Final Report of the
UGC Sponsored Major Research Project

on

Automatic Speech Recognition (ASR) Over VoIP and Wireless Networks

UGC Sanction Letter: 41-600/2012 (SR) Dated 18th July 2012

by

Prof.P.Laxminarayana

Contributors and Collaborators

M.Ram Reddy, S.Alivelu Mangamma
P.Gangadhar, S.Jagadish, A.V.Ramana and Mythilisharan

Research and Training Unit for Navigational Electronics
OSMANIA UNIVERSITY
HYDERABAD – 500 007
INDIA

Administrative and Financial Report of the
UGC Sponsored Major Research Project

on

Automatic Speech Recognition (ASR) Over VoIP and Wireless Networks

UGC Sanction Letter: 41-600/2012 (SR) Dated 18th July 2012

by

Prof.P.Laxminarayana

Contributors and Collaborators

M.Ram Reddy, S.Alivelu Mangamma
P.Gangadhar, S.Jagadish, A.V.Ramana and Mythilisharan

Research and Training Unit for Navigational Electronics
OSMANIA UNIVERSITY
HYDERABAD – 500 007
INDIA

Final Technical Report of the
UGC Sponsored Major Research Project

on

Automatic Speech Recognition (ASR) Over VoIP and Wireless Networks

UGC Sanction Letter: 41-600/2012 (SR) Dated 18th July 2012

by

Prof.P.Laxminarayana

Contributors and Collaborators

M.Ram Reddy, S.Alivelu Mangamma
P.Gangadhar, S.Jagadish, A.V.Ramana and Mythilisharan

Research and Training Unit for Navigational Electronics
OSMANIA UNIVERSITY
HYDERABAD – 500 007
INDIA

Table of contents

List of figures

List of Tables

ABSTRACT

Increasing processing power, communication bandwidth, and sophisticated Automatic Speech Recognition (ASR) algorithms are making simple to incorporate the applications of ASR in the handheld computers, mobile phones and VoIP devices. Therefore, lot of efforts are put by many people from research and industrial organizations, to make ASR as natural interface between humans and machines for services like enquiring for railway reservation, cost of commodity, and other speech enabled services with Remote Speech Recognition (RSR).

The implementation of client-server based ASR applications using communication networks can be divided to three modes. They are Embedded Speech Recognition (ESR), Distributed Speech Recognition (DSR) and Network Speech Recognition (NSR). In NSR, the user's speech is compressed using conventional speech coders, and is transmitted to the server for ASR. NSR won't require any changes to the existing infrastructure of network or to the mobile phones, except addition of a server for feature extraction and Speech Recognition. However, the speech coding for speech compression and channel noise will degrade the performance of ASR.

The main objective of the UGC sponsored Major Research Project "Automatic Speech Recognition (ASR) over VOIP and Wireless Networks", is to evaluate the performance of ASR over VoIP and Wireless Networks under different channel noise conditions with different speech and channel coding standards.

In this report, results obtained by the evaluation of the performance of the ASR using different channel noise conditions, when FR, EFR, HR and AMRNB speech and channel coding standards are used in the GSM wireless network are reported with critical analysis. A common ASR toolkit SPHINX and a common speech database TIMIT are used for evaluation of Automatic Speech Recognition.

Similarly the performance of ASR over VoIP networks due to pocket loss with different traffic/channel conditions are studied and reported in this report as Part-II.

Automatic Speech Recognition over GSM Networks with Narrowband Speech Under Different Channel Conditions

Technical Report No. NERTU/UGC-MRP/ASR/01

PART-I

M.Ram Reddy, P.Laxminarayana, S.Alivelu Mangamma
P.Gangadhar, and S.Jagadish

Osmania University
Hyderabad – 500 007
INDIA

1 Introduction

1.1 Significance of Automatic Speech Recognition

Speech is the natural and primary mode of communication among human beings. So, we prefer to have the speech as the communication/medium of interaction between computers and human beings. However, at present most of the communication/interaction with computers is done thorough input devices like keyboard or touch screens. Automatic Speech Recognition (ASR) can be used to interact with computers to retrieve the information or for giving the commands to computers by human beings using speech. ASR is the technology that allows a computer to identify the words that a person speaks into a microphone or telephone. Some of the expected applications of ASR in future include voice user interfaces such as voice dialing, control of domestic appliances, searching for a word in speech file, simple data entry and dictation systems. Ordinary illiterate people can also use the computers if, speech is the medium of interaction between computers and human beings. Automatic ASR is useful for the general public, particularly who are not able to type quickly.

1.2 ASR over Digital Communication Channels/ Networks

The increasing processing power, expanding communication bandwidth, and sophisticated ASR algorithms are making the incorporation of speech recognition applications simple in the handheld computers, mobile phones, and VoIP devices. Structurally, ASR system can be segregated into two parts: the front-end and the back-end. The acoustic front-end performs the extraction of features, whereas the back-end performs the pattern recognition operations based on the acoustic and language models. Based on the implementation of ASR modules, there are three modes of client-server based approaches; Embedded Speech Recognition (ESR), Distributed Speech Recognition (DSR) and Network speech Recognition (NSR).

In the Embedded Speech Recognition (ESR) mode, (Client-based), as shown in figure 1.1, both the front-end and back-end operations are implemented in the terminal (for example, mobile phone) itself. Such implementation avoids coding and transmission errors, though it may require high processing power and memory in the device.

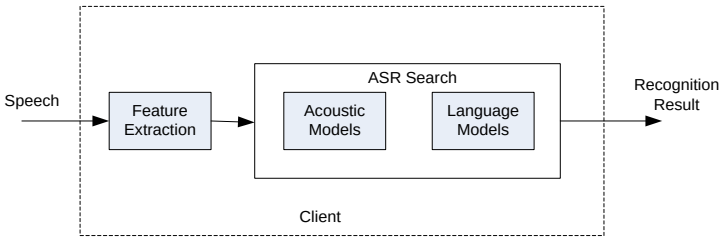

Figure 1.1: Client based ASR (ESR)

In Distributed Speech Recognition (DSR), (Client-Server based), as shown in figure 1.2, the ASR system is distributed between the client and server, where the features are extracted in the client device. The ASR features are compressed and transmitted to the server directly via a dedicated channel. ASR decoder at server takes the received features as input and gives the text as output.

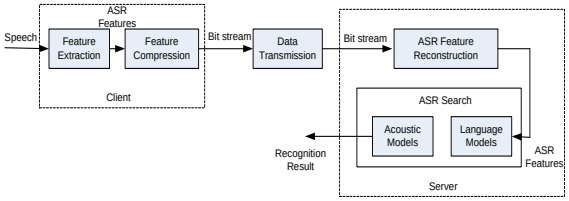

Figure 1.2: Client-Server based ASR (DSR)

In the Network Speech Recognition (NSR) mode as shown in figure 1.3, (Server-based), the user's speech is compressed using conventional speech coders, and is transmitted to the server. Compressed speech received at server will be decoded. ASR features will be extracted from the decoded speech. And then recognition task will be performed.

In bit-stream based NSR (as shown in figure 1.4), the server uses ASR features that are extracted directly from the speech coded parameters like Linear Predictive Coding (LPC), from the bit-stream, which avoids the step of reconstructing the speech from the coded speech parameters. However, separate algorithms are required for conversion of speech coding parameters into speech recognition parameters for ASR.

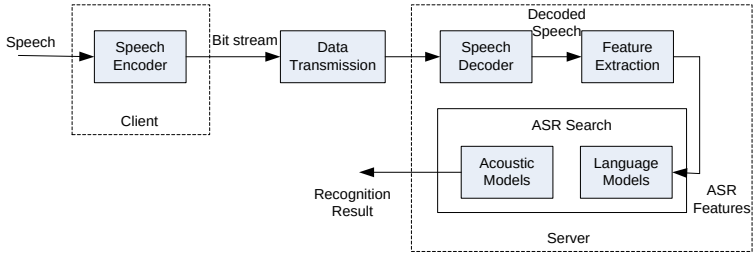

Figure 1.3: Server based ASR (NSR)

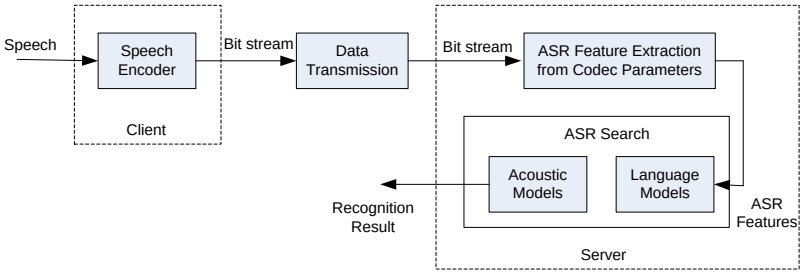

Figure 1.4: Server (Bit-stream) based ASR (NSR)

1.3 Network Speech Recognition (NSR)

There are various advantages and limitations that need to be considered in selecting a proper network configuration, among the above mentioned three or four methods, based on the application. At present NSR is the popular Remote Speech Recognition (RSR) due to the following reasons.

In the client based ESR, all the processing is performed in the client's device, to avoid the coding and transmission loss effects. As a downside, such a process requires huge amount of memory for storing the trained HMMs and enormous processing power for conducting the recognition process in the client's device itself. Moreover, it will be difficult to update all the client's devices, whenever the HMMs or the recognition software is enhanced for improvements, rather than updating one single server database.

In the DSR, the extracted ASR features at client devices will be directly encoded and transmitted through the networks using a different coding mechanism. The extracted features are decoded and used for recognition in the server to avoid the speech coding issues. But, the key barrier for deploying DSR is that it lacks the foundation in the existing mobile devices and requires additional algorithms for encoding and transmitting the features directly.

With recent advances in source coding, channel coding, and error concealment, a vast research work on this approach is aiming for achieving low bit-rate feature extraction techniques (DSR Codecs). A number of DSR standards have also been produced by the Aurora DSR working group in ETSI. Stronger motivation and consistent efforts are needed further to make the DSR is acceptable to all communities, developers and users.

Considering the above mentioned limitations in both ESR and DSR configurations, most of the present deployments prefer to adopt the server based NSR model for recognition process for better utilization of the existing infrastructure. The major advantage of NSR is the fact that numerous commercial applications are developed on the basis of speech coding. Such a development enables the clients to simply connect to a server having an ASR engine and related database or application. So NSR won't require any changes in the existing devices and networks. In addition, maintaining and updating the server will be easy at regular intervals.

1.4 Motivation: Effects of Speech and Channel Coding on RSR

The main difference between ASR and RSR systems is that RSR involves digital communication network placed between the user and recognition engine. The digital communication networks wired or wireless may not be fully reliable due to channel noise and adverse environmental conditions. This new paradigm of RSR involves important consequences for the users, since the users or clients using mobile phones or personal Device Assistants (PDAs) can interact with service. The introduction of mobility in the system (mobile phones and PDAs) implies in turn, that the user may access the service or system in adverse environments, where acoustic noise may severely degrade the system performance.

Similarly, NSR-based speech recognition will face problems such as introducing distortions due to speech transmission at low bit-rate coding, error-prone channels, conversion losses [i.e. LPC (codec) to MFCC (ASR) conversion], and mismatch between training and testing codecs. Therefore it is an interesting to find out the accuracy of NSR due to different channel conditions and using different codecs.

The transmission of speech data over network involves the application of various speech codecs. The main reasons for degradation of speech are speech coding and channel noise. Under the influence of speech coding algorithms, ASR performance degrades significantly. Similarly, there are different types of digital networks: wired and wireless, GSM, CDMA, and VoIP. Etc. There will be pocket loss in the VoIP and bit errors in the GSM or CDMA networks.

The demand for ASR system as an upcoming feature of VoIP and Wireless devices, is increasing day-by-day. So, the study of ASR performance under different environments and network parameters like background noise effects, speech coding, frame erasures/packet losses, delay, jitter effects, echo effects should be considered for designing the system.

1.5 Objectives of the project

In Network Speech Recognition (NSR), the user's speech is compressed using speech coders at client or mobile phone, and transmitted to the server. At the server, compressed speech will be decoded and later features extraction and recognition will be performed. NSR won't require any changes to the existing

infrastructure of network or to the mobile phones, except addition of a server for feature extraction and Speech Recognition. However, the adverse conditions of the channel will degrade the compressed speech and consequently the decoded speech or speech recognition. Channel coding techniques are used to enhance the degraded speech due to noise in the channel. However, many speech coding and channel coding standards are proposed for available bandwidth and channel conditions. Therefore, the effects of channel conditions and channel coding in NSR based RSR is an important study to be carried out.

Full rate (FR), Enhanced Full Rate (EFR) and Half Rate (HR) speech coding standards and corresponding channel coding standards are presently popular in the GSM communication. Therefore there is a need to analyze the performance of ASR due to different channel conditions, channel coding and speech coding standards.

Therefore the main objective of the UGC sponsored Major Research Project "Automatic Speech Recognition (ASR) over VOIP and Wireless Networks" is to study and Analysis of the ASR performance under different speech coding standards and network environment for Network Speech Recognition system for Sources of Degradation in Network Speech Recognition.

1. Sources of Degradation in Network Speech Recognition.
2. Analysis of ASR under noisy environment conditions with different Bit rates
3. Analysis of Degradation of ASR due to Various Speech/Audio Coding Standards at various bit rates.
4. Analysis of Degradation due to Packet loss and Channel Distortion.

The report gives the critical analysis on the results obtained in evaluation of performance of NSR based ASR over GSM networks due to different channel conditions at Full rate, Enhanced Full Rate, Half Rate and Adaptive Multi Rate (AMR) speech and channel coding standards. The performance of ASR over VoIP networks with critical analysis is reported in a separate report.

1.6 Organization of the Report

Next (second) chapter discusses GSM cellular networks. Third chapter explain in brief, the basic elements of digital communication systems, overview of various types of Speech Codecs, and the algorithms involved in the channel coding like Convolution coding, Interleaving, Bursting, Channel Simulation, De-Burst, De-Interleaving, Viterbi algorithm, Channel Decoder. Later fourth chapter gives the details of the implementation of ASR using CMU Sphinx tool and computation of error rate for words and sentences. Fifth chapter discuss the Experiment results and discussions on the performance of ASR over GSM Networks using FR, EFR, HR and AMR Narrow Band (NB) Speech codecs with different channel Conditions and finally sixth chapter gives Conclusion and Future Scope.

2 GSM Communication System

2.1 Digital Communication System

In general Digital communication system will have the following blocks.

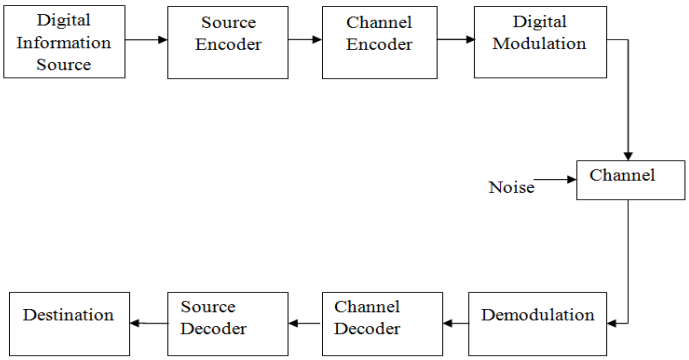

Figure 2.1: Basic Elements of Digital Communication system

Information source:

The source of information is human voice, television picture, teletype data etc. If input is analog signal then it is converted into digital by using sampler and quantizer. So, the source of Information is assumed to be Digital that is symbols, letters...etc.

Source Encoding:

If digital information coming out of source consisting lot of redundancy is transmitted without compression/coding, results in 'Improper utilization of bandwidth', hence results in poor efficiency in communication. The objective of the source encoder is to eliminate or reduce redundancy and compact in digital representation

Channel encoder/decoder:

Channel encoder and decoder are used to reduce the effects of channel noise. Channel coding is the process of adding controlled redundancy to the data to be transmitted, to detect and/or correct the errors caused by the channel noise at the receiver. Addition of redundancy increases bit rate and hence increases bandwidth. Decoder detects the errors in the received data and corrects the error. Example: error correcting codes like linear block codes, cyclic codes and convolution codes. The errors in the received signal are measured in terms of Bit Error Rate (BER).

Modulator/Demodulator:

Modulator converts the bit stream into a waveform with high frequencies suitable for transmission over a communication channel Example, ASK, FSK, PSK, QPSK, GMSK etc. Demodulator converts the waveform into originally transmitted digital data. However, the demodulated digital data signal and transmitted data signal may be different due to channel noise and other channel conditions. Optimum detectors like Viterbi detector are used to minimize the probability of errors.

Communication channel:

It is the media through which the signal can be transmitted. Example: Free space, co-axial cable, wave guide, optical fiber cable etc. The total signal attenuation can be distributed into path loss, shadowing and multi path fading. In this project of multi path fading is considered for channel simulation. Multipath fading in wireless communication systems is commonly modeled by Rayleigh distribution if the line of sight

component absent, if it is present then Racian distribution can be used. In mobile communication most of the cases line of sight path not exist hence Rayleigh fading channel is simulated for adding channel noise.

2.2 GSM Cellular Networks

A typical GSM network is shown in Figure 2.2. The GSM network can be viewed as consisting of three major parts:

- Mobile Switching Center (MSC),
- Base Station Controller (BSC) and
- Base Transceiver Station (BTS)

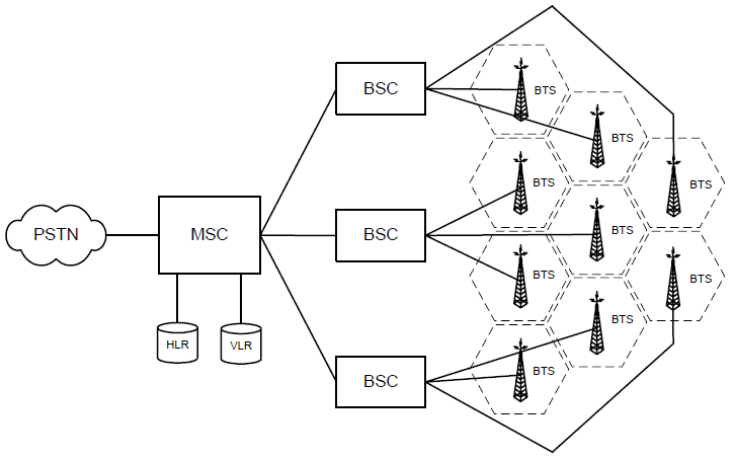

Figure 2.2: GSM Network Architecture

In a typical GSM network, there is a single MSC, a few BSCs and many BTSs. The equipment cost also decreases from MSC to BSC to BTS.

Mobile Switching Center (MSC) – Controls the call set up for incoming and outgoing calls. It is also Interfaces to the PSTN and other mobile networks. Usually there is one MSC in a network or possibly one in each major city. All calls must go through the MSC. The Home and Visitor Location Registers (HLR and VLR) and other back-office subsystems are considered to be part of the MSC.

Base Station Controller (BSC) – The BSC manages the radio resources for one or more BTSs. It handles radio channel setup, frequency hopping, and handovers. The BSC is the connection between the mobile and the MSC. The BSC also translates the full or half rate voice channel used over the radio link to the standard 64 Kbps channel used by the Public Switched Telephone Network (PSDN) or ISDN.

It assigns and releases frequencies and time slots for the Mobile Station. The BSC also handles inter cell handover. It controls the power transmission of the BSS and MS in its area. The function of the BSC is to allocate the necessary time slots between the BTS and the MSC. It is a switching device that handles the radio resources. Additional functions include:

- Control of frequency hopping
- Performing traffic concentration to reduce the number of lines from the MSC
- Providing an interface to the Operations and Maintenance Center for the BSS
- Reallocation of frequencies among BTSs
- Time and frequency synchronization

- Power management
- Time-delay measurements of received signals from the MS

Base-Station Transceiver Station (BTS) – Performs the actual transmission over the air to the mobile subscribers. The BTSs are located at the cellular towers throughout the coverage area. The BTS can contain one or more GSM radios, each of which supports eight GSM voice calls.

GSM Interfaces

All of the interfaces between the various components are carried using standard E1 bearer trunks to allow easier transmission over microwave, fiber or satellite. Figure 2.3 shows a diagram of the different interfaces in a GSM network.

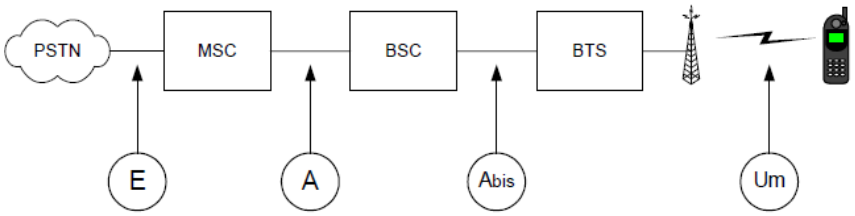

Figure 2.3: GSM Interfaces

The **Um** Interface is the air interface for the GSM mobile telephone standard. It is the interface between the mobile station (MS) and the Base transceiver station (BTS). The GSM network utilizes voice compression for the air (Um) interface between BTS and Mobile Stations (Handsets). Voice compression is performed by the FR, EFR, HR and AMR coding schemes. Channel coding is used in the digital communications to ensure the transmission is received with minimal or no errors. The various channel coding methods can be employed, adding additional binary digits into the transmission to minimize the errors in transmission and reception. When decoded on the receiving end, the transmission can be checked for errors that may have occurred and, in many cases, repaired. There are different channel coding standards based on channel environment. If there is more noise in the channel, more number of bits for channel coding and speech codec with low bit rate will be used.

The following are the narrowband and wideband speech coding standards approved by ETSI/3GPP for wireless mobile applications.

Table 2-1: GSM Speech codecs overview

Coding Standard	**Algorithm**	**Sampling Frequency (kHz)**	**Bit Rates (kbps)**
GSM - FR	RPE-LTP	8	13
GSM - EFR	ACELP	8	12.2
GSM - HR	VSELP	8	5.6
GSM - AMR	(Multi Rate) MR-ACELP	8	4.75, 5.15, 5.90, 6.70, 7.40, 7.95, 10.2, 12.2

GSM – AMR-WB	MRWB-ACELP	16	06.60, 08.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05, 23.85

The **Abis interface** is used to connect a BTS and a BSC. Since there are more BTSs in the network than other components, the Abis interface is the most common interface in a GSM network and is often implemented via satellite. The Abis interface contains compressed voice and GSM information. The BTS is a plain transceiver which receives information from the MS (mobile station) through the Um (air interface) and then converts it to a TDM ("PCM") based interface, the Abis interface, and sends it towards the BSC.

The A Interface is used to connect the BSC and MSC. It contains a maximum of 30 uncompressed voice channels plus a signaling channel for GSM call setup messages for the BSC and Mobile Subscribers. The E1 format is identical to the E interface and the same transmission and compression options apply to both the A and E interfaces.

The **E interface** is used to interface between the MSC and the PSTN, or between MSCs. The E interface is a “standard” E1, which is in common use in the PSTN for carrying telephone and data traffic. The E Interface consists of a 2.048 Mbps carrier with 32 timeslots for 30 voice channels @ 64 kbps each, an SS7 signaling channel and one timeslot for framing and alarms.

The major noise for degradation of speech is due to noise in the wireless channel i.e. mobile device and BTS. Hence the channel coding is essential for correcting the bit errors. The bit rate of compressed speech and the bits added to the compressed speech by the channel coding depends upon the channel noise. i.e. If there is more noise in the channel, speech will be compressed with low bit rate codec and more number of bits will be added for error correction by channel coding.

2.3 GSM Speech Coding:

GSM-FR operates at 8 kHz sampling rates and provides 13 kbps bit-rates. This is the widely used basic codec for cellular applications. GSM – EFR (Enhanced Full Rate) codec operates at 8 kHz sampling rates and provides bit-rates of 12.2 kbps. The coding scheme used is the Algebraic Code Excited Linear Prediction, ACELP. GSM – HR (Half Rate) coder is a 5.6 kbps VSELP (Vector Sum Excited Linear Prediction) coder that operates at 8 kHz sampling rates. This codec doubles the channel capacity when compared to the FR or EFR codecs. An advanced GSM-AMR (Adaptive Multi Rate) codec which supports multi rates 4.75, 5.15, 5.90, 6.70, 7.4, 7.95, 10.2, and 12.2 kbps, is standardized by the European Telecommunications Standards Institute (ETSI) and adopted by third-generation partnership project (3GPP).

In this report, FR, EFR, HR and AMR speech codecs, are considered for evaluation and TIMIT speech database is used for ASR performance evaluation. TIMIT is having, 16-kHz sampling frequency and 16-bits per sample speech files with WAV format. The WAV files are converted into RAW speech files by removing 44 bytes header at the starting of each speech file. As the GSM Speech codecs operates at 8-kHz sampling rates, 16-kHz raw file is down sampled to 8-kHz raw file by using rate conversion standard. Usage of rate conversion executable is given below

./rateconvert in_file.raw out_file.raw 0 1 0 0 2 0 1

Full Rate or FR or GSM-FR speech codec works based on the principles of Regular Pulse Excited - Linear Predictive Coder with a Long Term Predictor Loop (RPE-LPC). An LPC encoder fits a given speech signal against a set of vocal characteristics. The best-fit parameters are transmitted and used by the decoder to generate synthetic speech that is similar to the original. Information from previous samples is used to predict the current sample. The coefficients of the linear combination of the previous samples, plus an

encoded form of the residual, the difference between the predicted and actual sample, represent the signal. Speech is divided into 20 millisecond samples, each of which is encoded as 260 bits, giving a total bit rate of 13 kbps.

Half Rate or HR or GSM-HR codec, operating at 5.6 kbps and uses the Vector Sum Excited Linear Prediction (VSELP) algorithm. It requires half the bandwidth of the Full Rate codec, network capacity for voice traffic is doubled, at the expense of audio quality. It is recommended to use this codec when the battery is low as it may consume up to 30% less energy. The sampling rate is 8 kHz, frame length 160 samples (20 ms) and sub frame length 40 samples (5 ms).

Enhanced Full Rate or EFR or GSM-EFR is a speech coding standard that was developed in order to improve the quite poor quality of GSM-Full Rate (FR) codec. Working at 12.2 kbps the EFR provides wirelike quality in any noise free and background noise conditions. The sampling rate is 8000 sample/s leading to a bit rate for the encoded bit stream of 12.2 kbps. The coding scheme is the so-called Algebraic Code Excited Linear Prediction Coder (ACELP). The encoder is fed with data consisting of samples with a resolution of 13 bits left justified in a 16-bit word. The three least significant bits are set to 0. The decoder outputs data in the same format.

Speech coding standards, GSM-FR, GSM-HR and GSM-EFR which are used for ASR performance evaluations, are available at ETSI website. All the standards are downloaded and compiled to generate executables. The usage of executables is given below

./GSM_codec_ecoder.exe *in_file.raw* *out_file.cod*

2.4 GSM Channel Coding

The idea behind channel coding was developed due to the existence of errors on any given type of communication channel. Radio waves, electrical signals, and even light waves over fiber optic channels will have some amount of noise on the medium, as well as degradation of the signal that occurs over some distance. Being such a common problem in communications, one commonly used method is called automatic repeat request (ARQ), which simply involves the recipient checking the transmission for errors and asking for retransmission. This is referred to as backward error correction. Channel coding, on the other hand, is a forward error correction (FEC) technique. The sender prepares the bits for transmission using a algorithm known as an error-correcting code, which is decoded on the receiving end.

The first channel coding technique was created by a mathematician named Richard Hamming, who developed what's known as the Hamming code. This was the first forward error correction code, which is the inclusion of additional binary digits in the transmission that are called parity bits. The parity bits on the receiving end of the transmission will reveal if any errors have occurred in the transmission, where they are in the string of bits, and how to repair them in order to recover the original transmission.

The Hamming code falls into the family of channel coding methods referred to as block codes, Block codes typically involve the bits being collected into blocks of fixed lengths, which are then referred to as code words. Each code word is given the appropriate checking bits for decoding by the recipient.

Block code methods to increase the size of the transmission due to the added bits in the code word, which can have an effect on the channel's bandwidth.

Another channel coding method is known as a convolutional code. These methods are much faster and can encode a bit stream of any length. One commonly used code of this type is called the Viterbi code.

The drawback to this method is that as the length of the convolutional code increases, so does its complexity when decoding. In many cases, convolutional codes are used in combination with block codes in what's known as concatenated error correction codes.

2.4.1 Forward Error Correction

Forward error correction is a method of data transmission that allows receivers to detect and repair many kinds of errors in the information automatically. The process does not require communication with the transmitter. Instead, receivers independently manage errors, when possible. In situations where data becomes hopelessly corrupted, it may be necessary to request a retransmission to get a clean copy to use.

The process starts at the transmitter, which adds some extra bits to the message. The nature of the redundant data can vary, depending on the approach used to add data; some options include algebraic coding, the Viterbi decoding algorithm, and convolution coding. These create a pattern that the receiver can recognize and use to check the rest of the data.

If the transmission is clean, the check will show that there are no errors, and the receiver can deliver the data to the user. In the event there is a problem, the receiver uses forward error correction to compare the known redundant data against the apparently corrupted information and uses this analysis to fix the corrupted data and generate an output for the user. If the receiver cannot correct the error, it may indicate that the data is too corrupt, or it could include blank spots where it was not possible to restore the information.

2.4.2 Error Control Coding

Error control coding is a method to detect and possibly correct errors by introducing redundancy to the stream of bits to be sent to the channel. The Channel Encoder will add bits to the message bits to be transmitted systematically. After passing through the channel, the Channel decoder will detect and correct the errors. A simple example is to send ‘000’ (‘111’ correspondingly) instead of sending only one ‘0’ (‘1’ correspondingly) to the channel. Due to noise in the channel, the received bits may become ‘001’. However, since either ‘000’ or ‘111’ could have been sent. By majority logic decoding scheme, it will be decoded as ‘000’ and therefore the message has been a ‘0’.

In general the channel encoder will divides the input message bits into blocks of k messages bits and replaces each k message bits block with a n-bit code word by introducing (n-k) check bits to each message block. Some major codes include the Block Codes and Convolution Codes.

Once we have a compressed digital speech signal, we must add a number of bits for error control to protect the signal from interference. These bits are called redundancy bits. The GSM system uses convolution encoding to achieve this protection. The exact algorithms used differ for speech and for different data rates.

2.4.3 Convolution coding

Convolutional codes commonly specified by three parameters n, k, m

Where n = Number of Output bits

k = Number of Input bits

m = Number of memory registers

The quantity k/n called the code rate is a measure of the efficiency of the code. Often the manufacturers of convolutional chips specify the code by parameters n, k, L. The quantity L is called the constraint length of the code and is defined by

Constraint length L = k (m-1)

The constraint length L represents the number of bits in the encoder memory that affect the generation of the n output bits. Often the constraint length L is also referred to by the letter K. The structure of the convolutional encoder is shown in the Figure 2.4. Here k input bits are coded into n output bits. The constraint length of the code is 2. The output bits are produced by madulo-2 adders by adding present and past bits from the registers. The selection bits to be added to produce output is depends on the generator polynomial. Consider the Figure 2.4 as an example of 1/3 rate encoder, the first output bit has a generator polynomial of (1, 1, 1). The Second output bit has a generator polynomial (0, 1, 1) and third output bit has a polynomial of (1, 0, 1). The output bits are given by the equations

V_1=mod2 (u_1+u_0+u_{-1})

V_2=mod2 u_0+u_{-1})

V_3=mod2 (u_1+u_{-1})

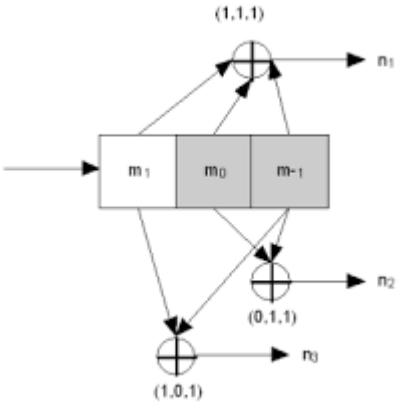

Figure 2.4: convolutional encoder (3, 1, 3)

A special form of convolutional code in which the output bits contain an easily recognizable sequence of the input bits is called the systematic form. This means, input bits are directly embedded into the coded sequence. The structure of Systematic convolutional code is same as Figure 2.4, but the one of the output is directly fed from the input. Systematic codes are often preferred over non systematic codes because they allow quick look. They also require less hardware for encoding.

The recursive systematic convolutional (RSC) encoder is obtained from the non-recursive non-systematic (conventional) convolutional encoder by feeding back one of its encoded outputs to its input. Figure 2.5 shows a recursive systematic convolutional encoder.

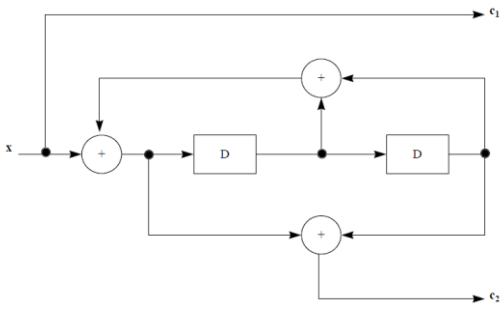

Figure 2.5: Recursive convolutional encoder

GSM FR, EFR and HR channel coding standards uses the simple conventional convolutional codes, where as the adaptive multi rate channel coding standard uses the recursive systematic convolutional codes in channel encoding.

2.5 Speech channel at full rate (TCH/FS and TCH/EFS)

The speech coder (whether Full rate or Enhanced full rate) delivers to the channel encoder a sequence of blocks of data. In case of a full rate and enhanced full rate speech TCH, one block of data corresponds to one speech frame.

