\( \pi = \{ \pi_i \} \): The initial state distribution

\[
\pi_i = P \left[ q_1 = i \right], \quad 1 \leq i \leq N \tag{2.19}
\]

The initial state distribution parameter is more relevant for HMM topologies such as ergodic. Since in ASR the left-to-right topology is preferred, it is typical that the initial state distribution is one for the first state and zero for the remaining states.

Given appropriate values of \( N, M, A, B \) and \( \pi \), the HMM can be used to generate an observation sequence \( O = (o_1, o_2, \ldots, o_T) \) (in which each observation \( o_t \) is one of the symbols from \( V \), and \( T \) is the number of observations in the sequence). The three steps of HMM are:

- Evaluation
- Decoding
- Parameter estimation

### 2.4.1.1 Evaluation

Evaluation is to find probability of generation of a given observation sequence by a given model. The recognition result will be the speech unit corresponding to the model that best matches among the different competing models. In this, we have to find the probability of observation sequence \( O = (o_1, o_2, \ldots, o_T) \) given the model \( \lambda \) i.e., \( P(O|\lambda) \). If \( q = (q_1, q_2, \ldots, q_T) \) is any fixed state sequence, one could in principle compute \( P(O|\lambda) \) by computing the joint probability, \( P(O, q|\lambda) \) for each possible state sequence \( q \) of length \( T \) and then summing all state sequences. Computationally this method is very costly. However, there is an efficient way of computing this probability using forward and backward algorithm.
Forward-Backward Algorithm

The Forward Probabilities:

Consider the forward variable $\alpha_t(i)$ defined as

$$\alpha_t(i) = P(o_1, o_2, \ldots, o_t, q_t = i | \lambda)$$

that is, the probability of the partial observation sequence, $o_1, o_2, \ldots, o_t$ (until time $t$) and state $i$ at time $t$, given the model $\lambda$. The variable $\alpha_t(i)$ can be solved inductively, as follows:

1. Initialization

$$\alpha_t(i) = \pi_i b_t(o_t), \quad 1 \leq i \leq N$$

2. Induction

$$\alpha_{t+1}(j) = \sum_{i=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}), \quad 1 \leq t \leq T-1; \quad 1 \leq j \leq N$$

where $N$ is the number of states in the model.

The Backward Probabilities:

Consider the backward variable $\beta_t(i)$ defined as

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \ldots, o_T | q_t = i, \lambda)$$

that is, the probability of the partial observation sequence from $t + 1$ to the end, given state $i$ at time $t$ and the model $\lambda$. Again, we can solve for $\beta_t(i)$ inductively as follows:

1. Initialization:

$$\beta_T(i) = 1, \quad 1 \leq i \leq N$$
3. Induction:

\[
\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(\omega_{t+1}) \beta_{t+1}(j), \quad t = T-1, T-2, \ldots, 1; \ 1 \leq i \leq N
\]  

(2.25)

The two forward and backward probabilities can be used to compute \( P(O|\lambda) \) according to equation (2.26)

\[
P(O|\lambda) = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(\omega_{t+1}) \beta_{t+1}(j)
\]  

(2.26)

2.4.1.2 Decoding

Decoding is to find the single best state sequence, \( q = (q_1, q_2, \ldots, q_T) \), for the given observation sequence \( O = (\omega_1, \omega_2, \ldots, \omega_T) \). Dynamic programming algorithm used for finding the best state sequence is Viterbi algorithm.

The Viterbi Algorithm

Let \( \delta_t(i) \) the best score (highest probability) along single path at time \( t \), which accounts for the first \( t \) observations and ends in state \( i \) be defined as:

\[
\delta_t(i) = \max_{q_1, q_2, \ldots, q_{t-1}} P[q_1, q_2, \ldots, q_{t-1}, q_t = i, \omega_1, \omega_2, \ldots, \omega_t | \lambda]
\]  

(2.27)

By induction, we have

\[
\delta_{t+1}(j) = \max_{i} \delta_t(i) a_{ij} \cdot b_j(\omega_{t+1})
\]  

(2.28)

To retrieve the best state sequence, procedure is as follows:

Preprocessing

\[
\bar{\pi}_i = \log (\pi_i), \quad 1 \leq i \leq N
\]  

(2.29a)

\[
\bar{b}_i(\omega_t) = \log [b_i(\omega_t)], \quad 1 \leq i \leq N; \ 1 \leq t \leq T
\]  

(2.29b)

