IMPROVED HYBRID MODEL OF HMM/GMM FOR SPEECH RECOGNITION

Poonam Bansal, Anuj Kant, Sumit Kumar, Akash Sharda, Shitij Gupta

Abstract: In this paper, we propose a speech recognition engine using hybrid model of Hidden Markov Model (HMM) and Gaussian Mixture Model (GMM). Both the models have been trained independently and the respective likelihood values have been considered jointly and input to a decision logic which provides net likelihood as the output. This hybrid model has been compared with the HMM model. Training and testing has been done by using a database of 20 Hindi words spoken by 80 different speakers. Recognition rates achieved by normal HMM are 83.5% and it gets increased to 85% by using the hybrid approach of HMM and GMM.

Keywords: Speech Recognition, GMM, HMM

ACM Classification Keywords: I5.4 Pattern Recognition- Applications- Conference proceedings

Conference: The paper is selected from International Conference "Intelligent Information and Engineering Systems" INFOS 2008, Varna, Bulgaria, June-July 2008

Introduction

Speech signal primarily conveys the words or message being spoken. Area of speech recognition is concerned with determining the underlying meaning in the utterance. Success in speech recognition depends on extracting and modeling the speech dependent characteristics which can effectively distinguish one word from another. A general pattern recognition system consists of 4 parts, the feature extractor, pattern trainer, pattern classifier and decision logic. One of the most common and successful feature extraction processes is MFCC [Kotnik, 2002] which has been implemented in this model. Of the various speech/ speaker recognition models available, the most commonly used are Artificial Neural Networks (ANN), Hidden Markov Model (HMM) and Gaussian Mixture Model (GMM). Presently HMM is widely used as one of the successful speech recognition process. HMM considers the speech signal as quasi- static for short durations and models these frames for recognition. It breaks the feature vector of the signal into a number of states and finds the probability of a signal to transit from one state to another [Rabiner, Juang, 1993]. Viterbi search, forward-backward and Baum-Welch algorithms are used for parameter estimation and optimization [Rabiner, 1989], [Rabiner, Juang, 1991]. GMM on the other hand considers a signal to contain different components that are independent of each other. These components represent the broad acoustic classes that represent certain vocal tract configurations [Reynolds, Rose, 1995]. Thus it is more inclined towards modeling features concerning to words having specific characteristics. Each component is optimized using EM algorithm [Dempster, 1977]. Various ways of combining these models have been proposed. One way is to use a 3 state HMM and implement GMM on the middle state to obtain the observation probabilities for that state [Rodriguez, 2003]. Here we have proposed a novel model combining GMM and HMM to provide more enhanced speech recognition engine, which we call as the hybrid model. We have proposed the extraction of likelihood value by combining the likelihood values of both the models using decision logic. In the rest of the paper we discuss individually about the HMM, the GMM and how the hybrid model was developed. At the end, we compare the HMM model with the hybrid model using a 20 word, 80 utterances data set.

Hidden Markov Model

Hidden Markov model (HMM) is a statistical model in which the system being modeled is assumed to be a Markov process with unknown parameters, and the challenge is to determine the hidden parameters from the observable parameters. The extracted model parameters can then be used to perform further analysis, for example for pattern/speech recognition applications. In a hidden Markov model, the states are not directly visible, but variables influenced by the state are visible. Each state has a probability distribution over the possible output tokens. Also, the state transitions are probabilistic in nature. Therefore, the sequence of tokens generated by an HMM gives some information about the sequence of states. The complete HMM model is denoted as $\lambda = (A, B, \pi)$.

HMM Model Parameters

1. A, the state probability distribution, A = {aij}.
2. B, the observation symbol probability density, B = {bj(k)}.
3. π, the initial state distribution, π = { πi}.

