TRAINING HIDDEN MARKOV MODELS USING POPULATION-BASED LEARNING

Bruce Maxwell

Dept. of Engineering
Swarthmore College
Swarthmore, PA 19081
maxwell@swarthmore.edu

Sven Anderson

Dept. of Computer Science
University of North Dakota
Grand Forks, ND 58202
anderson@cs.und.edu

ABSTRACT

Hidden Markov Models are commonly trained using algorithms derived from gradient-based methods such as the Baum-Welch procedure. We describe a representation of discrete observation HMMs that permits them to be trained using Population-Based Incremental Learning (PBIL), a variant of genetic learning. In this paper we examine the training speed and recognition performance of PBIL on a spoken digit recognition task. We demonstrate maximum likelihood training of HMMs can be achieved using PBIL, although the Baum-Welch procedure is significantly faster. However, unlike the Baum-Welch procedure, PBIL is easily extended to perform discriminative training. Furthermore, PBIL training extends to applications like speech intelligibility assessment that lack a differentiable or analytical optimization function.

1. INTRODUCTION

Hidden Markov models (HMMs) have been used successfully to model speech patterns for real time recognition of standard speech. The standard training method for HMM-based speech recognition systems is the Baum-Welch gradient descent algorithm. This algorithm is generally used to maximize the likelihood that each HMM generated its particular training set. Maximum likelihood estimation (MLE) training is possible because it is straightforward to calculate the derivative of the optimization function.

The HMM community has long recognized that the MLE criterion is not optimal for creating a set of maximally discriminating models. Other criteria, such as maximum mutual information (MMI) criterion, which maximizes both the likelihood of an HMM generating its training set and the discrimination between models, have been used to reduce error rates. Unfortunately, the MMI criterion, and others that might be used, do not have simple derivatives to calculate or are not numerically stable. In addition, as the community moves to harder modeling problems, it may wish to develop optimization functions that bear directly on a new problem, but which do not necessarily have a first derivative. Thus, there is a need for an alternative training method for HMMs.

Speech training applications are one example of a situation in which an alternative optimization function that directly measures speech intelligibility would be desirable. To rate speech intelligibility we currently rely on subjective expert listener judgments [1]. Scores from standard recognition algorithms do not correlate well with listener judgments [2]. Recently, progress has been made in adapting current algorithms to this task, however machine correlations of phoneme intelligibility reach only 80% of human listener correlations [3]. Ideally, we would like to translate expert listener judgments to an intelligibility function that we could train a set of HMMs to emulate. Clearly, this function will not have an easily calculable derivativein terms of the HMM parameters, so we must have an alternative training method.

This paper describes a new method of training HMMs using a variant of genetic algorithms called Population-Based Incremental Learning, Since the method relies only on a fitness function, it is possible to design arbitrary optimization criteria. In order to show that PBIL is a viable alternative to the Baum-Welch algorithm, we directly compare the performance of the two training algorithms on a digit recognition task using discrete HMMs.

2. POPULATION-BASED INCREMENTAL LEARNING

Population-based incremental learning (PBIL), developed by Baluja, is a variant of genetic algorithms (GA) that explicitly extracts the population statistics rather than maintaining a large number of samples [4]. Because PBIL does not have to maintain a large population, it requires less memory and runs faster than an equivalent genetic algorithm. On multiple standard GA problems tested by Baluja, PBIL matched the performance of or outperformed a comparable standard genetic algorithm [5].

With a GA, the variables to be optimized are represented as a bit string. A GA maintains numerous bit strings to represent different parts of the search space. The fundamental idea of PBIL is that the population within a genetic algorithm can be represented as a bit probability vector. For example, if 25% of the population has a 1 in the first bit position, then we can represent that bit with a probability of 25%. By examining all of the other bits across the population we generate a probability vector that explicitly maintains the statistics of the population. We can then recreate a statistically similar population using a random number generator and the bit probability vector.

The basic PBIL algorithm has following steps.

1. Initialize the bit probability vector elements to 0.5.

2. Generate N sample bit strings from the bit probability vector.
3. Keep track of the best sample found according to some evaluation function.
4. Update the probability vector using the rule $P_i = P_i(1.0 - LR) + LR \cdot Max_i$, where LR is a user-specified learning rate and Max_i is the value of bit i of the bit string with the maximum fitness.
5. Mutate the probability vector. For each bit probability indexed by j, adjust the bit probability using $P_j = P_j(1.0 - MS) + MS \cdot Max_i$ with probability MP, where MS is a mutation shift parameter.
6. Repeat from step 2 until the average fitness score converges.

