GA Approaches to HMM Optimization for Automatic Speech Recognition

Yara Pérez Maldonado, Santiago Omar Caballero Morales,
and Roberto Omar Cruz Ortega

Technological University of the Mixteca, UTM, Highway to Acatlima, Km. 2.5,
Huajuapan de Leon, Oaxaca, 69000
shara.luw@gmail.com, scaballero@mixteco.utm.mx, omarneon@hotmail.com

Abstract. Hidden Markov Models (HMMs) have been widely used for Automatic Speech Recognition (ASR). Iterative algorithms such as Forward - Backward or Baum-Welch are commonly used to locally optimize HMM parameters (i.e., observation and transition probabilities). However, finding more suitable transition probabilities for the HMMs, which may be phoneme-dependent, may be achievable with other techniques. In this paper we study the application of two Genetic Algorithms (GA) to accomplish this task, obtaining statistically significant improvements on un-adapted and adapted Speaker Independent HMMs when tested with different users.

Keywords: genetic algorithms, hidden markov model, automatic speech recognition.

1 Introduction

Hidden Markov Models (HMMs) are statistical model techniques which have been applied to pattern recognition problems such as automatic speech recognition [5,6,10]. An HMM has the following parameters:

1. $V = \{v_1,\ v_2,\ ...,\ v_M\}$, an output observation alphabet. This is used for **discrete** HMMs, where the observations are sequences of symbols (e.g., vector quantization symbols) rather than continuously-valued feature vectors. M is the size of the alphabet.
2. $Q = \{q_0,\ q_1,\ ...,\ q_N\}$, a set of states, where q_0 and q_N are non-emitting states (not associated with observations). Each state has associated a probability function which models the emission of certain observations ($B = \{b_i(\mathbf{o}_t)\}$).
3. $A = \{a_{01},\ a_{02},\ ...,\ a_{NN}\}$, a transition probability matrix A, where each a_{ij} represents the probability of moving from state i to state j. $\sum_{j=1}^{N} a_{ij} = 1\ \forall i$.
4. $B = \{b_i(\mathbf{o}_t)\}$, a set of observation likelihoods, also called the **emission probabilities**. Each term represents the probability of an observation vector $\mathbf{o}_t$ being generated from a state i.
5. $\pi = \{\pi_i\}$, an initial state distribution, where $\pi_i = Pr(q_0 = i),\ 1 \leq i \leq N$, and $\sum_{i=1}^{N} \pi_i = 1$.

J.A. Carrasco-Ochoa et al. (Eds.): MCPR 2012, LNCS 7329, pp. 313–322, 2012.

The notation $\lambda = (A, B, \pi)$ is used to indicate the parameter set of an HMM [5]. An example of a typical HMM structure for large vocabulary recognition is shown in Figure 1. This left-to-right structure is commonly used to model subword units (phonemes), which can be concatenated to form words [10].

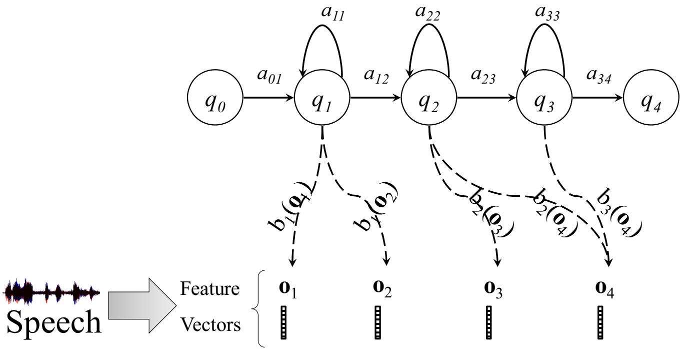


Fig. 1. Structure of a three-state left-to-right HMM

In a regular Markov model, the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters. In a Hidden Markov model, the state is not directly visible, but variables influenced by the state are visible (observations, i.e., feature vectors) [5]. Each state has a probability distribution over the possible output tokens. Therefore, the sequence of tokens generated by a HMM gives some information about the sequence of states.

There are three main problems associated with HMMs:

- The evaluation problem. Given the parameters of a model (λ), estimate the probability of a particular observation sequence ($Pr(\boldsymbol{O}|\lambda)$).
- The learning problem. Given a sequence of observations $\boldsymbol{o}_t$ from a training set, estimate/adjust the transition (A) and the emission (B) probabilities of an HMM to describe the data more accurately.
- The decoding problem. Given the parameters of the model, find the most likely sequence of hidden states $Q^* = \{q_1, q_2, ..., q_n\}$ that could have generated a given output sequence $O = \{\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_t\}$.

