A Fast Algorithm for HMM Training using Game Theory for Phoneme Recognition

J. Ujwala Rekha
Dept. of Computer Sci. & Engg.
JNTUH College of Engineering,
Hyderabad, Telangana, India

K. Shahu Chatrapati
Dept. of Computer Sci. & Engg.
JNTUH College of Engineering,
Manthani, Telangana, India

A Vinaya Babu
Dept. of Computer Sci. & Engg.
JNTUH College of Engineering,
Hyderabad, Telangana, India

ABSTRACT

Hidden Markov Models are widely used for modeling and predicting label sequences in ASR. In this paper, a game-theoretic approach for Hidden Markov Model training that is superior in terms of time-complexity over Baum-Welch algorithm is introduced. Furthermore, accuracy of recognition using proposed algorithm is comparable with that of Baum-Welch algorithm.

Keywords

HMM Training, Phoneme Recognition, Baum-Welch Algorithm

1. INTRODUCTION

The speech signal is a sequence of acoustic observation vectors, and learning from observation sequences is a fundamental problem in Automatic Speech Recognition (ASR) systems. It comprises two problems, namely segmenting observation sequences and annotating observation sequences. The most prevalent formalism for modeling and predicting label sequences in ASR systems is based on Hidden Markov Models (HMMs) and their variants. HMMs can be considered as a generalization of finite-state automata, where both the transitions between states and the generation of output symbols are dependent on probability distribution ([1],[2]). Learning from the sample data in HMM means estimating the transition probability matrix of the hidden states and emission probabilities of an observed sequence of data. Baum-Welch algorithm is one of the widely used algorithms to estimate these parameters. It uses dynamic programming to find the Maximum Likelihood (ML) estimate of the HMM parameters [3].

Dynamic programming structures the optimization problems into multiple stages, which are resolved in sequence, one stage at a time [4]. The solution of each one-stage problem aids in solving the next one-stage problem in the sequence. The states related to each stage of the optimization problem are the states of the process, and reflect the information required to gauge completely the consequences that the current decision has upon the future actions. Another characteristic of the dynamic programming is the recursive formulation of the problem. Despite the suitability of the dynamic programming in solving optimization problems, it becomes prohibitively expensive when there are more than two or three states in the model formulation limiting its applicability in practice.

Unlike Baum-Welch algorithm that uses the dynamic programming' formulation, an alternate method based on game theory is proposed. The proposed method estimates HMM parameters based on game theory which eliminates the need for exhaustive state space search. While HMMs are used to detect strategies in games [5], the proposed method uses games to train HMMs. Experiments are conducted to evaluate the proposed algorithm against Baum-Welch algorithm on TIMIT database [6]. The results demonstrate the superiority of the proposed algorithm over Baum-Welch algorithm in terms of computational complexity. Furthermore, the accuracy of the recognition using the proposed algorithm is comparable with that of Baum-Welch algorithm.

In Section 2, some background on HMMs and Baum-Welch re-estimation of parameters is presented. Game theory and Nash equilibrium concepts are discussed in Section 3. Section 4 describes the game-theoretic formulation for HMM training. Experiments and results are demonstrated in Section 5 and Section 6 presents conclusions.

2. HIDDEN MARKOV MODELS

The notations used in this paper follow the conventions used in the HTK book [7]. In HMMs, the sequential dependencies among the observation vectors are modeled as a Markov chain. A Markov model is a finite state automaton and makes transitions from some state i to another state j every time unit with probability a_{ij} as shown in Figure 1. At each time t that a state j is entered, it generates a speech vector o_t with the probability distribution $b_j(o_t)$. The joint probability that the observation sequence $O = (o_1, o_2, o_3, \ldots, o_T)$ is generated by the model M progressing across the state sequence $X = (x(0), x(1), x(2), \ldots, x(T))$ is computed as the product of the transition probabilities and the output probabilities given as follows

$$P(O, X \mid M) = a_{x(0)x(1)} b_{x(1)}(o_1) a_{x(1)x(2)} b_{x(2)}(o_2) \ldots \tag{1}$$

However, in HMMs, the observation sequence O is known while the underlying sequence of states X is hidden. Since X is hidden, the likelihood of observing sequence O can be approximated by considering the most likely state sequence calculated as follows

$$P(O \mid M) = \max_{x} \{ a_{x(0)x(1)} \prod_{t=1}^{T} b_{x(t)}(o_t) a_{x(t)x(t+1)} \} \tag{2}$$

Generally, the output distributions are a Gaussian Mixtures, generated with probability $b_j(o_t)$ given as follows

$$b_j(o_t) = \mathcal{N}(o; \mu, \Sigma) \tag{3}$$

where $\mathcal{N}(.; \mu, \Sigma)$ is a multivariate Gaussian with mean vector μ_j and covariance matrix Σ_j given as follows

$$\mathcal{N}(o;\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^n \left|\Sigma_j\right|}} e^{-\frac{1}{2}(o-\mu_j)' \Sigma_j^{-1}(o-\mu_j)} \tag{4}$$

where n is the dimensionality of O.