38
\[ \tilde{a}_{ij} = \log (a_{ij}), \quad 1 \leq i, \; j \leq N \] (2.29c)

**Initialization:**

\[ \delta_t(i) = \pi_i b_t(o_1), \quad 1 \leq i \leq N \] (2.30a)

\[ \tilde{\delta}_t(i) = \log (\delta_t(i)) = \tilde{\pi}_i + \tilde{b}_t(o_1) \quad 1 \leq i \leq N \] (2.30b)

\[ \psi_t(i) = 0, \quad 1 \leq i \leq N \] (2.30c)

\[ \psi_t(j) \] is an array, used to keep track the argument that maximizes equation (2.28)

**Recursion:**

\[ \tilde{\tilde{\delta}}_j(j) = \log (\delta_t(j)) = \max_{1 \leq i \leq N} [\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}] + \tilde{b}_t(o_j) \] (2.31a)

\[ \psi_t(j) = \arg \max_{1 \leq i \leq N} [\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}], \quad 2 \leq t \leq T ; 1 \leq j \leq N \] (2.31b)

**Termination**

\[ \tilde{P}^* = \max_{1 \leq i \leq N} [\tilde{\delta}_T(i)] \] (2.32a)

\[ q^*_T = \arg \max_{1 \leq i \leq N} [\tilde{\delta}_T(i)] \] (2.32b)

**Backtracking**

\[ q^*_t = \psi_{t+1}(q^*_{t+1}), \quad t = T-1, T-2, \ldots, 1 \] (2.33)

The array \( q^* \) contains the required best state sequence.

**2.4.1.3 Parameter Estimation (Training)**

The solution to the learning or training problem updates the parameters of a HMM \( \lambda_m \) in such a way that the new model \( \lambda_{m+1} \) gives a higher likelihood than \( \lambda_m \) for the training observation sequence. An iterative solution to this problem that is used extensively in speech recognition is Baum-Welch re-estimation. This type of method is used since there is no known way to analytically derive model parameters that maximize the probability
of an observation sequence. The Baum-Welch update algorithm is equivalent to the 
Expectation-Maximization (EM) algorithm.

The Baum-Welch re-estimation algorithm:

The Baum-Welch algorithm is a generalized expectation-maximization (GEM) 
algorithm. It can compute maximum likelihood estimates and posterior mode estimates 
for the parameters (transition and emission probabilities) of an HMM when given only 
emissions as training data.

The algorithm has two steps:

1. Calculating the forward probability and the backward probability for each HMM state.

2. On the basis of this, determining the frequency of the transition-emission pair values 
and dividing it by the probability of the entire string. This amounts to calculating the 
expected count of the particular transition-emission pair. Each time a particular transition 
is found, the value of the quotient of the transition divided by the probability of the entire 
string goes up and this value can then be made the new value of the transition. Baum-
Welch algorithm is the extensively used iterative procedure for choosing the model 
parameters. In this method, we start with some initial estimates of the model parameters 
and modify the model parameters to maximize the training observation sequence in an 
iterative manner till the model parameters reach a critical value. We define the variable 
$\xi(i,j)$ as the probability of being in state $i$ at time $t$, and state $j$ at time $t+1$, given the 
model and observation sequence.

$$\xi(i,j) = P(q_t = i, q_{t+1} = j \mid O, \lambda) \quad (2.34)$$
\[ \xi(i,j) = \frac{\alpha(i) \cdot \pi(i) \cdot \beta(i)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha(i) \cdot \pi(i) \cdot \beta(i)} \] (2.35)

\[ \gamma(i) \] is the probability of being in the state \( i \) at time \( t \), given the entire observation sequence \( O \), and the model \( \lambda \) and is defined as:

\[ \gamma(i) = P(q_t = i \mid O, \lambda) \] (2.36)

\[ \gamma(i) = \frac{\alpha(i) \cdot \beta(i)}{\sum_{i=1}^{N} \alpha(i) \cdot \beta(i)} \] (2.37)

By the definition of the variables \( \xi(i,j) \) and \( \gamma(i) \), the following relations are true,

\[ \gamma(i) = \sum_{j=1}^{N} \xi(i,j) \] (2.38)

\[ \sum_{t=1}^{T} \gamma(i) = \text{expected number of transitions from state } i \text{ in } O \]

\[ \sum_{t=1}^{T} \xi(i,j) = \text{expected number of transitions from state } i \text{ to state } j \text{ in } O \]