Standard algorithms such as Viterbi (for decoding) and Baum-Welch (learning) are used for these problems [5]. The use of Genetic Algorithms (GAs) for the optimization of the elements of an HMM has been explored [1,4,8]. A GA is a search heuristic that mimics the process of natural evolution and generates useful solutions to optimization problems [3]. In [1], GA optimization was performed for the observation probabilities and transition states for word HMMs, while in [8] finding an optimal HMM structure was the objective. Xiao *et al.* [9] used GA to optimize the number of states in the HMM models and its model parameters for web information extraction, obtaining important increases in precision rates when compared with Baum-Welch training.

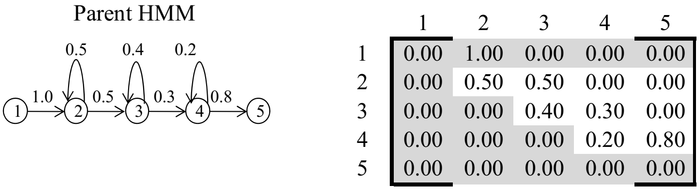
In this paper we focus on the application of GAs to optimize the transition probabilities (and thus, the internal structure of the HMMs) to improve speech recognition accuracy for a Speaker Independent (SI) system when used by test users. While in [1,8,4,9] a single GA was proposed, we explored on the application of two different GA approaches, giving a more depth insight about their effectiveness. Also, we tested with significant larger vocabulary (approximately 3000 words) and equally sized training and testing speech data.

Hence, this paper is structured as follows: in Section 2 we present the details of both GAs approaches, while in Section 3 we present details of the speech corpus used for the experiments. In Section 4 we present the convergence plots of the GAs with the speech data and the performance results of the optimized HMMs. Finally in Section 5 we present our conclusions and future work.

2 Genetic Algorithms

2.1 Approach 1

This approach consisted in the application of the GA on each phoneme's transition matrix, where the base structure to optimize is the one presented in Figure 1. As shown in Figure 2, each element in a phoneme's transition matrix was considered an individual of the Initial Population. This consisted of nine elements due to the following restrictions: (1) because of the continuous nature of speech, there must be transitions from states 1 to 2, 2 to 3, 3 to 4, and 4 to 5 independently of extra transitions in the original HMM; (2) all transitions must be from left-to-right (hence, the elements under the diagonal are not considered); (3) no direct transition from state 1 to 5 must exist, as this would lead to a 'Tee' model in the developing tool (HTK Toolkit [10]), thus, the original transitions from state 1 remained unchanged and were not considered.



	1	2	3	4	5
1	0.00	1.00	0.00	0.00	0.00
2	0.00	0.50	0.50	0.00	0.00
3	0.00	0.00	0.40	0.30	0.00
4	0.00	0.00	0.00	0.20	0.80
5	0.00	0.00	0.00	0.00	0.00

Fig. 2. Approach1: Initial Population

In contrast with common GA problems, the selection of parents for reproduction was not performed based on "fitness" measured by a cost function, it was directly based on the magnitude of the transition element. Hence, all the transition probabilities of the nine elements were normalized to 1.00, and cumulative probabilities were computed in order to perform *Roulette Wheel* selection. The arranged elements of the Initial Population and their normalized probabilities are shown in Table 1. Once that Roulette Wheel selection is performed, the elements are re-arranged according to the pair of individuals (parents) selected for reproduction. This is also shown in Table 1, where individual 9 would mate with 7, 1 with 8, 3 with 6, and 2 with 4. These couples will produce Offsprings by means of crossover, while the last element, 5, will produce an Offspring my means of mutation as shown in Figure 3. As we are dealing with probabilities (Dec), we converted these values into integer numbers (Int) so binary (Bin) chromosome representation were more representative of the problem's phenotype. For this, each probability was multiplied by 1000000000, and 23 bits were used to code the resulting integer number. This was due to HTK's definitions for transition probabilities considering up to nine digits after the point.

