models, using different random initializations in the K–means clustering. The correlation between development set and test set accuracy was indeed weak (correlation coefficient 0.16), and in this case the simple "pick-the-best-on-development-data" cross-validation strategy was particularly disappointing since the trial performing best on the development set performed worst on the test set. A second disadvantage of simple cross-validation is that computation is wasted, since the results of only one training trial are kept, while the models from the other trials are thrown away [1].

To counter the problems discussed above, an algorithm is needed which produces a mixture density function which can be proven to yield better classification accuracy, on average, than any randomly initialized density function trained using standard techniques. At the very least, it is desirable to show the new algorithm can reduce the value of some error function which is strongly correlated with classification accuracy. Aggregation is a technique which meets this criterion [1]. Aggregation improves the performance of classifiers which exhibit *uncertainty* or *instability* during their training phase. Aggregation has been applied to a variety of types of predictors and classifiers. For example, Breiman has shown the effectiveness of a specific type of aggregation known as *bagging* (or *bootstrap aggregating*) on linear regression predictors and on classification trees [2]. In [2], Breiman indicates aggregation could also be used on probabilistic classifiers.

In this paper, Breiman's work is extended by proving, both theoretically and empirically, how aggregation can be used to improve the performance and robustness of probabilistic classifiers. The theoretical development in Section 2 demonstrates that the mixture density function produced by aggregation produces a smaller error function than the average of the error functions of the individual models used in the aggregation. The theoretical advantages of aggregation are demonstrated empirically in Section 3 on speech classification and recognition tasks which utilize mixture Gaussian density functions.

2. THEORY

Aggregation of probabilistic classifiers is performed by averaging the outputs of a set of independently trained classifiers. The proof that follows will demonstrate that an aggregate classifier is guaranteed to exhibit an error metric which is equal to or better than the average error metric of the individual classifiers used during aggregation. Though the empirical evidence presented in this paper uses only mixture Gaussian classifiers, this proof is valid for any type of probabilistic classifier. This proof is also completely independent of the test data being presented to the classifier. Thus, the method is robust because it improves performance regardless of the test set being used.

To begin, assume a set of N different classifiers have been trained. In most of the experiments in this paper, multiple classifiers are generated from the same data set by using different random initializations in the K–means clustering prior to EM training of the mixtures. However, the proof does not depend in any way on how the classifiers are generated. This set will be called Φ and can be represented as:

$$\Phi = \{\vec{\varphi}_1(\vec{x}), \vec{\varphi}_2(\vec{x}), \ldots, \vec{\varphi}_N(\vec{x})\} \tag{1}$$

Within Φ, each classifier $\vec{\varphi}_n(\vec{x})$ takes an observation vector $\vec{x}$ as its input. The underlying class that $\vec{x}$ belongs to will be defined as $c(\vec{x})$. To classify $\vec{x}$, each $\vec{\varphi}_n(\vec{x})$ contains a probability density function for each of the D different classes from which $\vec{x}$ might be drawn. Each classifier $\vec{\varphi}_n(\vec{x})$ outputs a D dimensional vector containing the *a posteriori* probabilities of $\vec{x}$ belonging to each of the D classes. This output vector for $\vec{\varphi}_n(\vec{x})$ can be represented as

$$\vec{\varphi}_n(\vec{x}) = \begin{bmatrix} \varphi_{n,1}(\vec{x}) \\ \varphi_{n,2}(\vec{x}) \\ \vdots \\ \varphi_{n,D}(\vec{x}) \end{bmatrix} = \begin{bmatrix} \mathrm{P}_n(c(\vec{x}) = 1 \,|\, \vec{x}) \\ \mathrm{P}_n(c(\vec{x}) = 2 \,|\, \vec{x}) \\ \vdots \\ \mathrm{P}_n(c(\vec{x}) = D \,|\, \vec{x}) \end{bmatrix} \tag{2}$$