Using the above formulas and the concept of counting the event occurrences, the parameters of the model \( \lambda = (A, B, \pi) \) can be re-estimated as \( \tilde{\lambda} = (\tilde{A}, \tilde{B}, \tilde{\pi}) \), where:

\[ \tilde{\pi}_i = \gamma(i) \] (2.39)

\[ \tilde{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi(i,j)}{\sum_{t=1}^{T-1} \gamma(i)} \] (2.40)
\[ b_j(k) = \frac{\sum_{t=1}^{T} \gamma(j)}{\sum_{t=1}^{T} \gamma(j)} \]  \hspace{1cm} (2.41)

It has been proved that one of the following two statements is true for \( \lambda \) and \( \tilde{\lambda} \):

1. The initial model \( \lambda \) defines a critical point of the likelihood function in which case \( \tilde{\lambda} = \lambda \).

2. Model \( \tilde{\lambda} \) is more likely than the model \( \lambda \), in the sense that \( P(O| \lambda) < P(O| \tilde{\lambda}) \).

In case (1), we stop the iterative procedure declaring \( \lambda \) as the final trained model for the observation sequence O. In case (2), we replace the model \( \lambda \) by \( \tilde{\lambda} \) as the initial model for the next iteration. The iteration is stopped when \( \frac{P(O| \tilde{\lambda}) - P(O| \lambda)}{P(O| \lambda)} \) reaches some minimum value and then the model \( \tilde{\lambda} \) is declared as the final trained model for the observation sequence O.

### 2.5 Review of Noise Robust ASR System

Noise robustness is one of the most challenging problems in automatic speech recognition. The performance of automatic speech recognition (ASR) systems, trained with clean speech may drastically degrade in realistic environments. The main reason for this degradation is the acoustic mismatch between the training and test environments due to environmental effects. Many techniques have been proposed to overcome this degradation problem [36]. The robustness of the recognizer can be accomplished in three ways:

1. **Speech Enhancement**: by using speech enhancement techniques to increase the signal to noise ratio (SNR) [37]
(2) **Robust Feature Extraction**: by extracting the robust parametric representation of speech signal to minimize the effect of noise [38,39]

(3) **Model Compensation**: by using model adaptation techniques to dynamically adapt the speech recognition models to noisy speech [40-43].

Many methods have been proposed to increase the robustness of ASR systems. In the following sections some of the existing techniques falling in these categories are discussed.

### 2.5.1 Speech Enhancement

Techniques under this category try to enhance the speech specific aspects of the spectrum by suppressing the noise-specific aspects. An early method falling in this category is the spectral subtraction [37]. Spectral subtraction involves estimating the background noise spectrum, then subtracting it from the measured noisy speech spectrum. The background noise spectrum is estimated usually during non-speech periods. This type of estimation can prove to be a problem because the non-speech periods of the signal must also be reliably estimated. A related approach developed by Hirsch [44] estimates the noise level within a frequency sub-band by taking a histogram of spectral magnitudes over several successive time windows. Hence, care must be taken to compute the histogram over segments that have a sufficient number of non-speech segments. Another approach based on assumptions of a bimodal distribution of the total histogram of the logarithmic spectral energies is presented in [45,46]. Regardless of how the noise statistics are estimated, the true short-term spectrum of the noise for the specific segment being processed always shows finite variance. Thus the noise always overestimates or underestimates the true noise level [37]. This represents a fundamental problem with spectral subtraction and other transform-based methods. An improvement
over the spectral subtraction is the nonlinear spectral subtraction (NSS), which combines the spectral subtraction with noise masking. NSS has been used to improve the speech recognition performance in car noisy conditions [47]. In 2005, some of the authors used the concept of speech presence probability in conjunction with spectral subtraction to achieve noise robustness [48]. This was seen as a soft-decision spectral subtraction superior than hard-decision spectral subtraction [37]. In this the authors proposed three features based on the soft-spectral subtraction, which vary in their pre-processing steps and are termed POST-FILT, POWER-FILT and PSIL.

Another optimal technique for speech enhancement was suggested by Haykin [49]. It was based on adaptive filter theory named Wiener filtering. It is an optimal technique for the signal enhancement in the minimum mean square sense when the noise and the clean speech satisfy the following conditions:

- The noise and speech are statistically independent of each other and the noise is assumed to be stationary.

- Besides the speech plus noise audio channel, one has an access to the 'noise only' channel as well. This condition was required in order to estimate the noise power spectral density.

