DEVELOPMENT OF AN OPTIMAL EXTRACTED FEATURE CLASSIFICATION SCHEME IN VOICE RECOGNITION SYSTEM USING DYNAMIC CUCKOO SEARCH ALGORITHM

By

Sanusi Audee YUSUF

Department of Computer Engineering,
Faculty of Engineering,
Ahmadu Bello University,
Zaria, Nigeria

December, 2017

DEVELOPMENT OF AN OPTIMAL EXTRACTED FEATURE CLASSIFICATION SCHEME IN VOICE RECOGNITION SYSTEM USING DYNAMIC CUCKOO SEARCH ALGORITHM

By

Sanusi Audee YUSUF

P13EGCP8016
audeesy@gmail.com

A DISSERTATION SUBMITTED TO THE SCHOOL OF POSTGRADUATE STUDIES, AHMADU BELLO UNIVERSITY, ZARIA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF MASTER OF SCIENCE (M.Sc) DEGREE IN CONTROL ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING,

FACULTY OF ENGINEERING,

AHMADU BELLO UNIVERSITY,

ZARIA, NIGERIA

December, 2017

DECLARATION

I YUSUF AUDEE Sanusi hereby declare that the work in this dissertation entitled "Development of an Optimal Extracted Feature Classification Scheme in Voice Recognition System using Dynamic Cuckoo Search Algorithm" has been carried out by me in the Department of Computer Engineering. The information derived from literature has been duly acknowledged in the text and a list of references provided. No part of this dissertation was previously presented for another degree or diploma at this or any other institution.

Sanusi Yusuf Audee _____ _____
(Student) Signature Date

CERTIFICATION

This Dissertation entitled DEVELOPMENT OF AN OPTIMAL EXTRACTED FEATURE CLASSIFICATION SCHEME IN VOICE RECOGNITION SYSTEM USING DYNAMIC CUCKOO SEARCH ALGORITHM by YUSUF AUDEE Sanusi, meets the regulations governing the award of degree of Master of Science (MSc) in Control Engineering of the Ahmadu Bello University, and is approved for its contribution to knowledge and literary presentation.

Professor M. B. Mu'azu **(Chairman, Supervisory Committee)**	**Signature**	**Date**
Dr. Sani Man-Yahaya **(Member, Supervisory Committee)**	**Signature**	**Date**
Professor M. B. Mu'azu **(Head of Department)**	**Signature**	**Date**
Professor S. Z. Abubakar **(Dean, School of Postgraduate Studies)**	**Signature**	**Date**

DEDICATION

This dissertation is dedicated to my late father Alh. Yusuf Audi Malumfashi, my late beloved brother Alh. Muhammadu Hadi Yusuf Audi may Almighty Allah forgive their short comings and my mother Hajia Mariya Yusuf.

ACKNOWLEDGEMENT

All praises belong to Almighty Allah, we praised Him, seek for His assistance and forgiveness, and we take refuge of Allah from the evil of our souls and from our bad deeds. Whosoever Allah (SWT) guided remains guided, and whoever goes astray has no other guidance except from Allah. I thank Allah (SWT) for His Blessings and Guidance towards a successful completion of this work.

My deep and sincere appreciation with gratitude goes to my supervisor, the Chairman of my Supervisory committee, Professor Muhammad Bashir Mu'azu, for his immense contribution and guidance towards a successful completion of this work, despite his tight schedules his resilience is incomparable. You are indeed one in a million and a great mentor, I am indeed grateful having you as a supervisor, thank you very much sir and may Allah (SWT) increase your knowledge for the benefit of humanity. My appreciation also goes to my co-supervisor in person of Dr. Sani Man-Yahaya for his valuable input and persisting encouragement throughout the stages of the work, I really appreciate the training and pieces of advice received from you. I also wish to extend my special thanks to the members of Control & Computer Research Group for their valuable contributions, suggestions and constructive criticisms during the discussion stages of this work.

I want to acknowledge and appreciate the contribution of all the staff of Electrical and Computer Engineering Department, Ahmadu Bello University, namely: Prof. B. G. Bajoga, Professor M. B. Mu'azu, Prof. B. Jimoh, Prof. U.O Aliyu, Dr. K. A. Abu Bilal, Dr. A. M. S. Tekanyi, Dr. T. H. Sikiru, Dr. Y. Jibril, Dr. S. M. Sani, Dr. E. A. Adedokun, Engr. M. J. Musa, Engr. A. I. Abdullahi, Engr. Salawudeen A. Tijjani, Engr. Bashir Sadiq, Engr. Olaniyan Abdulrahman, Mallam Tukur Lawal, Mallam Mustapha Sulaiman, and all those whose names could not be mentioned.