A potential criticism of PBIL is that it does not maintain the second-order statistics (cross-correlations) of the population that a standard GA would implicitly maintain in its population. Thus, PBIL may not work as effectively in spaces where the second order statistics are an important clue in optimizing the bit-string. Given that the Baum-Welch training procedure for HMMs is based on first-order statistics, however, we hypothesize that PBIL will adequately train the HMM parameters. PBIL can potentially improve upon the standard Baum-Welch procedure by allowing the use of optimization criteria that are not analytical expressions or which do not have an easily calculable first derivative.

2.1. A Bit String Representation for HMMs

In order to use the PBIL algorithm to train HMMs, we must first represent an HMM as a bit string. The HMM parameters include the initial state probabilities π, the transition probabilities α, and the observation probabilities β. The constraints on these probabilities are that the π values must sum to one, and the α and β values for each state must each sum to one.

The obvious representation is to create a single probability vector out of all of the HMM parameters and then convert each probability to a specific number of bits, such as 16. This gives a precision of approximately $\frac{1}{65,000}$ for values between 0 and 1. To convert from the bit string to an HMM, we must then normalize the probabilities for each appropriate group. For example, we would divide the π probabilities by their sum to obtain a set of probabilities that sum to one.

This representation, however, is not a one-to-one mapping, and is ill-conditioned. There are, for example, many ways of representing a set of probabilities that are all $\frac{1}{N}$, where N is the number of probabilities in the group. So long as all of the numbers in the group are the same, it does not matter what their magnitudes are. This also demonstrates the ill-conditioned nature of the representation. If, for example, all of the numbers have a zero in all bit locations but the lowest bit, then the probabilities will all be $\frac{1}{N}$. If one number's highest bit is then switched to a 1, then that probability would become close to one, while all of the others would be approximately zero. These two shortcomings encourage us to look for a different mapping from a bit string to an HMM.

Table 1: Digit Data

Digit	OGIbet	No. Train Tokens	No. Test Tokens	Total
zero	zi:9roU	12	9	21
one	wen	168	84	252
two	tcthu	141	87	228
three	T9ri:	114	55	169
four	foU9r	111	47	158
five	faIv	96	53	149
six	sIkckhs	103	42	145
seven	sEven	21	5	26
eight1	eitcth	84	47	131
eight2	eitc	32	13	45
nine	naIn	102	50	152
Total		984	492	1,476

The representation we use is to explicitly include in the bit string only $N-1$ of the N probabilities for each related group. The $N-1$ 16-bit blocks are interpreted in the following manner. The first 16-bit block is interpreted as a probability in $[0,1]$. This number is subtracted from 1 and the remainder is stored as $SpaceLeft$. For each of the remaining $N-2$ 16-bit blocks the magnitude is interpreted as a value in $[0, SpaceLeft]$, and the calculated probability is subtracted from $SpaceLeft$. The N^{th} probability is given the value in $SpaceLeft$ after the other $N-1$ probabilities have been calculated.

This representation is a one-to-one mapping—so none of the bit string search space is duplicated—and better conditioned than the obvious representation described above. It also requires fewer bits to represent the HMM, which should result in faster convergence of the bit probability vector.

3. METHODS

We selected the simple, but realistic task of digit recognition for testing the capabilities of PBIL learning. The speech used for these tests was collected by the the Center for Speech and Language Understanding (CSLU) at the Oregon Graduate Institute and derives from 359 male and female native speakers of American English who were instructed to read a small set of addresses and telephone numbers. Speakers were recorded in quiet conditions and digitized at 8 kHz. Recorded utterances were segmented by CSLU researchers. We used the phonetic segmentations to automatically extract a subset of all digits from the continuously spoken utterances. A summary of the subset of digits selected for recognition experiments is shown in Table 1. Note that that the digit "eight" was pronounced with and without final aspiration; both forms were common and were therefore modeled separately. The resulting single-digit utterances are perceptually clear and distinct, but lack the complete context of natural, continuous speech. Therefore, they are probably much more confusable than digits spoken in isolation.

Table 2: Comparison of BW and PBIL trained HMMs on the MLE criterion.

Algorithm	Training % Correct	Testing % Correct
BW	77.5	70.5
PBIL	64.7	58.5
PBIL-MMI	58.9	55.8

3.1. Spoken Digits Data

For each utterance, 13 mel cepstral coefficients were calculated over 16 msec windows each 10 msec. A twenty symbol vector-quantization codebook was derived from the mel-scale cepstral coefficients of the set of training utterances. This codebook was used to create the input symbol sequence for each training and testing utterance. The average utterance duration was 260 msec (36 frames, $\sigma = \pm 12$). Two-thirds of the data was randomly selected for training models; the remaining one-third was used to test model generalization.

