EXPLOITING CORRELATIONS AMONG COMPETING MODELS WITH APPLICATION TO LARGE VOCABULARY SPEECH RECOGNITION

Ronald Rosenfeld, Xuedong Huang and Merrick Furst

School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

ABSTRACT

In a typical speech recognition system, computing the match between an incoming acoustic string and many competing models is computationally expensive. Once the highest ranking models are identified, all other match scores are discarded. We propose to make use of all computed scores by means of statistical inference. We view the match between an incoming acoustic string s and a model M_i as a random variable Y_i. The class-conditional distributions of $(Y_1, \ldots, Y_N)$ can be studied offline by sampling, and then used in a variety of ways. For example, the means of these distributions give rise to a natural measure of distance between models.

One of the most useful applications of these distributions is as a basis for a new Bayesian classifier. The latter can be used to significantly reduce search effort in large vocabularies, and to quickly obtain a short list of candidate words. An example HMM-based system shows promising results.

1 MOTIVATION AND OUTLINE

During the recognition phase of a typical speech recognition system, an incoming speech segment s is matched against a large number of competing models $M_1, M_2, \ldots, M_N$. The model or models that score the highest are then selected for further consideration.

There are many variations on this basic idea. The speech model may represent a phoneme, a syllable or a word. The competing models may represent the entire vocabulary, or only that part of it allowed by the grammar. The type of models used may vary, together with the matching process. Template-based models would typically be used with dynamic time warping [1] and some metric distance defined over frames. In HMM-based systems, a match is typically defined as the class-conditional log-probability ($\log P(s|M_i)$). The models themselves may be trained as Maximum Likelihood estimators, or else optimized for discrimination [2].

Common to all of these scenarios, however, are the following:

1. Computing the match for all the models is computationally expensive. For large vocabularies, it is prohibitive.
2. Once the best scoring models have been identified, all other match scores are discarded.

These observations suggest that a lot of computation is wasted in this process. Attempts have been made recently to overcome this problem by using fast preliminary search ([3, 4, 5, 6]). Here, we take a different approach to the problem. We propose to make use of *all* computed scores, by means of statistical inference. In order to use all the scores at run time, though, we must first analyze offline the statistical relationships between the models.

In section 2, we present a framework for such an analysis. Section 3 demonstrates the usefulness of this framework, by showing how it naturally gives rise to a measure of distance between models. In section 4 we develop the main application, reducing search time in large vocabularies.

2 FRAMEWORK

Let

$$Y_i(s) = \text{the match between acoustic string } s \text{ and model } M_i. \quad (1)$$

$Y_i(s)$ could be any reasonable measure of agreement between a model and an acoustic instance.

Consider the Y_i's as random variables. The distribution of $\mathbf{Y} \stackrel{\text{def}}{=} (Y_1, Y_2, \ldots Y_N)$ is determined by the population from which the s's are taken. Let

$$D_j(\mathbf{Y}) \stackrel{\text{def}}{=} P(\mathbf{Y}|s \in \text{speech-unit-}j). \quad (2)$$

$D_j(\mathbf{Y})$ is an N-dimensional distribution. It can be estimated by selecting examples of speech unit j and evaluating them by all the models. This can be done for all N distributions $D_1, D_2, \ldots D_N$.

We may also wish to consider the univariate distribution of each individual Y_i. We define:

$$D_j(Y_i) \stackrel{\text{def}}{=} P(Y_i|s \in \text{speech-unit-}j). \quad (3)$$

There are N^2 such distributions, which can be estimated in a similar way.

For generative models, if $M_1, M_2, \ldots, M_N$ are good models of speech units $1, 2, \ldots, N$, respectively, then D_j can be approximated by D_j^*, where

$$D_j^*(\mathbf{Y}) \stackrel{\text{def}}{=} P(\mathbf{Y}|M_j), \quad (4)$$

and similarly for $D_j^*(Y_i)$. $D_1^*, D_2^*, \ldots, D_N^*$ can be estimated in the same way as $D_1, D_2, \ldots, D_N$, except that the strings s are now generated by models $M_1, M_2, \ldots, M_N$, respectively.