Full rate coder each block contains 260 information bits, including 182 bits of class 1 (protected bits), and 78 bits of class 2 (no protection). The bits delivered by the speech coder are received in the order indicated in **[3GPP TS 46.010]** and have to be rearranged according to **[table 2 in 3GPP TS 45.003 version 13.0.0 Release 13]** before channel coding. The rearranged bits are labeled {d(0),d(1),...,d(259)}, defined in the order of decreasing importance. In practical FR codec will take 160 samples per frame (i.e. 320 bytes of data) and compresses it into 264 bits of data. However the present GSMFRENC.exe will give 264 bits per frame for making integer number of bytes. However channel coding will require only 260 bits. Basically to make 264 bits (33bytes) from 260 bits (32.5bytes), 4 bits were added at the starting. Therefore, the compressed file is having 264 bits per frame need to be converted to 260 bits per frame by removing 4-bits at the starting of every frame.

EFR coder each block contains 244 information bits. The block of 244 information bits, labeled {s(1).., s(244)}, passes through a preliminary stage, applied only to EFR which produces 260 bits corresponding to the 244 input bits and 16 redundancy bits. Those 16 redundancy bits correspond to 8 CRC bits and 8 repetition bits. The 260 bits, labeled {w(1)..w(260)}, have to be rearranged according to **[table 6 in 3GPP TS 45.003 version 13.0.0 Release 13]** before they are delivered to the channel encoding unit which is identical to that of the FR.

2.5.1 Preliminary channel coding only for GSM-EFR:

CRC generation

An 8-bit CRC is used for error-detection. These 8 parity bits (bits w253-w260) are generated by the cyclic generator polynomial: g(D) = D8 + D4 + D3 + D2 + 1 from the 65 most important bits (50 bits of class 1a and 15 bits of class 1b). These 65 bits {b(1)-b(65)} are taken from the **[table 5 in 3GPP TS 45.003 version 13.0.0 Release 13]** in the following order (read row by row, left to right):

Table 2-2: 65 Most Significant bits for 8-bit CRC generation for Error detection in GSM - EFR

s39	s40	s41	s42	s43	s44	s48	s87	s45	s2
s3	s8	s10	s18	s19	s24	s46	s47	s142	s143
s144	s145	s146	s147	s92	s93	s195	s196	s98	s137

s148	s94	s197	s149	s150	s95	s198	s4	s5	s11
s12	s16	s9	s6	s7	s13	s17	s20	s96	s199
s1	s14	s15	s21	s25	s26	s28	s151	s201	s190
s240	s88	s138	s191	s241					

The encoding is performed in a systematic form, which means that, in GF(2), the polynomial:

- b(1)D72 + b(2)D71 +...+b(65)D8 + p(1)D7 + p(2)D6 +...+ p(7)D1 + p(8)

- p(1) - p(8): the parity bits (w253-w260)

- b(1) - b(65) = the data bits from the table above

when divided by g(D), yields a remainder equal to 0.

Repetition of bits

The repeated bits are s70, s120, s173 and s223. They correspond to one of the bits in each of the PULSE_5, the most significant one not protected by the channel coding stage.

Generating output

The preliminary coded bits w(k) for k = 1 to 260 are hence defined by:

w(k) = s(k) for k = 1 to 71

w(k) = s(k-2) for k = 74 to 123

w(k) = s(k-4) for k = 126 to 178

w(k) = s(k-6) for k = 181 to s230

w(k) = s(k-8) for k = 233 to s252

Repetition bits:

w(k) = s(70) for k = 72 and 73

w(k) = s(120) for k = 124 and 125

w(k) = s(173) for k = 179 and 180

w(k) = s(223) for k = 231 and 232

Parity bits:

w(k = p(k-252) for k = 253 to 260

2.6 Channel coding for FR and EFR

The 260 bits block includes 182 bits of class 1(protected bits) and 78 bits of class 2 (no protection). The class 1 bits are further divided into the class 1a and class 1b, class 1a bits being protected by a cyclic code and the convolutional code whereas the class 1b bits are protected by the convolutional code only. Classification of bits is shown clearly in figure 2.7.

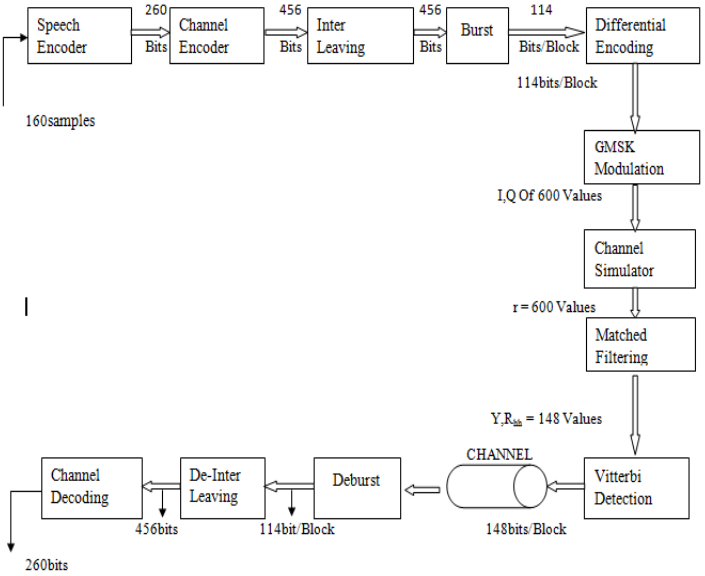

Figure 2.6: Block of elements in channel coding

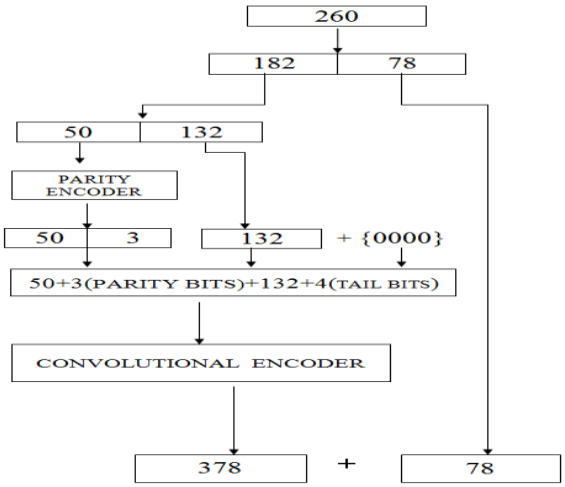

Figure 2.7: Classifications of bits in channel encoding of FR and EFR

2.6.1 Parity and tailing for a speech frame

Parity bits:

The first 50 bits of class 1 (known as class 1a for the EFR) are protected by three parity bits used for error detection. These parity bits are added to the 50 bits, according to a degenerate (shortened) cyclic code (53,50,2), using the generator polynomial: $g(D) = D^3 + D + 1$

Cyclic codes are linear codes, in addition to being linear, a cyclic shift, or rotate, of a codeword produces another codeword since the code used in GSM is a (53, 50) code, the generator polynomial used in the encoding is of degree 53-50 = 3. GSM chose to use cyclic encoding due to the ability to quickly determine if errors are present. The three redundancy bits produced by the cyclic encoder enable the receiver to quickly determine if an error was produced. If an error was produced the current 53 bit frame is discarded and replaced by the last known "good" frame.

The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

d(0)D52 + d(1)D51 +... + d(49)D3 + p(0)D2 + p(1)D+ p(2)

where p(0), p(1), p(2) are the parity bits,

when divided by g(D), yields a remainder equal to: 1 + D + D2

Tailing bits and reordering:

The information and parity bits of class 1 are reordered, defining 189 information + parity + tail bits of class 1, {u(0),u(1),...,u(188)} defined by:

u(k) = d(2k) and u(184-k) = d(2k+1) for k = 0,1,...,90

u(91+k) = p(k) for k = 0,1,2

u(k) = 0 for k = 185,186,187,188 (tail bits)

2.6.2 Convolutional encoder

The resulting 53 bits of the cyclic encoder are added to the 132 Class Ib bits (plus a tail of 4 extra bits) and encoded using the convolution encoder. The convolution encoder adds one redundancy bit for every bit that it sees based on the last four bits in the sequence. These four bits are added together using a modulo-2 adder. As a result, the convolution encoder encodes one input bit into two output bits. In GSM there are 4 flip-flops, and the convolution performed is of $D^4 + D^3 + 1$ and $D^4 + D^3 + D + 1$. GSM chose to employ a convolution encoder due to its ability to efficiently correct errors. In order to correct errors, GSM employs the use of Trellis Diagrams.

Once the convolution encoder has encoded the bits, a new bit sequence of 378 (2x(53+132+4=189) =378) bits is produced. These 378 bits are directly added to the 78 Class II bits (directly added since these bits are least sensitive to error). As a result, the channel encoded bit sequence is now 378+78=456 bits long. Therefore, each 20 ms burst produces 456 bits at a bit rate of 22.8 kbps. To further protect against bit errors, the 456 bit sequence is then diagonally interleaved.

The class 1 bits are encoded with the ½ rate convolutional code defined by the polynomials:

G0 = 1 + D3+ D4

G1 = 1 + D + D3+ D4

The coded bits {c(0), c(1),..., c(455)} are then defined by:

-class 1: c(2k) = u(k) + u(k-3) + u(k-4)

c(2k+1) = u(k) + u(k-1) + u(k-3) + u(k-4) for k = 0,1,...,188

u(k) = 0 for k < 0

-class 2: c(378+k) = d(182+k) for k = 0,1,....,77

2.6.3 Interleaving

Now, one problem remains all of this error detection and error correction coding will not do any good if the entire 456-bit block is lost. In order to reduce this, the bits are reordered and partitioned onto eight separate sub-blocks. If one sub-block is lost then only one-eighth of the data for each audio block is lost and those bits can be recovered using the convolution code on the receiving end. This is known as interleaving

The coded bits are reordered and interleaved according to the following rule:

i(B, j) = c(n, k), for k = 0,1,...,455

n = 0,1,...,N,N+1,...

$B = B_0 + 4n + (k \bmod 8)$

$j = 2((49k) \bmod 57) + ((k \bmod 8) \operatorname{div} 4)$

As per the standard given **[table 1 in 3GPP TS 45.003 version 13.0.0 Release 13],** the result of the interleaving is a distribution of the reordered 456 bits of a given data block, n = N, over 8 blocks using the even numbered bits of the first 4 blocks ($B = B_0 + 4N + 0, 1, 2, 3$) and odd numbered bits of the last 4 blocks ($B = B_0 + 4N + 4, 5, 6, 7$). The reordered bits of the following data block, n = N+1, use the even numbered bits of the blocks $B = B_0 + 4N + 4, 5, 6, 7$ ($B = B_0 + 4(N+1) + 0, 1, 2, 3$) and the odd numbered bits of the blocks $B = B_0 + 4(N+1) + 4, 5, 6, 7$. Continuing with the next data blocks shows that one block always carries 57 bits of data from one data block (n = N) and 57 bits of data from the next block (n = N+1), where the bits from the data block with the higher number always are the even numbered data bits, and those of the data block with the lower number are the odd numbered bits.

The block of coded data is interleaved "block diagonal", where a new data block starts every 4th block and is distributed over 8 blocks.

2.6.4 Mapping on a GSM Burst

To further protect against burst errors common to the radio interface, each sample is interleaved. The 456 bits output by the convolution encoder are divided into 8 blocks of 57 bits, and these blocks are transmitted in eight consecutive time slot bursts. Each 456-bit block is reordered and partitioned into 8 sub-blocks of 57 bits each. These eight 57-bit sub-blocks are then interleaved onto 8 separate bursts. The first four sub-blocks (0 through 3) are mapped onto the even bits of four consecutive bursts. The last four sub-blocks (4 through 7) are mapped onto the odd bits of the next 4 consecutive bursts. So, the entire block is spread out across 8 separate bursts.

2.7 Speech channel at Half Rate

A sequence of blocks of data is delivered by the speech coder to the channel encoder. In case of a half rate speech TCH, one block of data corresponds to one speech frame. Each block contains 112 bits, including 95 bits of class 1 (protected bits), and 17 bits of class 2 (no protection), as per **[tables 3a and 3b in 3GPP TS 45.003 version 13.0.0 Release 13]**.

The bits delivered by the speech coder are received in the order indicated in **[3GPP TS 46.020]** and have to be arranged according to either **[tables 3a or 3b in 3GPP TS 45.003 version 13.0.0 Release 13]** before channel encoding. The rearranged bits are labeled {d(0), d(1),...,d(111)}. Table 3a has to be taken if parameter Mode = 0 (which means that the speech encoder is in unvoiced mode), while table 3b has to be taken if parameter Mode = 1, 2 or 3 (which means that the speech encoder is in voiced mode).

2.8 Channel coding for HR

2.8.1 Parity and tailing for a speech frame

Parity bits:

The most significant 22 class 1 bits d(73), d(74),...,d(94) are protected by three parity bits used for error detection. These bits are added to the 22 bits, according to a cyclic code using the generator polynomial:

$$g(D) = D3 + D + 1$$

The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

d(73)D24 + d(74)D23 + ... + d(94)D3 + p(0)D2 + p(1)D + p(2)

where p(0), p(1), p(2) are the parity bits,

when divided by g(D), yields a remainder equal to: 1 + D + D2.

Tail bits and reordering:

The information and parity bits of class 1 are reordered, defining 104 information + parity + tail bits of class 1,

{u(0),u(1),...,u(103)} defined by:

u(k) = d(k) for k = 0,1,...,94

u(k) = p(k-95) for k = 95,96,97

u(k) = 0 for k = 98,99,...,103 (tail bits)

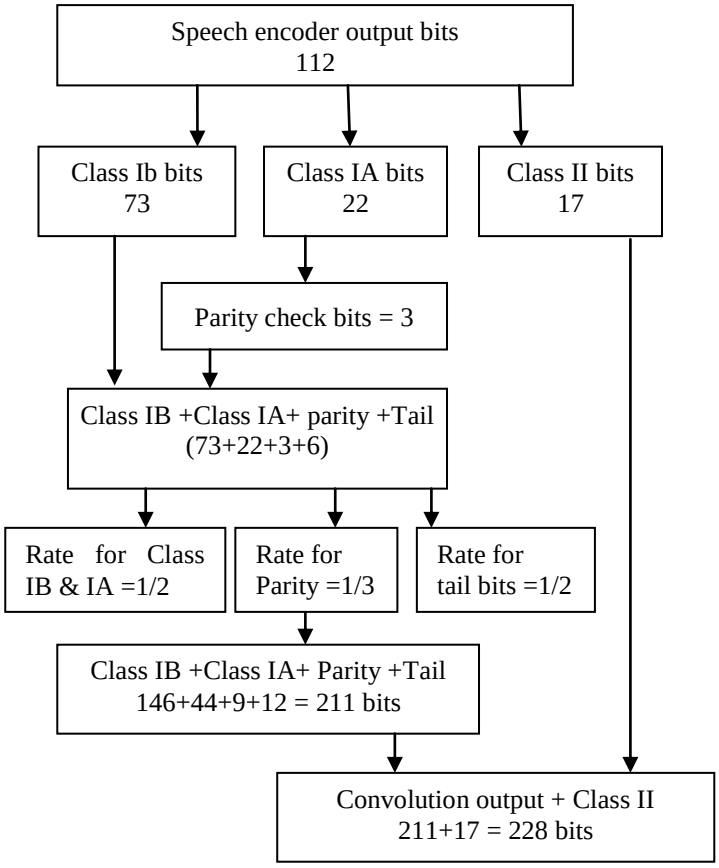

Figure 2.8: Bits classification for channel coding in HR

2.8.2 Convolutional enc

The class 1 bits are encoded with the punctured convolutional code defined by the mother polynomials:

G4 = 1 + D2 + D3 + D5 + D6

G5 = 1 + D + D4 + D6

G6 = 1 + D + D2 + D3 + D4 + D6

The coded bits {c(0), c(1),..., c(227)} are then defined by:

class 1 information bits:

$c(2k) = u(k)+u(k-2)+u(k-3)+ (k-5)+u(k-6)$

$c(2k+1) = u(k)+u(k-1)+u(k-2)+u(k-3)+u(k-4)+u(k-6)$ for $k = 0,1,...,94$; $u(k) = 0$ for $k<0$

parity bits:

$c(3k-95) = u(k)+u(k-2)+u(k-3)+u(k-5)+u(k-6)$

$c(3k-94) = u(k)+u(k-1)+u(k-4)+u(k-6)$

$c(3k-93) = u(k)+u(k-1)+u(k-2)+u(k-3)+u(k-4)+u(k-6)$ for $k = 95,96,97$

tail bits:

$c(2k+3) = u(k)+u(k-2)+u(k-3)+u(k-5)+u(k-6)$

$c(2k+4) = u(k)+u(k-1)+u(k-2)+u(k-3)+u(k-4)+u(k-6)$ for $k = 98,99,...,103$

class 2 information bits:

$c(k+211) = d(k+95)$ for $k = 0,1,...,16$

2.8.3 Interleaving

The coded bits are reordered and interleaved according to the following rule and is given the table 4 of **[ETSI TS 145 003 V13.2.0 (2016-08)].**

$i(B, j) = c(n, k)$ for $k = 0,1,...,227$

$n = 0, 1,..., N, N+1, ...$

$B = B_0 + 2n + b$

The values of b and j in dependence of k are given by table 4.

The result of the interleaving is a distribution of the reordered 228 bits of a given data block, n = N, over 4 blocks using the even numbered bits of the first 2 blocks ($B = B_0+2N+0,1$) and the odd numbered bits of the last 2 blocks ($B = B_0+2N+2,3$). The reordered bits of the following data block, n = N + 1, use the even numbered bits of the blocks B = B0 + 2N + 2,3 (B = B0+2(N+1)+0,1) and the odd numbered bits of the blocks B = B0 + 2(N+1) + 2,3. Continuing with the next data blocks shows that one block always carries 57 bits of data from one data block (n = N) and 57 bits from the next block (n = N+1), where the bits from the data block with the higher number always are the even numbered data bits, and those of the data block with the lower number are the odd numbered bits. The block of coded data is interleaved "block diagonal", where a new data block starts every 2nd block and is distributed over 4 blocks.

2.8.4 Mapping on a burst

The mapping is given by the rule:

$e(B, j) = i(B, j)$ and $e(B,59+j) = i(B,57+j)$ for $j = 0,1,...,56$

$e(B,57) = hl(B)$ and $e(B,58) = hu(B)$

The two bits, labeled hl(B) and hu(B) on burst number B are flags used for indication of control channel signaling. For each TCH/HS block not stolen for signaling purposes:

hu(B) = 0 for the first 2 bursts (indicating status of the even numbered bits)

hl(B) = 0 for the last 2 bursts (indicating status of the odd numbered bits)

For the use of hl(B) and hu(B) when a speech frame is stolen for signaling purposes.

2.9 Adaptive Multi Rate (AMR)

2.9.1 Coding of the in-band data

The two input in-band bits (id(0,1)) are coded to eight coded in-band bits (ic(0..7)). The encoded in-band bits are moved to the coded bits, c, as

c(k) = ic(k) for k = 0, 1, ..., 7.

2.9.2 Ordering according to subjective importance

The bits delivered by the speech encoder, {s(1),s(2),...,s(Ks)}, are rearranged according to subjective importance before channel coding. Tables 7 to 16 define the correct rearrangement for the speech codec modes 12.2 kbit/s, 10.2 kbit/s, 7.95 kbit/s, 7.40 kbit/s, 6.70 kbit/s, 5.90 kbit/s, 5.15 kbit/s and 4.75 kbit/s, respectively. In the tables speech codec parameters are numbered in the order they are delivered by the corresponding speech encoder according to 3GPP TS 26.090 and the rearranged bits are labelled {d(0),d(1),...,d(K_d-1)}, defined in the order of decreasing importance. Index K_d refers to the number of bits delivered by the speech encoder, see below:

Table 2-3: GSM AMR - NB Speech codecs

AMR Speech codec rates	No. of Output bits
TCH/AFS 12.2	244
TCH/AFS 10.2	204
TCH/AFS 7.95	159
TCH/AFS 7.4	148
TCH/AFS 6.7	134
TCH/AFS 5.9	118
TCH/AFS 5.15	103
TCH/AFS 4.75	95

The rearranged bits are further divided into two different classes to perform unequal error protection for different bits according to subjective importance.

The protection classes are:

1a - Data protected with the CRC and the convolution code.

1b - Data protected with the convolution code.

No unprotected bits are used.

The number of class 1 (sum of class 1a and 1b), class 1a and class 1b bits for each codec mode is shown in Table 2-4:

Table 2-4: Number of Bits in Each class of AMR NB Speech codecs

Codec mode	No. of speech bits from speech	No. of class 1 bits	No. of class 1a bits	No. of class 1b bits	CRC protected bits	No. of bits after first encoding $K_u=K_{d1a}+6+K_{d1}$

	codec	(K_d)	(K_{d1a})	(K_{d1b})	(class 1a) bits	b
TCH/AFS 12.2	244	244	81	163	81	250
TCH/AFS 10.2	204	204	65	139	65	210
TCH/AFS 7.95	159	159	75	84	75	165
TCH/AFS 7.4	148	148	61	87	61	154
TCH/AFS 6.7	134	134	55	79	55	140
TCH/AFS 5.9	118	118	55	63	55	124
TCH/AFS 5.15	103	103	49	54	49	109
TCH/AFS 4.75	95	95	39	56	39	101

2.9.3 Parity and tailing for a speech frame

Parity bits:

A 6-bit CRC is used for error-detection. These 6 parity bits are generated by the cyclic generator polynomial: g(D) = D6 + D5 + D3 + D2 + D1 + 1 from the first Kd1a bits of class 1, where Kd1a refers to number of bits in protection class 1a as shown above for each codec mode. The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

d(0)D(Kd1a+5) + d(1)D(Kd1a+4) +... + d(Kd1a-1)D(6) + p(0)D(5) +…+ p(4)D+ p(5)

where p(0), p(1) … p(5) are the parity bits, when divided by g(D), yields a remainder equal to:

1+ D + D2 + D3 + D4 + D5.

The information and parity bits are merged:

$u(k) = d(k)$ for $k = 0, 1, \ldots, K_{d1a}-1$

$u(k) = p(k-K_{d1a})$ for $k = K_{d1a}, K_{d1a}+1, \ldots, K_{d1a}+5$

$u(k) = d(k-6)$ for $k = K_{d1a}+6, K_{d1a}+7, \ldots, K_u-1$

Thus, after the first encoding step u(k) will be defined by the following contents for each codec mode:

TCH/AFS12.2:

$u(k) = d(k)$ for $k = 0, 1, \ldots, 80$

$u(k) = p(k-81)$ for $k = 81, 82, \ldots, 86$

$u(k) = d(k-6)$ for $k = 87, 88, \ldots, 249$

TCH/AFS10.2:

$u(k) = d(k)$ for $k = 0, 1, ..., 64$

$u(k) = p(k-65)$ for $k = 65, 66, ..., 70$

$u(k) = d(k-6)$ for $k = 71, 72, ..., 209$

TCH/AFS7.95:

$u(k) = d(k)$ for $k = 0, 1, \ldots, 74$

$u(k) = p(k-75)$ for $k = 75, 76, \ldots, 80$

$u(k) = d(k-6)$ for $k = 81, 82, \ldots, 164$

TCH/AFS7.4:

u(k) = d(k) for k = 0, 1, …, 60

u(k) = p(k-61) for k = 61, 62, …, 66

u(k) = d(k-6) for k = 67, 68, …, 153

TCH/AFS6.7:

u(k) = d(k) for k = 0, 1, …, 54

u(k) = p(k-55) for k = 55, 56, …, 60

u(k) = d(k-6) for k = 61, 62, …, 139

TCH/AFS5.9:

u(k) = d(k) for k = 0, 1, …, 54

u(k) = p(k-55) for k = 55, 56, …, 60

u(k) = d(k-6) for k = 61, 62, …, 123

TCH/AFS5.15:

u(k) = d(k) for k = 0, 1, …, 48

u(k) = p(k-49) for k = 49, 50, …, 54

u(k) = d(k-6) for k = 55, 56, …, 108

TCH/AFS4.75:

u(k) = d(k) for k = 0, 1, ..., 38

u(k) = p(k-39) for k = 39, 40, ..., 44

u(k) = d(k-6) for k = 45, 46, ..., 100

2.9.4 Convolutional encoder

The bits from the first encoding step (u(k)) are encoded with the recursive systematic convolutional codes as summarized below. The number of output bits after puncturing is 448 for all codec modes.

Table 2-5: Standards of convolution encoding in GSM AMR - NB Channel coding

Codec mode	Rate	No. of input bits for conv. encoder	No. of output bits from conv. encoder	No. of Punctured bits
TCH/AFS 12.2	½	250	508	60
TCH/AFS 10.2	1/3	210	642	194
TCH/AFS 7.95	1/3	165	513	65
TCH/AFS 7.4	1/3	154	474	26
TCH/AFS 6.7	¼	140	576	128
TCH/AFS 5.9	¼	124	520	72
TCH/AFS 5.15	1/5	109	565	117
TCH/AFS 4.75	1/5	101	535	87

Below the coding for each codec mode is specified in detail.

TCH/AFS12.2:

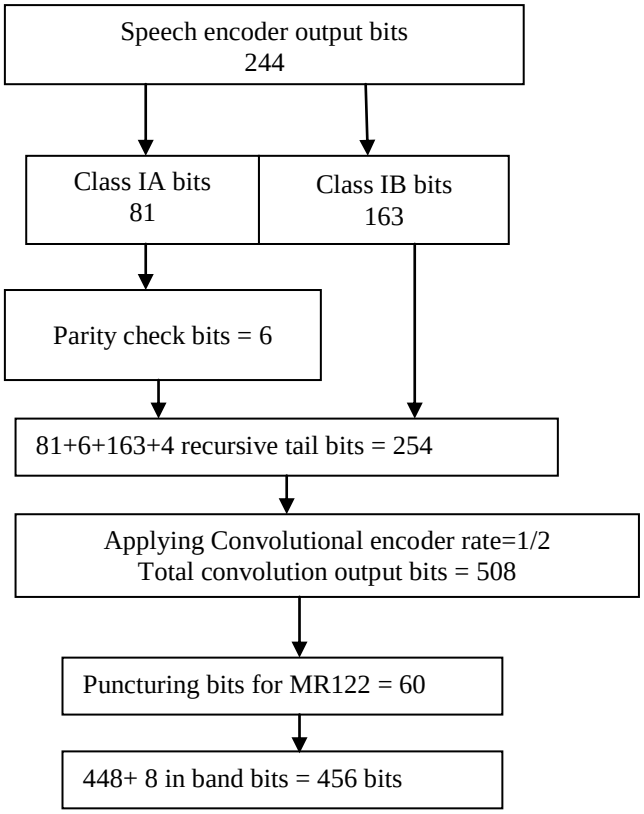

Figure 2.9: Bits classification for channel coding in GSM AMR-NB with mode MR122

The block of 250 bits {u(0)… u(249)} is encoded with the ½ rate convolutional code defined by the following polynomials:

G0/G0 = 1

G1/G0 = 1 + D + D3+ D4 / 1 + D3 + D4

resulting in 508 coded bits, {C(0)… C(507)} defined by:

r(k) = u(k) + r(k-3) + r(k-4)

C(2k) = u(k)

C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4) for k = 0, 1, ..., 249;

r(k) = 0 for k<0 and (for termination of the coder):

r(k) = 0

C(2k) = r(k-3) + r(k-4)

C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4) for k = 250, 251, ..., 253

The code is punctured in such a way that the following 60 coded bits:

C(321), C(325), C(329), C(333), C(337), C(341), C(345), C(349), C(353), C(357), C(361), C(363), C(365), C(369), C(373), C(377), C(379), C(381), C(385), C(389), C(393), C(395), C(397), C(401), C(405), C(409), C(411), C(413), C(417), C(421), C(425), C(427), C(429), C(433), C(437), C(441), C(443), C(445), C(449), C(453), C(457), C(459), C(461), C(465), C(469), C(473), C(475), C(477), C(481), C(485), C(489), C(491), C(493), C(495), C(497), C(499), C(501), C(503), C(505) and C(507) are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as

c(k+8) = P(k) for k = 0, 1, ..., 447.

TCH/AFS10.2:

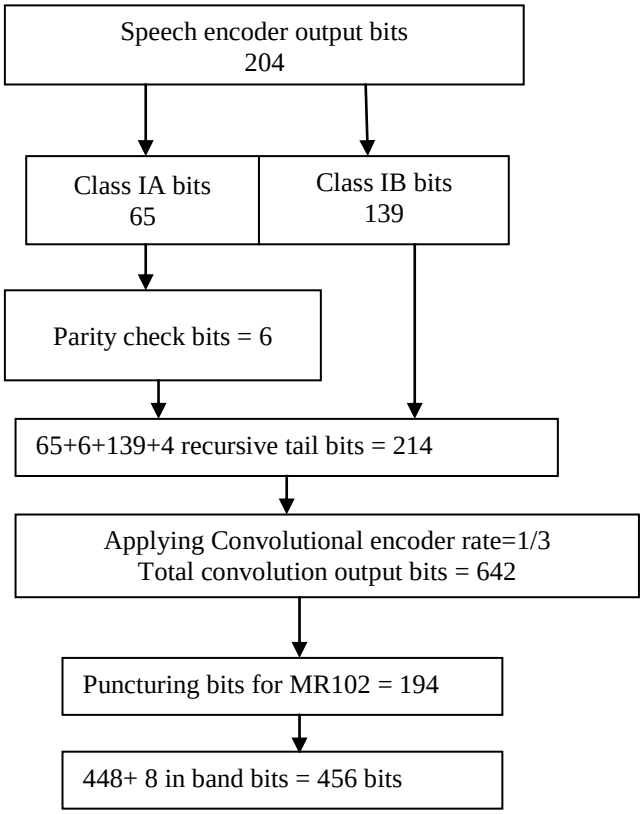

Figure 2.10: Bits classification for channel coding in GSM AMR-NB with mode MR102

The block of 210 bits {u(0)... u(209)} is encoded with the 1/3 rate convolutional code defined by the following polynomials:

G1/G3 = 1 + D + D3 + D4 / 1 + D + D2 + D3 + D4

G2/G3 = 1 + D2 + D4 / 1 + D + D2 + D3 + D4

G3/G3 = 1

resulting in 642 coded bits, {C(0)... C(641)} defined by:

r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)

C(3k) = r(k) + r(k-1) + r(k-3) + r(k-4)

C(3k+1) = r(k)+r(k-2)+r(k-4)

C(3k+2) = u(k) for k = 0, 1, ..., 209 and (for termination of the coder):

r(k) = 0

C(3k) = r(k)+r(k-1) + r(k-3) + r(k-4)

C(3k+1) = r(k)+r(k-2)+r(k-4)

C(3k+2) = r(k-1)+r(k-2)+r(k-3)+r(k-4) for k = 210, 211, ..., 213

The code is punctured in such a way that the following 194 bits:

C(1), C(4), C(7), C(10), C(16), C(19), C(22), C(28), C(31), C(34), C(40), C(43), C(46), C(52), C(55), C(58), C(64), C(67), C(70), C(76), C(79), C(82), C(88), C(91), C(94), C(100), C(103), C(106), C(112),

C(115), C(118), C(124), C(127), C(130), C(136), C(139), C(142), C(148), C(151), C(154), C(160), C(163), C(166), C(172), C(175), C(178), C(184), C(187), C(190), C(196), C(199), C(202), C(208), C(211), C(214), C(220), C(223), C(226), C(232), C(235), C(238), C(244), C(247), C(250), C(256), C(259), C(262), C(268), C(271), C(274), C(280), C(283), C(286), C(292), C(295), C(298), C(304), C(307), C(310), C(316), C(319), C(322), C(325), C(328), C(331), C(334), C(337), C(340), C(343), C(346), C(349), C(352), C(355), C(358), C(361), C(364), C(367), C(370), C(373), C(376), C(379), C(382), C(385), C(388), C(391), C(394), C(397), C(400), C(403), C(406), C(409), C(412), C(415), C(418), C(421), C(424), C(427), C(430), C(433), C(436), C(439), C(442), C(445), C(448), C(451), C(454), C(457), C(460), C(463), C(466), C(469), C(472), C(475), C(478), C(481), C(484), C(487), C(490), C(493), C(496), C(499), C(502), C(505), C(508), C(511), C(514), C(517), C(520), C(523), C(526), C(529), C(532), C(535), C(538), C(541), C(544), C(547), C(550), C(553), C(556), C(559), C(562), C(565), C(568), C(571), C(574), C(577), C(580), C(583), C(586), C(589), C(592), C(595), C(598), C(601), C(604), C(607), C(609), C(610), C(613), C(616), C(619), C(621), C(622), C(625), C(627), C(628), C(631), C(633), C(634), C(636), C(637), C(639) and C(640) are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as:

c(k+8) = P(k) for k = 0, 1, ..., 447.