Given an observation sequence O, training an HMM of N states, involves determining the transition probabilities a_{ij} and the emission probabilities $b_j(o_t)$ (for all $1 \le i \le N, 1 \le j \le N$ and $1 \le t \le T$) that maximizes the likelihood of $P(O|M)$. Determining the parameters of HMM is not trivial and there is no way to analytically solve for the values that maximizes the likelihood of (2). The common method is to train the HMM with the sample data by some iterative procedure until a local maximum is reached. Initially a rough guess of the parameters is done, then a set of re-estimation formula are applied iteratively, until the likelihood of the observation sequence is maximized.

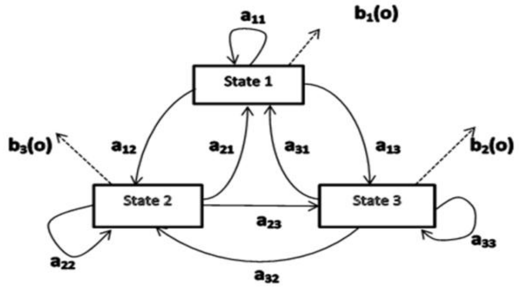

Figure 1. Illustration of Hidden Markov Model [12]

2.1 Baum-Welch Re-estimation

For K states and T vectors in an observation sequence K^T hidden state sequences need to be considered to find the Maximum Likelihood estimate of the parameters that maximize $P(O|M)$. For better understanding, Figure 2 gives the visualization of state transitions unfold over time. In contrast, Baum-Welch algorithm calculates only intermediary forward and backward probabilities at each step as shown in Figure 3, making it cost-effective than the naïve method that calculates all possible probabilities at every step of the observation sequence.

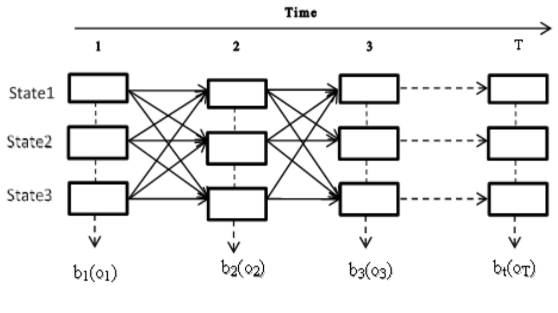

Figure 2. Visualization of State Transitions unfold over time [12].

The Baum-Welch algorithm employs Expectation Maximization (EM) algorithm to find the Maximum Likelihood estimate of the parameters of an HMM given a set of observed feature vectors. EM is an iterative method, alternating between two steps Expectation-Step (E-Step) and Maximization-Step (M-Step). The E-Step calculates expected likelihood of an observed sequence using current estimates of parameters and the M-Step computes new parameters maximizing the expected likelihood found in the E-Step.

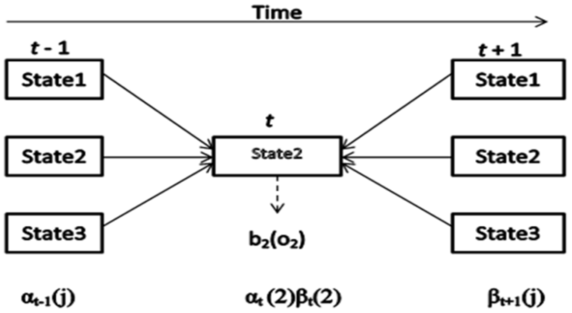

Figure 3. The illustration of the required forward and backward terms in order to compute the forward-backward variables for state 2 at time *t* [12].

