VOL 2 (2018) NO 2

e-ISSN : 2549-9904
ISSN : 2549-9610

INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION

Development of TTS Engine for Indian Accent using Modified HMM Algorithm

S. S. Gantayat [#]

[#] *Department of CSE, GMR Institute of Technology, Rajam, Andhra Pradesh, India*
E-mail: sasankosekhar.g@gmrit.org

***Abstract*— A text-to-speech (TTS) system converts normal language text into speech. An intelligent text-to-speech program allows people with visual impairments or reading disabilities, to listen to written works on a home computer. Many computer operating systems and day to day software applications like Adobe Reader have included text-to-speech systems. This paper is presented to show that how HMM can be used as a tool to convert text to speech.**

***Keywords*— K-means, Text-to-speech, Speech synthesis, HMM Algorithm.**

I. INTRODUCTION

A text to speech system is composed of two parts, a front-end and a back-end. The front end has two major tasks. First, it converts the raw text containing symbols like numbers and abbreviations into the equivalent of written-out words. This process is often called text normalization pre-processing or tokenization. The front-end then assigns phonetic transcriptions to each word and divides and marks the text into prosodic units, like phrases, clauses and sentences. The process of assigning phonetic transcriptions to words is called text-to-phoneme or grapheme-to-phoneme conversion. Phonetic transcriptions and prosody information together make up the symbolic linguistic representation that is output by the front-end. The back-end often referred to as synthesizer, it converts the symbolic linguistic representation into sound

HMM-based synthesis is a synthesis method based on Hidden Markov Models, also called Statistical Parametric Synthesis, which is a widely used model for any Speech Recognition and Synthesis applications. In this system, the frequency spectrum (vocal tract), fundamental frequency (vocal source), and duration (prosody) of speech are modeled simultaneously by HMMs. In the proposed project work it is considered to improve the HMM technique to the Indian accents.

Statistical parametric speech synthesis based on hidden Markov models (HMMs) is now well-established and can generate natural-sounding synthetic speech. In this framework, we have pioneered the development of the HMM Speech Synthesis System, HTS (H Triple S). This research started by developing algorithms for generating a smooth parameter trajectory from HMMs. Next, to simultaneously model the excitation parameters of speech as well as the spectral parameters, the multispace probability distribution (MSD) HMM was developed. Using the logarithm of the fundamental frequency and its dynamic and acceleration features as the excitation parameters, the MSD-HMM enabled us to treat the sequence, which is a mixture of one-dimensional real numbers for voiced regions and symbol strings for unvoiced regions, in a probabilistic framework. To simultaneously model the duration parameters for the spectral and excitation components of the model, the MSD hidden semi-Markov model (MSD-HSMM) was developed. The HSMM is an HMM having explicit state duration distributions instead of transition probabilities, to directly model duration; it can generate more appropriate temporal structures for speech. These basic systems employed a mel-cepstral vocoder with simple pulse or noise excitation, resulting in synthetic speech with a “buzzy” quality. To reduce buzziness, mixed or multi-band excitation techniques have been integrated into the basic systems to replace the simple pulse or noise excitation and have been evaluated . These basic systems also had another significant problem: the trajectories generated from the HMMs were excessively smooth due to statistical processing, resulting in synthetic speech with a “muffled” quality. To alleviate this problem, a parameter generation algorithm that considers the global variance (GV) of a trajectory to be generated was developed.

II. MODULES

Module 1: Creation of Speech Corpus (dictionary).
Module 2: Labeling.
Module 3: Prototype HMM (Training of HMMs).
Module 4: Speech Waveform Synthesizer (Model Set).

A. Creation of Speech Corpus

CMU Arctic databases designed for the purpose of speech synthesis research. These single speaker speech databases have been carefully recorded under studio conditions and consist of nearly 1150 phonetically balanced English utterances. They are distributed as free software, without restriction on commercial or non-commercial use. The Arctic corpus consists of four primary sets of recordings (3 male,1 female), plus several ancillary databases. Each database is distributed with automatically segmented phonetic labels. These extra files were derived using the standard voice building scripts of the Festvox system. In addition to phonetic labels, the databases provide complete support for the Festival Speech Synthesis System, including pre-built voices that may be used as is. Festival and Festvox are available at http://www.festvox.org. The Arctic speech corpus is available at http://www.festvox.org/cmu_arctic.

CMU Arctic is a set of single speaker databases that have been carefully recorded under studio conditions, packaged with associated information such as phonetic labels and pitchmark files. An Arctic "database" is a reading of the Arctic prompt set (plus associated files) by a single speaker in a specified style of delivery. This release of Arctic contains recordings by four separate speakers. When referring to the Arctic "corpus" we mean the entire collection of databases, including test sets. The databases have version numbers. As with computer code, version numbers indicate the level of maturity and stability. Numbers with a zero after the decimal point (e.g. version 1.0) are major releases intended to serve as a reference point for system development and evaluation. Minor releases are subject to change, allowing for more frequent additions, deletions, and improvements.