3.2. HMM Model Parameters

Each word was modeled by an HMM having 5 states and 20 observation symbols (130 parameters). For purposes of comparison, the number of states and number of observation types were held constant for all experiments. HMM initial state probabilities, transition probabilities, and observation probabilities were initially randomized subject to normalization to unity. HMMs were assumed ergodic (fully connected) rather than linear in order to demonstrate the generality of PBIL training. HMM states are implicitly associated with phonemes, but each word was modeled with an HMM having 5 states, corresponding to the number of phonemes in the longest digit, six (/sIkckhs/).

4. RESULTS

4.1. Maximum Likelihood Estimation (MLE) Criterion

The standard maximum likelihood criterion for training HMMs uses the training set of observations to derive model parameters. Model parameters (α, β, π) are chosen to maximize $P(O|\lambda)$, the likelihood of the observed sequence conditioned on the model. For Baum-Welch training, we used the standard re-estimation formulae modified for multiple observation sequences [6]. When training HMMs with PBIL, we used the Viterbi algorithm to estimate the fitness of models for each training utterance.

For each optimization, twenty-five randomly initialized optimizations were performed. We selected the optimal HMM for each digit based on the log-probability score over the entire set of training data. All of the optimal HMMs were then combined and used to test recognition accuracy. Average percent correct is shown in Table 2. The average scores indicate that PBIL solutions are substantially worse than those discovered by the BW procedure. The relative difference of performance on training versus testing data

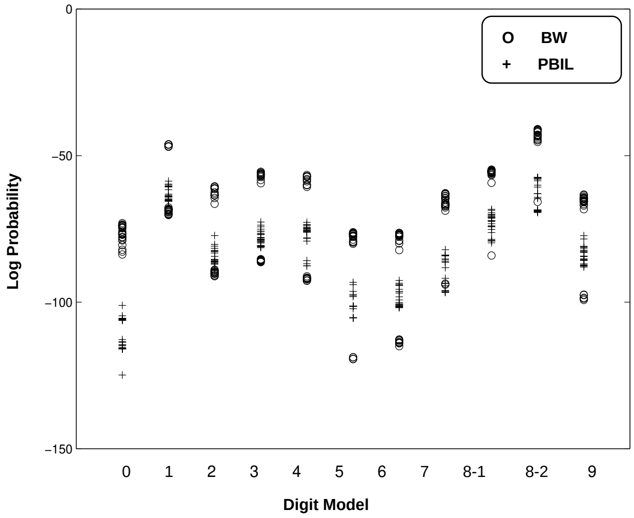

Figure 1: Final $log(P(O|\lambda))$ averaged over all training sequences for 25 BW and PBIL MLE optimization experiments.

indicates that the PBIL solution generalizes as well as the BW solution.

Confusion matrices for testing data are shown in Table 3. The models derived from each learning procedure exhibit a similar overall pattern of individual errors. The most confusable pairs are one vs. nine, two vs. three, four vs. five, and the two pronunciations of eight. Figure 1 presents the final log probability for each BW and PBIL experiment. The pattern of performance for each digit appears to be the same for the two training algorithms. However, the BW-trained HMMs tend to be bimodal, whereas those models derived from PBIL show less variation.

The computation time time required for an average BW optimization was about 13 seconds on a Pentium Pro 200 MHz processor. The PBIL optimizations were run for 1000 generations to ensure convergence, requiring an average of 86 seconds. However, nearly all models converged to a solution within 200 generations (about 20 seconds). Thus, Baum-Welch MLE optimization was approximately twice as fast as PBIL MLE optimization.

4.2. Maximum Mutual Information (MMI) Criterion

The maximum mutual information optimization criterion is a discriminative criterion and is given by

$$I_\Lambda(O^\nu, \nu) = logP(O^\nu|\lambda_\nu) - log \sum_{w=1}^{V} P(O^\nu|\lambda_w, w)P(w)$$

MMI is optimized over the set of all models represented by Λ. The primary advantage of the MMI criterion is that it attempts to maximize the likelihood of the observation sequence conditioned on the correct model while minimizing the likelihood of the wrong model for all the observation sequences. MMI involves all parameters for all models. We assume that the $P(O^\nu|\lambda_w, w) \approx P(O^\nu|\lambda_w)$, but scale the contribution of the second term by a small constant (0.1

Table 3: MLE Testing Data Confusion Matrix. BW results are shown on the left, PBIL results on the right.