Let the forward probability defined as $\alpha_j(t) = P(o_1, o_2, \ldots, o_t, x(t) = j \mid M)$, be the probability of observing observation sequence $o_1, o_2, \ldots, o_t$ and being in state j at time t. This can be calculated recursively as follows

$$\alpha_j(t) = \left[\sum_{i=1}^{N-1} \alpha_i(t-1) a_{ij}\right] b_j(o_t) \tag{5}$$

$$\alpha_1(1) = 1 \tag{6}$$

$$\alpha_j(1) = a_{1j} b_j(o_t) \qquad \text{for } 1 < \text{j} < \text{N} \tag{7}$$

Similarly, let the backward probability be defined as $\beta_j(t) = P(o_{t+1}, \cdots, o_T \mid x(t) = j, M)$, and interpreted as the probability of observing observation sequence $o_{t+1}, \cdots, o_T$ given state j at the time t. This can also be calculated recursively as follows

$$\beta_i(t) = \sum_{\text{j}=2}^{N-1} a_{ij} b_j(o_{t+1}) \beta_j(t+1) \tag{8}$$

$$\beta_i(t) = a_{iN} \qquad \text{for } 1 < \text{i} < \text{N} \tag{9}$$

Let $\xi_{ij}(t)$ denote the probability of the HMM being in state i at time t and moving to state j at the time $t+1$. Using Bayes law, it can be calculated as in (10).

$$\xi_{ij}(t) = P(x(t) = i, x(t+1) = j \mid O, M) = \frac{\alpha_i(t) a_{ij} b_j(o_{t+1}) \beta_j(t+1)}{P(O \mid M)} \tag{10}$$

If $\gamma_j(t)$ is the probability of being in state j at time t, then it can be calculated as follows

$$\gamma_j(t) = \frac{\alpha_j(t)\beta_j(t)}{P(O \mid M)} \quad (11)$$

From $\xi_{ij}(t)$ and $\gamma_j(t)$, $\hat{a}_{ij}$ can be calculated as given below

$$\hat{a}_{ij} = \frac{\text{expected no. of transitions from state i to j}}{\text{expected no. of transitions from state i}} = \frac{\sum_{t=1}^{T-1} \xi_{ij}(t)}{\sum_{t=1}^{T-1} \gamma_i(t)} \quad (12)$$

Finally, μ_j and Σ_j in (4) can be calculated from $\gamma_j(t)$ as follows

$$\hat{\mu}_j = \frac{\sum_{t=1}^{T} \gamma_j(t) o_t}{\sum_{t=1}^{T} \gamma_j(t)} \quad (13)$$

$$\hat{\Sigma}_j = \frac{\sum_{t=1}^{T} \gamma_j(t)(o_t - \mu_j)(o_t - \mu_j)'}{\sum_{t=1}^{T} \gamma_j(t)} \quad (14)$$

The Baum-Welch algorithm can now be summarized as follows. In the E-Step forward probabilities $\alpha_j(t)$ and backward probabilities $\beta_j(t)$ are calculated for each state j at each time unit t from the current estimates using equations (5) through (9). In the M-Step the values for $\xi_{ij}(t)$, $\hat{a}_{ij}$, $\gamma_j(t)$, $\hat{\mu}_j$ and $\hat{\Sigma}_j$ are re-estimated using equations (10) through (14). The above steps are repeated until $P(O \mid M)$ reaches a local maximum. The time complexity of Baum-Welch algorithm is given by $O(IN^2T)$ where I is the number of iterations. Though this algorithm is more efficient than naïve method, it can substantially slow down and can even make HMMs impractical to use if the observation sequence is very large.

3. GAME THEORY

A game consists of a set of N players, a set of possible strategies S_i for each player i and a payoff function $u_i : S_1 \times \cdots \times S_N \to R$ for each player i.

One of the cornerstones of the game theory is the solution concept called Nash equilibrium [8]. Let $s_i \in S_i$ be the strategy taken by a player i and s_{-i} denote the $(n-1)$ dimensional vector of the strategies taken by all other players. Nash equilibrium is a choice of strategies taken by the players such that the strategy of each player is the best response to the strategy of all other players. Mathematically, a joint strategy profile $s_i, \cdots, s_N$ is a Nash equilibrium if for all players $i \in N$

$$u_i(s_i, s_{-i}) \geq u_i(s_i^{'}, s_{-i}) \quad (15)$$

In other words, no player can increase his payoff by changing his own strategy.

4. GAME THEORETICAL HMM TRAINING

Training of an HMM can be thought of as T coordinating agents trying to maximize a joint reward function. Therefore, HMM training can be mapped to a game and in this section a game-theoretic formulation for HMM training is presented.