Table 1. Approach1: Roulette Wheel Selection of the Normalized Initial Population and Chromosome Representation

Initial Population				Roulette Wheel Selection			
Index	Dec	P(x)	Q(x)	Index	Dec	Int	Bin
1	0.50000	0.18519	0.18519	9	0.80000	800000000	11110100001001000000000
2	0.50000	0.18519	0.37037	7	0.00000	0	00000000000000000000000
3	0.00000	0.00000	0.37037	1	0.50000	500000000	.
4	0.00000	0.00000	0.37037	8	0.20000	200000000	.
5	0.40000	0.14815	0.51852	3	0.00000	0	.
6	0.30000	0.11111	0.62963	6	0.30000	300000000	.
7	0.00000	0.00000	0.62963	2	0.50000	500000000	.
8	0.20000	0.07407	0.70370	4	0.00000	0	.
9	0.80000	0.29630	1.00000	5	0.40000	400000000	01111010000100100000000
	2.70000	1.00000					

2-point crossover was performed with randomly selected crossover points (within the range 1-23, the length of the chromosome). For mutation, two points were also selected randomly, and the bits between these points were changed randomly from 1 to 0 and viceversa (thus, any of these bits could remain unchanged or get the opposite value) (see Figure 3). After these operators are applied the resulting Offsprings are decoded into the associated integer numbers and are divided by 1000000000 to get the phenotypes, which then are stored in the Offspring Population register. In order to load these probabilities into the HMM's transition matrix, these must be normalized to 1.00 (the sum of all the elements of each row in the transition matrix must be 1.00, with the exception of row 5 which must be 0.00). Also, if after the reproduction operators are applied any of the important transitions is deleted (i.e., 1 to 2, 2 to 3, 3 to 4, or 4 to 5), the missing probability gets a value of 1.00 prior to normalization. Finally, the normalized values are loaded into the HMM's transition matrix and ASR is performed using all HMMs. The *%Word Recognition Accuracy* [10][1] on training sentences then is considered the "Fitness Value" of this Offspring, measuring its effect on the whole set of HMMs. If this value is lower than the baseline performance then the GA is executed again (maximum of five times) unless a better accuracy is achieved. If the maximum number of iterations is reached and no better offspring was found, then the original HMM is kept and the process continues with the next phoneme's HMM transition matrix. The convergence plot of this approach is shown in Figure 5. As there are 44 phonemes in the SI

[1] %Word Recognition Accuracy = 100-%Word Error Rate.

HMMs set, and for each HMM the GA is executed up to five times, there are in total 44×5 = 220 iterations of the GA.

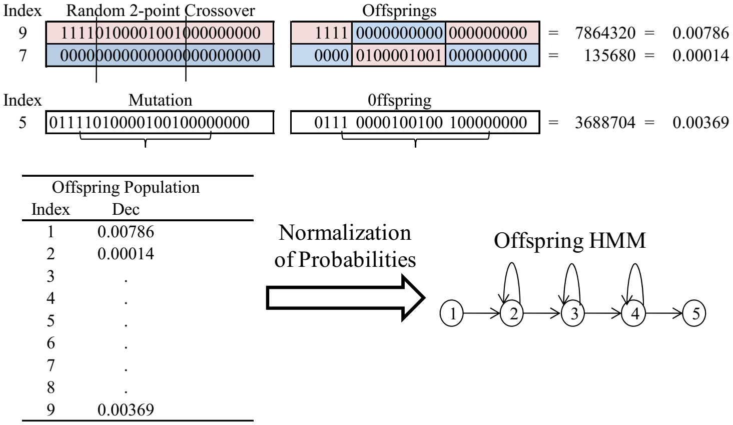


Fig. 3. Approach1: Crossover and Mutation Operators

2.2 Approach 2

In contrast with the previous approach, in this case each individual consists of the transition matrix of a phoneme, where the nine elements of the matrix form the genes of the chromosome representation. As shown in Figure 4, the Initial Population consists of 44 individuals (22 couples), and each individual is represented by nine genes which are also coded in binary form with 23 bits (hence, each individual was represented with 9×23=207 bits). In this case, the total probability (sum of all nine elements in a transition matrix) is used for selection of Parents for reproduction. As in the previous case, these values are normalized in order to compute the cumulative probabilities for the Roulette Wheel selection. This is presented in Table 2.

