Speech Recognition using HMMs

Saaketh Preetham[1], KedarNath[1], and Lakshmi[1]

IIIT-Hyderabad
{saaketh, kedar, lakshmi_g}@students.iiit.net

Abstract. This report presents an overview of speech recognition technology, software, development and applications. It begins with a description of how such systems work, and the level of accuracy that can be expected. A brief comparison of the most common systems is presented. This project deals with the study of the practical difficulties in implementing HMMs for speech recognition and how HTK ToolKit can be used to model HMMs to recognize speech samples to give excellent results for both isolated words as well as connected words.HMMs have been used for recognition as speech samples contain observation vectors of variable length which are best modeled by Hidden Marchov Models.

1 INTRODUCTION:

As people become ever more mobile and national and global economies ever more integrated, an ever larger population finds themselves needing to communicate in a language that is not their own. For Automatic Speech Recognition (ASR) this vast number of speakers of multiple languages implies the need to deal with accented speech, and indeed adapting to foreign-accented speech is an important problem in current speech recognition research

Methods for dealing with accented speech vary from simply collecting data in that accent and training a recognizer, to various ways of adapting recognizers trained on unaccented speech.

Speech recognition technologies allow computers equipped with a source of sound input, such as a microphone, to interpret human speech, e.g. for transcription or as an alternative method of interacting with a computer. It is the ability to match a speech pattern against a provided or acquired vocabulary.

Our focus in this work is on adaptation. We collected a database of Telugu numbers (for isolated word recognition) and English numbers (for connected word recognition), and then investigated a number of different methods for improving recognition.

There are various methods which can be applied for speech recognition.Some of them are

- *Comparison of feature Vectors* (Similarity Measure: Sequence Edit Distance): In this method, the means of each class in the training set are calculated. The test sample is classified into the class whose mean has the least sequence edit distance from it.Edit distance is the least cost of operations required in

order to match two sequences of feature vectors. Operations include insertion , deletion, replacement.

– *HMMs*: The Left-Right Model of the HMMs are used for speech recognition.The details are discussed later in the report.

– *Neural Networks*: Neural Networks can also be used for recognizing speech . A special kind of neural network called the Spiking Neural Network (SNN) can trained to perform isolated word recognition.

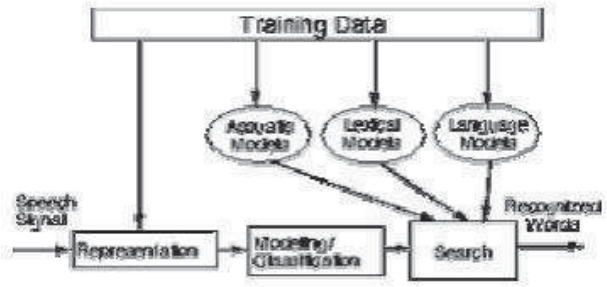
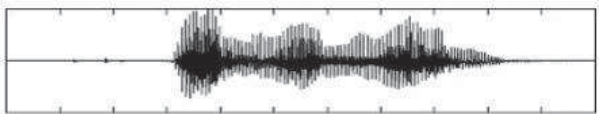


Fig. 1. Components of a typical speech recognition system.

A hybrid/variations of the above mentioned methods can also be used to achieve better results.We have implemented the Left-Right HMMs as the underlying state sequence associated with this model has the property that as time increases the state index increases (or stays the same), i.e., the states proceed from left to right. Clearly the left-right typeof HMM has the desirable property that it can readily model speech signals whose properties change overtime.

2 BACKGROUND WORK:

2.1 SPEECH

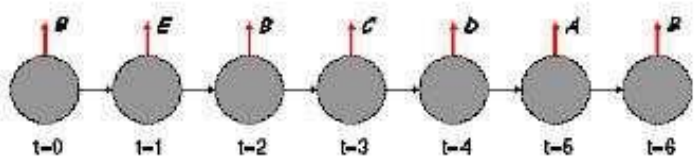


– Speech can be described as an act of producing voice through the use of the vocal cords.
– A speech signal can be understood as a sequence of phonemes or observations.
– Most useful parameters in speech processing are found in the frequency domain.
– Various defining features of speech can be captured by ceptral analysis
  • *MFCCs :* They contain Speech-related information.
  • *LPCCs :* They contain Speaker-related information.
– In general, the number of observations that describes a particular object is constant. But in case of speech , the number of feature vectors that constitute the observation sequence is of variable size.
– These kinds of sequences are best modeled by Hidden Markov Models.