TCH/AFS7.95:

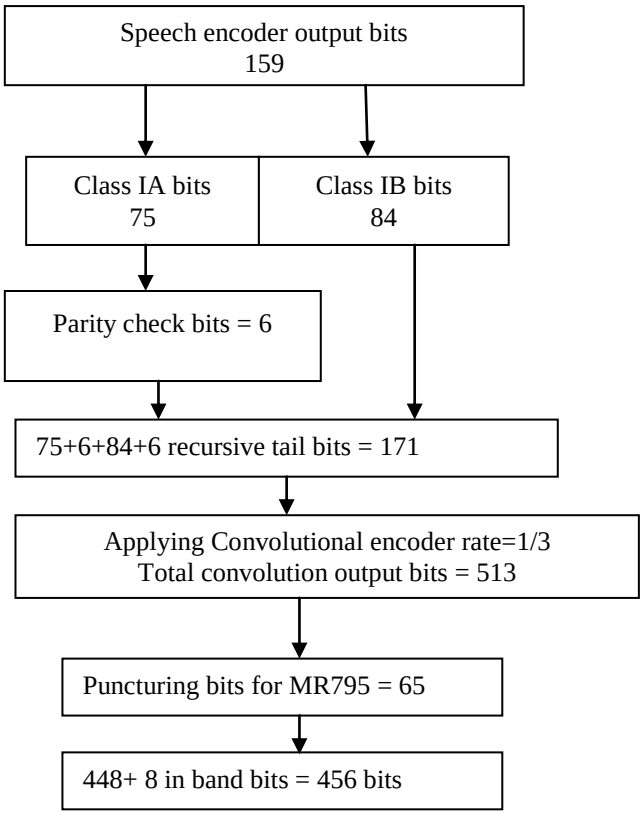

Figure 2.11: Bits classification for channel coding in GSM AMR-NB with mode MR795

The block of 165 bits {u(0)… u(164)} is encoded with the 1/3 rate convolutional code defined by the following polynomials:

G4/G4 =1

G5/G4 = 1 + D + D4 + D6/ 1 + D2 + D3 + D5 + D6

G6/G4 = 1 + D + D2 + D3 + D4 + D6/ 1 + D2 + D3 + D5 + D6

resulting in 513 coded bits, {C(0)… C(512)} defined by:

r(k) = u(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)

C(3k) = u(k)

C(3k+1) = r(k)+r(k-1)+r(k-4)+r(k-6)

C(3k+2) = r(k)+r(k-1)+ r(k-2)+r(k-3)+r(k-4)+r(k-6) for k = 0, 1, ..., 164;

r(k) = 0 for k<0 and (for termination of the coder):

r(k) = 0

C(3k) = r(k-2) + r(k-3) + r(k-5) + r(k-6)

C(3k+1) = r(k)+r(k-1)+r(k-4)+r(k-6)

C(3k+2) = r(k)+r(k-1)+ r(k-2)+r(k-3)+r(k-4)+r(k-6) for k = 165, 166, ..., 170

The code is punctured in such a way that the following 65 coded bits:

C(1), C(2), C(4), C(5), C(8), C(22), C(70), C(118), C(166), C(214), C(262), C(310), C(317), C(319), C(325), C(332), C(334), C(341), C(343), C(349), C(356), C(358), C(365), C(367), C(373), C(380), C(382), C(385), C(389), C(391), C(397), C(404), C(406), C(409), C(413), C(415), C(421), C(428), C(430), C(433), C(437), C(439), C(445), C(452), C(454), C(457), C(461), C(463), C(469), C(476), C(478), C(481), C(485), C(487), C(490), C(493), C(500), C(502), C(503), C(505), C(506), C(508), C(509), C(511) and C(512) are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as

c(k+8) = P(k) for k = 0, 1, ..., 447.

TCH/AFS7.4:

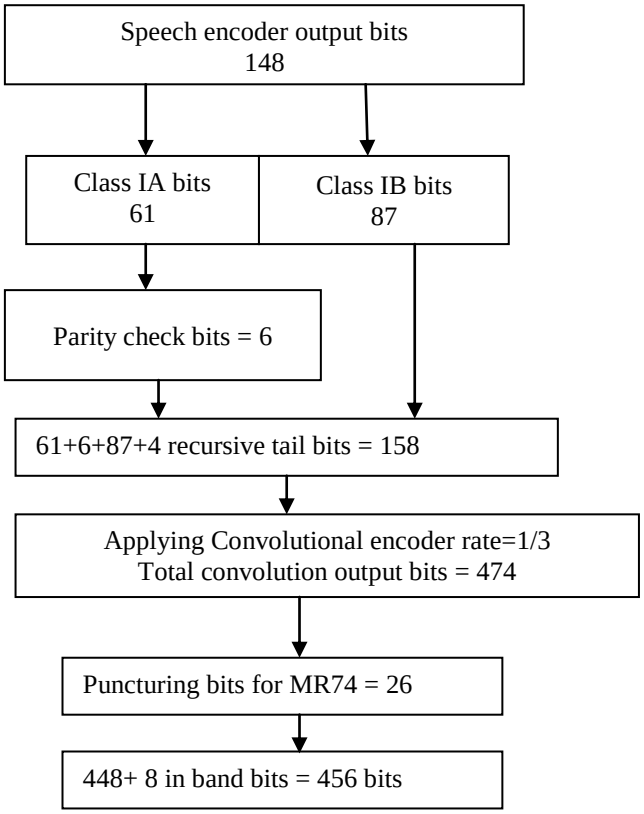

Figure 2.12: Bits classification for channel coding in GSM AMR-NB with mode MR74

The block of 154 bits {u(0)... u(153)} is encoded with the 1/3 rate convolutional code defined by the following polynomials:

G1/G3 = 1 + D + D3 + D4 / 1 + D + D2 + D3 + D4

G2/G3 = 1 + D2 + D4 / 1 + D + D2 + D3 + D4

G3/G3 = 1

resulting in 474 coded bits, {C(0)... C(473)} defined by:

r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)

C(3k) = r(k) + r(k-1) + r(k-3) + r(k-4)

C(3k+1) = r(k)+r(k-2)+r(k-4)

C(3k+2) = u(k) for k = 0, 1, ..., 153 and (for termination of the coder):

r(k) = 0

C(3k) = r(k)+r(k-1) + r(k-3) + r(k-4)

C(3k+1) = r(k)+r(k-2)+r(k-4)

C(3k+2) = r(k-1)+r(k-2)+r(k-3)+r(k-4) for k = 154, 155, ..., 157

The code is punctured in such a way that the following 26 bits:

C(0), C(355), C(361), C(367), C(373), C(379), C(385), C(391), C(397), C(403), C(409), C(415), C(421), C(427), C(433), C(439), C(445), C(451), C(457), C(460), C(463), C(466), C(468), C(469), C(471) and C(472) are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as:

c(k+8) = P(k) for k = 0, 1, ..., 447.

TCH/AFS6.7:

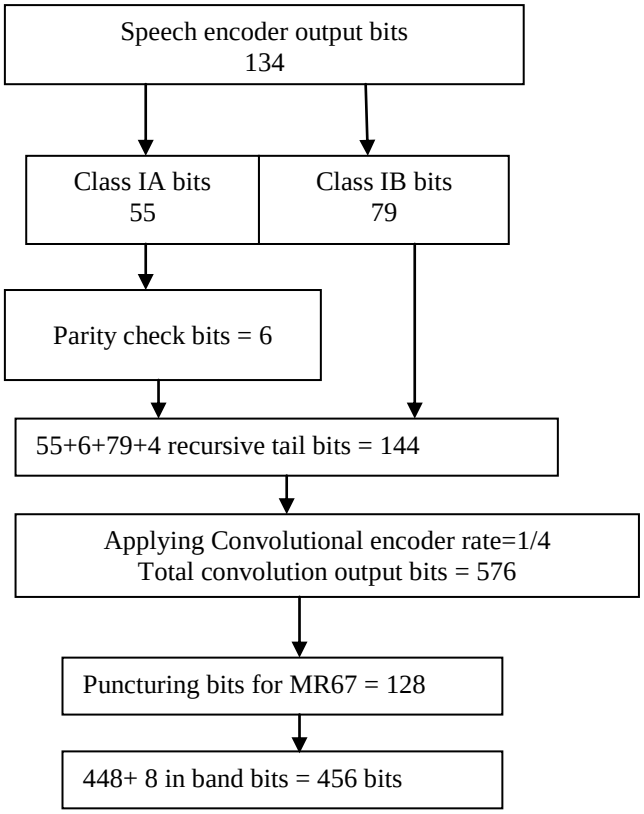

Figure 2.13: Bits classification for channel coding in GSM AMR-NB with mode MR167

The block of 140 bits {u(0)… u(139)} is encoded with the ¼ rate convolutional code defined by the following polynomials:

G1/G3 = 1 + D + D3 + D4 / 1 + D + D2 + D3 + D4

G2/G3 = 1 + D2 + D4 / 1 + D + D2 + D3 + D4

G3/G3 = 1

G3/G3 = 1

resulting in 576 coded bits, {C(0)… C(575)} defined by:

r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)

C(4k) = r(k) + r(k-1) + r(k-3) + r(k-4)

C(4k+1) = r(k)+r(k-2)+r(k-4)

C(4k+2) = u(k)

C(4k+3) = u(k) for k = 0, 1, ..., 139;

r(k) = 0 for k<0 and (for termination of the coder):

r(k) = 0

C(4k) = r(k)+r(k-1) + r(k-3) + r(k-4)

C(4k+1) = r(k)+r(k-2)+r(k-4)

C(4k+2) = r(k-1)+r(k-2)+r(k-3)+r(k-4)

C(4k+3) = r(k-1)+r(k-2)+r(k-3)+r(k-4) for k = 140, 141, ..., 143

The code is punctured in such a way that the following 128 coded bits:

C(1), C(3), C(7), C(11), C(15), C(27), C(39), C(55), C(67), C(79), C(95), C(107), C(119), C(135), C(147),C(159), C(175), C(187), C(199), C(215), C(227), C(239), C(255), C(267), C(279), C(287), C(291), C(295), C(299), C(303), C(307), C(311), C(315), C(319), C(323), C(327), C(331), C(335), C(339), C(343), C(347), C(351), C(355), C(359), C(363), C(367), C(369), C(371), C(375), C(377), C(379), C(383), C(385), C(387), C(391), C(393), C(395), C(399), C(401), C(403), C(407), C(409), C(411), C(415), C(417), C(419), C(423), C(425), C(427), C(431), C(433), C(435), C(439), C(441), C(443), C(447), C(449), C(451), C(455), C(457), C(459), C(463), C(465), C(467), C(471), C(473), C(475), C(479), C(481), C(483), C(487), C(489), C(491), C(495), C(497), C(499), C(503), C(505), C(507), C(511), C(513), C(515), C(519), C(521), C(523), C(527), C(529), C(531), C(535), C(537), C(539), C(543), C(545), C(547), C(549), C(551), C(553), C(555), C(557), C(559), C(561), C(563), C(565), C(567), C(569), C(571), C(573) and C(575) are not transmitted. The result is a block of 448 coded bits, P(0)...P(447) which are appended to the in-band bits in c as

c(k+8) = P(k) for k = 0, 1, ..., 447.

TCH/AFS5.9:

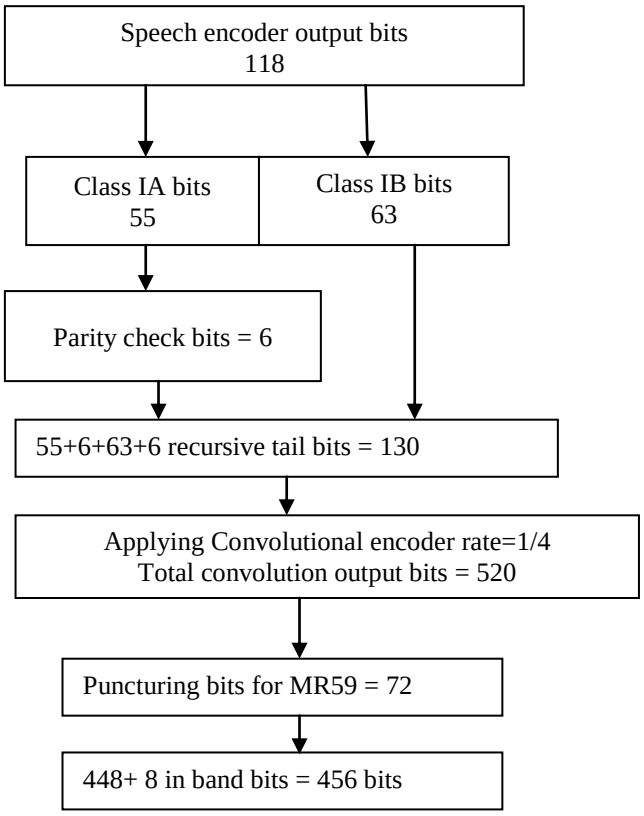

Figure 2.14: Bits classification for channel coding in GSM AMR-NB with mode MR59

The block of 124 bits {u(0)… u(123)} is encoded with the ¼ rate convolutional code defined by the following polynomials:

G4/G6 = 1 + D2 + D3 + D5 + D6 / 1 + D + D2 + D3 + D4 + D6

G5/G6 = 1 + D + D4 + D6 / 1 + D + D2 + D3 + D4 + D6

G6/G6 =1

G6/G6 =1

resulting in 520 coded bits, {C(0)… C(519)} defined by:

r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6)

C(4k) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)

C(4k+1) = r(k) + r(k-1) + r(k-4) + r(k-6)

C(4k+2) = u(k)

C(4k+3) = u(k) for k = 0, 1, ..., 123;

r(k) = 0 for k<0 and (for termination of the coder):

r(k) = 0

C(4k) = r(k)+r(k-2) + r(k-3) + r(k-5) + r(k-6)

C(4k+1) = r(k)+r(k-1)+r(k-4)+r(k-6)

C(4k+2) = r(k-1)+r(k-2)+ r(k-3)+r(k-4)+r(k-6)

C(4k+3) = r(k-1)+r(k-2)+ r(k-3)+r(k-4)+r(k-6) for k = 124, 125, ..., 129

The code is punctured in such a way that the following 72 coded bits:

C(0), C(1), C(3), C(5), C(7), C(11), C(15), C(31), C(47), C(63), C(79), C(95), C(111), C(127), C(143), C(159), C(175), C(191), C(207), C(223), C(239), C(255), C(271), C(287), C(303), C(319), C(327), C(331), C(335), C(343), C(347), C(351), C(359), C(363), C(367), C(375), C(379), C(383), C(391), C(395), C(399), C(407), C(411), C(415), C(423), C(427), C(431), C(439), C(443), C(447), C(455), C(459), C(463), C(467), C(471), C(475), C(479), C(483), C(487), C(491), C(495), C(499), C(503), C(507), C(509), C(511), C(512), C(513), C(515), C(516), C(517) and C(519) are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as

c(8+k) = P(k) for k = 0, 1, ..., 447.

TCH/AFS5.15:

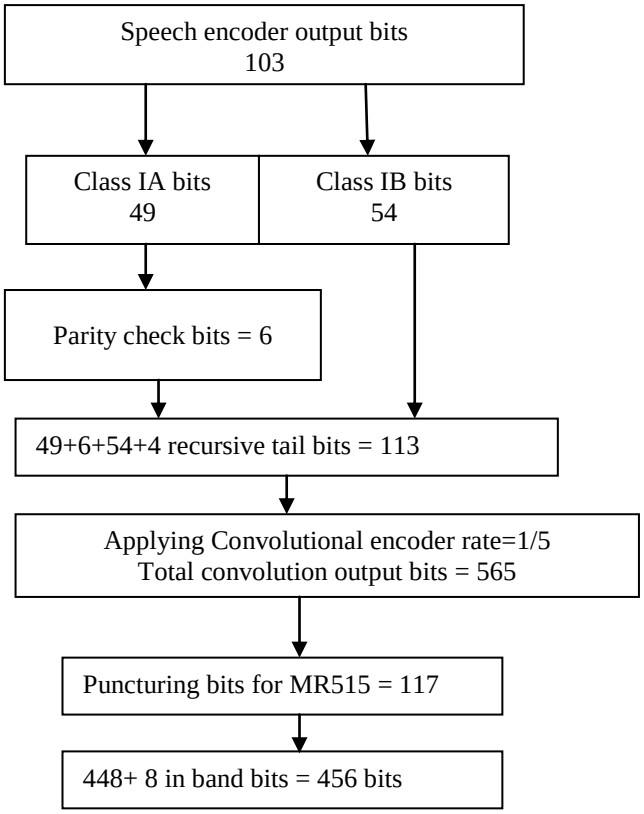

Figure 2.15: Bits classification for channel coding in GSM AMR-NB with mode MR122

The block of 109 bits {u(0)… u(108)} is encoded with the 1/5 rate convolutional code defined by the following polynomials:

G1/G3 = 1 + D + D3 + D4 / 1 + D + D2 + D3 + D4

G1/G3 = 1 + D + D3 + D4 / 1 + D + D2 + D3 + D4

G2/G3 = 1 + D2 + D4 / 1 + D + D2 + D3 + D4

G3/G3 = 1

G3/G3 = 1

resulting in 565 coded bits, {C(0)… C(564)} defined by:

r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)

C(5k) = r(k) + r(k-1) + r(k-3) + r(k-4)

C(5k+1) = r(k) + r(k-1) + r(k-3) + r(k-4)

C(5k+2) = r(k)+r(k-2)+r(k-4)

C(5k+3) = u(k)

C(5k+4) = u(k)

for k = 0, 1, ..., 108; r(k) = 0 for k<0 and (for termination of the coder):

r(k) = 0

C(5k) = r(k)+r(k-1) + r(k-3) + r(k-4)

C(5k+1) = r(k)+r(k-1) + r(k-3) + r(k-4)

C(5k+2) = r(k)+r(k-2)+r(k-4)

C(5k+3) = r(k-1)+r(k-2)+r(k-3)+r(k-4)

C(5k+4) = r(k-1)+r(k-2)+r(k-3)+r(k-4) for k = 109, 110, ..., 112

The code is punctured in such a way that the following 117 coded bits:

C(0), C(C(4), C(5), C(9), C(10), C(14), C(15), C(20), C(25), C(30), C(35), C(40), C(50), C(60), C(70), C(80), C(90), C(100), C(110), C(120), C(130), C(140), C(150), C(160), C(170), C(180),C(190), C(200), C(210), C(220), C(230), C(240), C(250), C(260), C(270), C(280),C(290), C(300), C(310), C(315), C(320), C(325), C(330), C(334), C(335), C(340), C(344), C(345), C(350), C(354), C(355), C(360), C(364), C(365), C(370), C(374), C(375), C(380), C(384), C(385), C(390), C(394), C(395), C(400), C(404), C(405), C(410), C(414), C(415), C(420), C(424), C(425), C(430), C(434), C(435), C(440), C(444), C(445), C(450), C(454), C(455), C(460), C(464), C(465), C(470), C(474), C(475), C(480), C(484), C(485), C(490), C(494), C(495), C(500), C(504), C(505), C(510), C(514), C(515), C(520), C(524), C(525), C(529), C(530), C(534), C(535), C(539), C(540), C(544), C(545), C(549), C(550), C(554), C(555), C(559), C(560) and C(564) are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as

c(8+k) = P(k) for k = 0, 1, ..., 447.

TCH/AFS4.75:

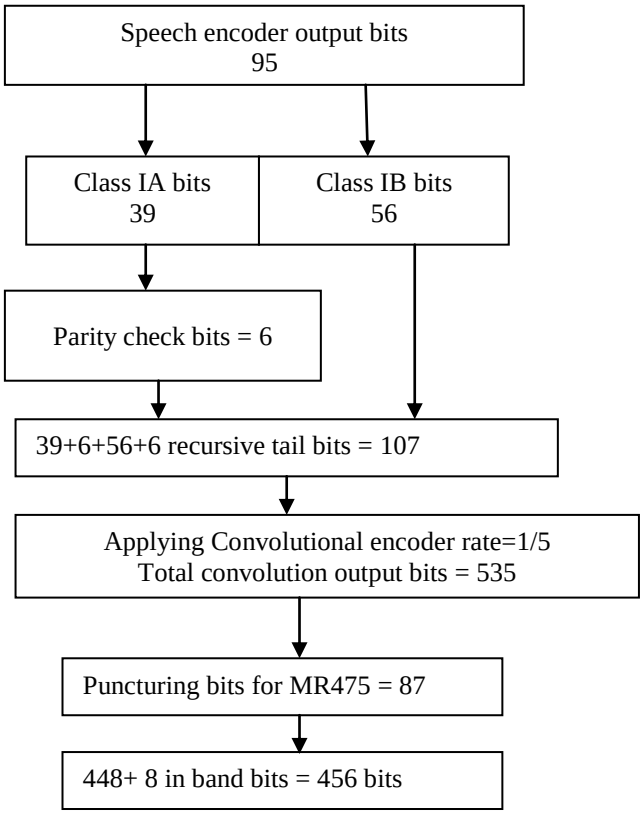

Figure 2.16: Bits classification for channel coding in GSM AMR-NB with mode MR122

The block of 101 bits {u(0)... u(100)} is encoded with the 1/3 rate convolutional code defined by the following polynomials:

G4/G6 = 1 + D2 + D3 + D5 + D6 / 1 + D + D2 + D3 + D4 + D6

G4/G6 = 1 + D2 + D3 + D5 + D6 / 1 + D + D2 + D3 + D4 + D6

G5/G6 = 1 + D + D4 + D6 / 1 + D + D2 + D3 + D4 + D6

G6/G6 =1

G6/G6 =1

resulting in 535 coded bits, {C(0)… C(534)} defined by:

r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6)

C(5k) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)

C(5k+1) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)

C(5k+2) = r(k) + r(k-1) + r(k-4) + r(k-6)

C(5k+3) = u(k)

C(5k+4) = u(k) for k = 0, 1, ..., 100; r(k) = 0 for k<0 and (for termination of the coder):

r(k) = 0

C(5k) = r(k)+r(k-2) + r(k-3) + r(k-5) + r(k-6)

C(5k+1) = r(k)+r(k-2) + r(k-3) + r(k-5) + r(k-6)

C(5k+2) = r(k)+r(k-1)+r(k-4)+r(k-6)

C(5k+3) = r(k-1)+r(k-2)+ r(k-3)+r(k-4)+r(k-6)

C(5k+4) = r(k-1)+r(k-2)+ r(k-3)+r(k-4)+r(k-6) for k = 101, 102, ..., 106

The code is punctured in such a way that the following 87 coded bits:

C(0), C(1), C(2), C(4), C(5), C(7), C(9), C(15), C(25), C(35), C(45), C(55), C(65), C(75), C(85), C(95), C(105), C(115), C(125), C(135), C(145), C(155), C(165), C(175), C(185), C(195), C(205), C(215), C(225), C(235), C(245), C(255), C(265), C(275), C(285), C(295), C(305), C(315), C(325), C(335), C(345), C(355), C(365), C(375), C(385), C(395), C(400), C(405), C(410), C(415), C(420), C(425), C(430), C(435), C(440), C(445), C(450), C(455), C(459), C(460), C(465), C(470), C(475), C(479), C(480), C(485), C(490), C(495), C(499), C(500), C(505), C(509), C(510), C(515), C(517), C(519), C(520), C(522), C(524), C(525), C(526), C(527), C(529), C(530), C(531), C(532) and C(534) are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the inband bits in c as

c(8+k) = P(k) for k = 0, 1, ..., 447.

2.9.5 Interleaving

The coded bits are reordered and interleaved according to the following rule and is given the table 4 of **[ETSI TS 145 003 V13.2.0 (2016-08)].**

i (B, j) = c(n, k) for k = 0,1,...,227

n = 0, 1,..., N, N+1, ...

$B = B_0 + 2n + b$

The values of b and j in dependence of k are given by table 4.

The result of the interleaving is a distribution of the reordered 228 bits of a given data block, n = N, over 4 blocks using the even numbered bits of the first 2 blocks ($B = B_0+2N+0,1$) and the odd numbered bits of the last 2 blocks ($B = B_0+2N+2,3$). The reordered bits of the following data block, n = N + 1, use

the even numbered bits of the blocks B = B0 + 2N + 2,3 (B = B0+2(N+1)+0,1) and the odd numbered bits of the blocks B = B0 + 2(N+1) + 2,3. Continuing with the next data blocks shows that one block always carries 57 bits of data from one data block (n = N) and 57 bits from the next block (n = N+1), where the bits from the data block with the higher number always are the even numbered data bits, and those of the data block with the lower number are the odd numbered bits. The block of coded data is interleaved "block diagonal", where a new data block starts every 2nd block and is distributed over 4 blocks.

2.9.6 Mapping on a burst

The mapping is given by the rule:

$$e(B, j) = i(B, j) \text{ and } e(B,59+j) = i(B,57+j) \text{ for } j = 0,1,...,56$$

$$e(B,57) = hl(B) \text{ and } e(B,58) = hu(B)$$

The two bits, labeled hl(B) and hu(B) on burst number B are flags used for indication of control channel signaling. For each TCH/HS block not stolen for signaling purposes:

$$hu(B) = 0 \text{ for the first 2 bursts (indicating status of the even numbered bits)}$$

$$hl(B) = 0 \text{ for the last 2 bursts (indicating status of the odd numbered bits)}$$

For the use of hl(B) and hu(B) when a speech frame is stolen for signaling purposes.

The interleaver shuffles the bits contained in the data blocks that are coming from the channel encoder, and distributes them over a number of bursts. The purpose of this procedure is to ensure that the errors that appear in a received data block are uncorrelated. The motivation for reducing the correlation between bit errors is that the convolution code used to protect the class I bits has better performance when errors are not correlated. Correlation between bit errors can occur in for example fading conditions. The block of size 456 bits coming out of channel encoder is spread onto eight bursts in subblocks of 57 bits each. A subblock is defined as either the odd or even-numbered bits of the coded data within one burst. The data are not put an ordered row into these subblocks, but are re-ordered before they are mapped onto the GSM bursts. This further decreases the possibility of a whole group of consecutive bits being destroyed in the radio channel. Convolutional codes are much better at repairing individual bit errors than they are at repairing burst errors. The 456 bits are subdivided onto the eight subblocks in the following way. Bit number 0 goes into subblock 1, bit number 1 goes into subblock 2, and so on until all eight subblocks are used up. Bit number 8 ends up in subblock number 1 again. The first four subblocks are put into the even numbered bits of four consecutive bursts, and the second four blocks are put into the odd numbered bits of the next consecutive four bursts. Upon receipt of the next speech block, a burst then holds contributions from two successive speech blocks 5.

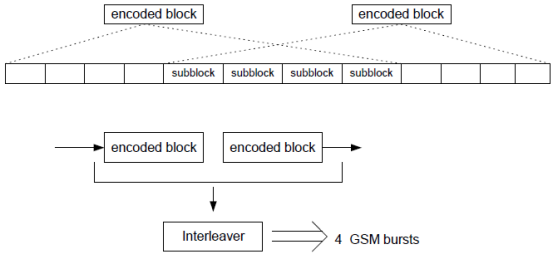

Figure 2.17: Interleaving and mapping on bursts

It can be realized that the interleaver can be implemented so that it operates at two code blocks at a time. For each interleaving pass four sets of encoded blocks are returned. These data are further

processed for the burst structure. Since two instances of encoded blocks contain two times 456 bits, and four set of output burst contain 456 bits, it is evident that all the bits contained in the input to the interleaver are not represented in the output. This is solved by passing each code block to the interleaver two times. In practice this is done by implementing a queue of code blocks.

2.10 Multiplexing

The input to the Multiplexer is TX _data, and the output is given in TX _burst. What the multiplexer does is to take TX _data from the interleaver, and place it appropriately in a frame structure.

GSM frame structure

There are two frequency bands of 25 MHz each that have been allocated for the use of GSM. The band 890 - 915 MHz is used for the uplink direction (from the mobile station to the base station). The band 935 - 960 MHz is used for the downlink direction (from the base station to the mobile station). By FDMA, the frequency band is divided into 200 kHz subbands, and by TDMA, each subband is divided into 120ms multi frames, and each multi frame is divided into 26 frames, the first two frames are used for controlling channel, and the rest are for traffic channel, and again, each frame is divided into 8 time bursts, each of which is approximately 0.577 ms. In GSM, there are 4 different types of bursts. A normal burst is used to carry speech and data information. Each burst consists of 3 tail bits at each end, 2 stealing bits, 2 data sequences of 57-bits, a 26-bit training sequence for equalization, and 8.25 guard bits. The implemented burst, referred to as a GSM normal burst, has the structure Displayed in figure

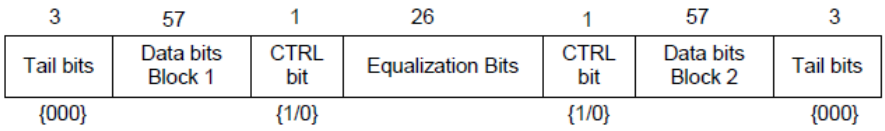

Figure 2.18: GSM normal burst structure

In one GSM burst, there are 3+57+1+26+1+57+3=148 bits. Of these 148 bits, 57*2=114 are data bits. The equalization bit sequence can be any of eight prescribed ones. The bit sequence selected in this project was:

EQUALIZATION_BITS = [0 0 1 0 0 1 0 1 1 1 0 0 0 0 1 0 0 0 1 0 0 1 0 1 1 1];

The tail bits are all zeros, control bits are selected as 1's. This is the structure that is modulated; on the receiver side, the 114 data bits are extracted from each burst and applied to de-interleaver.

2.11 Differential Encoding

The output from the GSM Burst is a binary {0, 1} bit sequence. This sequence is first mapped from the RTZ (Return to Zero) signal representation to a NRZ representation before being input to the GMSK-modulator. This task is accomplished by the differential encoding function.

GSM makes Use of the following combined differential encoding and level shifting scheme, where

$$d\{0,1\}, \ldots \ldots \ldots . a\{-1,1\}$$

$$d[n] = d[n] \oplus \mathrm{d[n-1]}$$

$$a[n] = 1 - 2 \times d\hat{}[n]$$

Hence, when calculating a [0] and thereby also d' [0], it may be assumed to d [-1] =1. This function generates differential encoding sequence each of 148 values per block.

2.12 GMSK Modulation

Gaussian Minimum Shift Keying, or to give it its full title Gaussian filtered Minimum Shift Keying, GMSK, is a form of modulation used in a variety of digital radio communications systems. It has advantages of being able to carry digital modulation while still using the spectrum efficiently. One of the problems with other forms of phase shift keying is that the sidebands extend outwards from the main carrier and these can cause interference to other radio communications systems using nearby channels. To overcome this, MSK and its derivative GMSK can be used.

MSK and also GMSK modulation are what is known as a continuous phase scheme. Here there are no phase discontinuities because the frequency changes occur at the carrier zero crossing points. This arises as a result of the unique factor of MSK that the frequency difference between the logical one and logical zero states is always equal to half the data rate. This can be expressed in terms of the modulation index, and it is always equal to 0.5.

The spectrum of an MSK signal consist sidebands extending beyond a bandwidth equal to the data rate. This can be reduced by passing the modulating signal through a low pass filter prior to applying it to the carrier. The requirements for the filter are that it should have a sharp cut-off, narrow bandwidth and its impulse response should show no overshoot. The ideal filter is known as a Gaussian filter which has a Gaussian shaped response to an impulse and no ringing. In this way the basic MSK signal is converted to GMSK modulation.

2.12.1 Generating GMSK modulation

There are two main ways in which GMSK modulation can be generated. The most obvious way is to filter the modulating signal using a Gaussian filter and then apply this to a frequency modulator where the modulation index is set to 0.5. This method is very simple and straightforward but it has the drawback that the modulation index must exactly equal 0.5. In practice this analogue method is not suitable because component tolerances drift and cannot be set exactly.

A second method is more widely used. Here what is known as a quadrature modulator is used. The term quadrature means that the phase of a signal is in quadrature or 90 degrees to another one. The quadrature modulator uses one signal that is said to be in-phase and another that is in quadrature to this. In view of the in-phase and quadrature elements this type of modulator is often said to be an I-Q modulator. Using this type of modulator the modulation index can be maintained at exactly 0.5 without the need for any settings or adjustments. This makes it much easier to use, and capable of providing the required level of performance without the need for adjustments. For demodulation the technique can be used in reverse.

2.12.2 Advantages of GMSK modulation

There are several advantages to the use of GMSK modulation for a radio communications system. One is obviously the improved spectral efficiency when compared to other phase shift keyed modes.

A further advantage of GMSK is that it can be amplified by a non-linear amplifier and remain undistorted This is because there are no elements of the signal that are carried as amplitude variations. This advantage is of particular importance when using small portable transmitters, such as those required by cellular technology. Non-linear amplifiers are more efficient in terms of the DC power input from the power rails that they convert into a radio frequency signal. This means that the power

consumption for a given output is much less, and this results in lower levels of battery consumption; a very important factor for cell phones.

A further advantage of GMSK modulation again arises from the fact that none of the information is carried as amplitude variations. This means that is immune to amplitude variations and therefore more resilient to noise, than some other forms of modulation, because most noise is mainly amplitude based.

Here GMSK modulator accepts a GSM burst bit sequence and performs a GMSK modulation of the sequence. It will generate in-phase (I) and quadrature-phase (Q) components, each of 600 values/block.

2.13 Channel Simulator

The channel simulator includes only noise by using variance value. This function will give received signal r (r= (I + noise) + j (Q + noise)) of 600 values per block. The channel simulator included in the GSMsim package to add noise.

In this project of multi path fading is considered for channel simulation. Multipath fading in wireless communication systems is commonly modeled by Rayleigh distribution if the line of sight component absent, if it is present then Rician distribution can be used. In mobile communication most of the cases line of sight path not exist hence Rayleigh fading channel is simulated for adding channel noise.

Gaussian Minimum Shift Keying modulated signal is passed through channel simulator to super impose with the noise using Rayleigh distribution by varying the variance. In-phase signal (I) and Quadrature signal (Q) are provided as individual input vectors to the simulator. r is the received signal predicted by the channel simulator as an output and this is the Complex baseband representation of the received GMSK modulated signal. The format is a row vector consisting of complex floating point numbers.

2.14 Matched Filtering

This function performs the tasks of channel impulse response estimation, bit synchronization, matched filtering and signal sample rate down conversion. This function gives 148values/block.