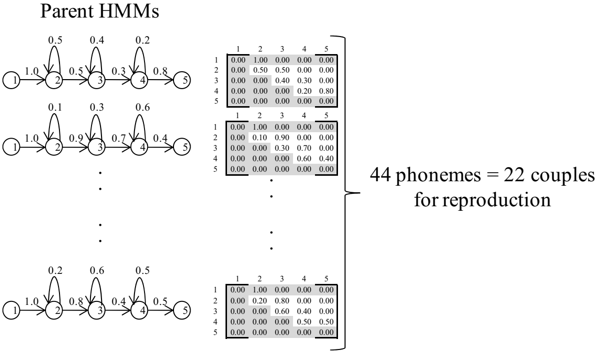


Fig. 4. Approach2: Initial Population

Table 2. Approach2: Roulette Wheel Selection of the Normalized Initial Population

Initial Population													Roulette Wheel Selection									
Index	Dec									Σ	P(x)	Q(x)	Index	Dec								
1	0.50	0.50	0.00	0.00	0.40	0.30	0.00	0.20	0.80	2.70000	0.08257	0.08257	1	0.50	0.50	0.00	0.00	0.40	0.30	0.00	0.20	0.80
2	0.10	0.90	0.00	0.00	0.30	0.70	0.00	0.60	0.40	3.00000	0.09174	0.17431	2	0.10	0.90	0.00	0.00	0.30	0.70	0.00	0.60	0.40
.		.		.		.		.		.	.	.	.									
.		.		.		.		.		.	.	.	.									
.		.		.		.		.		.	.	.	.									
.		.		.		.		.		.	.	.	.									
.		.		.		.		.		.	.	.	.									
.		.		.		.		.		.	.	.	.									
.		.		.		.		.		.	.	.	44	0.20	0.80	0.00	0.00	0.60	0.40	0.00	0.50	0.50
44	0.20	0.80	0.00	0.00	0.60	0.40	0.00	0.50	0.50	3.00000	0.09174	1.00000	6	0.10	0.90	0.00	0.00	0.70	0.30	0.00	0.70	0.30

Once the couples of Parents are selected, crossover and mutation is performed to produce Offsprings. For this case, one-point crossover was performed, where the point of crossover was randomly selected and could be set on any of the chromosome's 207 binary bits. After crossover, mutation is performed, consisting in the change of a randomly selected bit. Also, if the Offspring fails to comply with any restriction, mutation is performed to produce a valid Offspring. At the end, the Offspring is decoded into the original integer value and then divided by 1000000000 to get the phenotype. These values are then normalized to form valid transition probabilities (and thus, a valid transition matrix) to the associated phoneme.

In this approach, each Offspring is tested at a time in order to measure improvements in %Word Recognition Accuracy on the training speech data. If there is an improvement, the next Offspring is tested, otherwise Roulette Wheel selection is performed again. The Offspring that improves HMM performance is kept and the %Word Recognition Accuracy becomes the fitness value for that individual. Once that all 44 Offsprings are tested, the GA is executed again (up to five times) considering this population as the Initial Population. The convergence plot of this approach is shown in Figure 5. Because there are five main iterations of the GA, these were equally distributed for illustration purposes to compare with the previous approach.

3 Speech Data and Baseline Recogniser

A baseline Speaker-Independent (SI) speech recogniser was built using the HTK Toolkit [10]. The british-english Wall Street Journal (WSJ) database [7] was used for this purpose. The training set for the recogniser consisted of the WSJ data from 92 speakers in set *si_tr*. This was used to construct 45 monophone acoustic models. The models were a standard three state left-right topology with eight mixture components per state. The front-end used 12 MFCCs plus energy, delta and acceleration coefficients. A frame period of 10 msec with a Hamming window of 25 msec and 26 filter-bank channels were used.

The experiments were done with speech data from eight speakers of the development set *si_dt* of the same database. From each speaker, 60 sentences were selected, discarding adaptation sentences that were the same for all speakers (these were later used to measure the effect of the GA-Modified SI HMMs on adaptation performance, see Section 4.2). From this set, 30 were selected for GA training purposes (and fitness evaluation), and the remaining 30 for testing. Word bigram language models were estimated from the transcriptions of the *si_dt* speakers, and were the same for all the systems (Baseline SI HMMs and GA-Modified SI HMMs). In total, the vocabulary for GA optimization and testing consisted of 3015 different words.

4 Results

In comparison with [8], the size of the test corpus is significantly larger. In [8] the training data was twice the size of the test data, and also, the stimuli for both sets was the same. In this work, the speech data is more balanced (30 sentences for training, and 30 for testing), and the test vocabulary is different from the one used in the training data. Also the size of the vocabulary is significantly larger, while in [8] it consisted of 11 numerical words, in this work we used 3015 words.

In Figure 5 the mean convergence plots of the GA-Modified SI HMMs for the *si_dt* speakers are shown. While Approach 2 (App2) increases *%Word Recognition Accuracy* in the first iteration (from 70.92% to 71.75%), at the end of both GA executions Approach 1 (App1) performed better (increasing from 70.92% to 73.94%, while App2 increased to 73.02%).