My special thanks goes to Engr. Zaharuddeen Haruna, Umar Abubakar, Oyibo Prosper, Ajayi Ore-Ofe, Muktar Abubakar, Umar Musa for their valuable contributions and support towards the success of this work. Words can't express my appreciation; I am very grateful. My colleagues/course mates in Control

option, Engr. Abdulaziz Ango, Ibrahim Umar, Engr. Ugwo Gabby, Okoli, Babangida, and Sulaiman are worthy of mentioning for the friendly and peaceful coexistence enjoyed throughout our study period.

My special gratitude goes to the management of Federal College of Education Zaria, especially my super director, the Director of Works in person of Arc. U. I. Sulaiman (Mrs.) for the support and opportunity given to me to undergo this program, I am grateful.

Finally, my un-relented appreciation goes to my parents Late Alh. Yusuf Audi Malumfashi, Hajia Mariya and Hajia Hadiza for their, prayers, training, love and support, may Allah (SWT) make Jannatul Firdaus be your final abode. My brothers and sisters, I really appreciate your support and pieces of advice throughout my life endavour, I couldn't have made it without you, I love you all.

Last but certainly not the least, my appreciation goes to my wife Sabrah Salisu Shehu , my little children; Hafsat, Muhammad, Abdullahi and Maryam who endured my absence throughout the period I undertook this program, and for their constant prayers towards a successful completion of this work.

ABSTRACT

This research work is aimed at the development of an optimal extracted feature classification scheme in a voice recognition system using dynamic cuckoo search algorithm. This minimized error mismatch in the recognition process and increased accuracy of recognition. Standard voice dataset was obtained from English Language Speech Database for Speaker Recognition (ELSDSR) of the Technical University of Denmark (DTU), processed and key features of these voice data were extracted. A dynamic Cuckoo Search Algorithm (dCSA) was developed, which optimally classify the extracted feature vectors of the speech signals from the voice data for the voice recognition system (using the dataset obtained from ELSDSR database of the DTU. The performance of the developed Voice Recognition System (VRS) with dCSA-based scheme was compared with that of the standard CSA-based scheme using accuracy as performance metrics. The results of the dCSA-based classification scheme showed a recognition accuracy of 93.18% in the VRS when compared with that of the standard CSA-based classification scheme which records 90% accuracy. Simulation was carried out using MATLAB 2013b.

TABLE OF CONTENTS

CHAPTER THREE: MATERIALS AND METHODS

CHAPTER FOUR: RESULTS AND DISCUSSION

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

LIST OF ABBREVIATIONS

Acronyms	Definition
ACO	Ant Colony Optimization
ANN	Artificial Neural Networks
BA	Bat Algorithm
CSA	Cuckoo Search Algorithm
DE	Differential Evolution
DTU	Denmark Technical University
dCSA	Dynamic Cuckoo Search Algorithm
ELSDSR	English Language Speech Database for Speaker Recognition
ES	Evolution Strategies
FFT	Fast Fourier Transform
GA	Genetic Algorithm
GLA	Generalized Lloyd Algorithm
GUI	Graphic User Interface
HMM	Hidden Markov Model
HS	Harmony Search
Hz	Hertz
ICS	Improved Cuckoo Search
IMM	Informatics and Mathematical Modeling
LPCC	Linear Predictive Coding Coefficients
LVQ	Learning Vector Quantization
MATLAB	Matrices Laboratory
MFCC	Mel-Frequency Ceptral Coefficients
MOCS	Multi-Objective Cuckoo Search

MSE	Mean Square Error
NN	Nearest Neighbour
OBL	Opposition Based Learning
PBIL	Population-Based Incremental Learning
PCM	Pulse Code Modulation
PNN	Pairwise Nearest Neighbour
PSO	Particle Swarm Optimization
RLS	Randomised Local Search
SOM	Self Organizing Map
SVM	Support Vector Machine
VQ	Voice Quantization
VRS	Voice Recognition System

CHAPTER ONE

INTRODUCTION

1.1 Background of the Research

Voice denotes to sound produced in a person's larynx and articulated through the mouth, as speech or song, while speech refers to the ability to express thoughts and feelings by articulate sounds (Das & Nahar, 2016). Voice is used to express certain opinion or interest using specific words. These words are used for communication among individuals, which is the bridge that lays the foundation for the improved human relationships (Amarasinghe & Wimalaratne, 2017). In addition to human-human interaction, the spoken word is now extended through technological mediation such as telephony, movies, radio, television, computers and the Internet to finds a reflection in human-machine interaction as well. This gives rise to other interesting research topics like speech recognition, speaker identification, and voice recognition (Huang *et al.,* 2001). Research into voice recognition begun since the early 1960's (Juang & Rabiner, 2005).