The HMM training procedure requires codebook to estimate model parameters. In codebook, the large number of observational vectors of the training data is clustered into M observational vectors clusters using K-means iterative procedure. Based on this clustered observational vectors, estimates of the model parameters are generated during HMM training. The HMM training procedure tries to estimate the value of state probability distribution (A), observation symbol probability density (B), and initial state distribution (π). The observational vectors of training sequences are segmented for each of the N states, a maximum – likelihood estimates of the set of the observations that occur within each state j each of the observation vectors within a state is coded using the M – code – word codebook. Before proper estimation of the model parameters they are initialized to good initial estimates that are essential for rapid and proper convergence of the re-estimation formulas. The parameters are re-estimated using Viterbi Algorithm, Forward- Backward Algorithm and Baum/Welch Algorithm.

Forward- Backward Algorithm

Forward Algorithm

The forward variable αt(i) is defined as αt(i) = P(o1,o2,… ,ot,qT = i|λ) i.e. the probability of the partial observation sequence (until time t) and state i at time t, given the model λ. αt(i) is inductively computed by following steps:

Initialization:

$$\alpha_1(\iota) = \pi_i B_i(o_1), 1 \le \iota \le N \qquad (1)$$

Induction:

$$\alpha_{t+1}(j) = [\sum \alpha_t(i) a_{ij}] b_j(o_{t+1}), 1 \le t \le T-1 \qquad (2)$$

Termination:

$$\mathrm{P(O}|\lambda) \;=\; \sum \alpha_t(\mathrm{i}) \qquad (3)$$

Backward Algorithm

The backward variable βt(i) is defined as βt(i) = P(ot+1,ot+2,…,oT,qT = i|λ) i.e. the probability of the partial observation sequence from t+1 to the end, given the state i at time t and the model λ. βt(i) is inductively solved as follows:

Initialization:

$$\beta_t(i) = 1, 1 \le i \le N \qquad (4)$$

Induction:

$$\beta_{\mathrm{t}}(i) = \sum a_{ij} b_j b_j(o_{t+1})\ \beta_{t+1}(j),\ t = \mathrm{T}-1, \mathrm{T}-2, ..., 1, \quad 1 \le i \le \mathrm{N} \qquad (5)$$

Combining Forward and Backward variables, we get:

$$\mathrm{P(O}|\lambda) = \sum \alpha_t(i) \beta_t(i), 1 \le t \le \mathrm{T} \qquad (6)$$

The Viterbi Algorithm

To find the single best state sequence, for the given observation sequence, δt(i) = max P[q1q2…qt-1, qt = i, o1o2…ot|λ], δt(i) is the highest probability and ψt(i) is the track for path t.

Initialization:

$$\delta_1(i) = \pi_i b_i(o_1), 1 \le i \le N$$
$$\psi_1(i) = 0 \qquad (7)$$

Recursion:

$$\delta_t(j) = \max[\delta_{t-1}(i) a_{ij}] b_j(o_t), 2 \le t \le T, 1 \le j \le N \qquad (8)$$

$$\psi_t(t) = \arg\max[\delta_{t-1}(i)a_{ij}], 2 \le t \le T, 1 \le j \le N$$

Termination:

$$P^* = max\,[\delta_t(i)]1 \le i \le N$$
$$q_T^* = \arg\max\,[\delta_t(i)]1 \le i \le N \quad (9)$$

Path (state sequence) backtracking

$$qT^* = \psi_{t+1}(q^*_{t+1}), t = T-1, T-2, \ldots 1 \quad (10)$$

The Baum/Welch Algorithm

To adjust the model parameters to satisfy a certain optimization criteria is done by the Baum/Welch algorithm.

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)} \quad (11)$$

$$\gamma_t(i) = \sum_{j=1}^{N}\xi_t(i,j) \quad (12)$$

$$a_{ij} = \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} \quad (13)$$

$$b_{ij} = \frac{\sum_{t=1}^{T}\gamma_t(j)}{\sum_{t=1}^{T}\gamma_t(j)} \quad (14)$$

Using the final re-estimated A, B and π, the value of LIHMM is calculated with respect to all the word models available with the recognition engine by using Viterbi algorithm. The Viterbi algorithm takes model parameters and the observational vectors of the word as input and returns the value of matching with all particular word models. This is the likelihood values of the word (LIHMM) passed to hybrid training model.