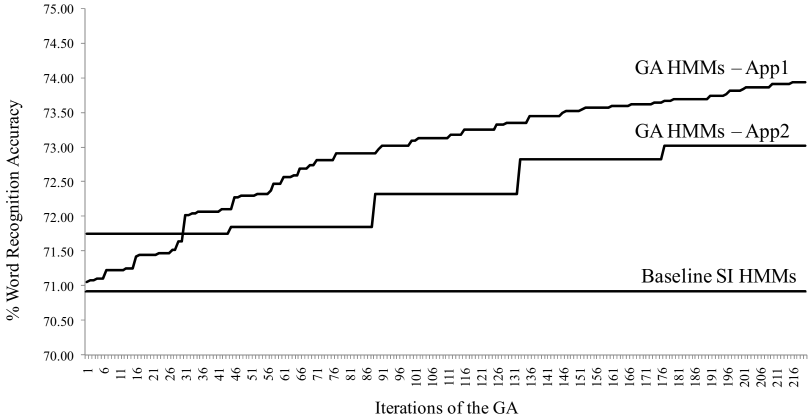


Fig. 5. Convergence plots of both GA approaches

4.1 Performance on the Un-Adapted Baseline

The performance on the test set (30 sentences) for each speaker when using the GA-Modified SI HMMs is shown in Figure 6. With Approach 1 (App1), significant improvements in recognition accuracy were achieved for speakers *c*3*f*, *c*3*j*, *c*31, *c*34 and *c*35. For speaker *c*3*d* there was null improvement, and for speakers *c3c* and *c3l* performance was lower than the baseline. On the other hand, performance was lower with the Approach 2 (App2), achieving only a significant gain for speakers *c3c* and *c3l* (although for this speaker, performance was not better than the baseline). A pair-match test [2] was applied to measure the statistical significance of these results. In Table 3 the results of this test are presented, where for App1 the improvements (reduction in errors) was statistically significant at the 0.05, 0.10, and 0.15 levels, while for App2 no difference was achieved and the number of errors increased.

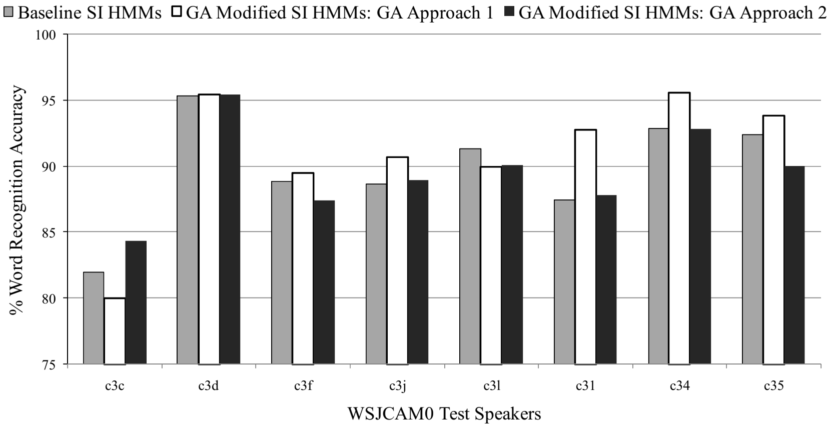


Fig. 6. Performance for all speakers with the test set of sentences and the GA-Modified SI HMMs

Table 3. Significance test

System	Errors	*p- value*		
Baseline	444	0.02555374	<	0.05, 0.10, 0.15
GA App1	401			
Baseline	444	0.63553898	>	0.15
GA App2	452			

4.2 Performance on the Adapted Baseline

Experiments were also performed to measure the effect of the GA-Modified SI HMMs on speaker adaptation. For this, we used the sub-set of sentences (18 in total) from each speaker defined for adaptation purposes which were not considered previously for GA training or testing. Maximum Likelihood Linear Regression (MLLR) [10] was used as the adaptation technique and it was applied on both, the Baseline SI HMMs and the GA-Modified SI HMMs. In Figure 7 the accuracy performance is presented. Note that for this experiment just the HMMs modified with the GA Approach 1 were used.