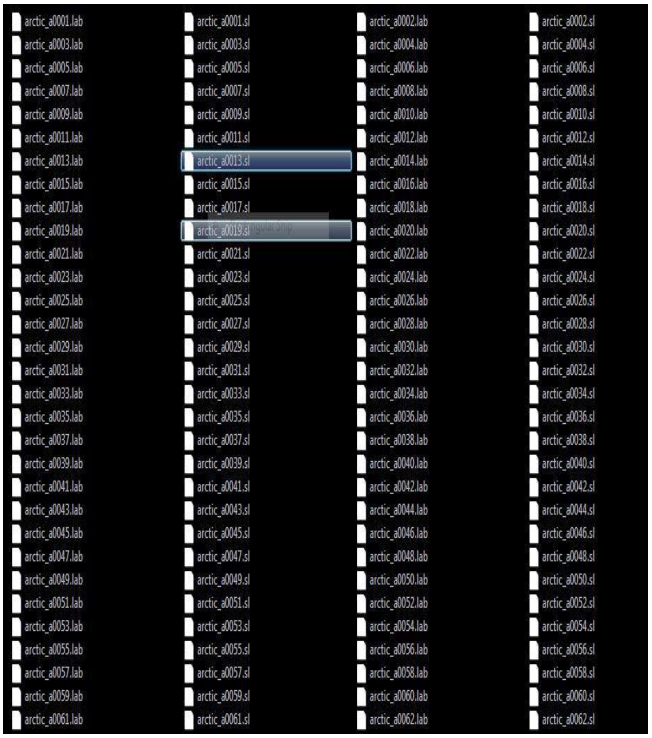
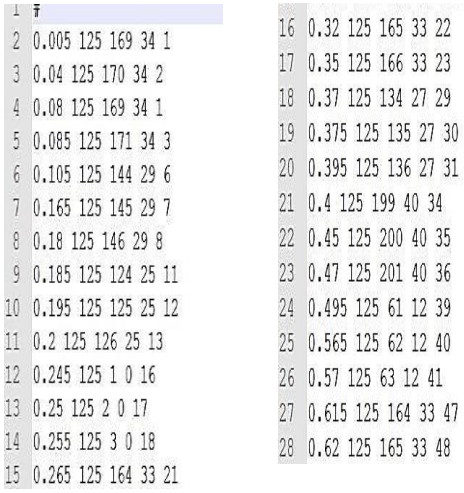


Fig. 1 Label of Dictionary



```
1  #
2  0.005 125 169 34 1
3  0.04 125 170 34 2
4  0.08 125 169 34 1
5  0.085 125 171 34 3
6  0.105 125 144 29 6
7  0.165 125 145 29 7
8  0.18 125 146 29 8
9  0.185 125 124 25 11
10 0.195 125 125 25 12
11 0.2 125 126 25 13
12 0.245 125 1 0 16
13 0.25 125 2 0 17
14 0.255 125 3 0 18
15 0.265 125 164 33 21
16 0.32 125 165 33 22
17 0.35 125 166 33 23
18 0.37 125 134 27 29
19 0.375 125 135 27 30
20 0.395 125 136 27 31
21 0.4 125 199 40 34
22 0.45 125 200 40 35
23 0.47 125 201 40 36
24 0.495 125 61 12 39
25 0.565 125 62 12 40
26 0.57 125 63 12 41
27 0.615 125 164 33 47
28 0.62 125 165 33 48
```

Fig. 2 Contents of Label of Speech File

Markov Model Proces

Classify weather into three states
– State 1: rain or snow
– State 2: cloudy
– State 3: sunny
By carefully examining the weather of some city for a long time, we found following weather change pattern.

TABLE 1
WEATHER PROBABILITY

		Tomorrow		
		snow	**cloudy**	**sunny**
today	**snow**	0.4	0.3	0.3
	cloudy	0.2	0.6	0.2
	sunny	0.1	0.1	0.8

At every state tomorrow weather is depending upon today state Visual representation.