2.2 MARKOV MODELS:

– Has N states: w1, w2 .. wN.
– Discrete time steps: t=0, t=1..
– On the tth timestep the system is in exactly one of the available states.
– At each time step, the next state is chosen randomly.
– The current state determines the probability distribution for the next.

2.3 HIDDEN MARKOV MODELS:



– Various states constitute this model
– These states are hidden.
– They emit various observations with different probabilities.
– Three basic problems with HMMs.

  • Given the observation sequence 0 = O1 O2. . * Or, and a model A = (A, 6, ir), how to compute P(O—A) (the probability of the observation sequence) efficiently ,given the model?
    It can also be calculated using the backward algorithm.

$$P(O|\lambda) = \sum_{\text{all } Q} P(O|Q, \lambda)\, P(Q|\lambda)$$
$$= \sum_{q_1, q_2, \cdots, q_T} \pi_{q_1} b_{q_1}(O_1)\, a_{q_1 q_2} b_{q_2}(O_2)$$
$$\cdots a_{q_{T-1} q_T} b_{q_T}(Q_T).$$

But this is computationally very expensive.

Hence , the forward algorithm is used.

$$\alpha_t(i) = P(O_1\, O_2 \cdots O_t,\, q_t = S_i|\lambda)$$
$$\alpha_1(i) = \pi_i b_i(O_1), \quad 1 \le i \le N.$$

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), \quad 1 \le t \le T-1$$
$$1 \le j \le N.$$

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i).$$

- Given the observation sequence 0 = 0, O2 . . . Or, and the model A,how to choose a corresponding state sequence Q = q1 q2. . . qJ which is optimal in some meaningful sense (i.e., best .explains. the observations)?

  The problem is to finding the "optimal" state sequence associated with the given observation sequence.The optimality criterion is to choose the states g, which are individually most likely.This is found by finding the state with maximum alpha value at each time interval.This is called the Viterbi Algorithm.

- How to adjust the model parameters A= (A, B, T) to maximize P(O—A)?

  The parameters are iteratively re-estimated using the forward-backward algorithm.

– Types:
  - ERGODIC MODEL:
    In the ergodic model there is a state transition from each state to every other state.

  - LEFT-RIGHT Model(BAKIS):
    * State transitions are possible only to higher states(limited number).

X = (A, B, *T*) have to be choosen such that *P(O|h)* is locally maximized using an iterative procedure such as the Baum-Welch method (or equivalently the EM (expectation-modification) method .

$$\alpha_j(t) \leftarrow \sum_{i=1}^{c} \alpha_i(t-1) a_{ij} b_{jk} v(t)$$

Alpha at time t gives the probability with which the model has generated the observation vector till time t.

$$\beta_i(t) \leftarrow \sum_{j=1}^{c} \beta_j(t+1) a_{ij} b_{jk} v(t+1)$$

Beta at time t gives the probability with which the model would generate the rest of the observation sequence.

$$\gamma_{ij}(t) = \frac{\alpha_i(t-1) a_{ij} b_{jk} \beta_j(t)}{P(V^T \mid \theta)}$$

Gama gives the probability with which the model has a transition from state i to state j at time t.

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \gamma_{ij}(t)}{\sum_{t=1}^{T} \sum_{k} \gamma_{ik}(t)} \qquad \hat{b}_{jk} = \frac{\sum_{t=1}^{T} b_{jk}(t)}{\sum_{t=1}^{T} \sum_{r} b_{jr}(t)}$$

* There are no transitions to states whose indices are lower than the current state.
* Initial state probabilities would be such that the state sequence always starts from state 1.
* Additional constraints are placed on state transitions such that large changes in state indices do not occur.
* The last state can have a transition to no other state other than itself.