4.1 Proposed Algorithm

Each agent corresponds to an observation vector $o_t \in O$ and can be associated with any of the N states. The game is played sequentially, where each agent t chooses to be associated with any of the N states. If the agent t at time t chooses to be associated with the state i, then $s_t(i) = 1$. In other words, if $s_t(i) = 1$, then HMM makes the transition to state i at time t generating observation vector o_t. The payoff agent t gains, is given as follows

$$u_t = \sum_i s_t(i) a_{x(t-1)x(t)} b_i(o_t) \quad (16)$$

Therefore, the strategy of any agent t is governed by the probability distribution and is dependent on the choices made by the previous agents. The best strategy for the agent t is the solution to the following optimization problem

$$\max_i u_t \quad (17)$$

subject to the constraints

$$s_t(i) \in \{0,1\} \quad \text{for all } 1 \leq t \leq T \text{ and } 1 \leq i \leq N \quad (18)$$

$$\sum_{i=1}^{N} s_t(i) = 1 \quad \text{for all } 1 < t < T \quad (19)$$

The joint reward function of all agents $P(O \mid M)$ can be rewritten as follows

$$P(O \mid M) = \prod_{t=1}^{T} u_t = \sum_{t=1}^{T} \ln(u_t) \quad (20)$$

Thus, maximizing each agent's payoff maximizes joint reward $P(O \mid M)$.

Similar to the Baum-Welch algorithm, there are two steps in our algorithm. In the E-Step, the game is played by T agents using current estimates to find the hidden state sequence $X = (x(0), x(1), x(2), \ldots, x(T))$ that maximizes each agent's payoff u_t, and the expected likelihood of the observed sequence $P(O \mid M)$ is calculated from (20). In the M-Step, the parameters are re-estimated/adjusted to maximize the expected likelihood found in the E-Step. The two steps are iterated until the Nash equilibrium is reached.

4.2 Existence of the Nash Equilibrium

Theorem 1 *A game has a Nash equilibrium if for all* $i \in I$, *the set* S_i *are non-empty, convex and compact subset of*

Euclidean space and the utility function u_i *is continuous and quasiconcave in each* S_i [9].

In the context of the game theoretic HMM training discussed in Section 4.1, all the conditions for the existence of Nash equilibrium are satisfied. Therefore, Nash equilibrium exists for the problem.

4.3 The Time Complexity of the Game Theoretical HMM Training

Each agent t $(1 \le t \le T)$ can be associated with any of the N states. Therefore, the time complexity of game theoretical HMM Training is $O(TNI)$ where I is the number of iterations.

5. EXPERIMENTS AND RESULTS

Before discussing the results of the experiments, a brief introduction of data set and experimental setup are given first.

5.1 Speech Database and Experimental Setup

The TIMIT database [6] is used to evaluate the performance of the proposed game-theoretic approach for training HMM against Baum-Welch algorithm. The TIMIT is a phonetically transcribed corpus containing a total of 6300 sentences - 10 sentences spoken by 630 speakers of different dialects of US divided into two sets for training and test consisting of 4620 and 1680 sentences respectively.

Experiments for phoneme recognition are conducted using proposed and Baum-Welch algorithm for HMM training. The original 61 TIMIT phonemes are mapped to 39 phonemes as in [10]. The Hidden Markov Model Tool Kit (HTK) is used in the experiments [11]. The HTK is a suite of tools used for ASR consisting of tools for data preparation, training, recognition and analysis. While in one set of experiments the available implementation of Baum-Welch algorithm provided with HTK training tools is used along with other tools, in another set of experiments the implementation of the proposed algorithm is integrated with HTKs tools of data preparation, recognition and analysis.

5.2 Results

The number of iterations for the convergence of the Baum-Welch algorithm and the proposed algorithm are investigated to measure the performance of the algorithms. The effect of the data set size and the length of observation sequence on the convergence of algorithm are analyzed and presented in Figure 4 and Figure 5 respectively. While there is no relation between the data set size and convergence of algorithms, there is an increase in the number of iterations with the increase in the length of observation sequence. It can be seen that the relation between the number of iterations and the length of the observation sequence in game-theoretic approach for HMM training is clearly linear. While convergence is slow with the increase in the observation sequence length in Baum-Welch algorithm also, the explicit relation between the number of iterations and observation sequence length cannot be established. Irrespective of data set size and observation sequence length, convergence of the proposed algorithm is superior to the Baum-Welch algorithm.