The Gaussian Mixture Model

The GMM can be viewed as a hybrid between parametric and non- parametric density models. Like a parametric model, it has structure and parameters that control the behavior of density in known ways. Like non-parametric model it has many degrees of freedom to allow arbitrary density modeling. The GMM density is defined as weighted sum of Gaussian densities:

$$p_{GM}(x) = \sum_{m=1}^{M} w_m\, g(x, \mu_m, C_m) \quad (15)$$

Here m is the Gaussian component (m=1…M), and M is the total number of Gaussian components. wm are the component probabilities (∑wm = 1), also called weights. We consider K-dimensional densities so the argument is a vector x = (x1, ... , xK)T . The component pdf, g(x, μm, Cm), is a K-dimensional Gaussian probability density function (pdf).

$$g(x, \mu_m, C_m) = \frac{1}{(2\pi)^{K/2}|C_m|^{1/2}} e^{-\frac{1}{2}(x-\mu_m)^T C_m^{-1}(x-\mu_m)} \quad (16)$$

where μm is the mean vector, and Cm is the covariance matrix.

Now, a Gaussian mixture model probability density function is completely defined by a parameter list given by

θ = {w1, μ1, C1... w1, μ1, C1}m m=1…M

Organizing the data for input to the GMM is important since the components of GMM play a vital role in the making of word models. For this purpose, we use K- Means clustering technique to break the data into 256 cluster centroids. These centroids are then grouped into sets of 32 and then passed into each component of GMM. As a result we obtain a set of 8 components for GMM. Once the component inputs are decided, the GMM modeling can be implemented.

EM Algorithm

The expectation maximization (EM) algorithm is an iterative method for calculating maximum likelihood distribution parameter estimates from incomplete data (elements missing in feature vectors). The EM update equations are used which gives a procedure to iteratively maximize the log-likelihood of the training data given the model. The EM algorithm is a two step process:

Estimation Step in which current iteration values of the mixture are utilized to determine the values for the next iteration

$$\gamma(m,t) = \frac{w_m^{(i)} g(X_t, \mu_m^{(i)}, C_m^{(i)})}{\sum_{j=1}^{M} w_j^{(i)} g(X_t, \mu_j^{(i)}, C_j^{(i)})} \quad (17)$$

Maximization step in which the predicted values are then maximized to obtain the real values for the next iteration.

$$\mu_m^{(i+1)} = \frac{\sum_{t=1}^{T} \gamma_{m,t} X_t}{\sum_{t=1}^{T} \gamma_{m,t}} \quad (18)$$

$$W_m^{(i+1)} = \sum_{t=1}^{T} \gamma_{m,t} \quad (19)$$

$$\lambda_{m,j}^{(i+1)} = \frac{\sum_{t=1}^{T} \gamma_{m,t} (x_{t,j} - \mu_{m,j}^{(i+1)})^2}{\sum_{t=1}^{T} \gamma_{m,t}} \quad (20)$$

EM algorithm is well known and highly appreciated for its numerical stabilities under a threshold values of λmin. Using the final re-estimated w, μ and C, the value of LIGMM is calculated with respect to all the word models available with the recognition engine as:

$$L_{GMM} = \frac{1}{T} \sum_{t=1}^{T} \log P_{GM}(x_t) \quad (21)$$

The Hybrid Model

The GMM/HMM hybrid model has the ability to find the joint maximum probability among all possible reference words W given the observation sequence O. In real case, the combination of the GMMs and the HMMs with a weighted coefficient may be a good scheme because of the difference in training methods. The i^{th} speaker independent GMM produces likelihood L^iGMM, I = 1, 2,…, W, where W is the number of words. The i^{th} speaker independent HMM also produces likelihood L^iHMM, I = 1, 2,…, W. All these likelihood values are passed to the so – called likelihood decision block, where they are transformed into the new combined likelihood L' (W):

$$L'(W) = (1 - x(W)) L^{i}_{GMM} + x(W) L^{i}_{HMM} \quad (22)$$

where x(W) denotes a weighting coefficient.