Voice recognition is a binary classification problem in which a person's identity is verified based on his/her voice (Zhang *et al.,* 2017). It has wide range of application area and plays a crucial role in the arena of forensics, security and biometric authentication for verifying or detecting the voice of a speaker from the group of speakers (Das & Nahar, 2016).

Human voice in general, carries much information such as gender, emotion and identity of the speaker. The objective of voice recognition is to decide which speaker is present based on the individual's utterance. A voice analysis is done after taking a sample of voice through microphone from a speaker (Muda *et al.,* 2010). The design of the system at the highest level contain two modules, feature extraction and feature matching. Feature extraction (which consist of data processing and extraction) is the process of extracting unique information from voice data that can later be used to identify the

speaker. Feature matching (which consist of feature classification and pattern matching) is the actual procedures of identifying the speaker by comparing the extracted voice data with a database of known speakers, and based on this a suitable decision is made (Price & Eydgahi, 2006).

In speaker recognition system, the main problem lies in the pattern recognition, and in a much broader view, this problem belongs to a generic topic (i.e. pattern recognition) in science and engineering with the aim of minimizing mismatch error and improve recognition accuracy (Kumar & Rao, 2011). The goal of pattern recognition is to classify objects of interest into one of a number of categories or classes. The objects of interest are generically called patterns, and in this research they are the sequences of acoustic vectors that are extracted from an input speech signal. The classes here refer to individual speakers (Kinnunen *et al.,* 2011).

Classification is the problem of identifying to which set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known (Tang *et al.,* 2014). Classification is an unsupervised technique in data clustering that aims at grouping similar samples into groups called clusters, each cluster has maximum within-cluster similarity and minimum between-cluster similarity based on certain similarity index (Aggarwal & Reddy, 2014). Hence, classification technique in any feature matching of voice recognition is an integral part that cannot be ignored, as it determines the recognition accuracy and performance of the system. However, most of the existing classification techniques used in voice recognition systems (VRS) were either classical or statistical methods that are prone to some challenges such as; determination of best sequence of a model states, adjustment of model parameters so as to best account for the observed signal, determination of the optimal training values etc.

However, nature inspired metaheuristic optimization algorithms are known to have a proven efficiency in solving many optimization problems (Yang, 2012). Optimization is a process of producing solutions

to a problem under constrained situations. Optimization methods were developed with the zeal to utilize available resources in the best way possible (Yılmaz & Küçüksille, 2015). Nature inspired computation techniques are derived from the study of natural system. Candidate solutions to the optimization problem play the role of individuals in a population, and the fitness function determines the quality of the solutions (Kamat & Karegowda, 2014). Nature inspired metaheuristic algorithms forms a significant part of modern global optimization algorithms, computational intelligence and soft computing. The growing reputation of metaheuristics and swarm intelligence has fascinated a great deal of consideration in engineering and industry, one of the reasons for this admiration is that nature-inspired metaheuristics are flexible and efficient, and such seemingly simple algorithms can deal with very complex optimization problems (Yang, 2012).

Cuckoo search algorithm (CSA) is also one of the nature inspired metaheuristic algorithm developed by Yang & Deb in 2009, based on the obligate brood parasitic behaviour of some cuckoo species in combination with the Levy flight behaviour of some birds and fruit flies. CSA has been proved to be an effective optimization algorithm when compared with other algorithms. It has been applied as an optimization algorithm for various tasks including finding optimal features, optimizing the parameters of various classifiers including Artificial Neural Network (ANN), Support Vector Machines (SVM) parameters, etc. (Kamat & Karegowda, 2014).

However, the standard CSA uses fixed value for both *pa* and α and the main drawback of this method appears in the number of iterations to find an optimal solution (Valian *et al.*,, 2011). A dynamic Cuckoo Search Algorithm (dCSA) will be developed to address these problems in the standard CSA by introducing inertia weight factor to the control parameters and increase its accuracy.

To extract features from voice signals in the Voice Recognition System (VRS), Mel-Frequency Cepstral Coefficients (MFCC) technique will be used to produce set of feature vectors. Subsequently, dynamic

CSA will be employed at the classification level of the feature matching stage of this research work, where it will be used to optimally classify the extracted feature vectors in order to improve recognition accuracy of the VRS.