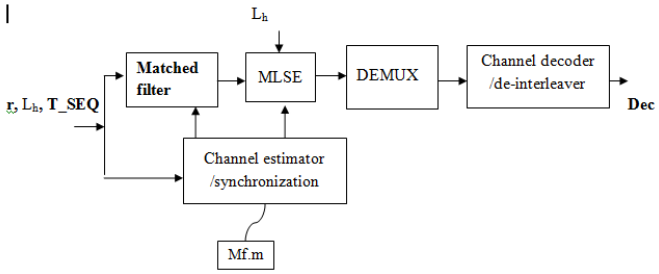

Figure 2.19: Illustration of how channel estimator/synchronization and matched filtering

From Figure 2.19, both the channel estimator and the matched filter have the sampled received signal, rr as input. rr is a sampled sequence which is expected to contain the received GSM burst. Also, the oversampling factor, OSR described as f_s/r_b with f_s being the sample frequency, and r_b the symbol rate, is input to both of these two blocks. Finally, these two blocks have L_h as input, where L_h is the desired length of the channel impulse response measured in bit time durations. The channel estimator passes an estimate of the channel impulse response, h, to the matched filter. Also, the channel estimator passes the sample number corresponding to the estimated beginning of the burst in r.

To interface correctly with the MLSE implementation mf.m must return a down sampled one sample per symbol version of the now matched filtered burst. Also, the MLSE requires information concerning

the matched filter. This information is supplied by also returning the impulse response autocorrelation, i.e. $R_{hh.}$

To understand the operation of mf.m, recall from earlier that a training sequence is inserted in each burst. The method used for obtaining synchronization is based on the mathematical properties of this training sequence.

The training sequence, TRAINING, used in *GSMsim* is as follows

$$TRAINING = [0,0,1,0,0,1,0,1,1,1,0,0,0,0,1,0,0,0,1,,0,0,1,0,1,1,1]$$

For which the following MSK-mapped equivalent, $T_{SEQ,}$ is used

$$T_{SEQ} = [1, J, 1, -J, 1, -J, -1, J, -1, -J, -1, -J, 1, J, 1, -J, 1, J, 1, -J, 1, -J, -1, J, -1, -J]$$

This sequence is one of eight predefined training sequences when a normal burst is considered. Now, from T_{SEQ} the central sixteen MSK-symbols are picked and referred to as $T_{SEQc.}$ If TSEQc is extended by placing five zeros in both ends, a sequence, T_{SEQe} is obtained.

2.15 Vitterbi Detection

The Vitterbi detector has been tested in two major tests. In the first test the detector is fed a sequence of non-distorted MSK-symbols. This is done using an OSR of 1 and the following impulse response $h = [1,0,0,0,0]$ And L_h is set to 5 and the corresponding value of $R_{hh} = [1,0,0,0,0]$ is used. With these settings the metric of the final survivor path should be 148. To realize this, observe that 148 transitions exist. Also, the metric gain for a single transition should equal

$$Gain(Y[n], s_a s_b) = \Re\{I^*[n]\, Y[n]\} - \Re\{I^*[n] \sum_{m=n-L_h}^{n-1} I[m] R_{hh}\,[n-m]$$

The result of the test is that the best path has the total metric 148, indicating correct operation. Also, the algorithm identifies this correctly, and does mapping from the survivor path and back to the transmitted binary symbols.

From this test, it is concluded that the metrics of a given previous survivor state is transferred correctly to the corresponding present state, and that the mapping from a path to binary information is correct. Thus, what may be referred to as the basic functionality and control flow within the algorithm is working as specified

This function does the actual detection of the received sequence. This function will give 148 bits /block. The output of the Vitterbi detector is Y, Rhh= 148bits/block is given to the channel.

2.16 De Multiplexing

From the channel 148 bits/block is given to the input of the De Multiplexing, the tasks of de-multiplexing, de-interleaving and decoding the data, are implemented in three separate blocks. This function accepts 148 bit/block and removes tail, ctrl and training bits and gives 114 data bits per each block.

The overall task of these three blocks is to regenerate the transmitted coded data blocks.

The input to the de-multiplexer is rx_burst (148bits/Block), which is output from the MLSE, The output from the de-multiplexer is the contents of the two data fields in a standard GSM burst.

These data are returned in a variable called rx_data. The de-multiplexer is simple in its function, since all that needs to be done is to locate the data fields in rx_burst and then copy these to rx_data.

The output of the deburst is having 114 bits/block that is total of four blocks having each one is 114 bits.

2.17 De Interleaving

This function takes 456 bits and uses the previous frame 456 bits and give 456 bits. The de-interleaver reconstructs the received encoded data, rx_enc from the received data, rx_data. The operation is the inverse of the interleaver, and may thus be considered as a reordering of the shuffled bits.

The de-interleaver operates according to the following two formulas

$$R = 4.B + (b\ mod\ 8)$$

$$r = 2.\big((49.b) mod\ 57\big) + \big((b\ mod\ 8) div\ 4\big)$$

Which provide the information that bit number b for rx_enc instance number B, may be retrieved from rx_data corresponding to burst number R at position r.

De-interleaver can be implemented so that it operates on eight sets of rx_data at a time. For each de-interleaving pass one instance of rx_enc is returned. Since rx_enc contains 456 bit, and eight sets of rx_data contain two times 456 bit, it is evident that all the bits contained in the input to the de-interleaver are not represented in the output. This is solved by passing each set of rx_data to the interleaver two times.

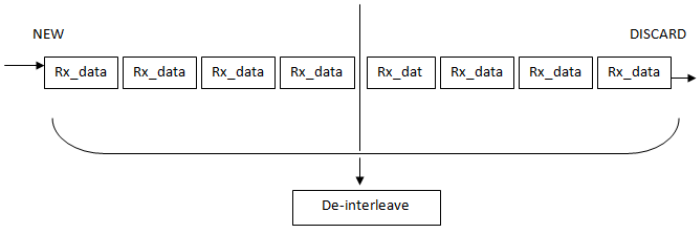

Figure 2.20: Operation of the de-interleaver aided by a queue

In the De Interleaving, the Received four 114 bits/block is forming into single block of 456 bits

2.18 Channel Decoder

This sequence of codewords passed through the channel decoder which attempts to reconstruct the original information sequence from the knowledge of the code used by the channel encoder and the redundancy contained in the received data. A Viterbi decoder uses the Viterbi algorithm for decoding a bit stream that has been encoded using convolutional code or trellis code. The Viterbi Algorithm (VA) was first proposed as a solution to the decoding of convolutional codes by Andrew J. Viterbi in 1967,

Performing Viterbi Decoding

The most important concept to aid in understanding the Viterbi algorithm is the trellis diagram. The Figure 2.21 below shows the empty trellis diagram for representing states and original code word for an input bit.

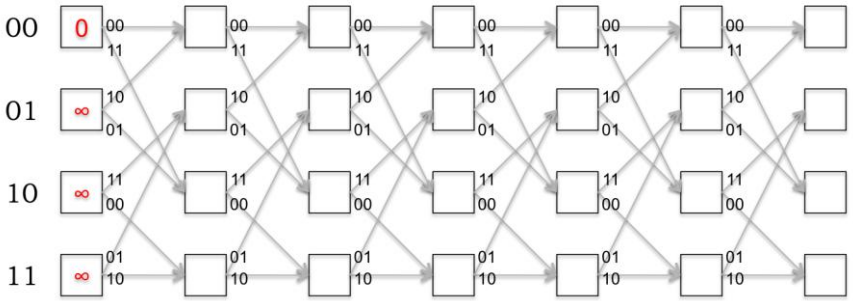

Figure 2.21: Trellis representation of convolutional code with rate = 1/2

The decoding algorithm uses two metrics: the branch metric (BM) and the path metric (PM). The branch metric is a measure of the "distance" between what was transmitted and what was received, and is defined for each arc in the trellis. In hard decision decoding, where we are given a sequence of digitized parity bits, the branch metric is the Hamming distance between the expected parity bits and the received ones. An example is shown in Figure 2.22, where the received bits are 00. For each state transition, the number on the arc shows the branch metric for that transition. Two of the branch metrics are 0, corresponding to the only states and transitions where the corresponding Hamming distance is 0. The other non-zero branch metrics correspond to cases when there are bit errors. The path metric is a value associated with a state in the trellis (i.e., a value associated with each node). For hard decision decoding, it corresponds to the Hamming distance over the most likely path from the initial state to the current state in the trellis. By "most likely", we mean the path with smallest Hamming distance between the initial state and the current state, measured over all possible paths between the two states. The path with the smallest Hamming distance minimizes the total number of bit errors, and is most likely when the BER is low.

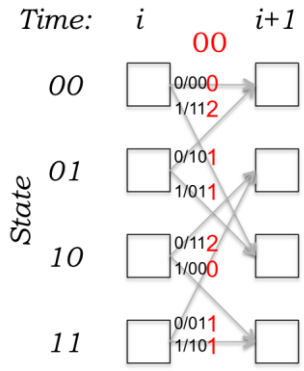

Figure 2.22 : The branch metric for hard decision decoding. e.g., the receiver gets the parity bits 00

The key insight in the Vitterbi algorithm is that the receiver can compute the path metric for a (state, time) pair incrementally using the path metrics of previously computed states and the branch metrics.

The decoding process begins with building the accumulated error metric for some number of received channel symbol pairs, and the history of what states preceded the states at each time instant t with the smallest accumulated error metric. Once this information is built up, the Vitterbi decoder is ready to recreate the sequence of bits that were input to the convolution encoder when the message was encoded for transmission. This is accomplished by the following steps:

1. First, select the initial state as the one having the smallest accumulated error metric and save the state number of that state.
2. Iteratively perform the following step until the beginning of the trellis is reached: Working backward through the state history table, for the currently selected state, select a new state by looking in the state history table for the predecessor to the current state. Save this state number as the new currently selected state and continue. This step is called traceback.
3. Now work forward through the list of selected states saved in the previous steps. Look up what input bit corresponds to a transition from each predecessor state to its successor state. That is the bit that must have been encoded by the convolutional encoder.

This function accepts 456 bits per frame and performs Vitterbi algorithm for corresponding rate convolution. It is correcting errors if any in the frame and the output of this function is again 260 bits /frame

Number of matrices and vectors are generated to help keep track of the different paths in the state trellis and the corresponding metrics. These variables are termed as STATE and MATRIC respectively. Also, to distinguish between legal and illegal state transitions, two matrices NEXT and PREVIOUS are set up to determine which two states a given state may switch to next and what states that are allowed to lead to a given state, respectively.

In order to enable the calculation of the metric a matrix, (DIBIT for half rate convolution) is set up. When, in the channel encoder, a transition from one state to another state occurs two bits, here referred to as dibits are outputs. (Possible four DIBIT bits combinations that are output for a given transition is stored in the DIBIT matrix) In close relation to this (DIBIT) matrix a BIT MATRIX is also required. The structure of BIT is just as that of the DIBIT matrix only here the content is the one bit binary input that is required for a given state transition. Hence, the BIT matrix is used in mapping state transition information to actual binary and decoded information.

The 456 bits are given input to the channel decoder and gives the output of 260 bits/frame after that again added the first four bits to the 260 bits and then given to the ASR Recognition unit.

Study of distortion due to speech compression, effects of channel noise on bit error rate and correction of bit errors using channel coding is the main objective of this project report. The simulation based experimental setup used to study the performance of ASR in a wireless network using different wireless speech codecs is explained in the next chapter.

3 ASR using CMU Sphinx

3.1 CMU Sphinx

CMU Sphinx, also called Sphinx in short, is the general term to describe a group of speech recognition systems developed at Carnegie Mellon University. SPHINX is one of the best and most versatile recognition systems in the world today. Sphinx has several speech recognizer components including SPHINX trainer for training and a SPHINX decoder for recognition.

Training: The process of learning about the sound units is called training.

Components provided for Training

- sphinxtools
- Training database.
- A set of transcripts for the database (in a single file)
- Language dictionary: legitimate words in the language are mapped sequences of sound units (or sub-word units).
- Filler dictionary: non-speech sounds are mapped to corresponding non-speech or speech-like sound units.

Decoding or Recognition: The process of using the knowledge acquired to deduce the most probable sequence of units in a given signal is called decoding, or simply recognition.

Components provided for Decoding

- sphinxtools
- The decoder source code
- The trained acoustic models and language model
- The language dictionary
- The filler dictionary
- The test data

Sphinx encompasses a number of software systems, described below

Sphinxbase

Sphinxbase is a continuous-speech, speaker-independent recognition system making use of hidden Markov acoustic models (HMMs) and an n-gram statistical language model. It was developed by Lee. It has been superseded in performance by subsequent versions.

Sphinx 2

A fast performance-oriented recognizer, originally developed by Xuedong Huang at Carnegie Mellon University, Sphinx 2 focuses on real-time recognition suitable for spoken language applications. As such it incorporates functionality such as end-pointing, partial hypothesis generation, dynamic language model switching and so on. It is used in dialog systems and language learning systems. It can be used in computer based PBX systems such as Asterisk. Sphinx 2 code has also been incorporated into a number of commercial products.

Sphinx 3

Sphinx 2 used a semi-continuous representation for acoustic modeling (i.e., a single set of Gaussians is used for all models, with individual models represented as a weight vector over these Gaussians). Sphinx 3 adopted the prevalent continuous HMM representation and has been used primarily for high-accuracy, non-real-time recognition. Recent developments (in algorithms and in hardware) have made Sphinx 3 "near" real-time recognition.

Sphinx 4

Sphinx 4 is a complete re-write of the Sphinx engine with the goal of providing a more flexible framework for research in speech recognition, written entirely in the Java programming language. Sun Microsystems supported the development of Sphinx 4 and contributed software engineering expertise to the project. Current development goals include

- developing a new (acoustic model) trainer
- implementing speaker adaptation (e.g. MLLR)
- improving configuration management
- creating a graph-based UI for graphical system design

PocketSphinx

A Current real-time decoder development is taking place in the Pocket Sphinx project and it is a version of Sphinx that can be used in embedded systems (e.g., based on an ARM processor).

It is designed to be fast and for real time speed written in C, supports desktop applications and mobile devices. It needs the library Sphinxbase. PocketSphinx is under active development and incorporates features such as fixed-point arithmetic and efficient algorithms for GMM computation.

3.2 Speech Data Base

We need to have good quality of Speech data base for getting more accurate recognition of ASR, and also requires large vocabulary, dictionary file which maps all the words presented in Speech into a sequence of phones. Here for this project, TIMIT data base which is readily available in the Department is used.

TIMIT database: This TIMIT data base was recorded at TI, transcribed at MIT, and has been maintained, verified, and prepared for CD-ROM production by the National Institute of Standards and Technology (NIST), recorded in the .WAV format and also contains the '.txt', '.wrd', '.phn' files for the recorded data. TIMIT has total of 6300 utterances, 4620 training and 1680 testing utterances, and 10 sentences spoken by each of 630 peoples (438 MALE, 198 FEMALE) from 8 major dialect regions of the United States. The dialect regions are

DR1: New England
DR2: Northern
DR3: North Midland
DR4: South Midland
DR5: Southern
DR6: New York City
DR7: Western
DR8: Army Brat (moved around)

TIMIT contains the following sub folders

TRAIN: Contains the training data of 4620 utterances.

TEST: Contains the test data of 1680 utterances.

DOC: Contains the phoncode.doc, prompts.txt, spkrinfo.txt, spkrsent.txt, testset.doc, timitdic.doc and timitdic.txt.

3.3 Building ASR and STEPs

Step 1 Installation of CMU Sphinx tools

First the required sphinx tools are to be downloaded from the website into a Directory named 'sphinx' and then to be retrieved from tar files *http://cmusphinx.sourceforge.net/wiki/download/*

- Sphinxbase-0.7: Library tool
- Sphinxtrain-1.0.7 :Trainer tool
- Sphinx3: Decoder

To retrieve from tar files use below command

tar –xvzf an4_raw.littleendian.tar.gz (If AN4 data base is used)
tar –xvzf sphinxbase-0.7.tar.gz
tar –xvzf sphinxtrain-1.0.7.tar.gz
tar –xvzf sphinx.nightly.tar.gz

Then these tools are installed by using following commands in the terminal

. /configure
make
make install (sudo make install, if any permission errors)

Obtaining software directly from the source code generally involves the above three commands.

Step 2 Setup for Training

After the installation of required tools, we have to setup the tutorial by copying all relevant executables and scripts from Sphinxtrain directory to TIMIT database. This can be done as follows

perl scripts_pl/setup_tutorial.pl TIMIT

And it also creates empty folders in TIMIT directory, which are used later in training and testing. The empty folders created in TIMIT directory, are

- bin: Contains executable files/tools.
- scripts_pl: Contains script files.
- etc: Contains configuration file and all required pre-language model files are to be copied in to it.
- Wav: Train and Test data to be copied into this directory.
- feat: To store feature values.
- Bwaccumdir: To store intermediate values.
- Model architecture: To store model definition file.
- Model parameters: To store trained acoustic models.
- Trees: To store decision tree values of each state of all phones.
- qmanager
- result: To store decoded output file.
- Logdir: To store output log files it contains debug information.

The below listed files are to be copied into the etc folder of TIMIT directory. These files are readily available in the department.

TIMIT.dic,
TIMIT.filler,
TIMIT_train.fileids,
TIMIT_test.fileids,
TIMIT_train.transcription,
TIMIT_test.transcription,
TIMIT.phone,
TIMIT.ug.lm.

Step 3 Computation of Feature parameters for speech data used in training

The system does not directly work with acoustic signals. The signals are first transformed into a sequence of feature vectors, which are used in place of the actual acoustic signals. Each recording can be transformed into a sequence of feature vectors by using a script file make_feat.pl and a control file. All control files containing the list of feature-set filenames with full paths to them. An example of the entries in this file:

TRAIN/DR1/FCJF0/SA1

TRAIN/DR1/FCJF0/SA2

Here, etc/sphinx_train.cfg can be changed as follows, because here, training is done for sphinx3 and raw files are used.

$CFG_HMM_TYPE = '.cont.'; # Sphinx III

#$CFG_HMM_TYPE = '.semi.'; # Pocketsphinx

$CFG_WAVFILE_EXTENSION = 'raw'; (from sph)

To perform this transformation, run the following command

root@ubuntu:/home/ganga/sphinx/TIMIT#perl scripts_pl/make_feats.pl -ctl etc/TIMIT_train.fileids

This script will compute, for each training utterance, a sequence of 13-dimensional vectors (feature vectors) consisting of the Mel-frequency cepstral coefficients (MFCCs). The MFCCs will be placed automatically in a directory called '**feat**'.

Step 4 Training for feature parameters obtained in step 3

To train the reference model or acoustic model, the following files are needed

1. Feature files (.mfc files)

2. A transcript file, in which the transcripts corresponding to the feature files are listed in exactly the same order as the feature filenames in the control file.

<s> she had your dark suit in greasy wash water all year </s> (SA1)
<s> don't ask me to carry an oily rag like that </s> (SA2)

3. A main dictionary, which has all the words in the transcripts mapped onto the acoustic units.

abbreviate ax b r iy1 v iy ey2 t
abdomen ae1 b d ax m ax n

4. A filler dictionary, which has the non-speech words, maps them to user-defined phones. This dictionary must at least have the entries

<s> SIL beginning-utterance silence
<sil> SIL within-utterance silence

</s> SIL end-utterance silence

5. A phonelist, which is a list of all acoustic units that we want to train models for them. All units in above two dictionaries must be listed in it; otherwise those acoustic units will not be permitted for training. Each phone must be listed on a separate line in the phone list, beginning from the left, with no extra spaces after the phone.

Example: aa
aa1
aa2
ae

Training the acoustic models is gone through various stages

Training for context independent phones

Training for Context Dependent models with untied states

Training for Context Dependent models with tied states

Some of the HMM states may have similar to each other in the triphones, so if we collect the data from all those states, we can train one global state(which is called *'senone'* and also called '*tiedstate'*). Here Decision trees are used to decide which of the HMM states of triphones are similar to each other.

At each stage the below steps are performed

Generation of model definition

Model definition file is to define a unique numerical identity to every state of every HMM that we are going to train, and to provide an order which will be followed in writing out the model parameters in the model parameter files.

Initializing model parameters: means, variances, transition matrices, mixture_weights

Baum-Welch algorithm

After initialization of statistical parameters, these can be re-estimated using forward-backward re-estimation algorithm called "Baum-Welch algorithm". This is an iterative re-estimation process, so we have to run the Baum-Welch re-estimation algorithm many times over the training data followed by normalization. Slightly better set of models will result iterations as compared previous iterations. Number of iterations depends upon the likelihood and convergence ratio.

Convergence ratio = current likelihood - previous likelihood, where as

Current likelihood is given by is equal to total likelihood/total frames

Normalization

At every stage of re-estimation, we have to normalize the HMM's parameters generated in the above step.

By running the below command in terminal, the above training procedure is done

root@ubuntu:/home/ganga/sphinx/TIMIT# perl scripts_pl/RunAll.pl

Step 5 Setup for Testing

All the executable and script files are copied into the directory where the testing is done. This can be done by below command

root@ubuntu:/home/ganga/sphinx/TIMIT# perl ../Sphinx3/ scripts/setup_sphinx3.pl -task TIMIT

Step 6 Computation of feature parameters for test utterances

Converting test utterance into the sequence of vectors is done by using the script file *'make_feat.pl'* and a control file having the list of file ids of test utterances, which are to be processed. The resultant .mfc files are stored in feat directory.

root@ubuntu:/home/ganga/sphinx/TIMIT# perl scripts_pl/make_feats.pl -ctl etc/TIMIT_test.fileids

Step 7 Decoding with sphinx3

This can be done in two stages

Stage 1 Decoding: The required input files for decoding

Feature values
Acoustic model
Language model
Dictionary and filler dictionary files
Transcription files of test data

Recognition is done and the decoded output is written into text file in the 'result' folder.

Stage 2 Computation of error rate

Computes the error rate by comparing the original data with decoded output and displays the error rate, accuracy in the terminal as given below

TOTAL Words: 14550 Correct: 14057 Errors: 641

TOTAL Percent correct = 96.61% Error = 4.41% Accuracy=95.59%

Total insertions 148, deletions 82, substitutions: 411

The above procedure is done by using the script *slave.pl*

root@ubuntu:/home/ganga/sphinx/TIMIT# perl scripts_pl/decode/slave.pl

3.4 Conclusions

The above procedure is used for training and obtaining the acoustic and language models using TIMIT training speech database. The obtained models are tested with the TIMIT test database and ASR accuracy is 95.59%. Where ASR accuracy is capability of an ASR system can be examined by measuring its word accuracy. Word accuracy is expressed as a percentage and represents the number of words recognized correctly out of the total number of words spoken. This ASR accuracy will be the baseline for comparing the performance of NSR with different channel conditions and channel coding.

4 Experimental Results

4.1 Introduction

At present, almost all the wire-line and wireless network deployments extensively use the narrowband speech codecs, which work at 8-kHz sampling rate. Considering the volume of the deployment of these codecs in the networks, they will continue to exist for longer times, until completely replaced with wideband codecs. Till such time, ASR engines must have to work with narrowband speech codecs in NSR configuration.

Several studies were also reported in the literature, on the performance of ASR using narrowband speech codecs. Therefore the studies on MoS and performance of ASR due to different channel conditions (noise) using full rate speech codec and the corresponding channel coding, are carried out in this project. The procedure and results obtained are given in this chapter.

4.2 MOS Evaluation Procedure

For MOS evaluation, TIMIT database (speech files) which consists of about 6300 different speech files with about 5 seconds duration each is used. The raw MOS scores are obtained by comparing the original TIMIT speech files with the degraded files which are obtained after encoding and decoding process using the PESQ algorithm, P.862 executable file.

The MOS-LQO scores can be obtained for different channel noise conditions by using Executable file given in ITU-T P.862. The average MOS value of all the speech files is taken as the reference MOS value for the respective individual codec. MoS measurement for both narrowband and wideband codec is shown in figures 4.1 and 4.2

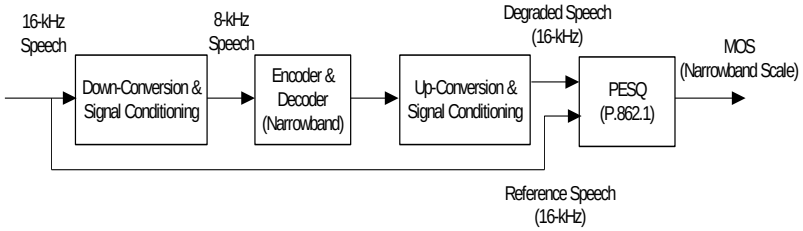

Figure 4.1: MOS measurement procedure for narrowband codecs

4.3 MOS evaluation for different channel noise conditions

Speech quality (MOS) is evaluated over the complete TIMIT speech files. Each speech file will generate a narrowband (P862.1) MOS value using the PESQ algorithms for the selected speech codec. The MOS values obtained for different channel noise conditions using narrowband speech codec are given in

4.4 Speech Recognition Setup for Narrowband Codecs

All the narrowband codecs work with 8 kHz sampling rates. 8-kHz trained HMM models are used for narrowband codec, the output of channel decoder is directly provided to the ASR system with 8-kHz trained models for the recognition analysis as shown in Figure 4.3.

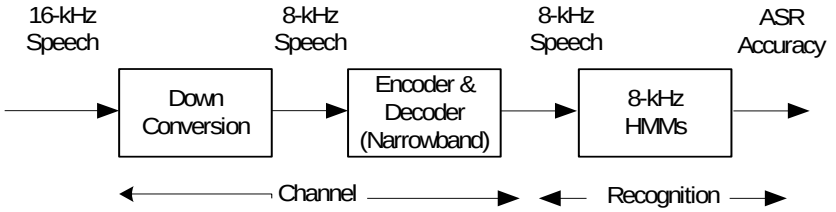

Figure 4.2: Recognition with 8-kHz trained models (HMM)

4.5 ASR Accuracy Measurement

The Word Accuracy Rate (WAR) for measuring the ASR performance is defined as

$$A = [H - (I+S+D) / H] * 100\ \%,$$ where

H is the total number of words,
I is the total number of Insertions
S is the total number of Substitutions, and
D is the total number of Deletions

The Word Error Rate (WER) is defined as

$$WER = (100 - A)\ \%$$

Word accuracy rate is defined as the ratio of correct recognized words to the total no of words
WAR=correct recognized words/total no of words

Word error rate is defined as the ratio of wrong recognized words to the total no of words
WER=wrong recognized words/total no of words

4.6 Performance Evaluation of ASR with Narrowband Codecs without channel noise and channel coding

The results reported in the literature are verified by rerunning the evaluation tests to study the performance of ASR due to Narrow Band speech codecs used in GSM. The following Table 4-1 gives the ASR accuracy obtained with GSM Narrow Band speech coded data, with the models trained using the un-coded narrowband speech data.

Table 4-1: Performance of ASR only with Narrow Band GSM Speech Codecs

GSM Speech Codec	ASR Accuracy (%)
Without Speech Codec	93.47
Full Rate	93.50
Half Rate	87.88
Enhanced Full Rate	93.57
AMR-NB-MR122	93.47
AMR-NB-MR102	93.57
AMR-NB-MR795	93.18
AMR-NB-MR74	93.44
AMR-NB-MR67	93.42
AMR-NB-MR59	93.31

AMR-NB-MR515	93.26
AMR-NB-MR475	93.22

4.7 Performance of ASR at different channel conditions using GSM speech and channel coding standards

4.7.1 GSM – FR, HR, EFR speech and channel coding standards

The testing speech data is encoded using GSM speech and channel coding standards (FR, HR, and EFR). Then noise is added to the encoded data to model/simulate the channel conditions. Later the encoded data with channel noise is decoded using same standards to obtain the speech data. This encoded-decoded speech data with different SNR levels is used for testing the accuracy of ASR with different channel conditions. The ASR models are obtained in this testing, using normal narrowband speech data without any speech or channel coding. The BER is computed between encoded speech bitstream before channel coding and decoded bitstream after channel simulation under different channel conditions. MoS is also computed between original narrowband speech and coded (Speech and channel) speech. The results are given in Table 4.2 and Figures 4.3 to 4.12.

Various tests are carried out to find out the accuracy of ASR, MoS and BER with different channel noise conditions and the results obtained are shown in Table 4.1

Table 4-2: ASR accuracy, MoS and BER for different Channel Noise conditions using FR, HR, and EFR Codecs in GSM,

Variance	SNR In dB	ASR Performance in (%)			MoS (Max=5)			Bit Error Rate		
		FR	HR	EFR	FR	HR	EFR	FR	HR	EFR
0.0001	36.99	94	87.6	92.9	3.45	2.15	3.85	0	0	0
0.0002	33.97	94	87.6	92.9	3.45	2.15	3.85	0	2.70×10^{-6}	7.20×10^{-7}
0.0003	32.21	93.9	87.31	92.8	3.4	2.14	3.84	3.76×10^{-5}	1.40×10^{-4}	3.70×10^{-5}
0.0004	30.96	93	86.63	92.9	3.16	2.11	3.81	1.94×10^{-4}	6.98×10^{-4}	1.98×10^{-4}
0.0005	30.00	92.1	83.97	93.2	2.85	2.02	3.75	4.92×10^{-4}	1.85×10^{-3}	5.23×10^{-4}
0.0006	29.20	89.4	81.35	92.8	2.47	1.93	3.68	1.02×10^{-3}	3.82×10^{-3}	1.02×10^{-3}
0.0007	28.53	86.6	77.99	92.6	2.16	1.8	3.57	1.66×10^{-3}	6.20×10^{-3}	1.71×10^{-3}
0.0008	27.95	83	71.64	92.2	1.89	1.65	3.46	2.52×10^{-3}	9.54×10^{-3}	2.55×10^{-3}
0.0009	27.44	76.6	66.1	91.8	1.67	1.49	3.34	3.50×10^{-3}	1.31×10^{-2}	3.55×10^{-3}
0.0010	26.98	70.9	59.51	91.3	1.49	1.36	3.19	4.71×10^{-3}	1.74×10^{-2}	4.77×10^{-3}
0.0011	26.57	61.8	51.34	90.2	1.33	1.22	3.03	6.14×10^{-3}	2.19×10^{-2}	6.21×10^{-3}
0.0012	26.19	54.6	46.12	88.6	1.19	1.1	2.88	7.84×10^{-3}	2.79×10^{-2}	7.91×10^{-3}
0.0013	25.85	44.6	37.93	86.5	1.08	0.99	2.70	9.69×10^{-3}	3.36×10^{-2}	9.91×10^{-3}
0.0014	25.52	37.9	31.66	84	0.99	0.9	2.5	1.17×10^{-2}	4.02×10^{-2}	1.19×10^{-2}
0.0015	25.22	29	26.63	81.7	0.91	0.8	2.38	1.44×10^{-2}	4.62×10^{-2}	1.43×10^{-2}

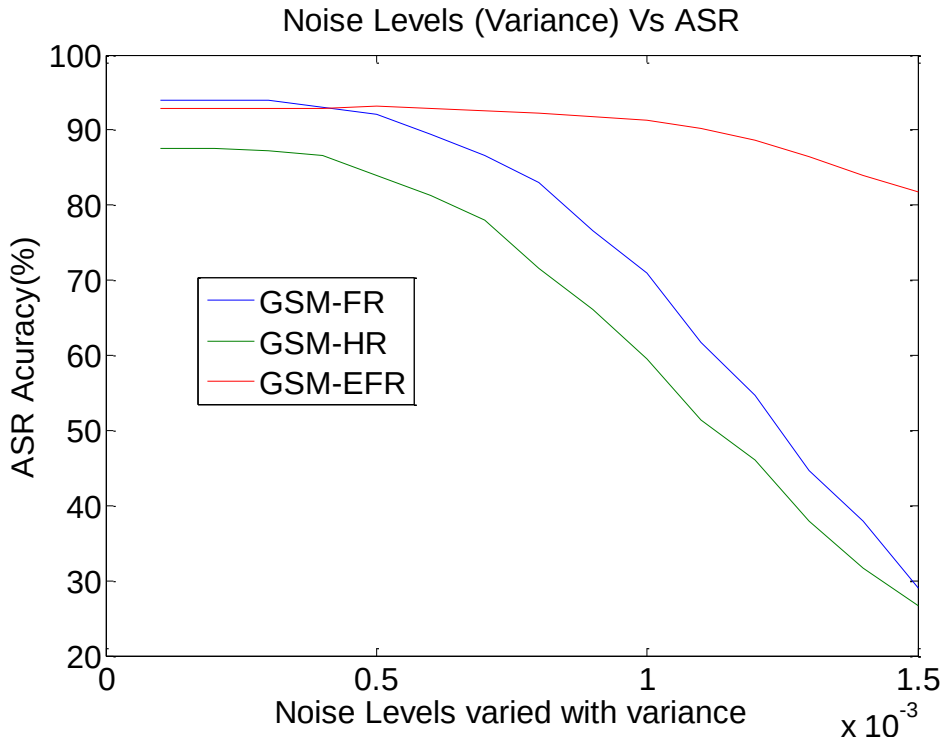

Figure 4.3: Plot between Noise Levels and ASR Accuracy for GSM - FR, HR, EFR

Analysis

ASR accuracy, MOS and Bit Error Rates are obtained by testing the speech and channel coding coded data with different channel noise conditions using the un-coded trained HMM models for 8-kHz speech. The following are the observations from the Table 4-1.

1) As the noise level is increasing in the channel, automatic speech recognition performance is not varying much in the Enhanced Full Rate, as compared to Full rate and Half rate speech and channel coding standards.

2) Although the MoS varying between 3.45 to 0.91 for FR and 2.15 to 0.8 for HR, the variation is little in EFR with same levels of noise, i.e. varies between 3.85 to 2.38 for EFR.

3) As the channel noise is increasing, Full Rate and Half Rate standards are giving exponentially decreased ASR performance.

4) From the Figure 4.3 it is observed that 3% to 6% of automatic speech recognition performance is decreased with GSM Half Rate speech and channel coding standard with respect to GSM Full Rate, even though half rate utilizing half the bandwidth of full rate.

5) Full rate and half rate codecs are produced more than 85% ASR performance at MoS is greater than 2. However if the MoS is less than 2, the ASR accuracy is decreasing drastically.