3 METHODOLOGY:

FEATURE EXTRACTION: Feature extraction involves convertion of speech waveform to some type of parametric representation. Most useful parameters in speech processing are found in the frequency domain. Since this project involves speech recognition, Mel scale cepstral analysis (MEL) is used to generate the mfcc co-efficients which characterize various speech sounds. Mel frequency cepstral coefficients result from the Discrete Cosine Transform of the filterbank spectrum (in dB).

MODEL: The Left-Right Hidden Marchov Model is used for recognition.

PROCEDURE: Two types of HMMs can be used for recognition.

- HMMs with *Discrete probability distribution*:
  - A codebook of discrete feature vectors is generated from the input sequences. The number of basic feature vectors is assumed to be 32-256.This is called Vector Quantization.
  - Discrete probability is assigned at each state for each of these basic feature vectors.
  - Probabilities are learned using the training set.
- HMMs with *Continuous probability distribution*:
  A Gaussian distribution is considered at each state.

Both isolated words and connected words can be recognized using hmms.

- HMM parameters are initialized randomly.
- The training set is used to train the different HMM models to recognize different pre-defined set of words using the *Baum-Welch algorithm* (Forward-Backward Learning)
- Each test sample is matched against the network of HMMs built above to output a transcription using the *VITERBI ALGORITHM.*

4 IMPLEMENTATION:

4.1 HMMs with Discrete probability distribution

VECTOR QUANTIZATION:

The procedure basically partitions the training vectors into M disjoint sets (where M is the size of the codebook),represents each such set by a single vector , which is generally the centroid of the vectors in the training set assigned to the mth region, and then iteratively optimizes the partition and the codebook (i.e., the centroids of each partition).

The K-Means Clustering procedure has been used for this purpose. Given the list of training data files(ceptral co-oeffiecients),the codebook.cpp program generates a sequence of codebook of feature vectors.

Associated with VQ is a distortion penalty since we are representing an entire region of the vector space by a single vector. Clearly it is advantageous to keep the distortion penalty as small as possible.However, this implies a large size codebook, and that leads to problems in implementing HMMs with a large number of parameters.

RECOGNITION:

Method: Using Hidden Markov Models. (Left-Right Model)
For isolated word recognition with a distinct HMM designed for each word in the Vocabulary, it is clear that a left-right model is more appropriate than an ergodic model, since we can then associate time with model states in a fairly straightforward manner. Furthermore,we can envision the physical meaning of the model states as distinct sounds (e.g., phonemes, syllables) of the word being modeled.

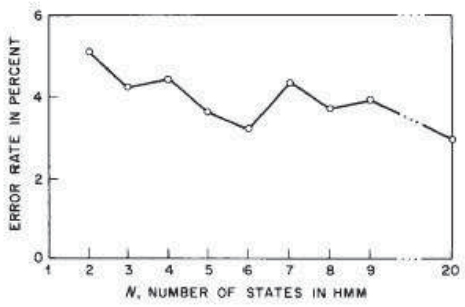
CHOICE OF MODEL PARAMETERS:

NSIZE:Number of hidden states.

- The number of states correspond roughly to the number of sounds (phonemes) within the word-hence models with 2-10 states are appropriate.
- The number of states correspond roughly to the average number of observations in a spoken version of the word, the so-called Bakis model. In this manner each state corresponds to an observation interval-i.e., about 15 ms for the analysis we use.

FVSIZE: Size of feature vector.
This depends upon the ceptral co-efficients being used. Lpccs: 17 Mfccs: 13

DELTA: Number of states forward that the trellis can move.Constraint put by the left-right model.



BTHRES: Minimum probability threshold with which a state emits an observation.