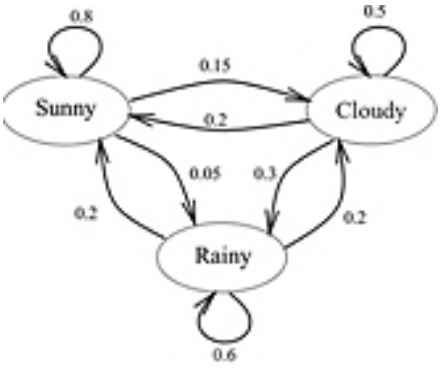


Fig. 3 State Diagram of Weather

Each state corresponds to one observation,
Sum of outgoing edge weights is one
Observable states, {1, 2, …..N}
Observed sequence, $\{q_1, q_2, \ldots q_n\}$
1st order Markov assumption
$P(q_t = j \mid q_t{-}1 = i, q_t{-}2 = k, \ldots..)$
$= P(q_t = j \mid q_t{-}1 = i)$

We denote the time instants associated with the state changes as t=1,2,3……n and actual state at time t as qt.

B. Markov Model: Sequence Probability

• Question: What is the probability that the weather for the next 7 days will be "sun-sun-rain-rain-sun-cloudy-sun" when today is sunny?

S : *rain*, *S* : *cloudy*, *S* : *sunny*

$P(O \mid model) = P(S_3,S_3,S_3,S_1,S_1,S_3,S_2,S_3 \mid model) =$
$P(S_3).P(S_3|S_3).P(S_3|S_3).P(S_1|S_3).P(S_1|S_1).P(S_3|S_1).$
$P(S_2|S_3).P(S_3|S_2) = \pi_3.a_{33}.a_{33}.a_{31}.a_{11}.a_{13}.a_{32}.a_{23}$
$=1.(0.8)(0.8)(0.1)(0.4)(0.3)(0.1)(0.2) = 1.536x10^{-4}$

C. HMM Algorithms

There are three algorithms used in HMM

1. Viterbi Algorithm,
2. Baum-Welch Algorithm,
3. K-Means.

1. Viterbi Algorithm

• Purpose – An analysis for internal processing result, The best, the most likely state sequence for Internal segmentation
– Alignment of observation and state transition
– Dynamic programming technique

Key idea – Span a lattice of N states and T times, to keep the probability and the previous node of the most probable path coming to each state i at time t

Introduction Phase:

$$\delta_1(i) = \pi_i b_i(x_1),\ \psi_1(i) = 0$$

Recursion Phase:

$$\delta_{t+1}(j) = \max_{1<=i<=N} \delta_t(i) a_{ij} b_j(x_{t+1})$$
$$\psi_{t+1}(j) = \text{argmax}_{1<=i<=N}\ \delta_t(i) a_{ij}$$

Termination Phase:

$$P^* = \max_{1<=i<=N} \delta_T(i)$$
$$q_T^* = \text{argmax}_{1<=1<=N} \delta_T(i)$$

Path backtracking Phase:

$$q_T^* = \psi_{t+1}(q_{t+1}^*),\ t = T-1,\ldots..1,$$

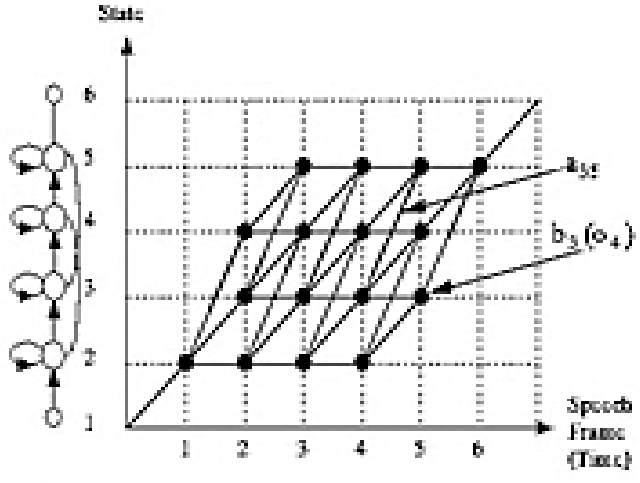


Fig 4. The Viterbi Algorithm for Isolated Word Recognition

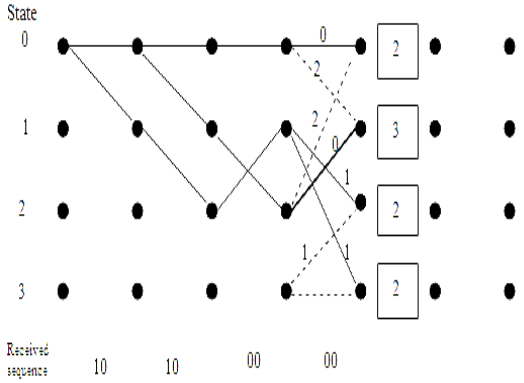


Fig. 5 Mapping of Model

2. Baum-Welch Algorithm

1. The Baum-Welch algorithm can be used to train an HMM to model a set of sequence data.
2. The algorithm starts with an initial model and iteratively updates it until convergence.
3. The algorithm is guaranteed to converge to an HMM that locally maximizes the probability of the training data given the model.