These definitions, and the following analysis, apply to models of any type. For empirical support, we chose to apply these ideas to a small SCHMM-based system [7] of 48 context-independent English phoneme models, as used in the baseline SPHINX system [8]. We chose SCHMM over the discrete model because distance between acoustic strings is better modeled in Continuous HMM or SCHMM (since they are not subject to VQ errors). For this system we define:

$$Y_i(s) = \frac{1}{|s|} \log P(s|M_i) \quad (5)$$

0-7803-0532-9/92 $3.00

	/ae/	/ax/	/ay/	/w/	/ng/	/g/	/sh/	/dd/
/ae/	7.68	5.39	4.53	-1.23	1.28	-1.72	-2.66	0.23
/ax/	-5.25	0.68	-10.35	-6.49	-5.15	-6.94	-10.24	-5.33
/ay/	6.76	5.13	8.95	0.13	0.80	-1.08	-2.36	-0.38
/w/	-9.55	-4.08	-10.45	1.55	-7.11	-5.99	-12.09	-6.48
/ng/	-3.70	-0.23	-5.93	-3.62	4.25	-1.81	-6.13	-0.37
/g/	-13.96	-9.20	-15.34	-8.71	-9.98	-0.44	-10.73	-2.60
/sh/	-5.53	-3.50	-6.00	-4.91	-3.60	-0.87	7.80	0.21
/dd/	-15.39	-12.41	-16.59	-13.40	-11.28	-8.59	-13.29	-0.19

Table 1: A submatrix of $E^* \stackrel{\text{def}}{=} E^*_{ji}$ (the means of the $D^*_j(Y_i)$'s). The diagonal entries are the row maxima but not necessarily the column maxima. See the text.

3 DERIVING A MEASURE OF DISTANCE BETWEEN MODELS

To illustrate the usefulness of our formalism, we now use it to derive a measure of distance between models.

Consider the means of the $D_j(Y_i)$'s:

$$E_{ji} \stackrel{\text{def}}{=} E[D_j(Y_i)] = \int P(s|\text{speech-unit-}j)\, Y_i(s)\, ds \qquad (6)$$

All N^2 such means can be estimated together in a matrix $E \stackrel{\text{def}}{=} \{E_{ji}\}$. E^* is defined similarly. Table 1 shows a submatrix of E^* for our example system. The diagonal elements are the row maxima. This is to be expected, since they were derived by evaluating strings using the same models that were used to generate them. Note, however, that this argument does not carry over to the columns; some diagonal elements are *not* the column maxima (e.g. $D_{\texttt{/ax/}}(Y_{\texttt{/ax/}})$). This reflects the fact that some models tend to generate more "agreeable" strings than others.

A rough feel for similarity between some phonemes can be gleaned from this data. For example, columns `/ae/` and `/ay/` are similar (compare them to column `/g/`), as are rows `/ae/` and `/ay/`. This corresponds to the similarity between these two vowels.

For a more rigorous measure of distance between models, consider how the off-diagonal means differ from the diagonal element. Let

$$\text{DIST}(j;i) \stackrel{\text{def}}{=} E_{jj} - E_{ji} \qquad (7)$$

and similarly for DIST*.

This measure can be used to cluster larger speech units. It is superior to phonemic clustering, which considers phonemes as atomic units. For example, `bat` and `pat` are more similar acoustically than phonetically[1].

Recall that, for our HMM based system, we defined $Y_i(s) \stackrel{\text{def}}{=} \frac{1}{|s|}\log P(s|M_i)$. For simplicity, let us write $P_i(s)$ for $P(s|M_i)$. Then

$$\begin{aligned}\text{DIST}^*(i;j) &\stackrel{\text{def}}{=} E^*_{jj} - E^*_{ji} \qquad (8)\\ &= \int \frac{1}{|s|}P_j(s)\log P_j(s)ds - \int \frac{1}{|s|}P_j(s)\log P_i(s)ds\\ &= \int \frac{1}{|s|}P_j(s)\log\frac{P_j(s)}{P_i(s)}ds\end{aligned}$$

[1] We are grateful to Raj Reddy for this example.