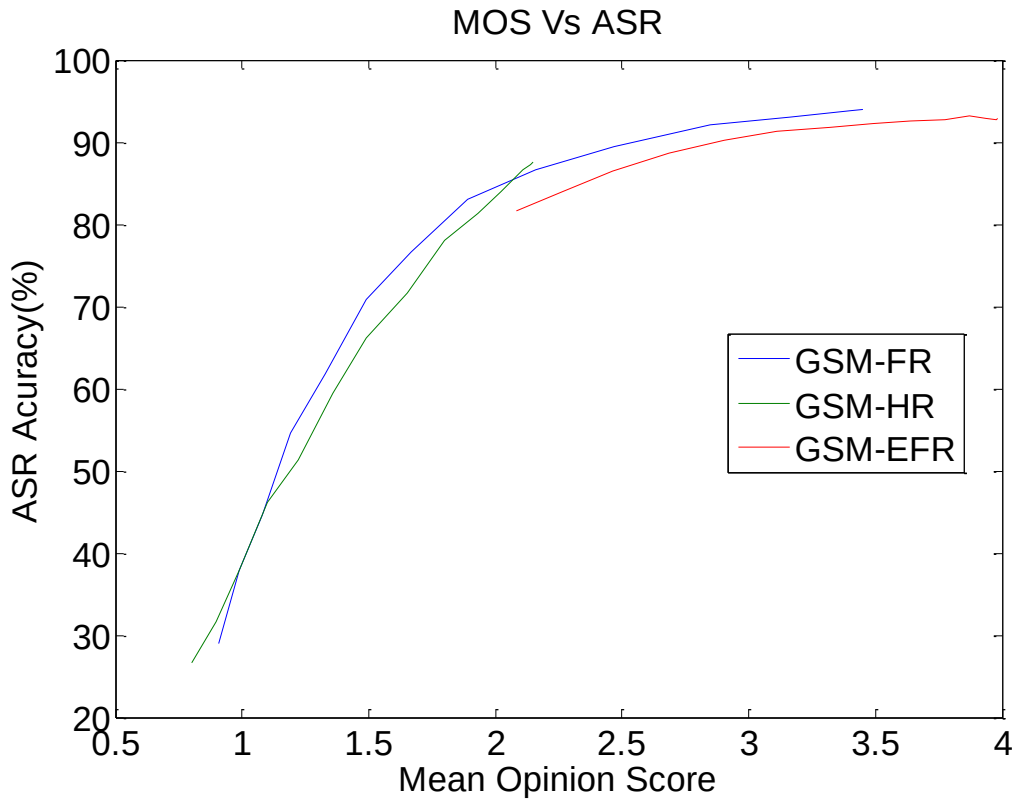

Figure 4.4: Plot between Mean Opinion Score and Automatic Speech Recognition for GSM - FR, HR, EFR

6) From the figure 4.5 it is observed that the Bit Error Rate, due to the communication channel is upto 10^{-3} ASR performance is almost constant. After that ASR is decreasing drastically.

7) From the figure 4.6 ASR accuracy (88% and 82%) is almost constant up to 29 dB and decreasing drastically with decrease of SNR for Full Rate and Half Rate. In the case of Enhanced full Rate, even if the SNR is 25dB, ASR accuracy is greater than 90%.

8) ASR accuracy is changed up 5 to 7% after the 24 dB in case of Enhanced full rate GSM Speech codec.

9) It is observed that MoS is decreasing very fast as compared to ASR accuracy with respect to increase in noise.

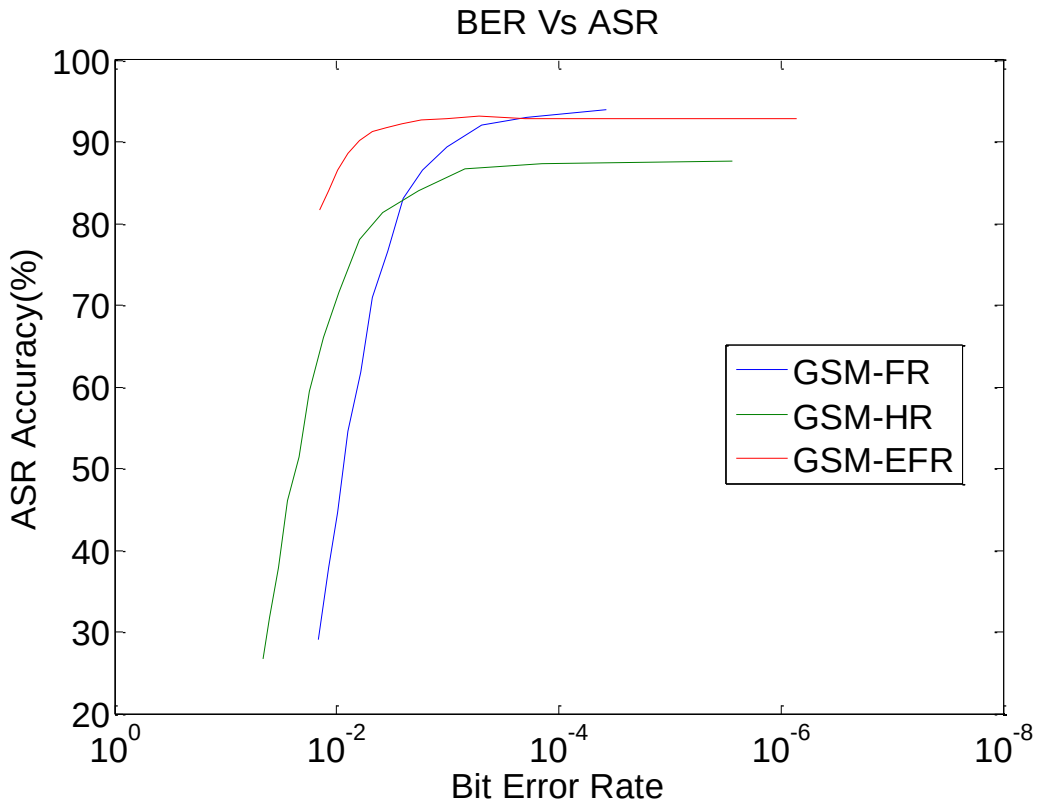

Figure 4.5: Plot between Bit Error Rate and Automatic Speech Recognition for GSM - FR, HR, EFR codecs

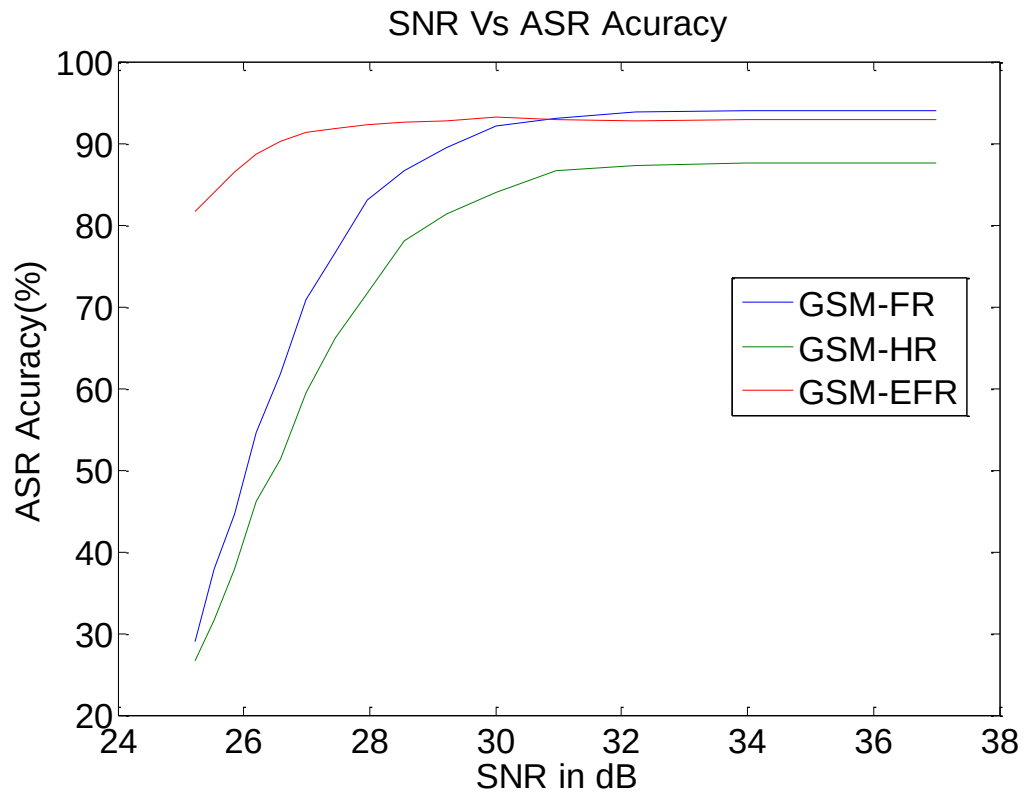

Figure 4.6: Plot Between Signal to Noise Ratio and Automatic Speech Recognition for GSM - FR, HR, EFR codecs

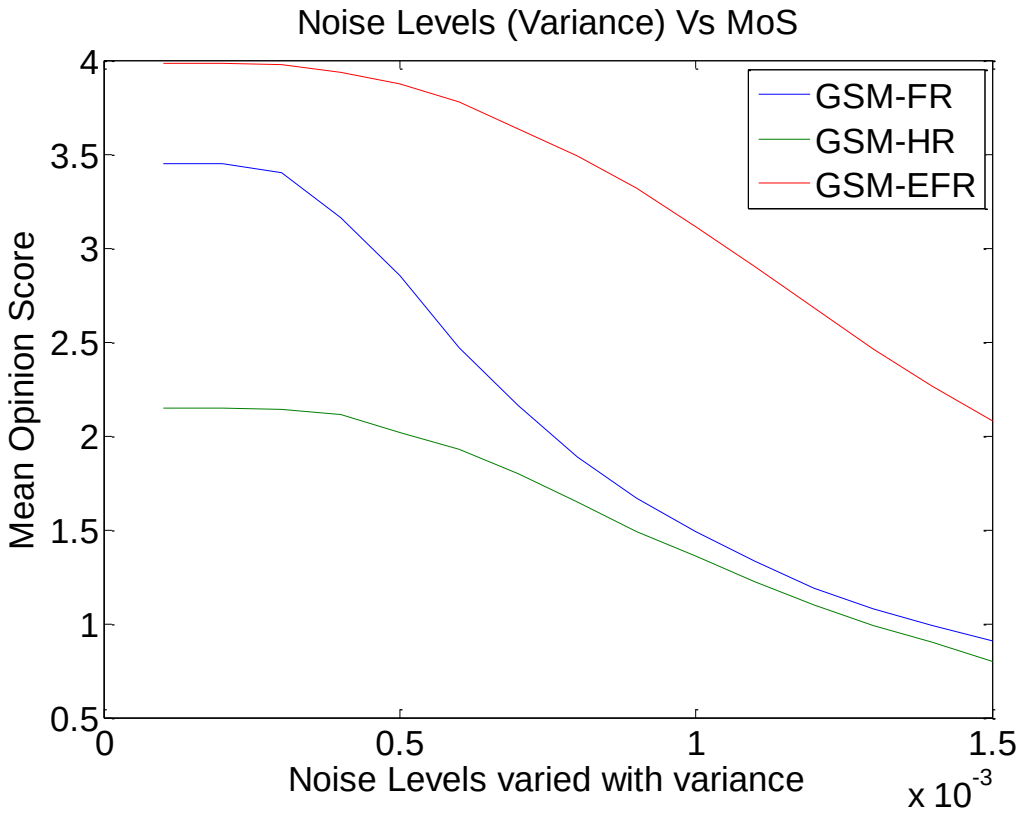

Figure 4.7: Plot between Noise Levels and MoS for GSM - FR, HR, EFR codecs

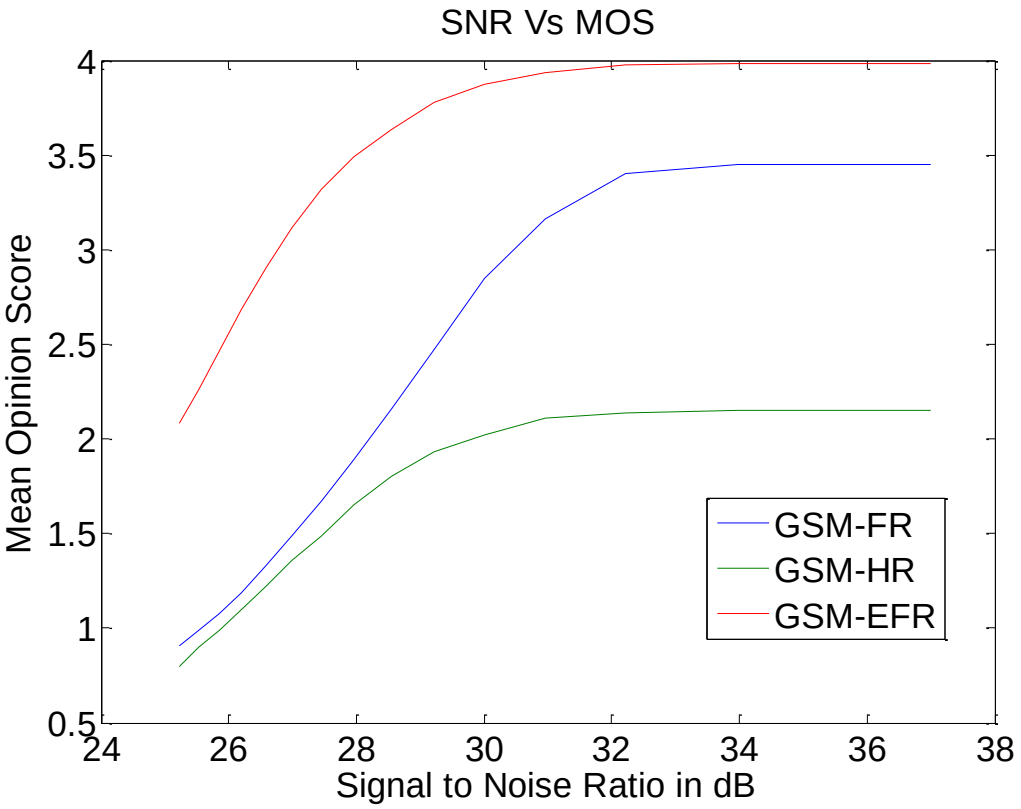

Figure 4.8: Plot between Signal to Noise and Ratio Mean Opinion Score for GSM - FR, HR, EFR codecs

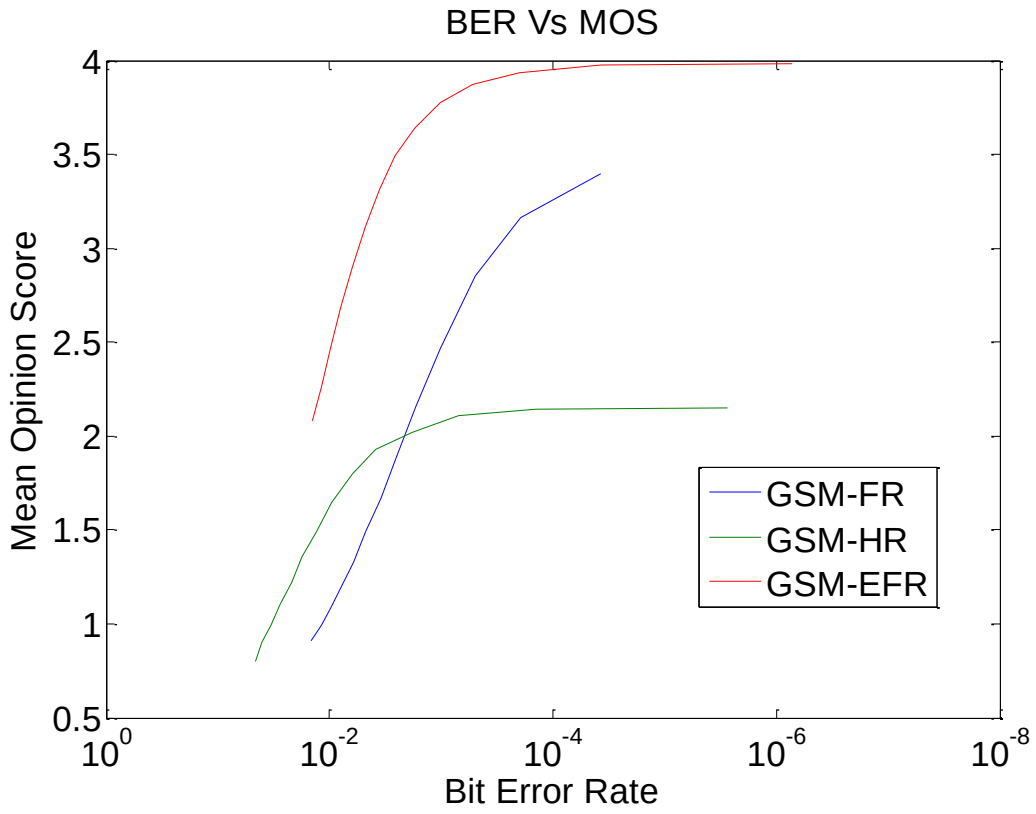

Figure 4.9: Plot between Bit Error Rate and Mean Opinion Score for GSM - FR, HR, EFR codecs

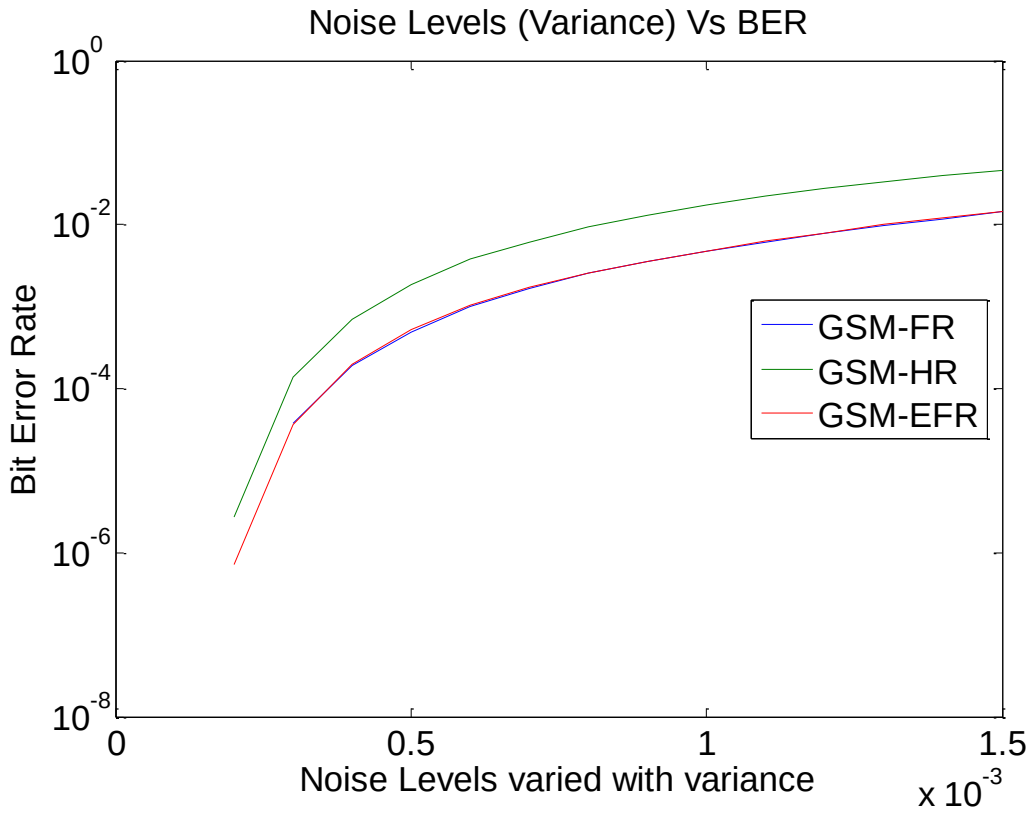

Figure 4.10: Plot between Noise Levels and Bit Error Rate for GSM - FR, HR, EFR codecs

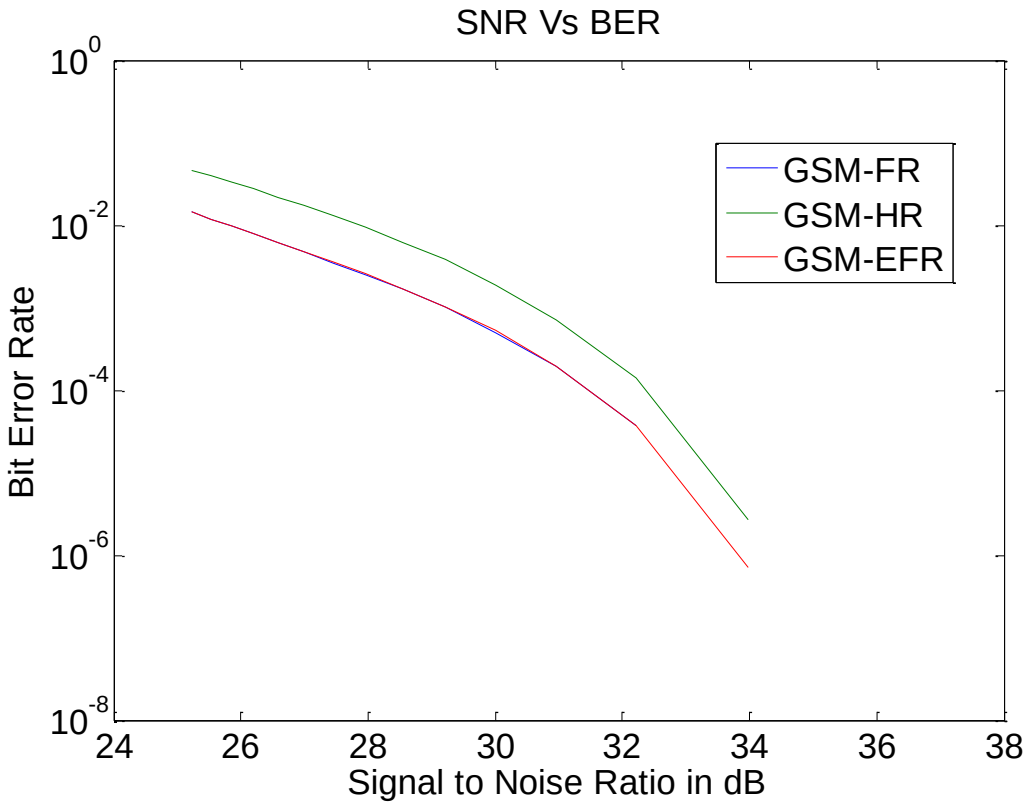

Figure 4.11: Plot between Signal to Noise Ratio and Bit Error Rate for GSM - FR, HR, EFR codecs

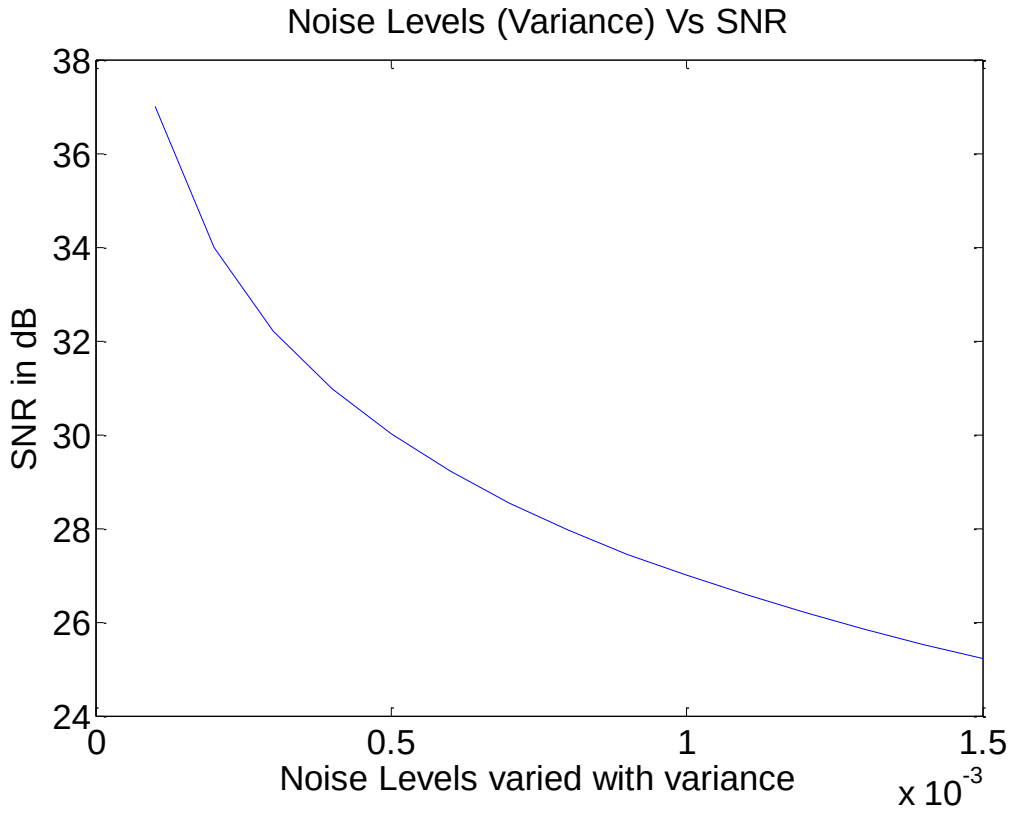

Figure 4.12: Plot between Noise Levels and Signal to Noise Ratio for GSM - FR, HR, EFR codecs

4.7.2 GSM – AMR speech and channel coding standards

Table 4-3: Automatic Speech Recognition for different Channel Noise conditions using GSM AMR-NB codecs

Variance	SNR	Automatic Speech Recognition (%)							
		MR122	MR102	MR795	MR74	MR67	MR59	MR515	MR475
0.0001	36.99	91.93	82.48	90.21	92.74	90.84	92.6	92.66	92.72
0.0002	33.98	91.93	82.48	90.21	92.74	90.84	92.6	92.66	92.72
0.0003	32.22	92.1	81.77	90.21	92.75	90.76	92.59	92.71	92.72
0.0004	30.97	92.14	81.51	90.24	92.73	90.76	92.58	92.7	92.71
0.0005	30	91.99	79.74	90.01	92.69	90.72	92.51	92.68	92.74
0.0006	29.21	91.8	77.18	89.19	92.94	90.51	92.47	92.6	92.63
0.0007	28.54	91.06	72.57	88.47	92.79	89.37	92.35	92.63	92.57
0.0008	27.96	90.73	67.32	87.91	92.71	89.18	92.19	92.37	92.59
0.0009	27.45	89.38	62.11	87.05	92.58	88.46	92.08	92.47	92.54
0.001	26.99	88.43	54.41	85.75	92.5	87.67	91.73	92.14	92.27
0.0011	26.58	86.12	46.05	84.63	92.25	86.51	91.36	91.65	91.97
0.0012	26.2	83.51	32.43	81.33	92.08	83.68	90.96	91.26	91.68
0.0013	25.85	82.94	28.9	80.07	92.08	82.78	90.82	91.14	91.52
0.0014	25.53	72.51	23.2	76.57	90.98	80.19	89.4	89.97	90.63
0.0015	25.23	70.04	19.4	74.6	90.82	79.09	89.24	89.82	90.36
0.0016	24.95	59.04	13.5	66.31	88.8	74.8	85.51	87.22	88.61
0.0017	24.68	54.4	11.2	63.39	88.12	72.49	84.89	86.27	88.17
0.0018	24.44	48.6	10.34	59.81	86.34	70.39	83.31	84.58	87.29

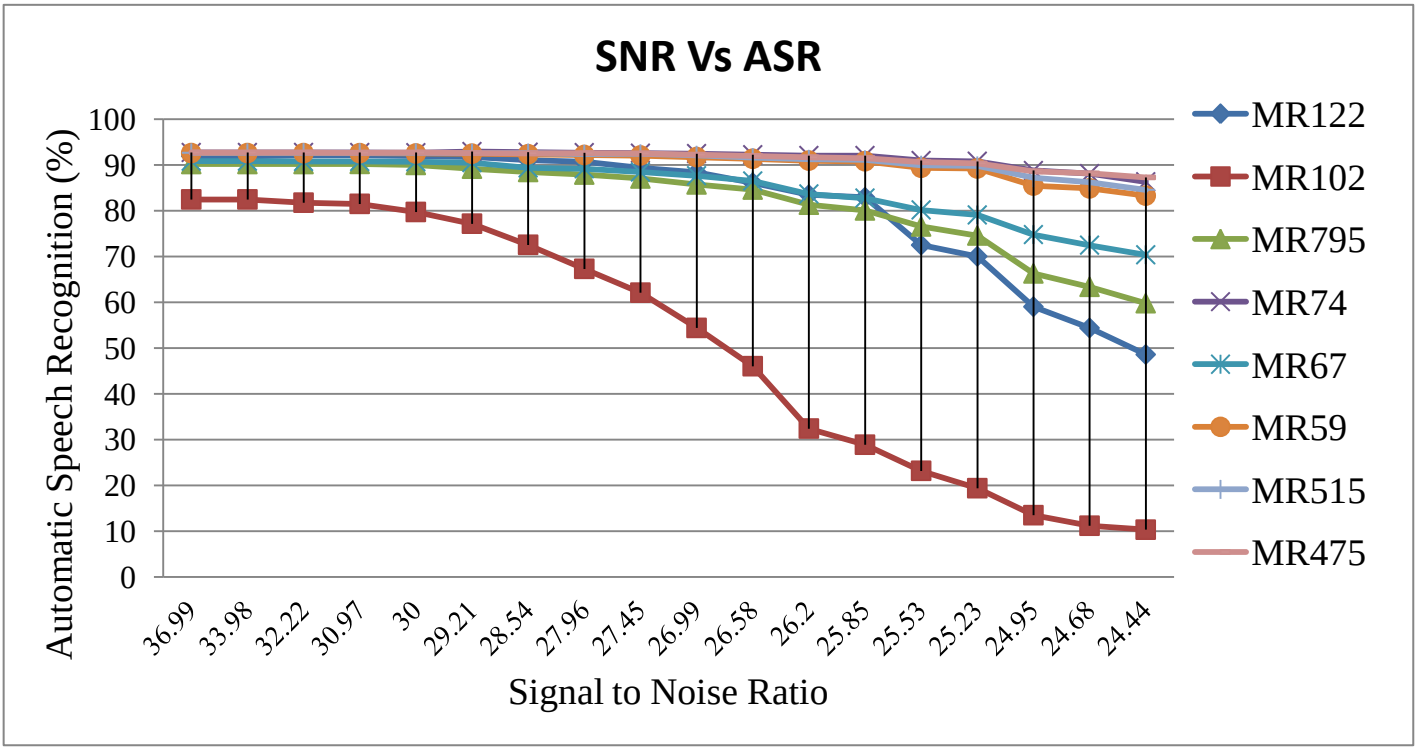

Figure 4.13: Plot between Signal to Noise Ratio and Automatic speech Recognition for AMR- NB codecs

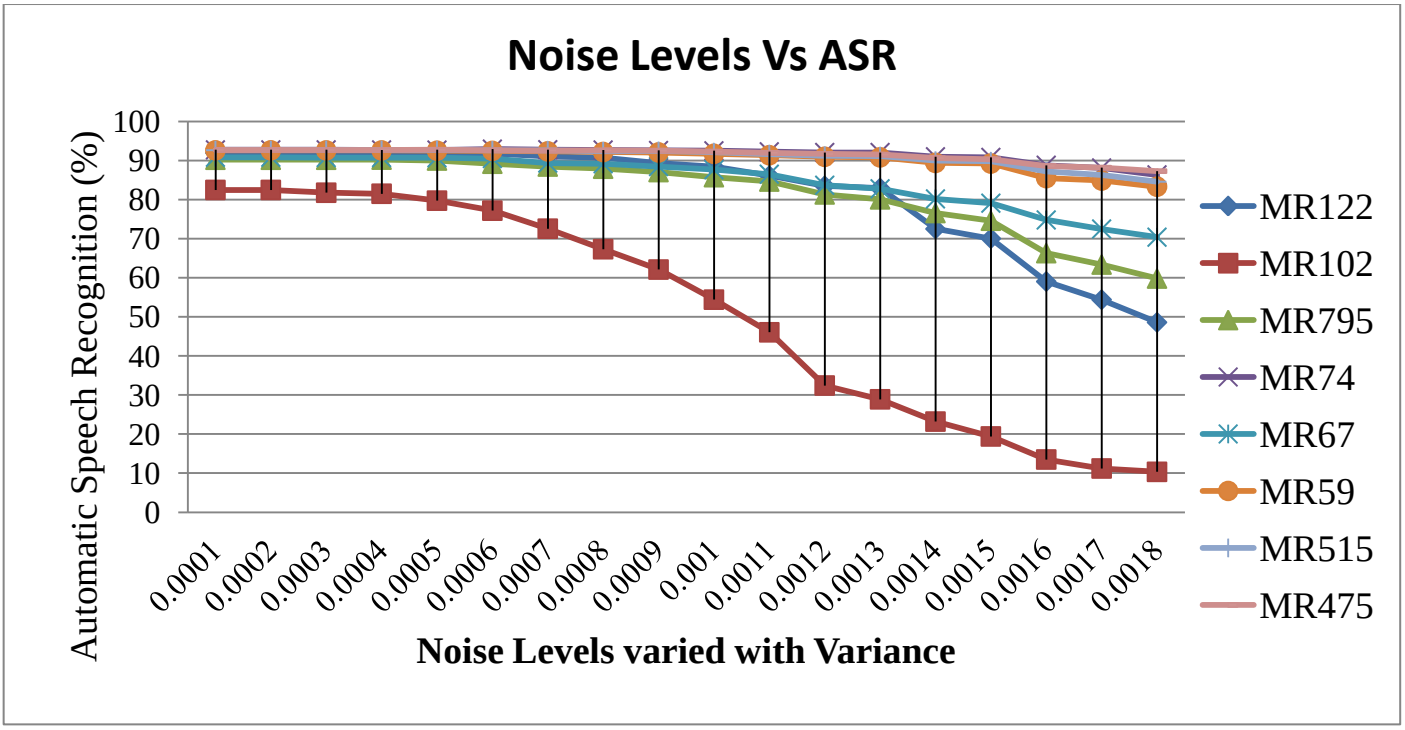

Figure 4.14: Plot between Noise Levels and Automatic Speech Recognition for different channel conditions using GSM AMR-NB codecs

Analysis:

Here the reported results of Automatic Speech Recognition are obtained under different channel conditions (different Signal to Noise Ratios) by testing the Adaptive Multi Rate Narrow band coded speech data with the trained models of un-coded speech data. In coding of speech data standard speech and channel coding algorithms are used which are specified by the ETSI and 3GPP. A Sphinx Speech Recognition tool kit is used for estimating the Automatic Speech Recognition performance with TIMIT speech database. From the obtained results the following observations are made:

1) In the Full Rate Channel, with Adaptive Multi Rate speech codec with modes MR74, MR59, MR515 and MR475, are producing automatic speech recognition performance of 83.31% to 87.29% at SNR of 24dB and 92.66% to 92.72% at SNR of 36dB. So the variation is large with low SNR as compared to high SNR.
2) MR122, MR102, MR795 and MR67 codec modes are highly varied Automatic speech recognition performance with decrease in signal to noise ratio when compared with the other codec modes.
3) MR102 is producing very bad ASR performance with the increase in noise when comparing with the other modes.
4) Enhanced full rate speech coding standard is used as one of the modes in adaptive multi rate speech codec, but due to the channel coding and puncturing in MR122 automatic speech recognition is reduced when comparing with EFR codec all levels of SNR.