The value of x is calculated during training of the Hybrid Model. In Hybrid Testing, the subset of training data is used and it's HMM & GMM likelihood values are calculated which are combined using weighing coefficient. Static values of weighted coefficient are also used in order to get higher recognition rate.

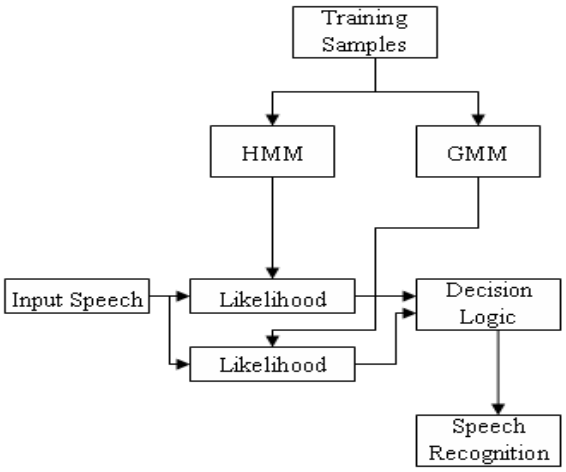

Figure 1: Hybrid Model Block Diagram

Results and Conclusions

The HMM model developed had been tested for different combinations of cluster sizes and number of states. The success rate (%) with different combinations of clusters and states are shown in figure 2. It was observed that as the cluster size increases success rate also increases, because of the reduction in quantization distortion , while increasing the number of clusters. Secondly the time complexity of the recognition engine is also computed with various cluster sizes, and it was observed that 3- state HMM model having 128 cluster size has the least time complexity as comparing with other combinations. Since its time complexity is small it can be useful in applications where speed is the deciding factor. The GMM/HMM Hybrid model created has 8 components and 256 clustering size in GMM and 3 states and 128 clustering size in HMM . Success rate with fixed as well as varying values of weighed coefficient was tested and is shown in figure 4. It was observed, that with the varying value of the weighted coefficient x ,the success rate of word recognition by using hybrid model has increased to 84.38% , which earlier was 83.6% with HMM alone. And by fixing the value of x to 0.9 success rate further increased to 85%. The complexity curve for different cluster size is shown in figure 3.