The last expression is similar to the "Asymmetric Divergence" — a well known measure of distance between two distributions[9]. The difference is in the presence of the $\frac{1}{|s|}$ factor. Asymmetric Divergence was proposed as a measure of distance between HMMs by Juang and Rabiner [10]. They derived it from information theoretic arguments. D'orta et al.[11] used their measure, with a sampling technique similar to ours, to cluster phonemes. In our derivation, both the measure and the estimation method naturally "fall out" of the definition of the $D_j(Y_i)$'s. More importantly, our definition is not limited to HMMs.

4 REDUCING SEARCH IN LARGE VOCABULARIES

4.1 Changing the Classifier

As a first step towards reducing the search effort, we replace the original Bayesian classifier $M_1, M_2, \ldots, M_N$ with a new one, $D_1(\mathbf{Y}), D_2(\mathbf{Y}), \ldots, D_N(\mathbf{Y})$.

Since $D_j(\mathbf{Y}(s))$ is an N-dimensional distribution, an unrestricted non-parametric estimation is impractical for even a large sample. We proceed by assuming that the individual $D_j(Y_i)$s are independent. This is clearly incorrect, as our data (and intuition) indicate. In making this assumption we are merely choosing to concentrate on the first-order statistics of the $D_j(Y_i)$'s.

Thus we are looking for the j that maximizes $\prod_i D_j(Y_i)$. This still leaves us with the problem of estimating the N^2 distributions $D_j(Y_i)$. What do they look like? Figure 1 shows histograms of selected $D^*_j(Y_i)$'s, each based on a sample of 1000 strings, which were generated from the appropriate model.

The distributions are well characterized by a Normal curve. This is true for all the distributions we checked. It is not difficult to see why this happens. Since the strings were generated from Hidden Markov Models, each frame in each codebook was drawn independently. Therefore, $\log P(s|M_i)$ is a sum of many independent events, hence the Gaussian.

The Normal shape of the distributions is welcome news, because they can be characterized well with only two parameters each: mean and standard deviation. These can be estimated accurately and reliably from a modest sample. Note that, in other models, if the $D_j(Y_i)$'s are not Gaussians, accurate characterization may be more difficult. However, the mean and standard deviation can still be

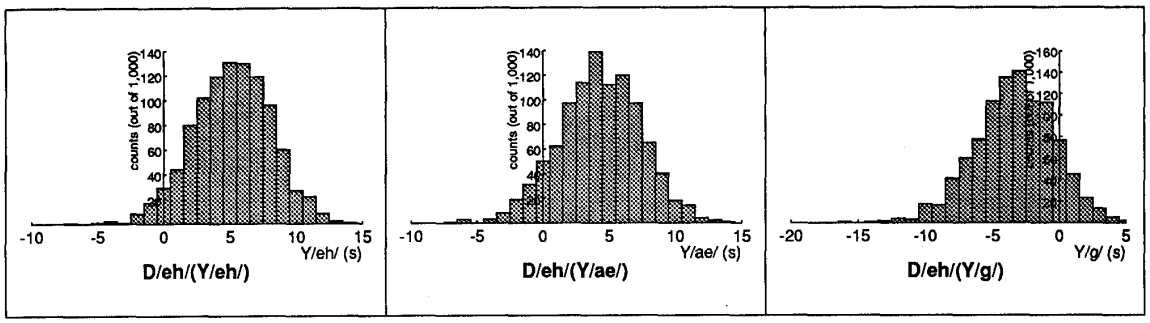

Figure 1: Histograms of some typical $D_j^*(Y_i)$'s, each based on a sample of 1000 strings generated from the HMM model M_j.

used to derive statistical bounds. The resulted inference is expected to be weaker, though.

Assuming $D_j(Y_i) \sim \mathcal{N}(E_{ji}, \sigma_{ji})$, classification can now be done by finding the j that minimizes:

$$-\log P_1(\mathbf{Y}|D_j) = \sum_{i=1}^{N} \left[\log \sigma_{ji} + \frac{(Y_i - E_{ji})^2}{2\sigma_{ji}^2} \right] \qquad (9)$$

Where the subscript "1" denotes the use of first-order statistics only.

4.2 Performance of the New Classifier

How good is the new classifier? If the M_i's are ML classifiers, some performance degradation is expected. Table 2 lists one possible measure of performance. In the first row, in 24% of the strings tested, the generating models was correctly given the highest P_1 value by the new classifier. In 37% of the cases, it was ranked among the top 2 contenders, and so on (the percentages are cumulative).