A constraint that bj(k) be greater than or equal to some minimum value BTHRESH is necessary to insure that even when the kth symbol never occurred in some state j in the training observation set, there is always a finite probability of its occurrence when scoring an unknown observation set. It can be seen that over a very broad range (10-l0 ¡= BTHRESH ¡= 10^-3) the average error rate remains at about a constant value; however, when BTHRESH is set to 0 , then the error rate increases sharply.

a[i][j] : Probability that there is a transition from state i to state i+j. 0¡=j¡C(delta of the left-right model). These values are initialized randomly using the rand() function such that Sigma j (aij)=1

b[j][k] : Probability that the jth state emit observation k. (k corresponds to one of the observations in the codebook.)
T : Size of the observation vector.This depends upon the input speech sample.
These probabilies are learned using the Baum-Welch forware-backward learning so that each model recognizes one word in the vocabulary.

IMPLEMENTATION DETAILS:

Training:

getAlpha : Values of the alpha are found using forward algoirthm.The actual values of alpha become so small that the system cannot handle them.In order to overcome this difficulty,the values are scaled so that the values are within reasonable range.
getBeta : Values of the alpha are found using backward algorithm.The values are beta are also scaled similar to alpha.
getGama : Values are found using the alpha and the beta values.
estimateA : Train the values of a using the gama values obtained using the present observation sequence.

estimateB : Train the values of b using the gama values obtained using the present observation sequence.
Train : Train the trellis to recogize the given class of observation sequence by adjusting the parameter values.The training procedure is continued till the values of the model parameters(A.s and B.s) converge, i.e. (Diff. between new values and old values is very small.)

Evaluation:
Returns the probability that the trellis generates a particular sequence of feature vectors. The system cannot handle the small values which result.Hence ,log probability is used.

Classification:
Bayes. Rule is used for classification. The test sample is labelled as belonging to the class corresponding to the model which evaluates the sample with the maximum probablity.

IMPLEMENTATION ISSUES:

- Various values such as alpha, beta etc are scaled so that the values are within the bounds such that system can handle them.
- A Threshold is used for the b[j][k] values so that is does not become zero.
- double values are used instead of floats so that there would not be any problem with precision of the number.
- Limiting the number of state transition for each state also reduces the computation complexity.
- Log probabilities are used so that the values are within bounds.

OBSERVATIONS:

When this implementation is tested on English Aphabet dataset , it is observed that the results are not satisfactory.
It can be seen that W (which has a set of phonemes very different from the rest of the alpabets) is recognized correctly, but B (which sounds very similar to other alphabets such as C,D,E,G,P,T etc) is labeled incorrectly as P.

* This is because there are many implementation issues which have to be answered in the training process.
* The elements in the data set are very similar to each other , hence the error rate is very high.
* The data set used for training is very small and therefore , the training is not complete.
* The codebook has been generated using the K-Means procedure which is not the best method.

Table 1. Test Results on the alphabets W and B

W	B
A -872.686	A -2496.59
B -659.497	B -2335.97
C -1146.02	C -2606.21
D -1442.92	D -2752.26
E -625.525	E -2640.41
F -1234.64	F -2433.38
G -780.291	G -2406.51
H -666.783	H -2560.42
I -590.755	I -2408.99
J -856.932	J -2393.41
K -1316.73	K -2475.13
L -1236.61	L -2573.23
M -1473.03	M -2616.19
N -915.292	N -2572.74
O -1302.41	O -2438.42
P -916.079	P -1714.52
Q -846.175	Q -2516.13
R -813.488	R -2581.7
S -1382.93	S -2562.87
T -1114.43	T -2630.34
U -906.666	U -2750.66
V -693.82	V -2571.57
W -528.659	W -2138.19
X -1319.39	X -2542.23
Y -982.41	Y -2505.36
Z -1340.81	Z -2581.18

* When a bigger data-set with words of bigger length was used , the values converged to very small values and resulted in nans.

In order to overcome all these practical difficulties , the standard HTK ToolKit has been used.

4.2 HMMs with CONTINUOUS PROBABILITY DISTRIBUTION:

CREATING TRAINING CORPUS:

The data set is split into training set and testing set and the corresponding lists are created. Word-level transcription is done for each training sample file.

FEATURE ANALYSIS:

A spectral analysis of the speech signal is performed to give observation vectors which can be used to train the HMMs which characterize various speech sounds.