In order to evaluate the performance of a classifier, an appropriate metric must be defined. Let $\vec{y}(\vec{x})$ be a D dimensional vector which indicates the reference class, or "correct answer," through a binary representation. That is,

$$\vec{y}(\vec{x}) = [y_1(\vec{x}), y_2(\vec{x}), \ldots, y_D(\vec{x})]^T \tag{3}$$

where

$$y_d(\vec{x}) = \begin{cases} 1 & \text{if c}(\vec{x}) = d \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Ideally, a classifier's *a posteriori* probability for the correct class should be as close to 1 as possible while all of the incorrect classes should have *a posteriori* probabilities as close to 0 as possible. The error metric used in this proof is the squared distance between the classifier's output and the "correct answer." Thus, the *error* when input $\vec{x}$ is presented to the n^{th} classifier is defined as

$$e_n(\vec{x}) = \|\vec{y}(\vec{x}) - \vec{\varphi}_n(\vec{x})\|^2 \tag{5}$$

This error can be expressed as

$$e_n(\vec{x}) = \sum_{d=1}^{D} \left(y_d(\vec{x}) - \varphi_{n,d}(\vec{x})\right)^2 \tag{6}$$

Using the error metric defined above, the mean error over all N classifiers in Φ for an input vector $\vec{x}$ is

$$e(\vec{x}) = \frac{1}{N} \sum_{n=1}^{N} \sum_{d=1}^{D} \left(y_d(\vec{x}) - \varphi_{n,d}(\vec{x})\right)^2 \tag{7}$$

For notational ease, the remainder of this development will drop the explicit dependence on the input vector $\vec{x}$. It should be understood that the analysis proceeds identically given any input vector. Continuing, the average error of the N classifiers expands to

$$e = \sum_{d=1}^{D} \left[y_d^2 - \left(\frac{2y_d}{N} \sum_{n=1}^{N} \varphi_{n,d} \right) + \left(\frac{1}{N} \sum_{n=1}^{N} \varphi_{n,d}^2 \right) \right] \tag{8}$$

The aggregate classifier, $\vec{\varphi}_A$, simply averages the outputs of all N classifiers in Φ. This is expressed as

$$\vec{\varphi}_A = \frac{1}{N} \sum_{n=1}^{N} \vec{\varphi}_n \tag{9}$$

The error for the aggregate classifier model is

$$e_A = \|\vec{y} - \vec{\varphi}_A\|^2 = \sum_{d=1}^{D} (y_d^2 - 2y_d\varphi_{A,d} + \varphi_{A,d}^2) \tag{10}$$

By substituting in the definition of $\vec{\varphi}_A(\vec{x})$ from (9) the error of the aggregate classifier can be rewritten as

$$e_A = \sum_{d=1}^{D} \left[y_d^2 - \left(\frac{2y_d}{N} \sum_{n=1}^{N} \varphi_{n,d} \right) + \left(\frac{1}{N} \sum_{n=1}^{N} \varphi_{n,d} \right)^2 \right] \tag{11}$$

By comparing the expressions in (8) and (11), it can be seen that e_A will be less than or equal to e if

$$\left(\frac{1}{N} \sum_{n=1}^{N} \varphi_{n,d} \right)^2 \leq \frac{1}{N} \sum_{n=1}^{N} \varphi_{n,d}^2 \tag{12}$$

In fact, this condition is always true for any arbitrary vector because it is a special case of the Cauchy–Schwarz inequality. Given any two vectors $\vec{a} = [a_1, a_2, \ldots a_N]^T$ and $\vec{b} = [b_1, b_2, \ldots b_N]^T$, then by the Cauchy–Schwartz inequality

$$\left| \sum_{n=1}^{N} a_n b_n \right|^2 \leq \left(\sum_{n=1}^{N} a_n^2 \right) \left(\sum_{n=1}^{N} b_n^2 \right) \tag{13}$$

Now let $b_n = 1$ for all n so that $\sum_{n=1}^{N} b_n^2 = N$ to obtain

$$\left(\sum_{n=1}^{N} a_n \right)^2 \leq N \sum_{n=1}^{N} a_n^2 \tag{14}$$

which is the desired result. Note that equality holds in (12) only if the $\varphi_{n,d}$ matrix is constant along each row n, i.e., every classifier is giving exactly the same *a posteriori* probabilities. Thus, in practical situations with classifiers that produce different probabilities, the inequality becomes strict.