How can these results be improved? The performance of any Bayesian classifier depends crucially on how well separated the class distributions are. In our context, this translates into the ratio of between-string variability to between-model variability. A close look at our data reveals very significant between-string variability. Some strings receive good scores from all the models, while others receive bad scores. There is very significant correlation between the various scores given to the same string. In fact, we found the pairwise correlation coefficients to lie in the range 0.93–0.99, regardless of the distribution from which the strings came, or the pair of models used for evaluation. This global correlation among the Y_i's of the same string means that some strings are "better acoustic segments" than others. This may be because some speech frames are further away from the codeword centers, resulting in weak matches with all models alike.

To get rid of most of this global correlation, we *normalize* the scores by subtracting, from each $Y_i(s)$, the average $\overline{Y}(s)$ of $Y_1(s), Y_2(s), \ldots, Y_N(s)$. The normalized results are listed in the second row of table 2. The improvement is indeed very significant.

4.3 Estimating the Scores

To avoid computing all N scores $Y_1(s), Y_2(s), \ldots, Y_N(s)$, we *estimate* $\log P_1(\mathbf{Y}(s)|M_j)$ and $\overline{Y}(s)$ using only a subset of the Y_i values. Performance degradation will depend on the sample size. The last part of table 2 shows the performance of the estimated normalized classifier, for different sample sizes. All samples were drawn randomly and independently for every test string.

4.4 Using the New Classifier

To quickly obtain short lists of candidate words, we use the following algorithm: compute $Y_i(s)$ for a small random subset of the models, and output the models ranked highest by this estimated classifier.

To reduce the effort in searching for the top ML model, we use the following probabilistic algorithm: if we desire, say, a 96% confidence in the classification, we compute $Y_i(s)$ for 10 randomly selected models, restrict our attention to the models that were ranked 1–15 by the estimated classifier, and choose the one with the highest Y_i among them. We only need to compute a total of 22 match scores on average, and no more than 25.

The two algorithms above save us some work over computing all $N = 48$ score values. For our small test system the savings are not dramatic, but they should increase considerably when the classifier is applied to real-world, large vocabulary systems:

1. For a given level of accuracy and confidence, the necessary sample size does not depend on the size of the population — it is only a function of the variance of the data, which is fixed. A sample of size 20 is large relative to our test vocabulary of 48 models, but represents significant savings for a vocabulary of, say, 1,000 items.
2. longer models (i.e. words as opposed to phonemes) are less confusable. Their between-model variance is therefore greater, resulting in better classification rate.

On the other hand, it is yet to be seen whether our approach will work on real speech and large vocabularies. Some possible problems are:

1. Real speech is different from the synthetic frames we generated for the experiment above. The distributions D_i are different from the D_i^*'s, and may be more difficult to characterize or to separate.
2. In a large vocabulary, a given entry is on the average confusable with more other entries than in our small test system.

	Top 1	Top 2	Top 3	Top 4	Top 7	Top 10	Top 15	Top 20
P_1 Ranking	24%	37%	46%	54%	70%	79%	88%	94%
Normalized P_1 Ranking	66%	80%	87%	91%	96%	98%	99.5%	99.8%
Estimated P_1 Ranking								
sample size = 16	54%	71%	79%	84%	92%	96%	98.6%	99.5%
sample size = 10	43%	59%	69%	75%	86%	91%	96%	98.0%
sample size = 8	37%	53%	62%	69%	81%	88%	94%	97%

Table 2: Performance of the D_i^* classifier, based on first-order statistics only. $N = 48$.

We plan to test these assertions when we implement this approach on a large vocabulary system, such as RM or WSJ.

4.5 Potential Improvements

The results discussed above are preliminary. The following can be used to try to achieve further improvement:

Judicious choice of the sample: In the experiments described above, we chose a new sample randomly for every string. Undoubtedly this is not optimal. We can use statistical analysis (e.g. multiple regression) to choose the subset that best predicts P_1 and $\overline{Y}$. This has the added advantage of allowing us to keep in memory only those columns of the $[E, \sigma]$ table that correspond to that subset. For large vocabularies, this represents a significant saving in memory requirements.