Table 4-4: Mean Opinion Score for different Channel Noise conditions using GSM AMR-NB codecs

Variance	SNR	Mean Opinion Score							
		MR122	MR102	MR795	MR74	MR67	MR59	MR515	MR475
0.0001	36.99	2.72	2.52	2.94	3.66	3.13	3.51	3.41	3.35
0.0002	33.98	2.72	2.52	2.94	3.66	3.13	3.51	3.41	3.35

0.0003	32.22	2.71	2.48	2.93	3.66	3.11	3.51	3.41	3.35
0.0004	30.97	2.71	2.47	2.92	3.66	3.11	3.51	3.41	3.35
0.0005	30	2.71	2.42	2.9	3.66	3.09	3.5	3.41	3.35
0.0006	29.21	2.69	2.36	2.86	3.65	3.07	3.49	3.39	3.34
0.0007	28.54	2.63	2.24	2.81	3.63	3.02	3.48	3.38	3.32
0.0008	27.96	2.6	2.12	2.73	3.62	2.97	3.46	3.35	3.31
0.0009	27.45	2.53	2.02	2.66	3.6	2.92	3.43	3.32	3.29
0.001	26.99	2.45	1.89	2.58	3.57	2.86	3.38	3.27	3.26
0.0011	26.58	2.31	1.79	2.49	3.48	2.79	3.31	3.19	3.22
0.0012	26.2	2.18	1.62	2.35	3.39	2.69	3.24	3.09	3.17
0.0013	25.85	2.14	1.57	2.31	3.34	2.65	3.21	3.06	3.15
0.0014	25.53	1.87	1.46	2.16	3.13	2.56	3.04	2.91	3.07
0.0015	25.23	1.81	1.41	2.13	3.1	2.52	3.01	2.88	3.05
0.0016	24.95	1.61	1.31	1.91	2.82	2.38	2.69	2.66	2.89
0.0017	24.68	1.57	1.25	1.84	2.76	2.33	2.63	2.59	2.85
0.0018	24.44	1.45	1.22	1.75	2.62	2.29	2.52	2.51	2.8

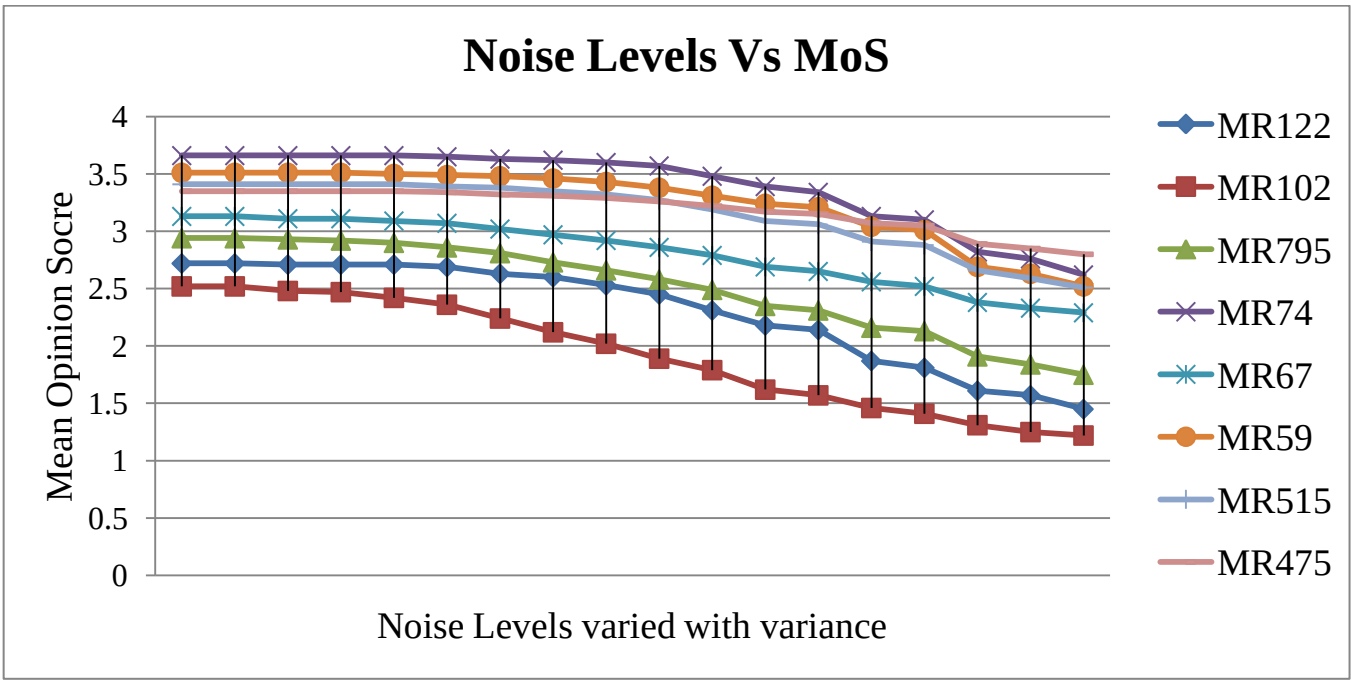

Figure 4.15: Plot between Noise Levels and Mean Opinion Score for different channel noise using GSM AMR-NB codecs

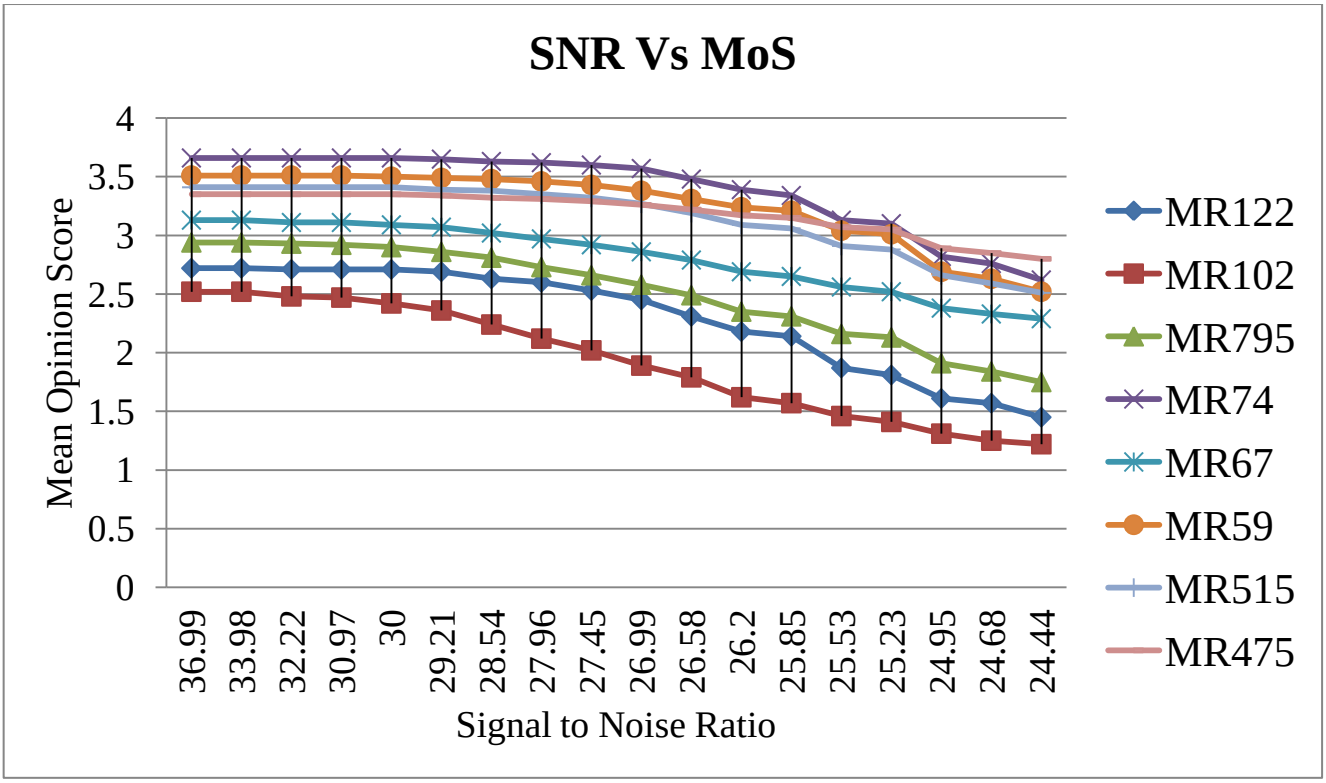

Figure 4.16: Plot between Signal to Noise Ratio and Mean Opinion Score for different channel noise conditions using GSM AMR-NB Codecs

Analysis:

Mean Opinion Scores are estimated with standard p.862 (PESQ) by comparing the speech samples of AMR narrow band with different modes coded with original speech samples. TIMIT speech database is used for this purpose. From the obtained results the following observations are made:

1) Low bit rate modes of AMR NB codec are performing with high MoS values. This means low bit rate modes are having with efficient channel coding
2) Because of the puncturing high bit rate modes are losing most of the information bits so they results in poor quality in the presence of noise.

Table 4-5: Bit Error Rates for different Channel Noise conditions using GSM AMR-NB codecs

Variance	**SNR**	**Bit Error Rate**							
		MR122	**MR102**	**MR795**	**MR74**	**MR67**	**MR59**	**MR515**	**MR475**
0.0001	36.99	0.03557	0.02893	0.00487	0.00075	0.00884	0.00017	0	0.00028
0.0002	33.98	0.03557	0.02893	0.00487	0.00075	0.00884	0.00017	0	0.00028
0.0003	32.22	0.03578	0.02999	0.00512	0.00079	0.0091	0.00018	0.0000189	0.00029
0.0004	30.97	0.03583	0.03019	0.00517	0.0008	0.00915	0.00018	0.000024	0.00029
0.0005	30	0.03621	0.03169	0.00556	0.00086	0.00956	0.00019	0.0000559	0.0003
0.0006	29.21	0.03683	0.03386	0.00616	0.00098	0.01021	0.00022	0.000129	0.00033
0.0007	28.54	0.03801	0.03789	0.00729	0.00125	0.01143	0.00029	0.000294	0.00039
0.0008	27.96	0.03908	0.04218	0.00849	0.00147	0.0127	0.00034	0.00045	0.00044
0.0009	27.45	0.04042	0.04567	0.00964	0.00173	0.01393	0.00041	0.000639	0.00051
0.001	26.99	0.04243	0.05151	0.01143	0.00217	0.01582	0.00057	0.000932	0.00064
0.0011	26.58	0.04495	0.05724	0.01359	0.00296	0.01805	0.00093	0.00154	0.00089

0.0012	26.2	0.04871	0.06855	0.01757	0.00426	0.02196	0.00147	0.00259	0.00122
0.0013	25.85	0.05011	0.07227	0.01887	0.00475	0.02333	0.00167	0.00296	0.00135
0.0014	25.53	0.05531	0.07901	0.02281	0.00695	0.02714	0.00316	0.00486	0.00211
0.0015	25.23	0.05714	0.08242	0.02408	0.00737	0.02854	0.0033	0.00519	0.00222
0.0016	24.95	0.0632	0.09045	0.02952	0.01106	0.03358	0.00677	0.00854	0.0041
0.0017	24.68	0.06576	0.09629	0.03228	0.0124	0.03617	0.00764	0.00981	0.00457
0.0018	24.44	0.06917	0.09966	0.03489	0.0143	0.03839	0.00911	0.0112	0.00527

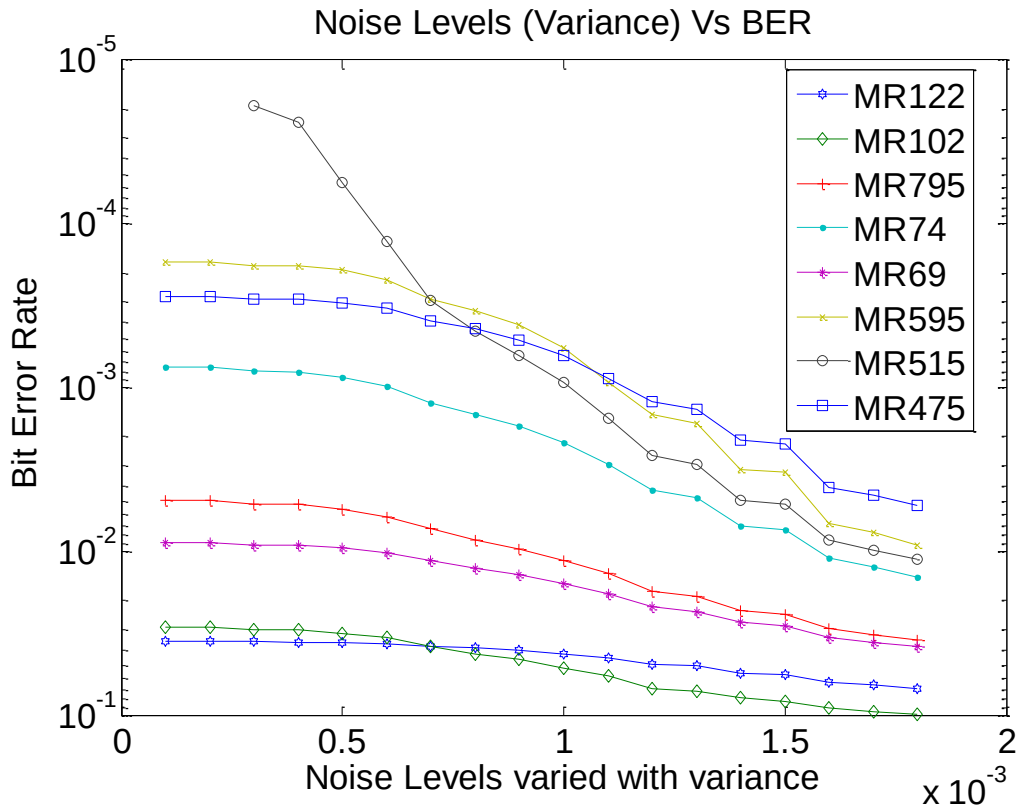

Figure 4.17: Plot between Noise levels and Bit Error Rate for different channel noise conditions using GSM AMR-NB Codecs

Analysis:

Bit Error Rates are calculated for each and every frame in the speech file by comparing the before and after the channel coding, because the bit errors will occur due to channel noise. From the obtained results following observations were made:

1) In Adaptive multi rate MR475, MR74, MR595, MR515 are very resistive to the bit errors in channel due to the channel coding mechanism in those modes.
2) Rests of the modes are mainly affected by the puncturing mechanism in the channel coding process, because they coded with low rate Recursive convolutional codes. So puncturing is forced to remove the information bits.
3) Automatic speech recognition performance is constant upto the BER is 10^{-2}. After this ASR performance is decreasing drastically.
4) After the BER 10^{-2} ASR is very sensitive to bit errors.

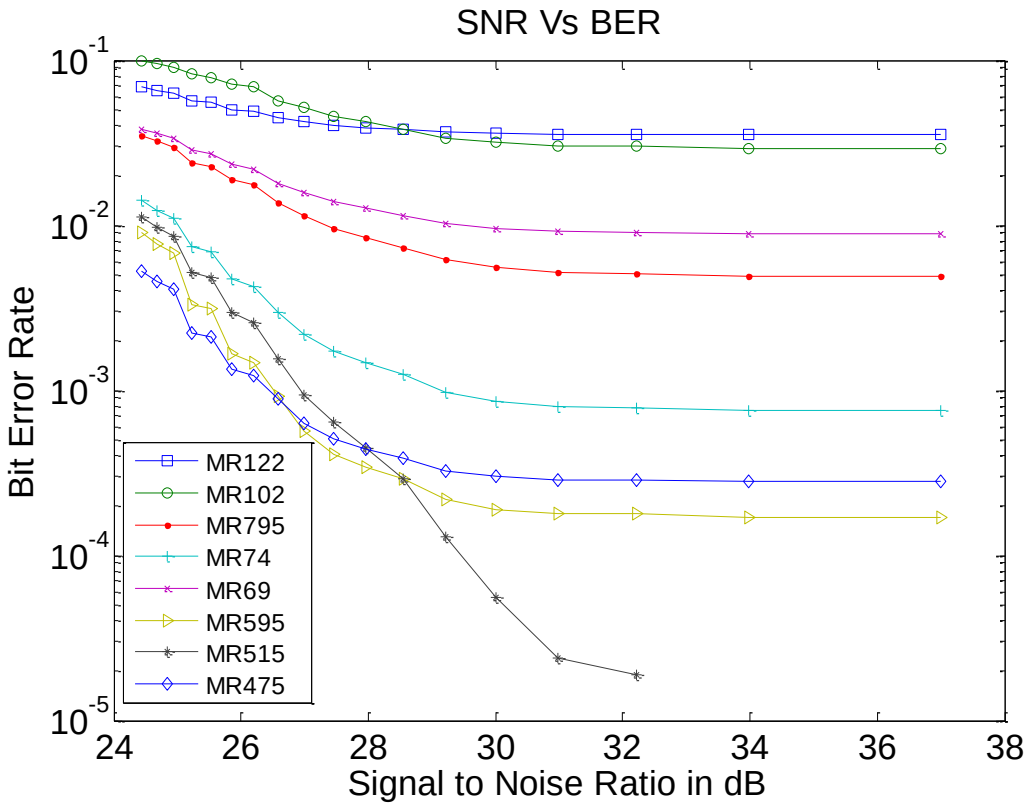

Figure 4.18: Plot between Signal to Noise Ratio and Bit Error Rate for different channel noise conditions using GSM AMR-NB Codecs

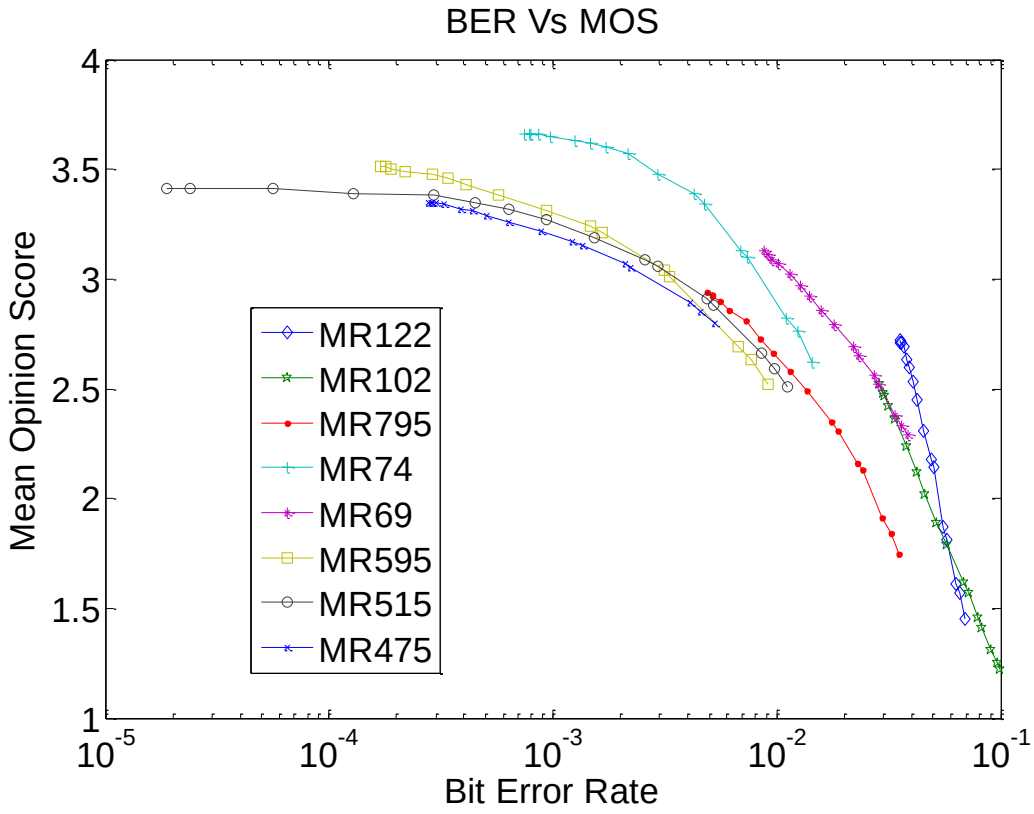

Figure 4.19: Plot between Bit Error Rate and Mean Opinion Score for different channel noise conditions using GSM AMR-NB Codecs

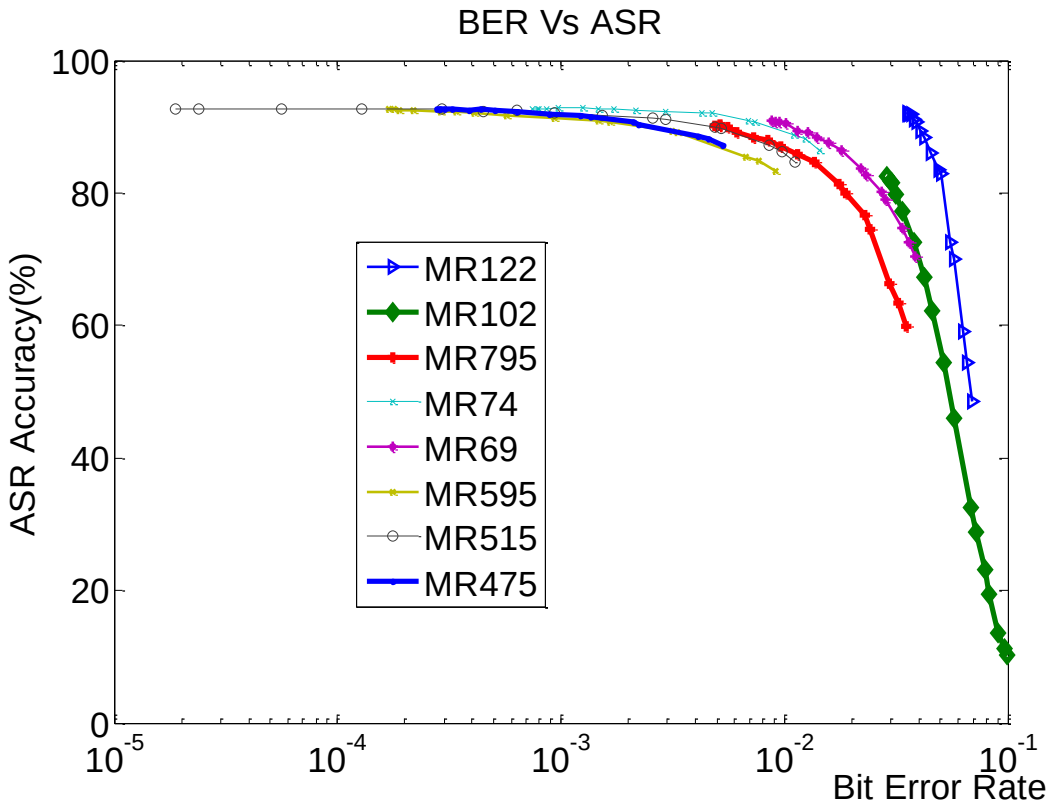

Figure 4.20: Plot between Bit Error Rate and Automatic Speech Recognition for different channel noises using GSM AMR-NB codecs

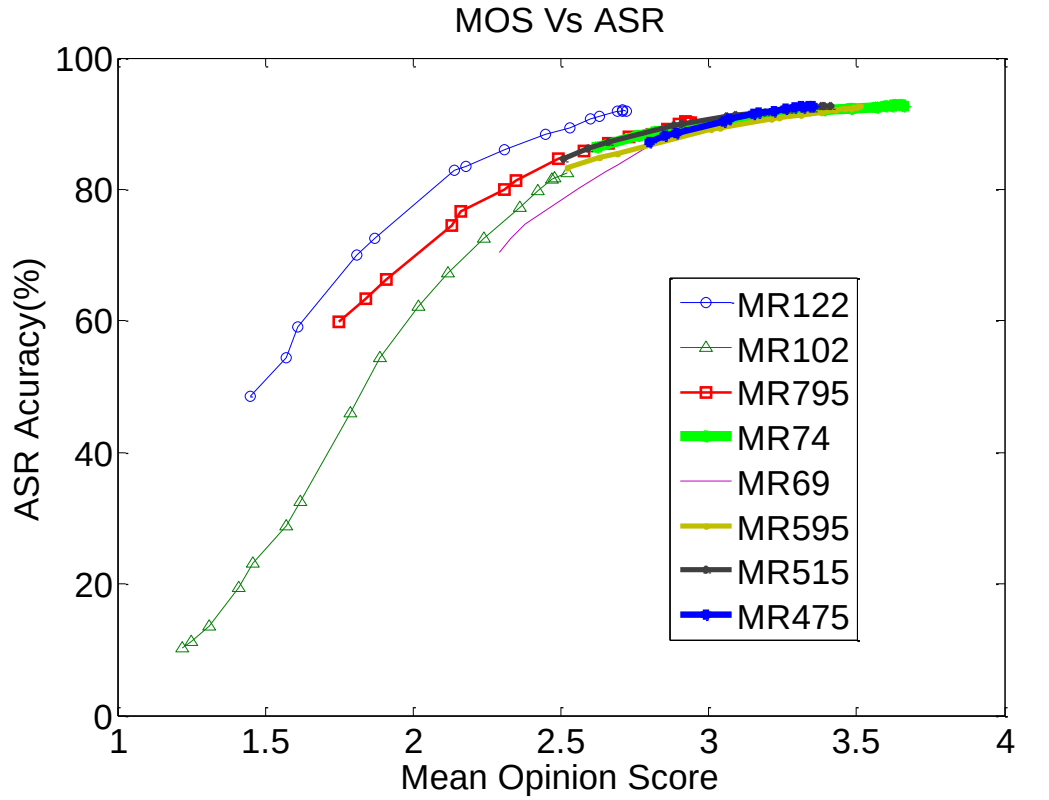

Figure 4.21: Plot between Mean Opinion Score and Automatic Speech Recognition for different channel noises using GSM AMR-NB codecs

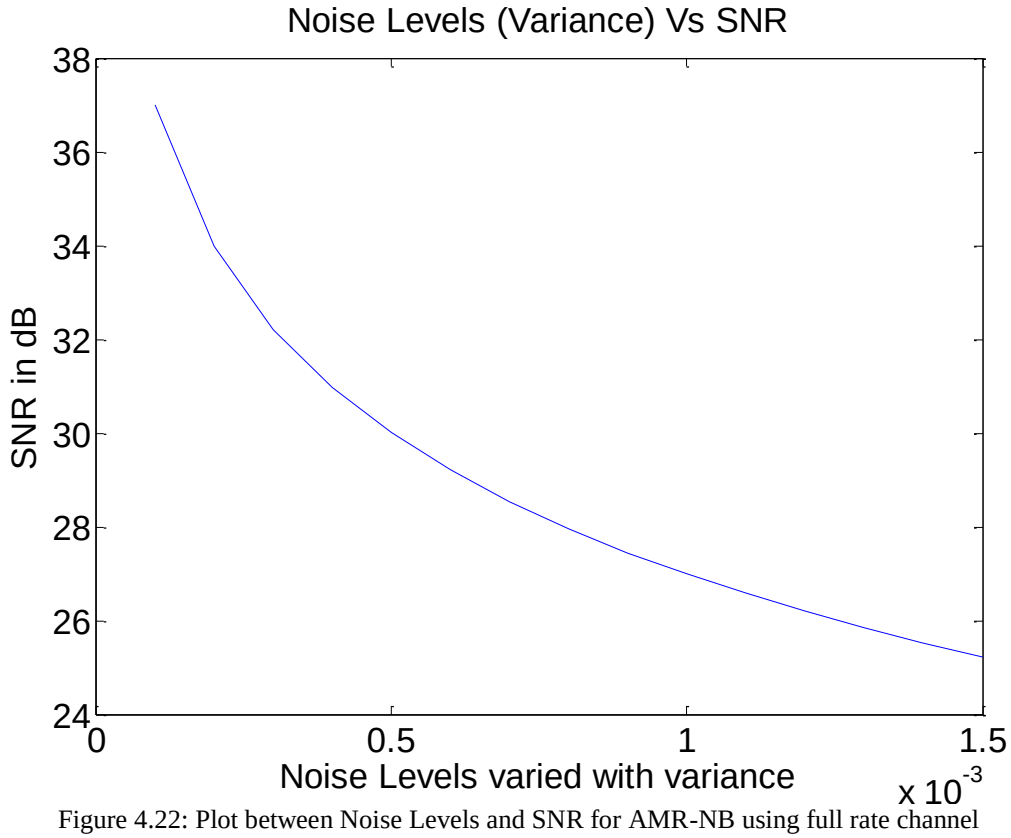

Figure 4.22: Plot between Noise Levels and SNR for AMR-NB using full rate channel

4.8 Conclusions

Various tests have been performed to find the ASR accuracy, Mean Opinion Score and Bit Error Rates for GSM Narrow Band Speech codec using standard speech and channel coding algorithms specified by the ETSI and 3GPP. Original speech data is used to train the ASR models and as reference files to find the results. Low bit rate codec modes from AMR Narrow band codec are producing good results in the presence of noise due to channel coding mechanisms. High bit rate modes are unable compete with the Enhanced full rate speech codec due to puncturing.

5 Conclusions

The performance of Automatic Speech Recognition over GSM Networks for Different Channel Conditions with Narrowband Speech codecs, are carried out using simulations and the results obtained are analyzed and given in this report. Open source ASR tool (Sphnix-3), TIMIT Speech database of LDC and ETSI,ITU-T and 3GPP standards are taken for these studies.

The testing speech data is encoded using GSM speech and channel coding standards. Then noise is added to the encoded data to model/simulate the channel conditions. Later the encoded data with channel noise is decoded using decoding standards to get back the speech data. This encoded-decoded speech data with different SNR levels in the channel coding, is used for testing the accuracy of ASR with different channel conditions. The ASR models are obtained in this testing, using normal narrowband speech data without any speech or channel coding. The BER is computed between encoded speech bit-stream before channel coding and decoded bit-stream after channel simulation under different channel conditions. MoS is also computed between original narrowband speech and received speech after speech and channel coding.

ASR accuracy, MOS and Bit Error Rates are obtained for different channel noise conditions with different speech and channel coding standards, using the un-coded trained HMM models for 8-kHz speech. The following are the observations are made from the obtained results.

The performance of ASR for FR, HR and EFR

1) As the noise level is increasing in the channel, automatic speech recognition performance is not varying much in the Enhanced Full Rate, as compared to Full rate and Half rate speech and channel coding standards.
2) Although the MoS varying between 3.45 to 0.91 for FR and 2.15 to 0.8 for HR, the variation is little in EFR with same levels of noise, i.e. varies between 3.85 to 2.38 for EFR.
3) As the channel noise is increasing, Full Rate and Half Rate standards are giving exponentially decreased ASR performance.
4) It is observed that 3% to 6% of automatic speech recognition performance is decreased with GSM Half Rate speech and channel coding standard with respect to GSM Full Rate, even though half rate utilizing half the bandwidth of the full rate standard.
5) Full rate and half rate codecs are producing more than an ASR accuracy of 85% at MoS greater than 2. However if the MoS is less than 2, the ASR accuracy is decreasing drastically.
6) It is observed that the performance of ASR is not effected till Bit Error Rate is up to 10^{-3}, due to the channel noise. However if the BER is greater than 10^{-3}, ASR accuracy is decreasing drastically.
7) The ASR accuracy is almost constant (88% and 82%) up to 29 dB and decreasing drastically with further decrease of SNR for Full Rate and Half Rate.
8) In the case of Enhanced full Rate, even if the SNR is 25dB, ASR accuracy is greater than 90%. ASR accuracy is decreased by 5 to 7% after 24 dB of SNR.
9) It is observed that MoS is decreasing very fast as compared to ASR accuracy with respect to increase in noise for all standards.