#	0	1	2	3	4	5	6	7	8-1	8-2	9	#	0	1	2	3	4	5	6	7	8-1	8-2	9
0	4	1	0	2	0	1	0	1	0	0	0	0	2	0	2	4	0	0	0	1	0	0	0
1	3	60	2	0	4	3	0	0	0	0	12	1	6	58	0	1	2	5	0	0	0	0	12
2	1	0	65	8	2	2	6	0	3	0	0	2	7	1	50	15	1	0	6	0	5	0	2
3	3	2	9	34	1	2	0	0	3	0	1	3	1	2	21	27	0	1	0	0	0	2	1
4	1	2	2	0	40	2	0	0	0	0	0	4	8	2	2	1	30	3	0	1	0	0	0
5	2	2	2	0	7	34	0	3	0	0	3	5	2	2	0	4	9	24	0	8	0	1	3
6	1	0	1	0	0	1	36	2	1	0	0	6	1	0	0	3	0	3	29	1	3	1	1
7	0	0	0	0	1	0	0	4	0	0	0	7	2	0	0	0	0	0	1	2	0	0	0
8-1	1	0	6	1	0	0	0	0	30	8	1	8-1	0	0	9	3	0	0	7	0	22	5	1
8-2	0	0	2	0	0	0	0	0	7	4	0	8-2	0	0	2	1	0	0	0	0	5	5	0
9	1	9	1	1	0	1	1	0	0	0	36	9	1	6	1	4	0	0	0	0	0	5	38

here). We have found that the second, discriminative term is sensitive to inaccurate initial models and can prevent learning in the early stages of PBIL training when individual models are acquired. After basic models are acquired, the discriminative term appears to become a more significant factor. The average performance of PBIL-MMI optimization is shown in Table 2. Comparison with the MLE recognition scores shows that the MMI recognition scores are much lower than the those obtained via PBIL-MLE or BW-MLE.

5. DISCUSSION

The experiments reported in this paper were motivated by pilot studies in which PBIL-optimized HMMs were used to recognize several hundred circles, squares, and triangles drawn by hand with a computer mouse. The present findings are in disagreement those results. In those experiments BW and PBIL derived equivalently good HMMs when MLE optimization was used. Moreover, we found that the discriminative PBIL-MMI procedure reduced the total error rate from 5.1% to 0.9% relative to the BW-MLE procedure, a five-fold improvement in accuracy. We therefore believe that the present results reflect the larger set of parameters to be optimized, and expect improved performance will eventually be achieved with PBIL-MMI. The larger set of parameters in the speech recognition experiments led to longer bit strings. One difficulty with long bit strings is an exponential growth in the size of the search space. Limits to bit string length might be obtained by using a bit string code that expands every generation. This would have the advantage of providing low resolution at the earliest stages of training but high resolution much later when it is desirable.

In the pilot experiments we also explored several PBIL modifications that were not explored here. In particular, Step 3 of the algorithm can use several best strings and/or move the probability vector away from the least promising individuals. Baluja [5] has found these modifications to yield improved performance on many tasks, but this was not confirmed in our pilots. Thus, these modifications were not included in the present study.

6. SUMMARY

We compared population-based learning with the standard Baum-Welch procedure for training HMMs on a speech recognition task and found that it provided substantially worse performance. We intend to explore other representations of HMM and the application of PBIL to the training of models for which the optimization function is empirically derived from listener assessments of speech intelligibility.

7. ACKNOWLEDGMENTS

This work was partially supported by NSF EPSCoR grant OSR-9452892. The Center for Spoken Language Understanding of the Oregon Graduate Institute of Technology collected and analyzed the speech data used in this report. The Baum-Welch implementation of HMMs was adapted from a program written by Tapas Kanungo, University of Maryland. We thank Mizuho Schwalm and Joel Emter for their assistance with data preparation and programming, respectively.

8. REFERENCES

[1] C. S. Watson, D. Kewley-Port, D. J. Reed, and D. Maki, "The Indiana Speech Training Aid (ISTRA) I: Comparisons between human and computer-based evaluation of speech quality," J. Speech and Hearing Res., vol. 32, pp. 245–251, 1989.

[2] S. E. Anderson and D. Kewley-Port, "Evaluation of speech recognizers for speech training applications," IEEE Transactions on Speech and Audio Processing, vol. 3, pp. 229–241, 1995.

[3] Y. Kim, H. Franco, and L. Neumeyer, "Automatic pronunciation scoring of specific phone segments for language instruction," in Proc. Eurospeech, vol. 2, (Rhodes, Greece), pp. 645–648, 1997.

[4] S. Baluja and R. Caruana, "Removing the the genetics from the standard genetic algorithm," in The International Conference on Machine Learning 1995 (ML-95) (A. Prieditis and S. Russel, eds.), (San Mateo, CA), pp. 38–46, Morgan Kaufman Publishers, 1995.

[5] S. Baluja, "An empirical comparison of seven iterative and evolutionary heuristics for static function optimization," in Proceedings of the Eleventh International Conference on Systems Engineering, (University of Nevada at Las Vegas), pp. 692–697, 1996.

[6] L. Rabiner and B. Juang, Fundamentals of Speech Recognition. Englewood Cliffs, NJ: Prentice-Hall, 1993.