Baum-Welch Reestimation (EM)

– Iterative procedures that locally maximizes
– Convergence proven
– MLE statistic estimation

The following table shows a one run of the Baum-Welch algorithm:

'*a*' is considered as start and ending symbols of the sequence string and through multiple iterations the probability values will be converges.

$e \to a$	$a \to b$	$b \to a$	$a \to b$	$b \to b$	$b \to e$	P(path)	$q \xrightarrow{a} r$	$r \xrightarrow{b} q$	$q \xrightarrow{a} q$	$q \xrightarrow{b} q$
q	r	q	r	q	q	0.00077	0.00154	0.00154	0	0.00077
q	r	q	q	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	r	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	q	q	q	0.02548	0.0	0.000	0.05096	0.07644
Rounded Total →						0.035	0.01	0.01	0.06	0.095
New Probabilities (P) →							0.06	1.0	0.36	0.581
State sequences										

Fig 6 Example of Baum-Welch Algorithm

3. K-Means Clustering

HMM kind of algorithm is well known to be sensitive to its initialization point. The problem of this initialization point choice is addressed in this paper: a model with a very large number of states which describe training sequences with accuracy is first constructed. The number of states is then reduced using a k-mean algorithm on the state. This algorithm is compared to other methods based on a k-mean algorithm on the data with numerical simulations.

The clustering algorithm partitions a dataset into a fixed number of clusters supplied by theuser. Hidden Markov Model (HMM) based clustering method, which identifies a

suitable number of clusters in a given dataset without using prior knowledge about the number of clusters. Initially, the dataset is partitioned into windows of fixed size based on the HMM log likelihood values. This provides a framework for identifying the most appropriate number of clusters (windows of varying sizes). After determining the number of clusters, the data values are then labelled and allocated to clusters.

D. Features of Audio

The following features will be extracted from the audio file for training the HMM to get the speech.

1. Energy Entropy Standard Deviation (std)
2. Signal Energy Std by Mean(average) Ratio
3. Zero Crossing Rate Std
4. Spectral Rolloff Std
5. Spectral Centroid Std
6. Spectral Flux Std by Mean Ratio

III. CONCLUSION

In this paper it is shown how HMM can be used to generate speech from the text file. HMM uses dictionaries, labelling and the three different algorithms like K-means, Viterbi Algorithm and Baum-Welch Algorithm to train and after that the speech will be generated from the text files. The HMM can be used for image processing and forecasting also.

REFERENCES

[1] Junichi Yamagishi, Takashi Nose, Heiga Zen, Zhen-Hua Ling, Tomoki Toda, Keiichi Tokuda, Simon King and Steve Renals,"Robust Speaker-Adaptive HMM-Based Text-to-Speech Synthesis", IEEE Transactions on Audio, Speech, and Language Processing, Vol.17, No. 6, August 2009

[2] Agni Dika1, Adnan Maxhuni1, Avni Rexhepi, "The principles of designing of algorithm for speech synthesis from texts written in Albanian language", IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 3, No 3, May 2012

[3] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," Proceedings of the IEEE, vol. 77, no. 2, pp. 257–286, February 1989.

[4] A. W. Black, K. Lenzo, Building voices in the Festival speech synthesis system, 2000, http://festvox.org/bsv.

[5] Kevin Murphy, "HMM toolbox for Matlab", freely downloadable SW written in Matlab, http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html

[6] Juang BH, Rabiner LR, "Mixture Autoregressive Hidden Markov Models for Speech Signals", IEEE Trans Acoustics, Speech and Signal Processing 33: 1404-1413,1985

[7] Qystein Birkenes,Tomoko Matsui, Kunio Tanabe, Sabato Marco Siniscalchi, Tor Andre Myrvoll, and Magne Hallstein Johnsen, "Penalized Logistic Regression with HMM LogLikelihood Regressors for Speech Recognition", IEEE Transactions on Audio, Speech, and Language Processing Vol. 18, No. 6, pp. 1440-1454, August 2010.

[8] Tokuda, K., Nankaku, Y., Toda, T., Zen, H., Yamagishi, J., Oura, K.,"Speech synthesis based on hidden markov models", In: Proceedings of the IEEE, Vol. 101(5), pp. 1234–1252 (2013)

[9] HMM-based Speech Synthesis System (HTS). http://hts.sp.nitech.ac.jp/

[10] L.E. Baum, T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains", Ann. Math. Stat., 37 (1966), pp. 1554-1563

[11] Olivier Cappé Eric Moulines Tobias Rydén, Inference in Hidden Markov Mode,l Springer Series in Statistics, 2005.

[12] Y. Ariki, M.A. Jack, "Enhanced time duration constraints in hidden Markov modelling for phoneme recognition", Electronics Letters, 25 (13) (22 June 1989), pp. 824-825.