This derivation proves that, for *any* input token $\vec{x}$, the error e_A of the aggregate classifier created from the classifier set Φ is always smaller than the average error e of the N individual classifiers in Φ, provided that the N classifiers do not all produce the same *a posteriori* probabilities.

3. EXPERIMENTS

Overview: The effectiveness of aggregation will be demonstrated with results obtained on three different tasks: phonetic classification, phonetic recognition, and word recognition. Each experiment utilizes the SUMMIT [3] recognition system. This system uses mixture Gaussian acoustic models to score segment-based phonetic units. For each experiment, 24 individual sets of acoustic models were independently trained. These 24 individual trials were then used to create sets of independent models for each aggregation trial. Thus, the models from 24 individual training trials are used to create 12 2–fold aggregation trials, 8 3–fold aggregation trials, etc., until only one trial of 24–fold aggregation is performed.

Phonetic Classification: For the task of phonetic classification, the SUMMIT system was tested on the TIMIT corpus using context-independent phonetic segment models. The classifier and measurements used for these experiments are similar, but not identical, to those described in [4]. The classifier used a 71 dimensional feature vector to model the segments of 61 different phones. The feature vector contains MFCC and energy averages and derivatives, duration, zero-crossing rate and fundamental frequency. The classifier uses mixtures of full-covariance Gaussian density functions. In Table 1, there are two different context-independent phonetic classification (CI-PC) conditions. In CI-PC1, each individual training trial requires a minimum of approximately 500 training tokens for each Gaussian kernel. In CI-PC2, only 300 tokens are required thus allowing more mixture components per model.

Phonetic Recognition: For the phonetic recognition task, SUMMIT was tested on the TIMIT corpus using context-independent phonetic segment models and context-dependent diphone boundary models. A 77 dimensional feature vector is used to model the segments of 61 different phonetic units. The 77 dimensions are primarily composed of averages and derivative of MFCC's and the total energy. For each segment model, a diagonal Gaussian mixture is created using a maximum of 50 Gaussian kernels. A 50 dimensional feature vector is used to model the boundaries for each of 983 diphone units. This feature vector contains MFCC and energy measurements taken from regions surrounding each boundary. For each boundary model a diagonal Gaussian mixture is created using a maximum of 50 Gaussian kernels. All models were trained on a 462 speaker training set. Phonetic recognition accuracies were computed on both a 50 speaker development test set (400 utts) and on the 24 speaker core test (192 utts). The recognizer used for these experiments is described in detail in [3].

Word Recognition: For the word recognition task, SUMMIT was tested on the DARPA Resource Management corpus. The recognizer used the same measurement set as the TIMIT phonetic recognizer used above. The phonetic unit set contained 67 phonetic units instead of 61, and the diphone unit set contained 558 diphone units instead of 983. Each segment model contained a maximum of 100 Gaussian kernels while each boundary model contained a maximum of 50 Gaussian kernels. To handle words, the system utilized a pronunciation network which accounted for multiple alternative pronunciations for each word. The system used the standard word pair grammar provided for the task (perplexity 60). The system was trained on all 120 speakers in the full training set (4400 utts) and tested on all 40 speakers in the full test set (1200 utts).

Results: A summary of the results on the three tasks is presented in Table 1. This table shows the average accuracy of N-fold aggregation over M independent trials on the three different tasks. As can be seen, aggregation improves accuracy of the acoustic models in all three cases. For phonetic classification, there are two different conditions labeled CI–PC1 and CI–PC2. CI–PC2 used more mixture components per class than CI–PC1, thus giving CI–PC2 the greater possibility of overfitting the training data. This is confirmed by the fact that CI-PC2 performs worse on average than CI-PC1 for 1-fold aggregation. However, the damaging effects of overfitting in CI-PC2 are largely overcome when multiple models are aggregated. For 24-fold aggregation, CI-PC2 achieves comparable performance to CI-PC1. Similarly, the CD-WR models use more mixture components than the CD-PR models, and likewise show a larger reduction in error rate using aggregation. These result indicates that aggregation improves robustness to sources of overfitting. Thus, when aggregation is used, it is not so important that there be a perfect balance between fitting and smoothing the training data on each model training trial. Instead, one can err somewhat on the side of overfitting, and aggregation will appropriately smooth the models.