Using higher-order statistics: So far we discussed and exploited only the first-order behavior of the distributions $D_j(\mathbf{Y})$. Higher-order statistics can also be employed. Much more information can be gleaned from even the second-order behavior alone. If two models are similar, than a string scoring well (badly) on one is likely to score well (badly) on the other. For two very different models, a good score on one implies a bad score on the other. These deductions are based on a generalized form of the Triangle Inequality, although they do not require that the distance between the models be a metric. An elimination algorithm similar to that reported in [12] can then be used to implement Fast Search.

Better normalization: The normalization we used in order to reduce the global correlation is an ad-hoc subtraction of the string's average score. Better methods may be possible, leading to lower within-string variance, and hence to better performance[2].

Better modeling of the $D_j(\mathbf{Y})$'s: This may be particularly useful when the distributions are estimated from real speech samples $(D_1, D_2, \ldots, D_N)$ and not from strings generated by the models $(D_1^*, D_2^*, \ldots, D_N^*)$. We expect the former to match the Gaussian curve less well than the latter do.

Other statistics of s: We can view $\mathbf{Y}(s)$ as a set of statistics of the acoustic string s, which reduce its dimensionality to a reasonable level. There is no reason why $\mathbf{Y}$ should not include other statistics of s as well. One plausible candidate is the string's length, namely the number of speech frames it has. Other statistics can be suggested.

[2]We did try, unsuccessfully, to use the *length* of the strings (number of frames) to normalize the scores. We found very little correlation between the two.

ACKNOWLEDGMENTS

We are grateful to Raj Reddy, Kai-Fu Lee, Fil Alleva and Dan Julin for helpful comments and encouragement. This research was sponsored by the Defense Advanced Research Projects Agency (DOD), ARPA Order No. 7239, under contract number N00039-91-C-0158.

References

[1] Sakoe, H., and Chiba, S., "Dynamic programming algorithm optimization for spoken word recognition." *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-26(1):43–49, February 1978.

[2] Bahl, L. R., Brown, P. F., de Souza, P. V., and Mercer, R. L., "A New Algorithm for the Estimation of Hidden Markov Parameters". *ICASSP 88*, pp. 493–496, April 1988.

[3] Bahl, L., Bakis, R., de Souze, P., and Mercer, R. "Obtaining Candidate Words by Polling in a Large Vocabulary Speech Recognition System". *Proc. ICASSP 88*, pp. 489–492, New-York, NY, April 1988.

[4] Bahl, L., Gopalakrishnan, P.S., Kanevsky, D., and Nahamoo, D. "Matrix Fast Match: A Fast Method for Identifying a Short List of Candidate Words for Decoding". *Proc. ICASSP 89*, pp. 345–347, Glasgow, Scotland, May 1989.

[5] Bahl, L., de Souze, P., Gopalakrishnan, P.S., Kanevsky, D., and Nahamoo, D. "Constructing Groups of Acoustically Confusable Words". Proc. ICASSP 89, pp. 345–347, Glasgow, Scotland, May 1989.

[6] Aubert, X. "Fast Look-Ahead Pruning Strategies in Continuous Speech Recognition". *Proc. ICASSP 89*, pp. 659–662, Glasgow, Scotland, May 1989.

[7] Huang, X., Lee, K., and Hon, H. "On Semi-Continuous Hidden Markov Models". *ICASSP 90*, pp. 689–692, 1990.

[8] Lee, K.F. *Large-Vocabulary Speaker-Independent Continuous Speech Recognition: The SPHINX System.* PhD thesis, Computer Science Department, Carnegie Mellon University, April 1988.

[9] Kullback, S. *Information Theory and Statistics.* New York, Wiley, 1959.

[10] Jaung, B. H., and Rabiner, L. R., (1985). "A Probabilistic Distance Measure for Hidden Markov Models". *AT&T Technical Journal* 64(2).

[11] D'orta, P., Ferretti, M., and Scarci, S., "Phoneme Classification for Real Time Speech Recognition of Italian". *Proc. ICASSP 87*, pp. 81–84, Dallas, TX, 1987.

[12] Vidal, E., Rulot, H., Casacuberta, F., and Benedí, J. M. "On the Use of Metric-Space Search Algorithm (AESA) for Fast DTW-Based Recognition of Isolated Words". *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-36, pp. 651–656, May 1988.