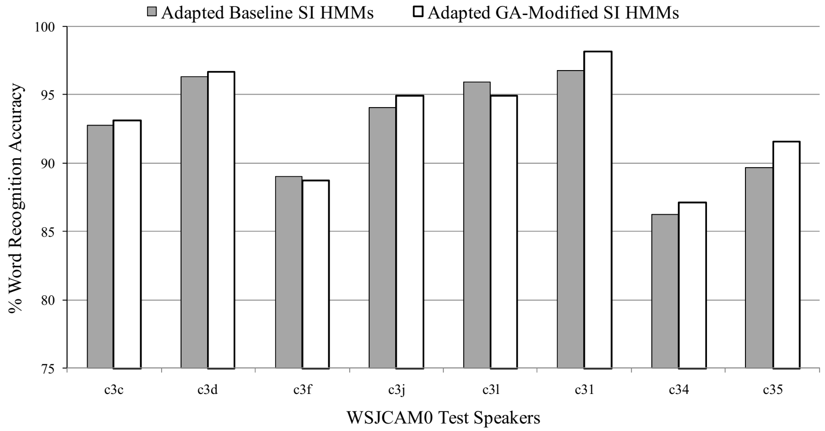


Fig. 7. Performance for all speakers with the test set of sentences and the MLLR-Adapted GA-Modified SI HMMs

As in the un-adapted case, significant improvements in accuracy were achieved for speakers *c*3*d*, *c*3*j*, *c*31, *c*34, and *c*35. In both cases, *c*3*l*'s performance was lower than the baseline, however *c*3*c* achieved improvement, although small, with the adapted GA-Modified SI HMMs. The opposite happened with *c*3*f*, whose performance decreased. In general, the gains in accuracy were statistically significant at the 0.15 level with a *p*-value of 0.143713. Thus, the improvements achieved on SI HMMs with the GA Approach 1 produced significant improvements on adaptation performance.

5 Conclusions and Future Work

In this paper we presented two Genetic Algorithm (GA) approaches to improve transition probabilities and structures within SI HMMs for speech recognition. In the first approach, each phoneme's HMM is optimized one at a time, applying the reproduction operators on sub-sets of the HMM's transition matrix elements (thus, Parents consisted of different elements of the HMM's transition matrix). The second approach consisted of optimization of all HMMs at once, applying the reproduction operators on pairs of HMM's transition matrices (thus, Parents consisted of complete HMM's transition matrix).

The first approach performed better than the second approach, achieving statistically significant improvements in recognition accuracy in experiments with un-adapted and adapted Speaker-Independent (SI) HMMs. However, while improvements were obtained consistently for 80% of the test speakers, no improvement was achieved for the remaining 20%. This can be due to significant acoustic differences or the effect of language model restrictions during the execution of the GA estimates. Hence, more analysis and research will be performed to cover the future work:

- to test the GA Approach 1 with other operators for reproduction and selection;
- integrate the influence of the language model in the recognition process in the design of the GA;
- to increase the test set of speakers and achieve higher improvement in recognition accuracy;
- to apply the GA on other HMM parameters as the number of states and the observation probabilities associated to each HMM state.

References

1. Chan, C.W., Kwong, S., Man, K.F., Tang, K.S.: Optimization of hmm topology and its model parameters by genetic algorithms. Pattern Recognition 34, 509–522 (2001)
2. Gillick, L., Cox, S.J.: Some statistical issues in the comparison of speech recognition algorithms. In: Proc. IEEE Conf. on Acoustics, Speech and Signal Processing, pp. 532–535 (1989)
3. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Co. (1989)
4. Hong, Q.Y., Kwong, S.: A genetic classification method for speaker recognition. Engineering Applications of Artificial Intelligence 18, 13–19 (2005)
5. Jurafsky, D., Martin, J.H.: Speech and Language Processing. Prentice Hall, Pearson (2009)
6. Rabiner, L.: A tutorial on hidden markov models and selected applications in speech recognition. Proc. IEEE 37, 257–286 (1989)
7. Robinson, T.: WSJCAM0: A british english speech corpus for large vocabulary continuous speech recognition. In: Proc. IEEE Conf. on Acoustics, Speech and Signal Processing, pp. 81–84 (1995)
8. Takara, T., Iha, Y., Nagayama, I.: Selection of the optimal structure of the continuous hmm using the genetic algorithm. In: Proceedings of ICSLP 1998 (1998)
9. Xiao, J., Zou, L., Li, C.: Optimization of hidden markov model by a genetic algorithm for web information extraction. In: Proc. of the International Conference on Intelligent Systems and Knowledge Engineering, ISKE 2007 (2007)
10. Young, S., Woodland, P.: The HTK Book (for HTK Version 3.4). Cambridge University Engineering Department (2006)