Figure 1 shows the results on the task of phonetic recognition on the development set in more detail. In this figure, the average results for N-fold aggregation are presented as well as the best and worst individual trials for each N. As can be seen, the average

Task	Test Set	Average Error Rate (%) M=24 N=1	M=6 N=4	M=1 N=24	% Error Reduct.
CI-PC1	dev	21.2	20.0	19.6	7.1
CI-PC1	core	22.1	20.7	20.2	8.3
CI-PC2	core	23.2	21.3	20.4	12.0
CD-PR	dev	28.1	27.3	27.0	3.6
CD-PR	core	29.3	28.4	28.1	4.0
CD-WR	test	4.5	4.2	4.0	12.0

Table 1: Average accuracy of M trials of N–fold model aggregation for the tasks of context-independent phonetic classification on TIMIT (CI-PC1 and CI-PC2), context-dependent phonetic recognition on TIMIT (CD-PR), and context-dependent word recognition on Resource Management(CD-WR).

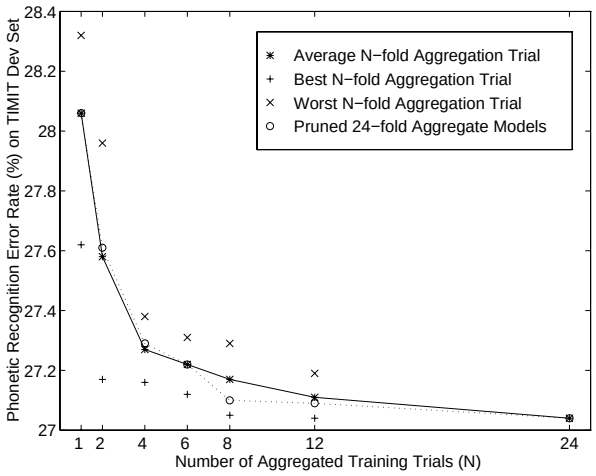

Figure 1: The effect of aggregation on the TIMIT context-dependent phonetic recognition task on the development set.

error rate performance over twelve 2-fold trials (27.58%) is better than the best single 1-fold trial (27.62%). Furthermore, the worst trial from six 4-fold trials (27.38%) is still better than the best single 1-fold trial (27.62%). Similar performance curves were observed for both phonetic classification and word recognition.

4. MODEL PRUNING

We have investigated the possibility of reducing the variance of the performance of a randomly trained N-fold model through the use of pruned tree-structured probability density functions [6]. We have constructed a hierarchical tree of the Gaussian kernels in the 24-fold models used for phonetic recognition. Pruning is performed by utilizing the model of a branch node as a replacement for the mixture of models of the leaf nodes which emanate from it. The branch nodes each contain a single Gaussian density function which estimates the actual mixture of Gaussians produced by the leaf nodes. It is hoped that the single Gaussian of a branch node provides a reasonable approximation of the likelihood function yielded by its leaf nodes. As pruning progresses backwards towards the root of the tree, less computation is required but more approximation is performed.

For our experiments we create a tree structure density with bottom-up clustering using a weighted Bhattacharyya distance. Given two Gaussian density functions, g_1, and g_2, the Bhattacharyya distance, $\mathrm{b}(g_1, g_2)$, is

$$\frac{1}{8}(\mu_1-\mu_2)^t\left(\frac{\Sigma_1+\Sigma_2}{2}\right)^{-1}(\mu_1-\mu_2)+\frac{1}{2}\ln\frac{|(\Sigma_1+\Sigma_2)/2|}{|\Sigma_1|^{\frac{1}{2}}+|\Sigma_2|^{\frac{1}{2}}} \quad (15)$$

Here μ_1 and Σ_1 are the mean and covariance of Gaussian 1 and μ_2 and Σ_2 are the mean and covariance of Gaussian 2. The weighted Bhattacharyya function used during clustering is

$$\mathrm{d}(g_1, g_2) = \frac{w_1+w_2}{w_1 w_2}\mathrm{b}(g_1, g_2) \quad (16)$$

Here, w_1 and w_2 are the mixture component weights. By using this weighting scheme, the clustering is biased toward clustering mixture components which have similar weights.