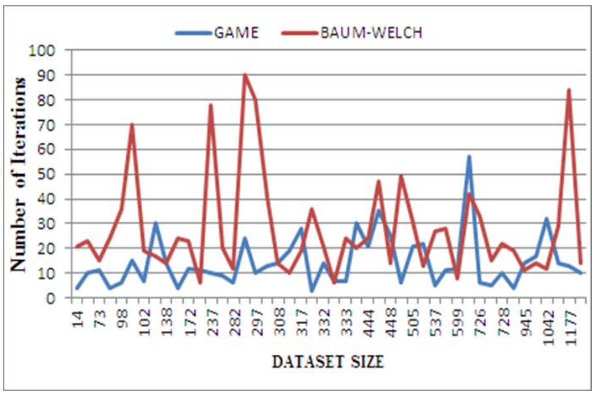

Figure 4. Data set size Vs. Number of iterations

In order to evaluate the effectiveness of our training algorithm against the Baum-Welch algorithm accuracy of phoneme recognition is measured both on training and test set of TIMIT and the results are presented in Table 1. It can be seen that the accuracy of the phoneme recognition system with proposed training algorithm is comparable with that of BW algorithm.

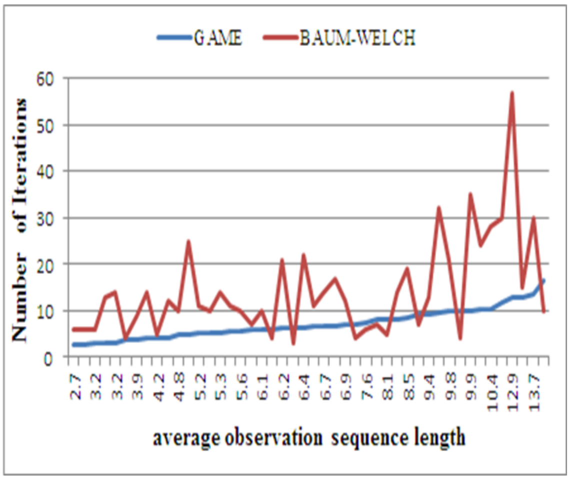

Figure 5. Average observation sequence length Vs. Number of Iterations

Table 1. Comparison of the Accuracies

	BAUM-WELCH	**GAME-THEORY**
TRAINING SET	92.51%	89.99%
TEST SET	91.52%	89.72%

6. CONCLUSIONS

A game-theoretic method for estimation of HMM parameters is proposed. To verify the efficiency and effectiveness of the proposed method, experimental comparisons with Baum-Welch algorithm are done for phoneme recognition. Results show that the proposed algorithm converges faster than Baum-Welch algorithm and the accuracy of recognition using the proposed training algorithm is comparable with that of the Baum-Welch algorithm.

7. REFERENCES

[1] Stolcke, A., & Omohundro, S. (1993). Hidden Markov model induction by Bayesian model merging. Advances in neural information processing systems, 11-11.

[2] Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE, 77(2), 257-286.

[3] Baum, L. E., Petrie, T., Soules, G. & Weiss, N. (1970), ‘A maximization technique occurring in the statistical analysis of probabilistic functions in Markov chains’, The Annals of Mathematical Statistics 41(1), 164–171.

[4] Bradley, S. P., & Arnoldo, C. (1977). Hax, and Thomas L. Magnanti. Applied Mathematical Programming.

[5] Shachat, Jason, J. Todd Swarthout, and Lijia Wei (2012). A hidden Markov model for the detection of pure and mixed strategy play in games. No. 1202. Xiamen Unversity, The Wang Yanan Institute for Studies in Economics, Finance and Economics Experimental Laboratory.

[6] Lemel, L., Kassel, R., & Seneff, S. (1986). Speech database development: Design and analysis. In Proc. DARPA Speech Recognition Workshop, Report no. SAIC-86/1546.

[7] Evermann, G., Kershaw, D., Moore, G., Odell, J., Ollason, D., Valtchev, V., & Woodland, P. (1997). The HTK book (Vol. 2). Cambridge: Entropic Cambridge Research Laboratory.

[8] Nash, J. (1951). Non-cooperative games. Annals of mathematics, 286-295.

[9] Fudenberg D, Tirole J. (1991). Game Theory. Cambridge, MA: MIT Press

[10] Lee, K. F. and Hon, H. W. (1989), “Speaker-Independent Phoneme Recognition Using Hidden Markov Models”, IEEE Transactions on Acoustics, Speech, and Signal Processing, 37(12), pp. 1641-1648.

[11] HTK3. Retrieved December 5, 2014, from http://htk.eng.cam.ac.uk/

[12] Pfundstein, G. (2011). Hidden Markov Models with Generalised Emission Distribution for the Analysis of High-Dimensional, Non-Euclidean Data (Dissertation, Institut für Statistik).