HCOPY is configured to automatically convert its input into MFCC vectors.

UNIT MATCHING SYSTEM:

First a choice of speech recognition unit must be made. Possibilities include linguistically based sub-word units such as phones (or phone-like units), diphones, demisyllables, and syllables, as well as derivative units such as fenemes, fenones, and acoustic units. Other possibilities include whole word units, and even units which correspond to a group of 2 or more words (e.g., and an, in the, of a, etc). Typically each such unit is characterized by some type of HMM whose parameters are estimated from a training set of speech data. The unit matching system provides the likelihoods of a match of all sequences of speech recognition units to the unknown input speech.
Monophones have been used in this implementation.

LEXICAL DECODING:

This process places constraints on the unit matching system so that the paths investigated are those corresponding to sequences of speech units which are in a word dictionary (a lexicon). This procedure implies that the speech recognition word vocabulary must be specified in terms of the basic units chosen for recognition.
This dictionary of words in the training set is extracted from standard reference pronunciation Dictionaries such as the BEEP Dictionary . The HDMan command of the HTK TookKit is used for this purpose.This command also gives the list of monophones present in the training set.
HDMan -m -w wlist -n monophones1 -l dlog dict beep

SYNTACTIC ANALYSIS:

This process, much like lexical decoding, places further constraints on the unit matching system so that the paths investigated are those corresponding to speech units which comprise words (lexical decoding) and for which the words are in a proper sequence as specified by a word grammar.

In order to obtain the monophones , a dictionary has to be build containing the possible pronunciations of words in the training set. One reference pronunciation Dictionary is required like BEEP Dictionary(for English words).

Phone-level transcription is done using the set of monophones , the dictionary and the word-level transcription files.This is very helpful while training the hmm models using phones as the basic unit.

HMM INITIALIZATION AND TRAINING: HMM DEFINITION(FLAT-START INITIALIZATION):

A topology for each hmm is choosen a priori.It specifies
- Number of states
- Form of the observation functions (associated with each state)
- Disposition of transitions between states

This is specified in the proto file.

HMM models are initialized to a standard values.This is done using the HCompV command which scans a set of data files, computes the global mean and variance and sets all of the Gaussians in a given HMM to have the same mean and variance.

HCompV -C config -f 0.01 -m -S train.scp -M hmm0 proto

It also generates a variance floor macro (called vFloors) which is equal to 0.01 times the global variance. This is a vector of values which will be used to set a floor on the variances estimated in the subsequent steps.This is done so that the parameter values are within acceptable bounds. A Master Macro File (MMF) called hmmdefs containing a copy for each of the required monophone HMMs is constructed by manually copying the prototype and relabeling it for each required monophone.These are used for subsequent training.
HMM TRAINING:

The flat start monophones are re-estimated using the embedded re-estimation tool HEREST .This procedure is repeated a number of times so that the model would learn the training data.

The amount of computation required by an order of magnitude is reduced by pruning. This limits the range of state alignments that the forward-backward algorithm includes in its summation. HEREST has an option of setting auto-incrementing pruning threshold.

Extra transitions are added in the silence model to make the model more robust by allowing individual states to absorb the various impulsive noises in the training data. The backward skip allows this to happen without committing the model to transit to the following word. The silence models are fixed using the HHEd command.

The parameters are re-estimated many times using the HERest command which essentially implements the Baum-Welch Algorithm.

RECOGNIZER EVALUATION:

* An input speech signal input.wav is first transformed into a series of .acoustical vectors. (here MFCCs) with tool HCopy, in the same way as what was done with the training data.
* The input observation is then process by a Viterbi algorithm, which matches it against the recogniser.s Markov models. This is done by tool HVite:

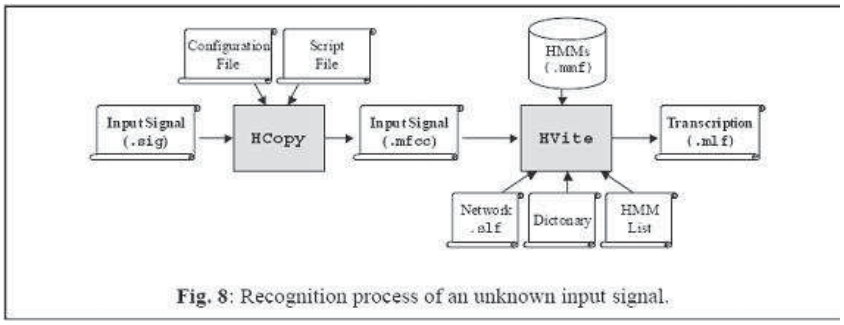


Fig. 8: Recognition process of an unknown input signal.