To test the effectiveness of the tree-structured pruning, the 24-fold set of TIMIT phonetic recognition models was pruned until it was the same size as a 12-fold model, then an 8-fold model, etc. The pruned models were tested on the TIMIT development set to compare their performance against standard N-fold aggregation. The results are also shown on Figure 1. As can be seen, the pruned 24-fold models obtain performances which are comparable to, if not better than, the average of the N-fold models of the same size. A similar curve is observed on the TIMIT core test set. This empirical result suggests that aggregating many models followed by pruning is advantageous because it is likely to achieve the *expected* N-fold performance.

In the future, we hope to further utilize the tree structure to reduce computation by performing the fast likelihood approximation technique suggested by Watanabe, *et al.* [6]. In this approach, pruning of low scoring branches is done dynamically during testing. Aggregated models may be well suited to this type of approach since many of the Gaussian kernels from different training trials may be similar and exhibit large overlap with each other. Gaussian kernels such as these might well be approximated with a single Gaussian without severely degrading their likelihood estimates.

5. CONCLUSIONS

In this paper, we have presented the theoretical foundation for the model aggregation technique. We have also evaluated this technique on several standard speech recognition tasks using mixture Gaussian models. In each case, aggregation was found to produce significant improvements over standard training techniques, with observed error rate reductions of up to 12%. Aggregation performs particularly well when the standard training techniques produce models that are overfit to the training data. Additionally, these improvements are robust to changes in the test set. A tree-structured pruning method was also introduced. Experiments indicate that the pruned models retain some beneficial robustness properties of the aggregated models. This aggregate-then-prune technique is a potential replacement for the standard cross-validation strategy of choosing the single trial that performs best on a development set. Furthermore, dynamic pruning could lead to reduced computational requirements with minimal degradation in performance.

There are a large number of possible generalizations and extensions of this work. In a sense, model aggregation is equivalent to linear interpolation of models, where the interpolation is performed using equal weights. Thus, any situation where models are interpolated can be viewed as a form of aggregation. This encompasses common techniques such as the interpolation of speaker-independent and gender-specific models, or the interpolation of context-independent and context-dependent models. In recent experiments using SUMMIT, aggregation was fruitfully applied to models trained using different segmentations of the speech signal. Thus, a wide variety of techniques may be used in order to generate classifiers producing different posterior probabilities. One set of techniques can be obtained by perturbing the classifier structure or type. Another set of techniques emerges from perturbing the learning set through weighting, resampling, or generating new learning sets through alternative preprocessing of the same underlying input data. In addition to these extensions, more sophisticated schemes for combining classifiers have been developed [5]. Aggregation remains attractive because of its simplicity, and because the empirical evidence indicates its effectiveness in reducing the error rate in typical speech classification and recognition tasks.

6. REFERENCES

[1] C. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[2] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.

[3] J. Glass, J. Chang, and M. McCandless, "A probabilistic framework for feature-based speech recognition," in *ICSLP96* (Philadelphia), pp. 2277–2280, 1996.

[4] A. Halberstadt and J. Glass, "Heterogeneous measurements for phonetic classification," in *EUROSPEECH97* (Rhodes, Greece), pp. 401–404, September 1997.

[5] R. Jacobs, "Methods for combining experts' probability assessments," *Neural Computation*, vol. 7, pp. 867–888, 1995.

[6] T. Watanabe, K. Shinoda, K. Takagi, and E. Yamada, "Speech recognition using tree-structured probability density function," in *ICSLP94* (Yokohama), pp. 223–226, 1994.