The performance of ASR for FR, HR and EFR

1) In the Full Rate Channel, with Adaptive Multi Rate speech codec with modes MR74, MR59, MR515 and MR475, are producing ASR accuracy between 83.31% to 87.29% at SNR of 24dB and 92.66% to 92.72% at SNR of 36dB. So the variation is large with low SNR as compared to high SNR between different rates of AMR standard.
2) The performance ASR with the increase in noise is very bad with MR102 when comparing with the other modes. This may attributed to more puncturing in channel coding, to adjust the number of bits to fit in to Full Rate frame length.
3) Enhanced full rate speech coding standard is used as one of the modes in adaptive multi rate speech codec, but due to the channel coding and puncturing in MR122 automatic speech recognition is reduced when comparing with EFR codec at all levels of SNR.
4) Low bit rate modes of AMR NB codec are performing with high MoS values. This means low bit rate modes are having with efficient channel coding
5) Because of the puncturing, high bit rate modes are losing most of the information bits. So the MoS is very poor in the presence of noise.
5) In Adaptive multi rate MR475, MR74, MR595, MR515 are very resistive to the bit errors in channel due to the channel coding mechanism in those modes. Rest of the modes (MR122, and MR102) is mainly affected by the puncturing mechanism in the channel coding process, because they coded with low rate Recursive Convolutional codes. So puncturing is forced to remove the information bits.
6) The performance is almost constant up to the BER of 10^{-2}. ASR accuracy is very much sensitive to bit errors more than BER 10^{-2}.

In the presence of low channel noise, GSM-FR and EFR speech codecs are preferable and in the presence of high channel noise GSM-AMR is preferable due to channel coding mechanism present in it.

6 References

1. Arne Norre Ekstrøm & Jan H. Mikkelsen, "A MATLAB Implementation of a GSM Simulation Platform", Institute of Electronic Systems, Division of Telecommunications, Aalborg University, December, 1997
2. "Digital cellular telecommunications system (Phase 2+)", Channel coding (GSM 05.03), ETSI August 1996
3. A.V. Ramana on "Effects of Speech Coding on the Performance of Automatic Speech Recognition over Wired and Wireless Networks", Ph.D. Thesis, NERTU, OU, August-2012
4. ITU-T Recommandations, http://www.itu.int
5. ETSI Standards, http://www.etsi.org
6. 3GPP standards, http://3gpp.org
7. Rabiner, L.R. and Juang, B.H. (2003), Fundamentals of Speech Recognition, Pearson Education
8. Antonio M.Peinado and Jose C.Segura "Speech Recognition over Digital Channels robustness and standards"
9. P.Thirumal Reddy, P.Laxminarayana and A.V. Ramana, A Tutorial on "Building CMU Sphinx on Linux", Research and Training Unit for Navigational Electronics (NERTU), Osmania University
10. Lecture Notes of Short course on "Automatic Speech Recognition" held at NERTU, Osmania University, during 7th to 11 November 2012
11. Arthur Chan, Evandro Gouvea, Rita Singh, Mosur Ravishankar, Ronald Rosenfeld, Yitao Sun, David Huggins-Daines, Mike Seltzer, (third draft)The Hieroglyphs: "Building Speech Applications Using CMU Sphinx and Related Resources", March 11, 2007.
12. A.V.Ramana, **P.Laxminarayana**, P.Mythilisharan, "**Speech Enhancement of Coded Speech for ASR**", **Under Major Revision after Review.**
13. Mythilisharan, A.V.Ramana, **P.Laxminarayana** *"***Performance of ASR over VoIP and GSM networks using HMMs Trained with Wideband and Narrowband Speech Data**", ***Communicated for possible presentation*** at International Conference on Signal Processing and Communications, IISc, Bangalore, 22-25, July 2012
14. A.V.Ramana, **P.Laxminarayana**, P.Mythilisharan, "**Investigation of ASR Recognition Performance and Mean Opinion Scores for Different Standard Speech and Audio Codecs**", **Accepted by IETE Journal of Research** and likely to appear **in March-April, 2012**
15. A.V.Ramana, **P.Laxminarayana** , Mythilisharan, *"***Investigation of Speech Coding Effects on different Speech Sounds in Automatic Speech Recognition***"*, Proceedings of the First Indo-Japan Conference on Perception and Machine Intelligence (Permin'12), 12-13, January 2012, CDAC, Kolkata**.**
16. K. Alavenu , **P. Laxminarayana,** P. Mythilisharan, K. Hariprasad, S. Alavelu Mangamma, **"Front end processing for Automatic Speech Recognition in real time on a Blackfin Processor**", Proceedings of the First International Conference on Advances in Computing And Communication, 8-10, April 2011, NIT, Hamirpur, India pp 482-487
17. A.V.Ramana, **P.Laxminarayana** , Mythilisharan, *"***Investigation of ASR Accuracy under Transcoding with Narrowband and Wideband Speech Codecs***"*, Proceedings of the First International Conference on Advances in Computing And Communication, 8-10, April 2011, NIT, Hamirpur, India pp 492-498.

18. A.V.Ramana, **P.Laxminarayana** , Mythilisharan, *"***Investigation of the Effects of Wideband Speech and Audio coding on the ASR Performance in VoIP and Wireless Networks***",* Proceedings of the National Conference on Advancements in Signal Processing and Integrated Networks, 24-25, February 2011, Amity University, Noida, India.

19. A.V.Ramana, **P.Laxminarayana** and MythiliSharan ***"A New Frequency Domain Adaptive Speech Segmentation Technique",* National Symposium on Acoustics",** National Symposium on Acoustics, (NSA-09), November 25-26, 2009 at RCI-Hyderabad.

20. A.V.Ramana, **P.Laxminarayana** and MythiliSharan, **"Effects of speech coding on ASR performance in VOIP networks**", National conference on advances in signal processing, November 20-21, 2009 at Andhra University, Vishakhapatnam.

21. A.V.Ramana, **P.Laxminarayana** and P.MythiliSharan, "**Real-Time Continuous Speech Recognition with HMM based Word Models for Telugu***",* Proceedings of the International Conference on Advanced Computing Technologies (ICACT '08), December 26-27, 2008, GRIET, Hyderabad.

22. A.V.Ramana, **P.Laxminarayana**, P.Mythilisharana, *"***Real-Time ASR with HMM Word Models for Telugu***",* Proceedings of the International Conference on Recent Advances in Communication Engineering, December 20-23, 2008, Osmania University.

Websites

23. http://www.speech.cs.cmu.edu/sphinxman/fr4.html.

24. http://cmusphinx.sourceforge.net/wiki/tutorialpocketsphinx#windows.

25. http://www.speech.cs.cmu.edu/sphinx/tutorial.html.

26. http://www.speech.cs.cmu.edu/databases/an4/.

27. http://www.ldc.upenn.edu.

28. http://en.wikipedia.org/wiki/Microsoft_Visual_Studio.

ASR over VoIP Networks under Different Channel/Traffic Conditions

Technical Report No. NERTU/UGC-MRP/ASR/01

PART-II

P.Laxminarayana, A.V.Ramana and Mythilisharan

Osmania University
Hyderabad – 500 007
INDIA

7 ASR over VoIP Networks under Different Traffic Conditions

7.1 Introduction to VoIP Networks

Internet telephony refers to communications services such as voice, facsimile, and/or voice-messaging applications that are transported via the Internet, rather than the Public Switched Telephone Network (PSTN). The basic steps involved in originating an Internet telephone call are conversion of the analog voice signal to digital format and compression/translation of the signal into Internet protocol (IP) packets for transmission over the Internet; the process is reversed at the receiving end.

VoIP solutions aimed at unified communications services that treat all communications like phone calls, faxes, voice mail, e-mail, web conferences and more, as discrete units that all can be delivered via any means and to any handset, including cell phones as shown in Figure 7.1. VoIP runs both voice and data communications over a single network, which can significantly reduce the infrastructure costs 10.

Initially, one of the main drivers in developing VoIP was the potential to cut the cost of telephone calls. Traditional voice calls, running over the PSTN, are made using circuit switching, where a dedicated circuit or channel is set up between two points before the users talk to one another just like old-fashioned operators, plugging in the wires to connect two callers. The advantage of this is that once the circuit is set up, the call quality is very good, because it is running over a dedicated line. But this type of switching is expensive because the network needs a great bandwidth which is mostly under-used.

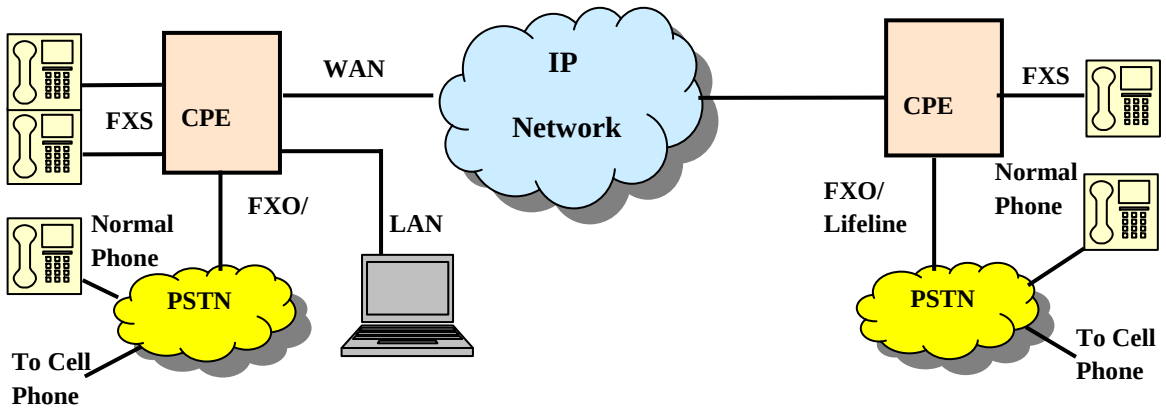

Figure 7.1: A Typical VoIP Network 1

The development of VoIP represents a major change in telecommunications. VoIP uses IP protocols, originally designed for the Internet, to break voice calls up into digital 'packets'. In order for a call to take place the separate packets travel over an IP network and are re-assembled at the other end. The major breakthrough was in being able to transmit the voice calls in the same way as data, though they are much more sensitive to any time delays or problems on the network.

Packetized voice also enables much more efficient use of the network because bandwidth is only used when something is actually being transmitted. Also, the network can handle connections from many applications and many users at the same time, unlike the dedicated circuit-switch approach. This greater efficiency is one of the main reasons that all major carriers are changing their own networks as IP-enabled.

The basic process involved in a VoIP call is as follows:

- Conversion of the caller's analogue voice signal into a digital format
- Compression and translation of the digital signal into discrete Internet Protocol packets
- Transmission of the packets over the Internet or other IP-based network
- Reverse translation of packets into an analogue voice signal for the call recipient.

The digitization and transmission of the analogue voice as a stream of packets is carried out over a digital data network that can carry data packets using IP and other Internet related protocols. The compression process is carried out by a codec, a voice-encoding algorithm, which allows the call to be transmitted over the IP network within the network's available bandwidth.

In common with other modern telecommunications systems, VoIP is structured around two key components:

- The *bearer* (the actual voice being sent over the network) and
- The *signaling* (additional messaging necessary to control and handle other call elements such as the dialed digits for the destination).

Both these components make use of a variety of standards and protocols. The fundamental building block for a VoIP call is, as with other Internet-based applications, the layered/tiered architecture of the Internet as shown in Table 7.1. Lower layers concentrate on the physical transmission and delivery of data (Physical, Data, IP, UDP/TCP, RTP/RTCP layers (1 to 5 respectively)), and higher layers on tasks such as applications and controlling the communication (e.g. signaling (SIP or H.323)).

Table 7-1: VoIP Protocol stack and comparison with the OSI model

VoIP Protocol	Layer	Common Interface Equivalent	OSI Model
VoIP Appl., SIP	7	HTTP	Application
H.323	6		Presentation
RTP, RTCP	5	SSL	Session
UDP	4	TCP	Transport
IP	3	IP	Network
Data	2	Ethernet	Data
Physical	1	xDSL (100Base-T)	Physical

7.2 Protocols

Higher layer in the stack, the session, presentation and application layers are used to handle the signaling required for a telecommunications system. These protocols control the communication

between the two end points of a call e.g. the call set-up and signaling necessary to transmit traffic over an IP network. The following are the major signaling standards for VoIP:

- H.323 [2] and
- Session Initiation Protocol (SIP) [3]

H.323: This is the protocol ratified in 1996 by the ITU, and is a well-established protocol for handling voice, video and data conferencing over packet-based networks.

SIP: SIP is a text-based, open protocol, ratified by the Internet Engineering Task Force (IETF), for telephone calls over IP and has recently gained rapid uptake in spite of the fact that H.323 is the more established protocol. SIP does a similar job to H.323, but was specifically designed for the Internet.

7.3 Speech Codecs

Prior to the transmission of a voice call across an IP-based network a person's voice (which is an analogue sound wave) must be converted to a digital form and encoded. A certain amount of data compression can also take place in order to save bandwidth during the subsequent transmission. On receipt of the voice data at the other end, this process must be reversed.

A number of different voice-encoding algorithms (codecs) are used which have been standardized by the ITU-T as a series of recommendations known as the G-series. Codecs differ in the algorithms they use for sampling the analogue voice signal and the sophistication of the compression used. This in turn determines the amount of digital bandwidth required for the encoded sample. However, ultimately there is a trade-off between the sophistication of the algorithms, the amount of bandwidth required and the quality of the voice signal received. Recently, various wideband codecs are also under deployment for improved voice quality within the same bandwidth.

The following are some of narrowband and wideband speech coding standards most popularly used in VoIP and other wired networks.

G.711

In ISDN, PSTN and VoIP the de facto standard for speech coding is G.711 [4]. This exists in two flavors; μ-law and A-law. μ-law is mainly used in North America and Japan, whereas A-law is used in Europe and the remainder of the world. These standards encode 14-bit linear Pulse Code Modulation (PCM) samples into 8-bit logarithmic samples. The signal is sampled at 8 kHz which gives a bit-rate of 64 kbps.

G.726

G.726 [5] is the Adaptive Differential PCM (ADPCM) based waveform codec, works on a sample basis by making use of a short term correlations between the samples. G.726 codec supports the voice compressed bit rates of 40, 32, 24 and 16-kbps, operates at 8-kHz and the voice quality as good as G.711.

G.729

G.729 [6] [7] is the Code Excited Linear Prediction (CELP) based compression algorithm which makes use of the human vocal tract models. G.729 is supported by many vendors for compressed voice operating at 8 kHz and provides 8-kbps bit rates. With quality just below that of G.711, it is the second most commonly implemented standard.

G.723.1A

G.723.1A [8] was once the recommended compression standard. It operates at 6.3 and 5.3 kbps. Although this standard reduces bandwidth consumption, voice is noticeably poorer than G.729 and is not very popular nowadays.

G.722 (WB)

G.722 [9] operates at 16-kHz sampling rates and provides 64 kbps bit rates. While the narrowband coding standards deliver an analog sound range of 3.4 kHz, wideband codecs such as G.722 processes 7-kHz speech to offer high-fidelity speech. This is the basic and simpler speech codec in the wideband range of codecs based on the Sub-band ADPCM (SB-ADPCM) pronciples.

G.711.1 (WB)

The G.711.1 [10] coder is a wideband (50-7000 Hz) extension of ITU-T Rec. G.711. The encoder input and decoder outputs are sampled at 16 kHz by default, but 8-kHz sampling is also supported. When sampled at 16 kHz, the output of the G.711.1 coder can encode signal with a bandwidth of 50-7000 Hz at 80 and 96 kbps, and for 8-kHz sampling, the output may produce signal with a bandwidth ranging from 50 up to 4000 Hz, operating at 64 and 80 kbit/s. At 64 kbit/s, G.711.1 is compatible with G.711, hence an efficient deployment in existing G.711 based Voice over IP (VoIP) infrastructures is possible.

G.729.1 (WB)

The G.729EV [11] coder is an 8-32 kbit/s scalable wideband (50-7000 Hz) extension of ITU-T Rec. G.729. In G.729EV, Layer 1 is the *core layer* corresponding to a bit rate of 8 kbps. This layer is compliant with G.729 bit-stream, which makes G.729EV interoperable with G.729. Layer 2 is a narrowband enhancement layer adding 4 kbps, while Layers 3 to 12 are wideband enhancement layers adding 20 kbps with steps of 2 kbps.

G.722.2 (WB)

G.722.2 codec (also known as WB-AMR [12]) operates in wideband range (50-7000 Hz) and generates bit rates at 6.60, 8.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 and 23.85 kbit/s. WB-AMR standard is selected by both 3GPP (2000) and ITU-T (2001).

The following Table 7.2 summarizes the ITU-T approved and more popularly used speech coding standards with their corresponding sampling and bit rates for wireline applications.

Table 7-2: ITU-T approved narrowband and wideband Speech Codecs

Coding Standard	Algorithm	Sampling Frequency (kHz)	Bit Rates (kbps)

G.711 (A /U)	Companded PCM	8	64
G.726	ADPCM	8	16/24/ 32/40
G.729	CS-ACELP	8	8
G.723.1A	ACELP / MP-MLQ	8	5.3 / 6.3
G.722 (WB)	SB-ADPCM	16	48, 56, 64
G.711.1 (WB)	Companded PCM , MDCT	16	64, 80, 96
G.729.1 (WB)	CELP, TD-BWE, TDAC	16	8 - 32
G.722.2 (AMR-WB)	(Multi Rate) MRWB-ACELP	16	6.60, 8.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05, 23.85

7.4 Packet Drop in VoIP Networks

However in the VoIP networks, addressing the problem of packet drop is challenging. The communication bandwidth between two phones/users with VoIP depends on the total available bandwidth divided by number of users. Similarly there will be many routes to send or receive the packets from one user to user. VoIP system (internet) estimate the traffic in the network and it will plan to send each packet in the possible route which will have less traffic and takes less time. If we want to communicate using speech in real time, the there is the possibility of getting packets in different order with respect to time. However we have to play them in the original sequence to get back the speech. So, if a packet coming late we have to wait to play the speech or we have to make alternate arrangement substitution to packet. If we wait for longer time there is chance to accumulation of already received packets and some time that delay will make the decoded speech unnatural. So if any packet is delayed by certain amount of time, the packet is assumed to be lost and substitution will be done. So the concealment algorithms or standards for solving the problem of packet loss are available in the literature. So it is proposed to study the performance of ASR due to packet loss with different traffic condition and standard packet loss concealment algorithms. As there was long gap of one year between the submission of project proposal and approval, this part of work was carried out before the project is started.

The encoded speech files with any specific codec are altered by removing some of the frames uniformly, based on the packet drop requirement i.e. 1%, 2% and 5% rates. For example, for 5% uniform packet drops, about 5% packets are removed from the total available packets i.e. every 20^{th} packet is removed from the encoded speech file and filled with zeroes or appropriate numbers (based on the codec requirement) with appropriate frame lengths.

The in-built Packet Loss Concealment (PLC) algorithms associated with most of the codecs (G.711 Appendix-I, in case of G.711 and G.726 codecs) will try to re-generate the lost frames from the stored parameters of the previously received best frames.

8 Experimental Results

8.1 Testing of the Coded data with Un-coded Trained (8-kHz) HMMs

When the 16-kHz speech based HMMs are used, the coded test data is mismatched with the un-coded trained models because of the narrowband coding and the ASR accuracies are reduced as shown in Figure 8.1. As all the narrowband codecs work at 8-kHz sampling rates, it is interesting to find out the ASR accuracies when using 8-kHz speech based models. This can improve the ASR performance when NB codecs are used. But the only additional requirement would be to maintain separate 8-kHz speech based HMMs. The results obtained for ASR recognition accuracies with 8-kHz un-coded trained HMMs for wireline and wireless codecs are given in Table 8.1 and Table 8.2 respectively.

Table 8.1: Testing the wireline coded data (NB Codecs) with un-coded trained models (8kHzHMMs)

Coded Data used in Testing	CI	CD-1gu	CD-2gu	CD-4gu	CD-8gu
Un-coded	85.895	91.281	94.139	94.594	94.683
G.711	85.44	90.682	93.409	94.118	94.098
G.726	85.882	91.006	93.402	94.408	94.601
G.729	84.725	90.758	93.34	94.415	94.532

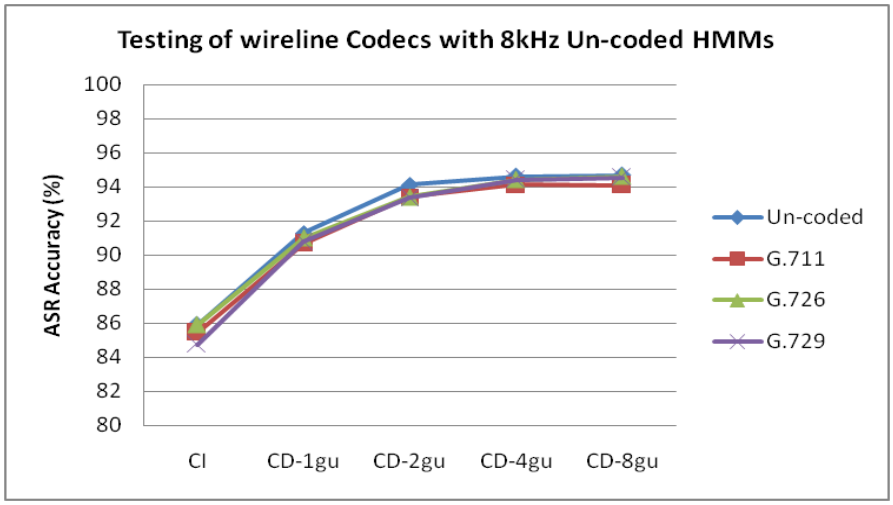

Figure 8.1: Testing the wireline coded data (NB Codecs) with un-coded trained models (8kHzHMMs)

8.2 Testing of the Coded data with G.711-Coded Models (8-kHz HMMs)

The 8-kHz un-coded and coded speech data i.e. coded with all other wireline codecs like G.711, G.726 and G.729 is tested with the G.711 coded models (8-kHz trained HMMs) for the CI, and CD tied tri-phone models with 1, 2, 4 and 8 Gaussians per state, and are reported in the Table 8.2.

Table 8-2: Testing the wireline coded data (NB Codecs) with G.711-coded trained models (8kHzHMMs)

Coded Data	CI	CD-1gau	CD-2gau	CD-4gau	CD-8gau

used in Testing					
Un-coded	86.11	91.41	94.29	95.23	95.89
G.711	89.54	93.86	95.31	96.22	96.51
G.726	90.43	94.33	95.77	96.48	96.83
G.729	85.19	90.86	94.03	94.98	95.43

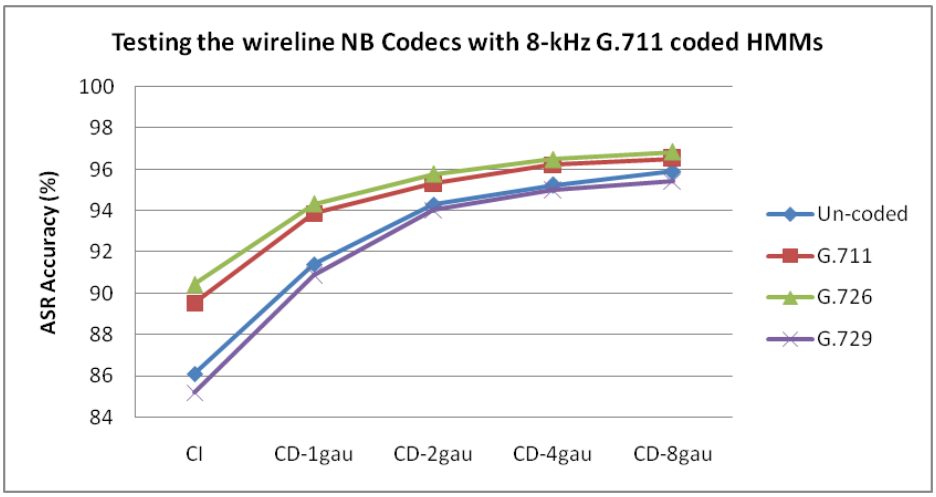

Figure 8.2 the wireline coded data (NB Codecs) with G.711-coded trained models (8kHzHMMs)

8.3 Testing of the Coded data with G.711-Coded Models (16-kHz HMMs)

The 8-kHz coded speech data i.e. coded with all other wireline codecs like G.711, G.726 and G.729, after up-conversion to 16-kHz, is also tested with the G.711 coded models (16-kHz, G.711 coded trained HMMs) for the CI, and CD tied tri-phone models with 1, 2, 4 and 8 Gaussians per state, and are reported in the Table 8.3.

Table 8.3: Testing the wireline coded data (NB Codecs) with G.711-coded trained models (16kHzHMMs)

Coded Data used in Testing	CI	CD-1gau	CD-2gau	CD-4gau	CD-8gau
G.711	84.78	91.11	92.58	93.75	94.25
G.726	84.70	91.35	92.75	93.86	94.47
G.729	78.55	86.89	89.59	91.34	92.30

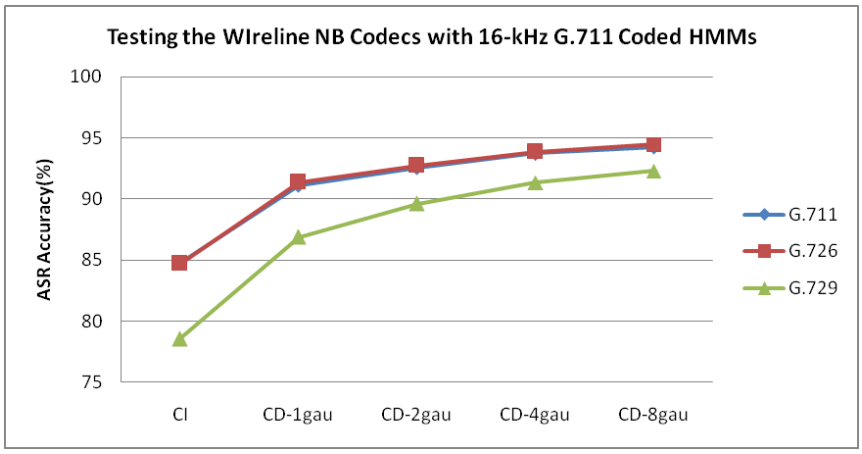

Figure 8.3: Testing the wireline coded data (NB Codecs) with G.711-coded trained models (16kHzHMMs)

8.4 Testing of the Coded data with G.729-Coded Models (8-kHz HMMs)

The 8-kHz un-coded and coded speech data i.e. coded with all other wireline codecs like G.711, G.726 and G.729 is tested with the G.729 coded models (8-kHz HMMs) for the CI, and CD tied tri-phone models with 1, 2, 4 and 8 Gaussians per state, and are reported in the Table 8.4.

Table 8.4: Testing the wireline coded data (NB Codecs) with G.729-coded trained models (8kHzHMMs)

Coded Data used in Testing	CI	CD-1gau	CD-2gau	CD-4gau	CD-8gau
Un-coded	89.54	93.95	95.63	96.5	96.56
G.711	89.208	93.409	95.22	96.281	96.494
G.726	89.063	93.767	95.241	96.716	96.57
G.729	89.952	94.656	96.026	96.853	96.942

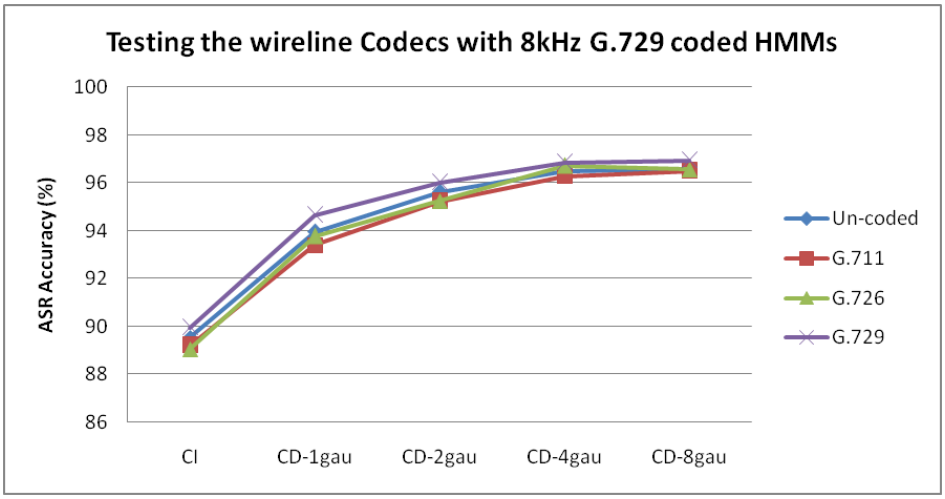

Figure 8.4: Testing the wireline coded data (NB Codecs) with G.729-coded trained models (8kHzHMMs)

8.5 Testing of the Coded data with G.729-Coded Models (16-kHz HMMs)

The 8-kHz coded speech data i.e. coded with all other wireline codecs like G.711, G.726 and G.729 is tested with the G.729 coded models (16-kHz HMMs) for the CI, and CD tied tri-phone models with 1, 2, 4 and 8 Gaussians per state, and are reported in the Table 8.5.

Table 8.5: Testing the wireline coded data (NB Codecs) with G.729-coded trained models (16kHzHMMs)

Coded Data used in Testing	CI	CD-1gau	CD-2gau	CD-4gau	CD-8gau
G.711	85.93	91.14	92.48	93.68	94.14
G.726	85.06	90.67	92.24	93.51	94.18
G.729	84.88	91.18	92.83	93.78	94.47

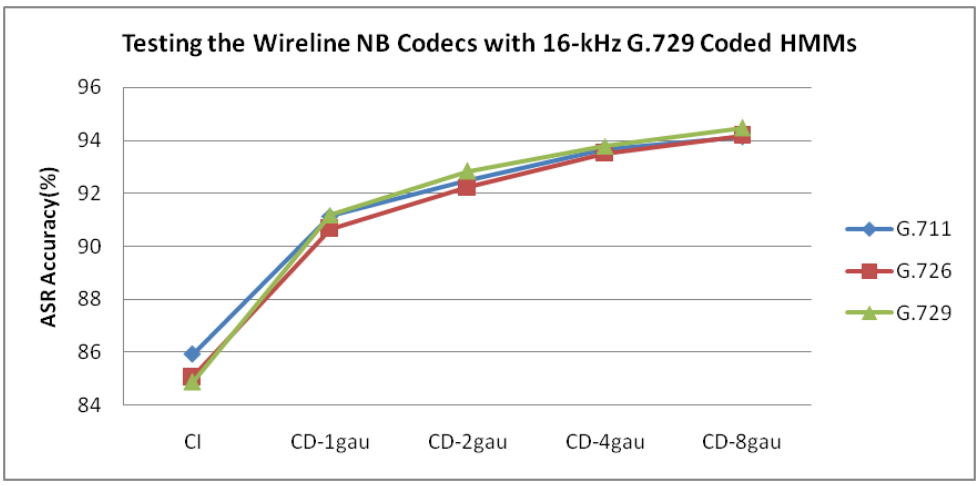

Figure 8.5: Testing the wireline coded data (NB Codecs) with G.729-coded trained models (16kHzHMMs)

9 Performance of ASR with Packet Drops and Transcoding in the Networks

9.1 Introduction

VoIP and GSM networks improve the speech quality, with the use of wideband codecs (7-kHz bandwidth) when compared to narrowband codecs (3.4-kHz bandwidth). On the other hand, there are a number of factors that contribute to the quality of the speech perceived by the user. VoIP and GSM networks may introduce degradations such as packet loss or frame erasures, and delay 10. The characteristics that influence the quality of speech perceived by the user are:

- Speech coding algorithms
- Transcoding and Tandeming of the codecs
- Packet loss or frame erasures
- Delay and jitter
- Echo
- Terminal characteristics (e.g. Frequency response, noise).

In the following sections brief explanation is given about the major network impediments, other than speech codecs, which can influence the speech quality and hence the performance of ASR.

9.2 Factors influence the overall speech performance

9.2.1 Codec Bit Rate

The speech coding degrades the user perceived speech quality. The amount of degradation is not only the function of the compression ratio, but also on the coding principles used and other implementation aspects.

9.2.2 Transcoding and Tandeming

If the voice is transmitted across the different networks, or within the same network, passing through different codecs is known to be Transcoding. If the voice is passing through the same codec multiple times, then it is known to be Tandeming.

In the modern communications, more focus is being put towards the convergence of different networks like cellular, xDSL (VoIP) and other wireless technologies like Wi-Fi and DECT. There is a possibility of using different codecs in these different networks till one or two common codecs are adapted across all the networks. Till such time, all the operators in different networks must be able to inter-operate with each other by transcoding feature as they use different speech codecs.

One of the example scenarios, as shown in Figure 7., is that a DECT phone connected to a VoIP Residential Gateway (RG), calling to a GSM mobile user over IP network. In this case, DECT phone may use one of the wireline codecs like G.711, G.726 or G.722, and VoIP RG may use G.729 or G.722 over the IP network but GSM network use one of the wireless codecs like EFR, HR or AMR-NB codecs. When considering an end-to-end call from DECT phone to mobile phone, the speech has to undergo at least 2-3 different transcoding combinations in the path, which surely affect the overall speech quality (MOS) as well as the ASR performance.

9.3 Packet Loss due to Delay and jitter

In the packet data networks, the data packets may traverse through different nodes, and likely to be get dropped if they incur more delays in the travel path. The major delay sources of a multimedia connection in an IP network and data recovery techniques are explained briefly below.

9.3.1 Transmitting terminal delay

The main sources of the transmitting delay are the speech processing (codec) and the speech packetization. The frame-based algorithms segment the speech signals into frames that typically range from 10 to 30 msec. To reduce the effect of packet loss, Forward Error Control (FEC) / Packet Loss Concealment (PLC) techniques can be used. To do so, some codecs include an extra time window called look-ahead. The minimum delay introduced by a frame-based algorithm is *2 × frame size + look-ahead.*

9.3.2 Network delay

The network delay of a connection between a terminal in the fixed circuit-switched network and a GSM terminal is approximately 100ms. The delay introduced by a connection via a geostationary satellite is approximately 270 ms.