The output transcriptions can be compared with the actual labels and the performance of the method can be tested. HResults command does this.

5 RESULTS:

On ISOLATED WORDS: Data Set Used: Telugu digits (0 . 9) Training set contains 79 wav files with approximately 8 files of each number. A telugu pronunciation dictionary has been used.

aaru aa r u
aidu ai d u
eidzoo ei dz oo
enimidi e n i m i d i
muudzu m uu dz u
naalugu n aa l u g u
okatzi o k a tz i
rendzu r e n dz u
sunnaa s u nn aa
tommidi t o mm i d i
sil [] sil

———————— Overall Results ————————
SENT : %Correct=100.00 [H=79, S=0, N=79]
WORD : %Corr=100.00, Acc=100.00 [H=79, D=0, S=0, I=0, N=79]

It is observed that hmm trained using HTK ToolKit gives 100% results on isolated words.
On CONNECTED WORDS: The above described system can be extended to recognize connected words.A grammar is used to specify the word network so as to improve the accuracy.A short pause (sp) is introduced to mark the break between the words.
Data Set Used: Miscellaneous English numbers and words. There are 225 training wav files.

————————- Overall Results ————————-
SENT: %Correct=21.00 [H=16, S=84, N=100]
WORD: %Corr=74.72, Acc=49.67 [H=337, D=16, S=98, I=113, N=451]

-
%Correct: Number of utterances matched.
%Corr : Number of words matched.
%Acc : Accuracy

6 CONCLUSIONS:

* HMMs are ideal for speech recognition as the observation vector of each input speech signal is of variable length.

* There are many implementation issues which are difficult to address while implementing HMMs.

* HMMs with Discrete probability distribution do not give properly results when implemented directly as a proper codebook has to be obtained. Generation by simple methods such as K-Means Cluster-

ing results in erroneous codebooks.

* When phones are used as the basic units of the HMM recognizer, the results are accurate.

* Phone-level transcription and word-level transcription result in better training.

* Performance of isolated word recognition is better than that on connected word recognition.

* In connected word recognition , even though the words are recognized correctly, even if one word is labeled incorrectly , the whole speech sample is considered to be labeled incorrectly.This brings down the performace.

* There are short pauses (sp) between words in the speech samples of connected word forms. Even if there is a small disturbance , it is considered to be some word and incorrectly labels the silence.

* If the data-set used for training contains wav files of words spoken by a single person , then the recognizer.s performance on recorded data of another person is poor.This is because not enough variations (such as pitch , stress etc)are present in the training set to train the recognizer properly.

* Directly recorded speech samples without any processing done ,contain a lot of noise which results in improper recognition.

7 REFERENCES:

* A Tutorial on Hidden Markov Models andSelected Applications in Speech Recognition - LAWRENCE R. RABINER, FELLOW, IEEE HTKBook
* HTKTrain: A Package for Automatic Speech Segmentation - Sathish Chandra Pammi, Venkatesh Keri
* HTK (v.3.1): Basic Tutorial -Nicolas Moreau
* J. G. Wilpon, C. H. Lee, and L. R. Rabiner. "Connected digit recognition based on improved acoustic resolution." Computer Speech and Language. Study on Noisy Speech Recognition (Linear Predictive Coding Prediction Error). by FENG Chenglin, WU Shuzhen
* http://cslu.cse.ogi.edu/HLTsurvey/ch1node4.html
* http://www.faqs.org/docs/Linux-HOWTO/Speech-Recognition-HOWTO.htmlUSES