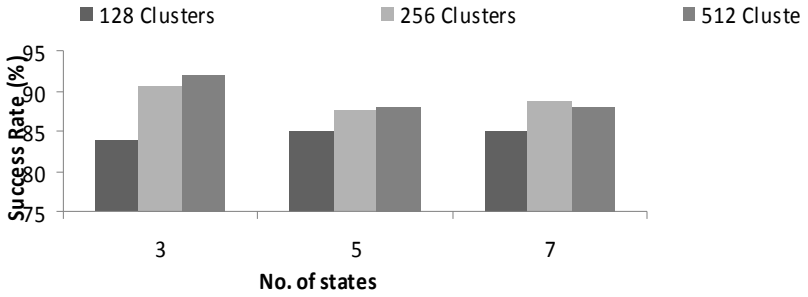

Figure 2: Success Rate verses different combinations of clusters and states

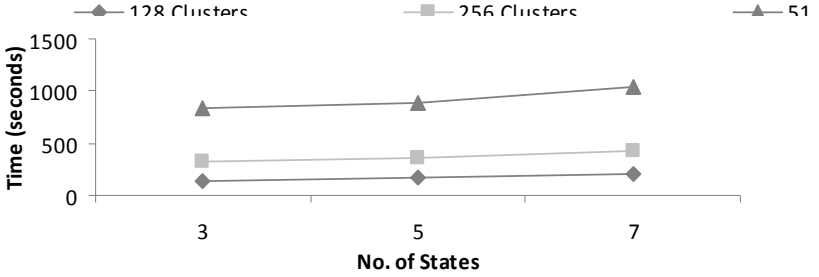

Figure 3: Complexity v/s clusters and states

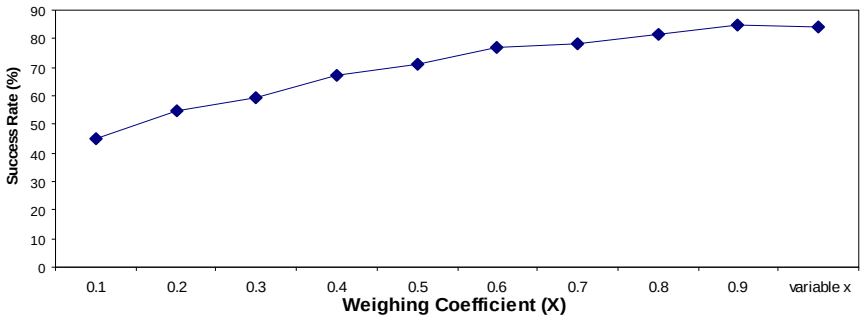

Figure 4: Success Rate v/s weighing coefficient

Bibliography

[Dempster, 1977] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. In: Journal of the Royal Statistical Society: Series B, 39(1) pp. 1–38, November 1977.

[Kotnik, 2002] B. Kotnik, D. Vlaj, Z. Kačič, B. Horvat. Robust MFCC Feature Extraction Algorithm Using Efficient Additive and Convolutional Noise Reduction procedures .In: ICSLP'02 Proceedings, Denver, Colorado, USA pp 445- 448, 2002.

[Rabiner, Juang, 1993] L. Rabiner and B. H. Juang. Fundamentals of Speech Recognition. Prentice Hall, New Jersey, 1993.

[Rabiner, Juang, 1991] B. H. Juang, L. R. Rabiner. Hidden Markov Models for Speech Recognition. In: Technometrics, Vol. 33, No. 3, pp. 251-272, August 1991.

[Rabiner, 1989] L. Rabiner. A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, In: Proceedings of IEEE Volume 77 No. 2 pp 257-286, February 1989.

[Reynolds, Rose, 1995] D. A. Reynolds and R.C. Rose. Robust text-independent speaker identification using Gaussian mixture speaker models. In: IEEE Transactions on Speech and Audio Processing, 3(1), pp. 72–83, 1995

[Rodriguez, 2003] E. Rodriguez, B. Ruiz, A. G. Crespo, F. Garcia. Speech/Speaker Recognition Using a HMM/GMM Hybrid Model. In: Proceedings of the First International Conference on Audio- and Video-Based Biometric Person Authentication, pp. 227- 234, April 2003

Authors' Information

Poonam Bansal – Assistant Professor , Department of Computer Science and Engineering, Amity School Of Engineering and Technology, 580, Delhi Palam Vihar Road, Bijwasan, New Delhi 110061, India ; Member ITHEA. e-mail: pbansal89@yahoo.co.in

Anuj Kant – Department of Computer Science and Engineering, Amity School Of Engineering and Technology, 580, Delhi Palam Vihar Road, Bijwasan, New Delhi 110061, India.

Akash Sharda – Department of Computer Science and Engineering, Amity School Of Engineering and Technology, 580, Delhi Palam Vihar Road, Bijwasan, New Delhi 110061, India.

Sumit Kumar – Department of Computer Science and Engineering, Amity School Of Engineering and Technology, 580, Delhi Palam Vihar Road, Bijwasan, New Delhi 110061, India.

Shitij Gupta – Department of Computer Science and Engineering, Amity School Of Engineering and Technology, 580, Delhi Palam Vihar Road, Bijwasan, New Delhi 110061, India.