The major limitation of the Wiener filter is that in the most of the practical scenarios, one does not have an access to the 'noise only' channel. This leads to the problem of estimation of the noise power spectral density which is required to estimate the Wiener filter. Other technique that does not rely on estimating the noise spectrum is noise masking [50]. By adding a constant to each of the spectral components when calculating the cepstrum, the cepstral space can be made insensitive to changes in noise as shown in equation (2.41).
\[ \text{ceptr} = r (\log(C + xe^{j\alpha})) \] (2.41)

where, \( r \) is a cosine transform, \( xe^{j\alpha} \) is the speech spectrum, and \( C \) is the masking level. The disadvantage of this technique is that in clean conditions, as the masking level \( C \) is increased performance degrades.

### 2.5.2 Robust Feature Extraction

The ideal aim of the feature extraction in ASR systems is to extract representations from the speech signal that carries only the linguistic information. The knowledge of the human speech perception mechanism and the human speech production apparatus is utilized to transform the power spectrum to feature vectors that may be useful for the ASR. The goal of these transformations is to emphasize the linguistic information in the signal and to suppress the undesired variabilities present in the signal that do not carry any information about the linguistic message conveyed by the signal. Some of the feature vector sets like Mel frequency cepstral coefficients (MFCC) [15], linear predictive cepstral coefficients (LPCC) [51] and perceptual linear prediction (PLP) [22] have been shown to be quite successful for the clean speech ASR. They are usually combined with their first and second order derivatives [52,53]. There are other various methods reported for feature extraction i.e. principal component analysis (PCA) [54-56], independent component analysis (ICA) [57,58], and linear discriminant analysis (LDA) [59]. In [54,55], the subspace method based on PCA was applied to speech signals in the time domain for noisy speech enhancement, and cepstral features from enhanced speech showed robustness in noisy speech recognition. ICA [58] was applied to speech data in the time or time-frequency domain, and gave good performance in phoneme recognition tasks. In [59], LDA that was applied to speech data in the time-frequency domain showed better performance than combined linear discriminants in the temporal and
spectral domain in continuous digit recognition task. Comparative experimental results using data-driven methods based on PCA, ICA, and LDA in phoneme recognition tasks are described in [60]. Another technique of robust feature extraction is Cepstral Mean Normalization also known as cepstral mean subtraction (CMS), attempts to remove convolutional distortion from the speech signal by subtracting the mean from the cepstral features. Any stationary spectral distortion such as the microphone response is removed by subtracting the dimension mean since convolutional effects are additive in the cepstral domain [61].

Kajarekar et al., [62] have observed in experiments that cepstral mean normalization (CMN) also reduces speaker variability. They note that speaker variability may not only be concentrated in the DC component, but is distributed over the very low frequency band of the modulation spectrum. Another robust and fast technique for robust feature extraction is the relative spectral (RASTA) feature processing [63]. RASTA attempts to remove the irrelevant components by band pass filtering time trajectories of feature vectors between two non-linearities. By band pass filtering, RASTA not only removes the DC component like CMN does, but also influences the speech spectrum. The band pass filter used in RASTA processing is usually of an IIR type. This kind of filter enhances the transitions between speech segments, which may make the resulting feature more perceptually meaningful. Summerfield [64] performed experiments that showed the perception of a speech-like sound by humans depends on the difference between the spectra of the current sound and the preceding sound.

Distorting components can be either convolutional or additive in the signal domain. Convolutional distortions, such as the choice of microphone can be removed when the compressing non-linearity is a logarithm and the decompressing non-linearity is anti-logarithm. A compressive non-linearity such as \( y = \ln (1 + \alpha x) \) is used by Hermansky [38]
to address a combination of additive and convolutional distortions where \( J \) is a signal
dependent coefficient that can be either constant or adapted. Contributions for feature
extraction using autocorrelation domain has been reported in various techniques. One of
the techniques uses short-term modified coherence (SMC) [65]. It is an all-pole modeling
of the autocorrelation sequence followed by a spectral shaper. Modeling the
autocorrelation function which is less affected by noise than the original signal [66] has
been found to be more robust to additive noise. In addition, it may be assumed that only
the first few autocorrelation parameters are affected by the noise, a very good assumption
for near white noise sources. Removing these from the parameter estimation process
further improves the robustness. Another similar approach, one-sided autocorrelation
linear predictive coding (OSALPC) [67] has been found to give even better performance
than SMC with a variety of speech parameters in the car environment. This form of
speech parameters has been tested on the alphabets and digits. All showed improvements
using SMC cepstra over LPCC. However, as with many techniques, it has not been tested
on larger vocabulary systems, nor compared with other compensation techniques.