9.3.3 Receiving terminal delay

The main delay source at the receiving terminal is the jitter buffer. Jitter is defined as the delay variation caused by queuing in network elements or by routing the packets along different network paths.

The speech packets that are sent from the transmitting terminal at constant intervals are received at variable intervals. Packets might even be out of order. This delay variation needs to be removed and packets need to be reordered before replaying the audible signal to the human user. This is achieved by inserting a buffer (jitter buffer or playout buffer) at the receiving terminal. The jitter buffer size needs to match the amount of jitter at the receiving terminal. When the jitter buffer is too short, packets which arrive too late will be lost. On the other hand, a long jitter buffer increases the end-to-end delay perceived by the user.

The size of jitter buffer can be fixed or adaptive. In most scenarios, an adaptive jitter buffer is preferable because the jitter characteristics may depend on the actual connection and traffic scenario. However, there are challenges related to adaptive jitter buffers. It is important that the algorithm detects rapid changes in the jitter. In most implementations the adjustment of the jitter buffer size takes place during pauses in the speech.

9.3.4 Packet Loss Concealment (PLC)

VoIP frames have to traverse an IP network, which is unreliable. Frames may be dropped as a result of network congestion or data corruption. The terminal leaves a gap in the voice stream if a voice frame is missing. If too many frames are lost, the speech will sound choppy with syllables or words missing. One possible recovery strategy is to replay the previous voice samples. This works well if only a few samples are missing.

PLC is a technique for replacing or re-generating the speech for lost or unacceptably delayed packets with a prediction based on previously received good speech packets. Whereas any voice repair technology would have difficulty in recovering from extreme packet loss in abnormal conditions, packet

loss concealment is designed to perform intelligent restoration of lost or delayed packets for a large majority of congested network scenarios.

Packet loss degrades the perceived speech quality. The amount of degradation depends on the robustness of the speech codec, and whether or not protection mechanisms such as PLC are implemented along with the codecs. There are various PLC schemes available for speech codecs to work in the case of packet drops as given below:

- Pitch-based PLC as per ITU-T G.711 Appendix-I [ITU-T-G.711 (1999)].
- LP-based PLC as per T1.521a-2000 Appendix-B [T1-521a (2000)].
- Hybrid techniques make use of several combinations of techniques like pitch, LP, interpolation, and time frequency modifications

G.711 and G.726 uses pitch based PLC as per ITU-T Appendix-I whereas G.729A, EFR, AMR and other CELP based codecs have in-built hybrid technique based PLC algorithms. If packet drop exceeds about 3%, hybrid techniques are more helpful.

Much of the analysis was not found in the literature about the ASR performance with the speech and audio codecs under packet drop conditions as well as transcoding and tandeming combinations. In this chapter, the ASR performance and the speech quality (MOS) are evaluated for different NB, WB speech codecs under different packet drops and transcoding combinations for better comparison, in the following sections.

9.4 Experimental Setup

9.4.1 HMMs used in ASR

As explained in section 3.2, the 16-kHz un-coded speech based ASR models (HMMs) are created first with TIMIT speech database. The Context Dependent (CD) HMMs with 8 Gaussians per state are considered for the evaluation of each codec.

9.4.2 Packet Drop Simulation

As explained in section 7.4, the encoded speech file is altered by removing some of the frames uniformly based on the packet drop requirement i.e. 1%, 2% and 5% drop rates. For example, for 5% drops about 5% packets are removed i.e. every 20^{th} packet is removed from the speech file and filled with zeroes or appropriate numbers with appropriate frame lengths based on the codec selected.

9.4.3 MOS Evaluation with Packet drops

For all of these codecs the speech quality (MOS) is evaluated first as explained in section 4.3, over the complete TIMIT speech database. The average MOS value of all the individual speech files is taken as the reference MOS value using P.862.1 and P.862.2 mapping for the individual narrowband and wideband codecs respectively. The MOS values obtained for all the widely used narrowband and wideband speech codecs with different uniform packet drops are given in Tables 9.1 and 9.2 respectively.

9.5 MOS V/s ASR Accuracy under Packet Drops

9.5.1 ASR Recognition Performance for Narrowband Codecs

The original speech files in the TIMIT database are sampled at 16 kHz. Based on the recognition setup given in section 3.5, when the 16-kHz ASR trained models are used for the ASR performance analysis

for narrowband codecs, the following sequence of operations are used: The speech data is down sampled to 8-kHz first as narrowband codecs work with 8 kHz sampling rates only. The data is encoded with the respective coding scheme. Some of the encoded data frames are erased uniformly based on the packet drop requirement and then decoded. The Packet Loss Concealment (PLC) algorithms associated with speech decoders will try to re-generate appropriate speech frames in place of erased frames based on the stored previous good speech samples. The decoded speech is up-sampled back to 16 kHz before giving it to the ASR system for recognition analysis. Table 9.1 shows the MOS and ASR recognition accuracies for the narrowband codecs with different packet drop rates.

Table 9.1: MOS Vs ASR Recognition accuracy for Narrowband Codecs with different packet drops

Codec (@ kbps) used for Testing	Packet Drop (%)	MOS-LQO (Avg.)	ASR Accuracy (%) (Un-coded 16-kHz Training)
G.711 (PCMA) @ 64	0	**4.455**	92.10
	1	4.173	91.99
	2	3.850	91.89
	5	3.017	91.70
G.726 @ 32	0	**4.066**	91.35
	1	3.673	90.85
	2	3.250	90.26
	5	2.561	87.60
G.729A @ 8	0	**3.875**	92.52
	1	3.471	91.55
	2	3.045	91.23
	5	2.392	90.89
GSM-FR @ 13	0	**3.732**	91.19
	1	3.700	90.73
	2	3.144	87.69
	5	2.178	76.52
GSM-EFR@ 12.2	0	**4.229**	92.52

	1	3.577	90.97
	2	2.853	88.23
	5	1.622	73.69
GSM-HR @ 5.6	0	**3.522**	89.92
	1	3.242	85.75
	2	2.783	82.44
	5	2.045	67.22
AMR @ 4.75	0	**3.416**	90.50
	1	3.021	89.78
	2	2.456	89.80
	5	1.666	89.79
AMR @ 12.2	0	**4.229**	93.18
	1	3.658	92.40
	2	2.838	92.26
	5	1.756	92.40

Analysis:

Figure 9.1 is showing the MOS values and Figure 9.1 is showing the ASR accuracies for all the narrowband codecs with different packet drop rates. The following are the observations made from the results:

- The ASR performance of the AMR codec with 12.2 kbps rate is good under all packet drop rates, when compared to all other narrowband codecs. Though the MOS value is dropping more at 2 and 5% rates than other codecs, the ASR accuracy is maintained good.
- The ASR accuracies are also comparatively good for G.711, G.729AB along with AMR codec for all packet drops.
- FR, EFR and HR codecs are performing worst at 5% packet drops when compared to lower packet drop rates.

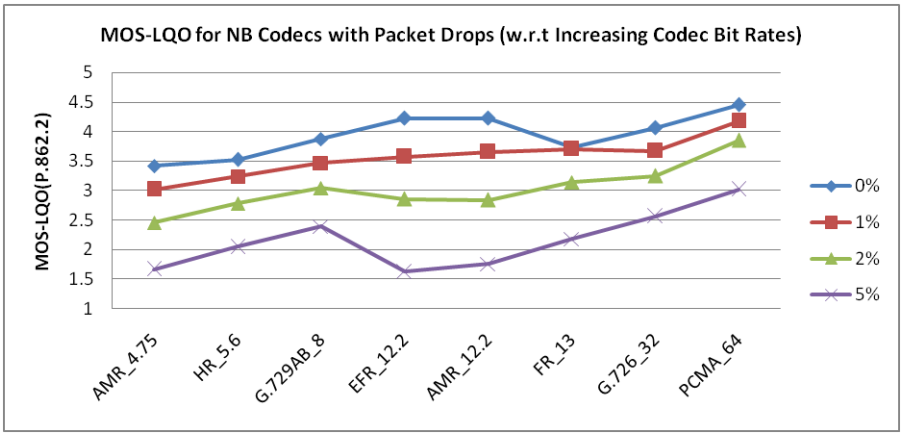

Figure 9.1MOS scores with different packet drops for narrowband codecs

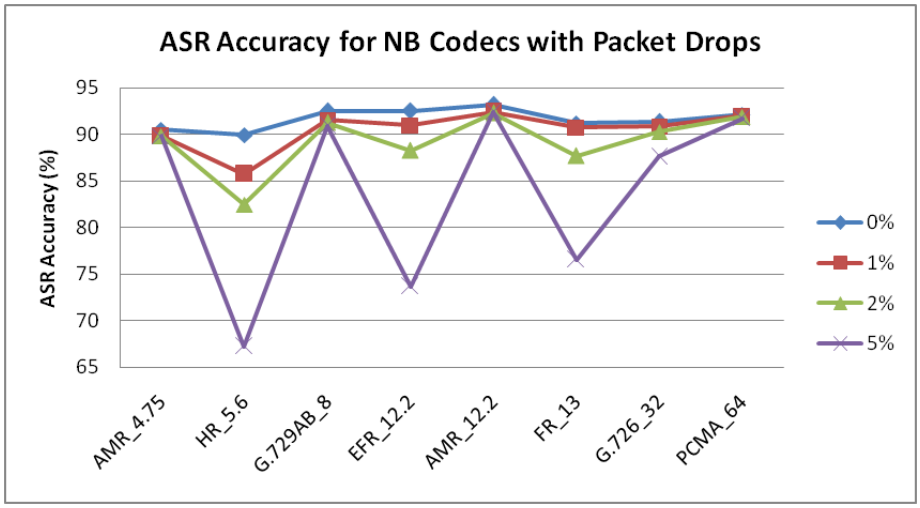

Figure 9.2: ASR Accuracy with different packet drops for narrowband codecs

9.5.2 ASR Recognition Performance for Wideband Codecs

Based on the recognition setup given in section 3.7, when the 16-kHz ASR trained models are used for the ASR performance analysis for wideband codecs, the following sequence of operations are used: The 16-kHz speech data is encoded with the respective wideband coding scheme. Some of the encoded data frames are erased uniformly based on the packet drop requirement and then decoded. The Packet Loss Concealment (PLC) algorithms associated with speech decoders will try to re-generate appropriate speech frames in place of erased frames based on the stored previous good speech samples. The decoded speech is given to the ASR system for recognition analysis. Table 9.2 shows the MOS and ASR recognition accuracies for the narrowband codecs with different packet drop rates.

Table 9.2: MOS and ASR recognition accuracies for the wideband codecs with different packet drop rates

Codec (@kbps) used for Testing	Packet Drop (%)	MOS-LQO (Avg.)	ASR Accuracy (%) (Un-coded 16-kHz Training)
G.722 @ 64	0	**4.271**	98.90
	1	3.926	98.29
	2	3.554	98.16

	5	2.868	98.15
G.729.1 @ 32	0	**4.053**	98.13
	1	3.088	97.50
	2	2.311	96.54
	5	1.422	95.84
WBAMR @ 6.60	0	**2.961**	98.05
	1	2.678	97.75
	2	2.286	97.87
	5	1.661	95.26
WBAMR @ 23.85	0	**4.125**	98.10
	1	3.583	97.52
	2	2.823	96.32
	5	1.728	95.81
Speex @ 8	0	**3.125**	98.30
	1	3.101	98.03
	2	2.861	97.90
	5	2.589	97.80
Speex @ 16	0	**3.617**	98.35
	1	3.585	98.29
	2	3.286	98.25
	5	2.930	98.12

Analysis:

Figure 9.3 is showing the MOS values and Figure 9.4 is showing the ASR accuracies for all the wideband codecs with different packet drop rates. The following observations are made from results:

- The ASR performance of Speex codec at 8 and 16 kbps rates is highly stable for all the packet drop rates when compared to all other codecs, though the MOS-LQO values are poor.
- The ASR accuracies are comparatively good for G.722 @64 under all packet drops.

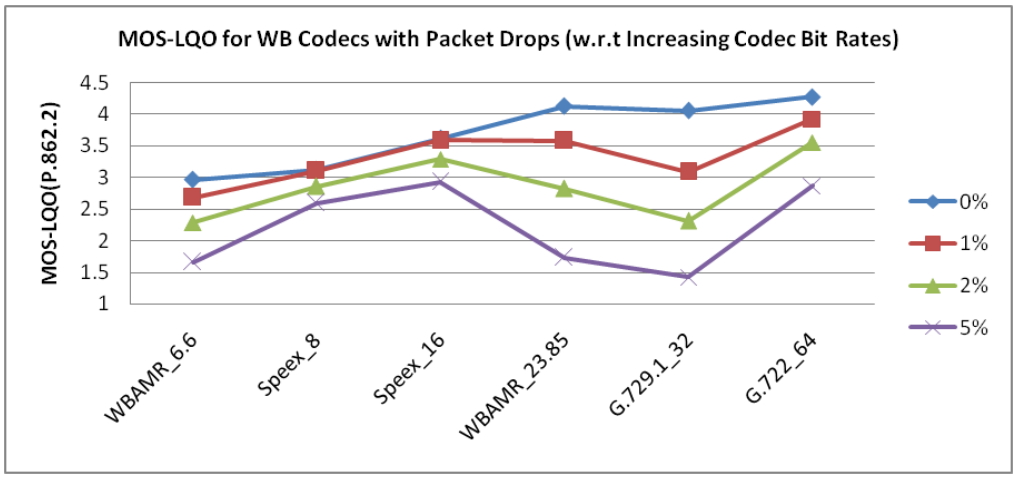

Figure 9.3: MOS scores with different packet drops for wideband codecs

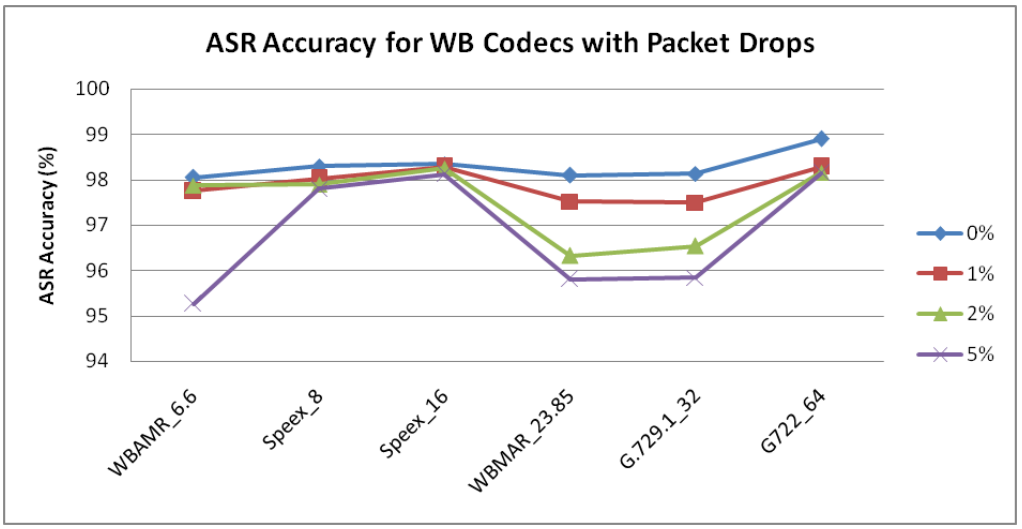

Figure 9.4: ASR Accuracy with different packet drops for wideband codecs

9.6 ASR Performance for Transcoding and Tandeming

9.6.1 ASR Performance with Narrowband Codecs

Each of the narrowband speech codec is combined with same or other narrowband codec, or with wideband codecs in some cases, to form the tandeming or transcoding combinations respectively. The encoded-decoded speech data from the first codec is once again encoded-decoded with the second selected codec.

Table 9.3 shows the MOS (narrowband MOS scale, P.862.1) and ASR recognition accuracies for the different individual narrowband codecs and tandeming/transcoding combinations with other narrowband, or some of the wideband, codecs with different bit rates. The ASR accuracies are measured for both 8-kHz and 16-kHz trained CD-8gaussians HMMs.

Table 9.3: MOS Vs ASR Recognition accuracy for Narrowband Codecs with different tandeming/transcoding combinations

Base Codec (@ kbps) used for testing	Coding scheme used for Transcoding	MOS-LQO (Avg.) (P.862.1)	ASR Accuracy (%) (Un-coded 16-kHz Trained Models)	ASR Accuracy (%) (Un-coded 8-kHz Trained models)
Un-coded (Reference)		**4.491**	**98.23**	**97.12**
G.711 (PCMA)	-	4.455	92.10	95.71
	PCMA	4.455	92.07	95.72
	PCMU	4.437	92.03	95.78
G.711 (PCMU)	-	4.459	92.83	96.24
	PCMU	4.459	93.19	96.29
G.726	-	4.066	91.37	96.88
	G.726	3.863	90.54	95.37
G.729AB	-	3.875	92.52	95.89
	G.729AB	3.360	86.55	92.81
	PCMA	3.892	88.86	94.52
	G.726	3.762	88.19	94.35
	EFR	3.704	87.53	93.45
	HR	3.065	81.92	89.97
	AMR@4.75	3.044	86.54	91.54

	AMR@12.2	3.692	87.78	93.32
	G.722	3.881	88.79	93.98
	WBAMR@6.6	3.266	83.22	90.55
	WBAMR@ 23.85	3.823	89.18	92.54
GSM-FR	-	3.732	91.19	95.31
	FR	3.353	87.76	94.64
	HR	3.098	81.81	90.51
	EFR	3.683	88.69	94.82
GSM-EFR	-	4.229	92.52	95.06
	EFR	4.011	89.85	94.90
	HR	3.408	85.34	92.08
GSM-HR	-	3.522	89.92	94.31
	HR	2.858	74.39	85.12
AMR @ 4.75	-	3.416	90.50	95.21
	AMR@4.75	2.828	83.45	93.40
	EFR	3.294	85.08	91.72
	HR	2.832	76.89	86.25
AMR @ 12.2	-	4.229	93.12	96.05
	AMR@12.2	4.018	89.51	94.88
	EFR	3.996	89.75	94.75
	HR	3.401	85.48	92.48

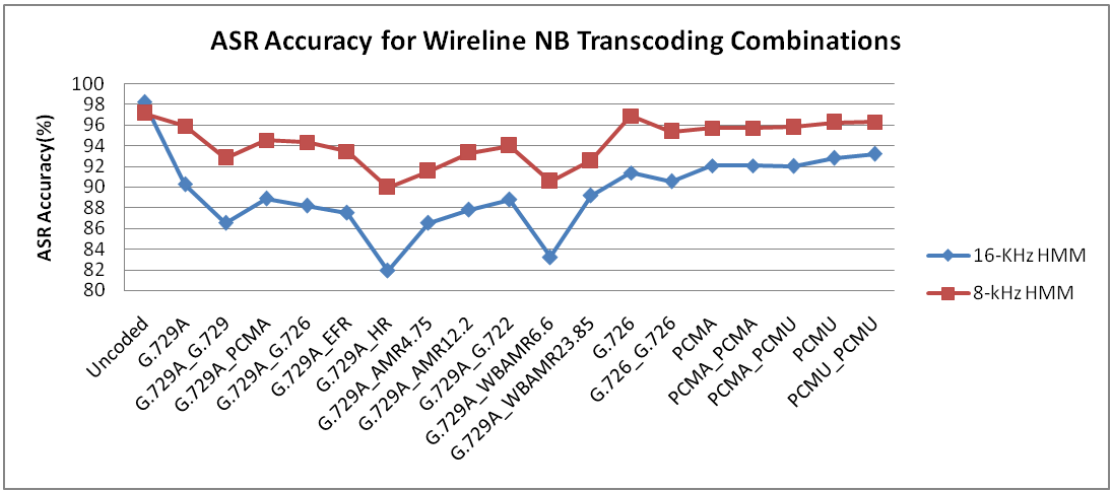

Figure 9.5: ASR Accuracy for wireline NB trans-coding combinations with 16-kHz and 8-kHz un-coded trained models

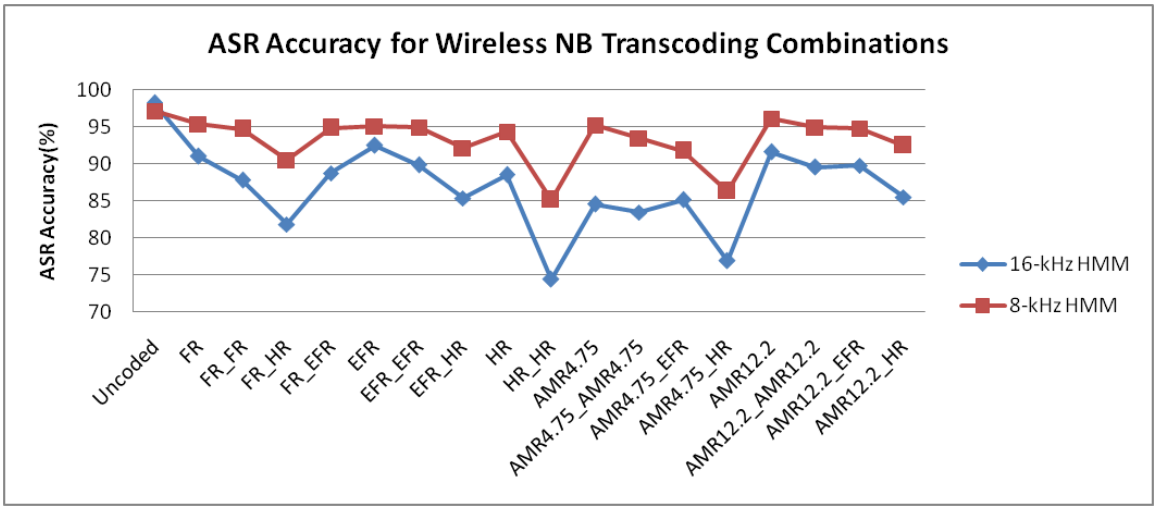

Figure 9.6: ASR Accuracy for wireless NB trans-coding combinations with 16-kHz and 8-kHz un-coded trained models

Analysis:

Figure 9.5 and Figure 9.6 show the ASR accuracies for different wireline and wireless narrowband transcoding combinations respectively. From the results obtained, it can be observed that

- The MOS values (P862.1) are reducing when two different narrowband codecs are used in tandeming or transcoding, especially the MOS is reducing more when using any of the lower bit rate codec in the transcoding combination (e.g. HR@5.6, or AMR@4.75 as one of the codec).
- When 16-kHz un-coded trained models are used for recognition, the ASR accuracy is varying between 74 to 92 % for different NB transcoding combinations, which is about 6-24 % deviation from the baseline (reference) ASR accuracy (98.23%).
- But when 8-kHz un-coded trained models are used for recognition, the ASR accuracy is improved and varying between 85 to 97 % for the same NB transcoding combinations, which is about 0-12 % deviation only from the baseline (reference) ASR accuracy (97.12%).

9.6.2 ASR Performance with Wideband Codecs

Each of the wideband speech codec is combined with same or other wideband codec, or some of the narrowband codecs, to form the tandeming or transcoding combinations respectively. The encoded-

decoded speech data from the first codec is once again encoded-decoded with the second selected codec.

Table 9.4 shows the MOS (wideband MOS scale, P.862.2) and ASR recognition accuracies for the different individual wideband codecs and tandeming/transcoding combinations with other wideband and some of the narrowband codecs with different bit rates. The ASR accuracies are measured for both 8-kHz and 16-kHz trained CD-8gaussian HMMs.

Table 9.4: MOS Vs ASR Recognition accuracy for Wideband Codecs with different transcoding combinations

Base Codec (@kbps) used for testing	Coding scheme used for Transcoding	MOS-LQO (Avg.) (P.862.2)	ASR Accuracy (%) (Un-coded 16-kHz Trained models	ASR Accuracy (%) (Un-coded 8-kHz Trained models)
Un-coded (Reference)		**4.644**	**98.23**	**97.12**
G.722	-	4.267	98.20	96.36
	PCMA	3.482	91.77	95.79
	G.726	2.917	91.12	95.65
	G.729AB	2.844	92.12	95.82
	AMR@4.75	2.449	89.55	94.61
	AMR@12.2	3.147	92.83	95.94
	G.722	3.958	98.17	96.45
	WBAMR@6.6	2.840	98.30	96.14
	WBAMR@23.85	3.863	98.13	96.28
	G.729.1@16	3.526	98.13	
	G.729.1@32	3.823	98.18	96.42
G.729.1 @ 16	-	3.722	97.95	
	G.729.1@16	3.332	97.78	
G.729.1 @ 32	-	4.053	98.13	96.23
	G.729.1@32	3.781	98.03	95.98
WBAMR @ 6.60	-	2.961	98.05	96.16

	WBMAR@6.60	2.411	97.99	95.17
	G.729.1@16	2.671	97.96	
	G.729.1@32	2.859	98.26	
	Speex@16	2.623	98.17	
	Speex@32	2.834	98.33	
WBAMR @ 23.85	-	4.125	98.12	96.34
	WBAMR@23.85	3.853	98.15	96.27
	G.729.1@16	3.526	98.04	
	G.729.1@32	3.872	98.01	96.18
	Speex@16	3.319	98.11	
	Speex@32	3.836	98.20	96.30
Speex @ 16	-	3.617	98.33	96.15
	Speex@16	3.055	98.05	95.90
Speex @ 32	-	4.112	98.10	
	Speex@32	3.797	98.19	

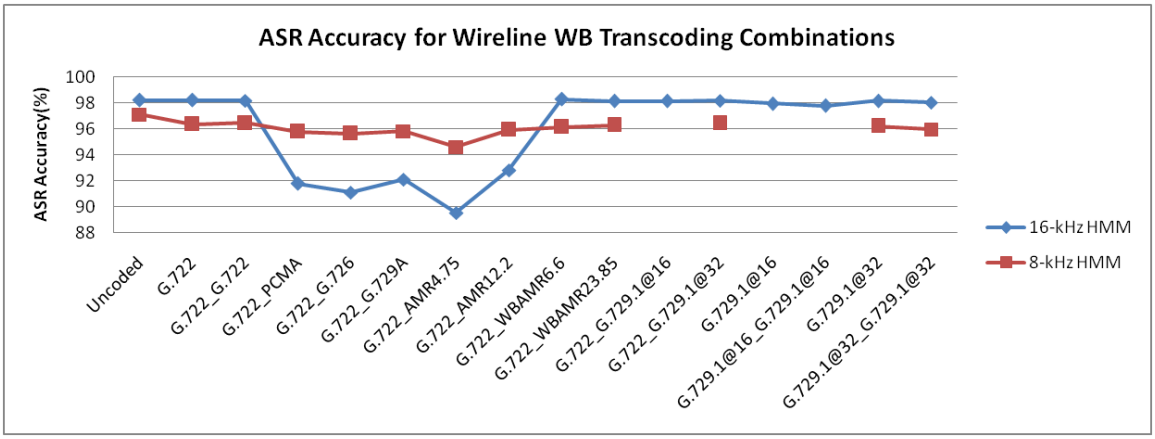

Figure 9.7: ASR Accuracy for wireline WB trans-coding combinations with 16-kHz and 8-kHz un-coded trained models

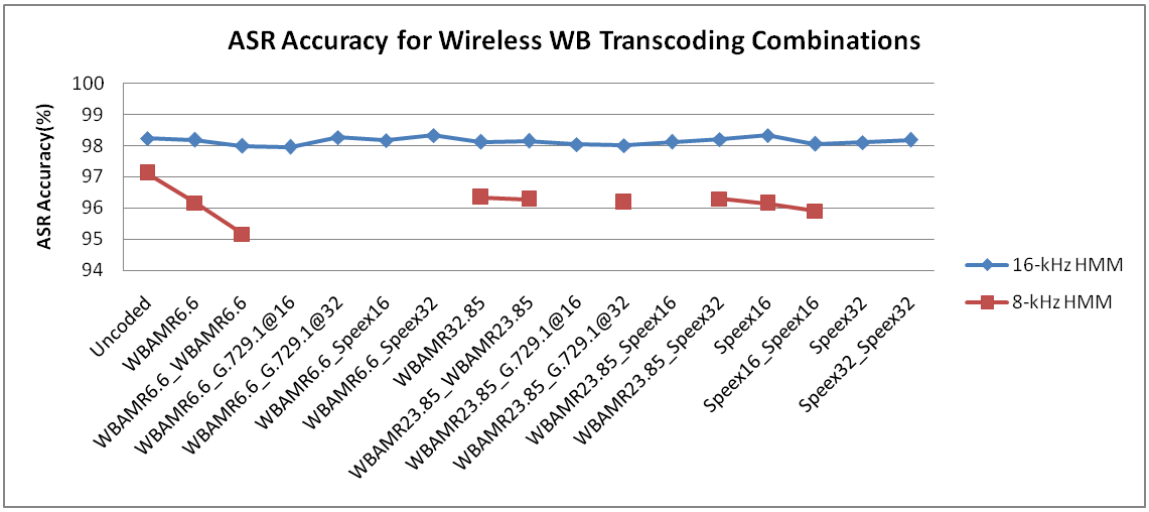

Figure 9.8: ASR Accuracy for wireless WB trans-coding combinations with 16-kHz and 8-kHz un-coded trained models

Analysis:

Figure 9.7 and Figure 9.8 show the ASR accuracies for different wireline and wireless wideband transcoding combinations respectively. From the results obtained, it can be observed that

- The MOS values (P862.2) are reducing when any one of the low bit rate codec is used in the transcoding combination (e.g. either WBAMR@6.60, or G729.1@16 as one of the codec).
- The MOS is also reducing when wideband codecs are transcoded with any of the narrowband codecs (In this case, up-sampling and down-sampling are used when NB codecs are used with WB codecs accordingly).
- When 16-kHz un-coded trained models are used for recognition, the ASR accuracy is varying just within 2% only for almost all wideband transcoding combinations, from the baseline (reference) ASR accuracy. But when wideband codecs are transcoded with narrowband codecs, the ASR accuracy is going less by about 6-8% from the baseline reference ASR accuracy.
- When 8-kHz un-coded trained models are used for recognition, the ASR accuracy is going down by about 2-3% for wideband transcoding combinations when compared to 16-kHz models. But the wideband and narrowband transcoding combinations are improved by about 3% w.r.t. their 16-kHz model values.

9.7 Summary

The ASR performance for the narrowband and wideband codecs under different rates of packet drops and transcoding combinations can be summarized as follows:

- From the narrowband codecs, the AMR and G.729A codecs performing well when compared to all other narrowband codecs under all packet drop rates considered, though the MOS values are lower at higher packet drop rates when compared to other codecs.
- From the wideband codecs, the Speex and G.722 are performing well over all other codecs under all packet drop rates in the wideband mode.
- The ASR recognition performance is not reducing as the MOS values are reducing, even at higher packet drop rates.

- When two different codecs are transcoded, though the MOS values are going very much down, the ASR accuracies are not going down in the similar way, especially for wideband transcoding combinations.
- So, poor MOS is not a true indication of poor ASR performance

10 References

[1] Sivannarayana Nagireddy, "VoIP Voice and Fax signal processing", Wiley publishers, 2008

[2] ITU-T Rec. H.323, Packet based Multimedia Communications Systems, 2006

[3] [RFC 3261], "SIP: Session Initiation Protocol", IETF, 2002

[4] ITU-T Recommendation, G.711 – Pulse Code Modulation (PCM) of Voice Frequencies, 1988

[5] ITU-T Recommendation, G.726: 40, 32, 24, 16 kbit/Adaptive Differential Pulse Code Modulation (ADPCM), 1990

[6] ITU-T Recommendation, G.729 (01/2007), "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction", 1996

[7] ITU-T Recommendation, G.729 – Annex A, "Reduced complexity 8 kbit/s CS-ACELP speech codec"

[8] ITU-T Recommendation, G.723.1 – "Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kbit/s", 1996

[9] ITU-T Recommendation, G.722: 7 kHz Audio - Coding within 64 kbit/s, 1993

[10] ITU-T Recommendation G.711.1, (03/2008), "Wideband embedded extension for G.711 pulse code modulation"

[11] ITU-T Recommendation G.729.1, (05/2006), "G.729-based embedded variable bit-rate coder: An 8-32 kbit/s scalable wideband coder bitstream interoperable with G.729"

[12] 3GPP TS 26.190: "AMR Wideband speech codec; Transcoding functions" (3GPP TS 26.190 version 8.0.0 Release 8), 2009

[13] Ramachandran Ramjee, Jim Kurose, Don Towsley, "Adaptive Playout Mechanisms for Packetized Audio Applications in Wide-Area Networks", INFOCOM-94, Networking for Global Communications, 13th proceedings IEEE, pp. 680-688, Vol.2, Toronto-Canada, 1994

11 Publications

1. M. Ram Reddy, P. Laxminarayana, “Performance of Automatic Speech Recognition over GSM Networks under Different Channel Conditions and Different Narrowband Speech Coding Standards”, Communicated to IEEE Transactions on Audio, Speech and Language Processing.

2. Reddy.M.R., Mangamma S.A., Laxminarayana.P., Ramana.A.V., Gangadhar.P., “Effects of channel conditions on the performance of ASR over GSM networks using full rate speech codec”, International Conference on Computer and Communications Technologies (ICCCT), 11-13 Dec. 2014, Hyderabad, pp. 1 - 6, DOI: 10.1109/ICCCT2.2014.7066729.

3. M. Ram Reddy, P. Laxminarayana, A. V. Ramana et.al., “Transcription of Telugu TV News using ASR”, International Conference on Advances in Computing, Communications and Informatics (ICACCI) 10-13, August 2015, Kochi, pp. 1542-1545, IEEE 978-1-4799-8792-4/15

4. Mythilisharan, A.V.Ramana, P.Laxminarayana, “Enhancement Of ASR Performance Using Mixed Narrowband And Wideband Models”, International Conference on Communication, VLSI and Signal Processing, 20-22 February, 2013, SIT, Thumkur. pp 10-15