2.5.3 Model Compensation

Model compensation techniques operate in the pattern matching stage instead of the
feature parameterization stage. Major techniques include the maximum likelihood linear
regression (MLLR) [68], the model decomposition [69], parallel model combination
(PMC) [70], and the structural maximum a posteriori (SMAP) method [71].

G. Rahim et al.,[72] investigated a signal bias removal (SBR) method based on
maximum likelihood estimation for the minimization of the undesirable effects which
occur in telephone speech recognition system such as ambient noise, channel distortions
etc. A maximum likelihood (ML) stochastic matching approach [73] to decrease the
acoustic mismatch between test utterances and a given set of speech models was proposed to reduce the recognition performance degradation caused by distortions in the test utterances and/or the model set.

2.6 Database and Recognition System Used

A standard database is required for the development, evaluation and benchmarking of automatic speech recognition and synthesis systems whereas on the other hand it facilitates a systematic style of acoustic-phonetic correlates of a language. In this section, the database that was used to train the HMMs system and to perform the recognition test is discussed. After that the recognition system used in this thesis is outlined.

Two databases have been used in this work. The first database, TIFR-200 has been used to build and test word recognizer. The second database, NOISEX-92, contains several noises that were added to the signal to simulate noisy conditions. The choice of training data has great influence on system performance. TIFR -200 is a Hindi speech database which is widely used by research institutes and industry.

2.6.1 TIFR Database

This is a product of the Tata Institute of Fundamental Research (TIFR). TIFR-200 contains 30,000 speech data files from 30 speakers. The database has been developed in the studio environment under high quality recording conditions using a Sennheiser microphone model MD421 and a tape recorder model Philips AF6121.

Following are the characteristics of the database:
Language Used: Standard Hindi

Vocabulary Size: A set of 200 most frequently occurring Hindi words

Speakers: 30 speakers

Utterances: (15 male, 15 female and 5 children) 5 repetitions each

Audio Recording: Recording on a cassette tape in studio SNR > 40dB

Digitization: 16KHz. Sampling, 16 bit quantization.

2.6.2 NOISEX-92 Database

Noises from the NOISEX-92 database have been used for the purpose of conduct of experiments [74]. The NOISEX-92 database contains eleven different noises:

- Speech babble noise
- Machine gun noise
- Pink noise
- HF Channel noise
- F16 noise
- Car noise
- Factory noise
- Engine room noise
- Jet Cockpit noise
- Military vehicle noise
- Tank noise

All these noises were recorded at a sampling frequency of 20 kHz and down-sampled to a frequency of 16 kHz. A 16-bit quantization was used to represent the samples. In our experiments, we have selected three of these noises plus white Gaussian noise. The
selected noises are:

1. **Babble noise**: This noise distribution in frequency is equivalent to the long term average speech spectrum. The source of babble noise is 100 people speaking in a canteen. The room radius is over two meters. Therefore, individual voices are only slightly audible.

2. **Factory noise**: Noise is recorded on factory floors near plate-cutting and electrical welding equipment.

3. **F16 noise**: The noise was recorded at the co-pilot's seat in a two-seat F-16, travelling at a speed of 500 knots, and an altitude of 300-600 feet. It was found that the flight condition had only a minor effect on the noise.

These noises have been selected because they are representative of noise conditions that can occur in real world applications.

### 2.6.3 Recognition System Used

In this research work, the optimum seven-state, Bakis model has been used as a recognition system [75]. Vector Quantization has been used to map each continuous observation vector into a discrete codebook index. Once the codebook of vectors has been obtained, the mapping between continuous vectors and codebook indices has been done by nearest neighbor computation.

The recognition rate used is calculated as follows:

\[
\text{Recognition rate} = \frac{\text{No. of successful detection of word}}{\text{No. of words in testing set}}
\]
It is further mentioned that instead of using multicondition training, the acoustic model parameters have been trained using only the clean utterances while testing has been done on noisy as well as clean speech utterances. For the noisy speech recognition, different noises from the NOISEX-92 database are added to the test-set’s clean speech utterances. The noise types considered are white noise, factory noise, babble noise and F-16 aircraft cockpit noise. These noises are added to the clean utterances of only the test set at varying SNRs of 40dB, 20dB, 15dB, 10dB, 5dB and 0dB.