1.2 Motivation

Speech is a complex signal produced as a result of numerous transformations arising at several different levels, due to mixture of anatomical variances inherent in the vocal tracts of different individuals. These inherent differences (unique features) are extracted from the speech signal for further analysis. For the past six decades, researchers explored the utilization of these differences from the speech signal for various applications such as, forensics investigation, security system, biometric check, voice recognition, crime detection, etc. Methodologies adopted by the researchers for classification and matching of these unique features in order to reduce mismatch error were mostly classical and statistical methods, this include; Hidden Markov Model (HMM), Voice Quantization (VQ) and Dynamic Time Warping (DTW). These techniques have some associated problems that hinders classification process, which in turn affect recognition accuracy. However, metaheuristic algorithms, especially those based on swarm intelligence are remarkably efficient and have many advantages over traditional and deterministic methods. Also metaheuristic algorithms are problem independent, as they can be applied to solve different kind of problems. Hence, better classification is expected.

Thus, this research work offers the development of a metaheuristic search algorithm named as the dynamic cuckoo search algorithm to determine an optimal extracted feature classification in voice recognition system.

1.3 Significance of Research

The significance of the research is the development of an optimal extracted classification scheme in voice recognition system using dynamic cuckoo search algorithm, which improve the clustering in the classificat

ion technique and better recognition accuracy. This has not been done by previous researchers.

1.4 Statement of Problem

Classification is an integral part of VRS that identifies to which set of categories a new observation belongs. Existing classification techniques have some challenges in the determination of best sequence of model states, determination of optimal training values and adjustment of model parameters to best account for the observed signal.

Hence, to address these challenges of classification in voice recognition system, a Computational Intelligence based classification scheme using a dynamic cuckoo search algorithm is developed in order to increase the accuracy of the system.

1.5 Aim and Objectives

The aim of this research work is to develop an optimal extracted feature classification scheme in voice recognition system using dynamic cuckoo search algorithm.

This aim was accomplished by the following objectives:

1. Obtaining a standard voice data from English Language Speech Database for Speaker Recognition (ELSDSR) database of the Technical University of Denmark (DTU), process and extract key features for voice recognition system (VRS)**.**
2. Developing a dynamic Cuckoo Search Algorithm (dCSA) for optimal extracted feature classification scheme in voice recognition system using same dataset obtained from ELSDSR database of DTU**.**

3. Validating by comparing the performance of the VRS with a standard CSA-based scheme and the dCSA-based scheme using accuracy as performance metrics in order to determine improvement in the VRS.

1.6 Methodology

The methodologies adopted are as follows:

1. Development of speakers' database for voice recognition system by:
    a) Obtaining a standard voice dataset from ELSDSR database of DTU for training.
    b) Extracting key features of the voice signal with MFCC.
    c) Training of the extracted features for storage in a voice database.
2. Development of dynamic Cuckoo search algorithm (dCSA) by:
    a) Initializing random population of *n* host nest and Cuckoo parameters.
    b) Introducing random inertia weight to the control parameters (*pa* and α)
    c) Generating new solutions by Lèvy flight.
    d) Evaluating fitness of the new solution and comparing with a randomly chosen nest, retaining the best solution.
    e) Performing local search to replace worst nest with new one and keeping the best solution.
3. Comparison of the standard CSA with the dynamic CSA using ten (10) standard optimisation test functions (i.e. Michaelwicz, De Jong, Easom, Shubert, Griewangk, Ackley, Rastrigin, Sphere, Rosenbrock and Schwefel).
4. Application of the dynamic CSA to VRS by:
    a) Repeating step 1(a & b) above,
    b) Classifying the extracted features using dCSA for matching and identification.
    c) Testing of speakers for recognition.
5. Validation of the performance of VRS with CSA-based scheme and dCSA-based scheme using Accuracy as the performance metric.

1.7 Dissertation Organization

The general introduction has been presented in Chapter One. The rest of the chapters are structured as follows: Detailed review of related literature and relevant fundamental concepts about Voice itself, how a speech or sound is produced. Voice recognition systems, categories and types of voice recognition system, speech processing, speech signal acquisition, process of speaker recognition. Feature extraction process and its techniques, classification and feature matching, classification process and its different techniques in VRS. Metaheuristic optimization algorithms, Cuckoo search algorithm (CSA), inertia weight factor, dynamic Cuckoo search algorithm (dCSA), optimization test functions are carried out in Chapter Two. Likewise, an in-depth approach and relevant mathematical models describing the development of an optimal extracted feature classification scheme in VRS using dCSA are presented in Chapter Three. Furthermore, analysis, performance and discussion of the results obtained are shown in Chapter Four. Finally, summary, conclusion and recommendations for further work makes up Chapter Five. The list of cited references, transcript of audio messages used during the training/testing session and MATLAB codes are all provided at the appendices of this dissertation.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This chapter comprises of the review of fundamental concepts and the review of similar works. The review of fundamental concepts dwells on overview of concepts and theories that establish the basis of the study while the review of similar works dwells on review of literature to establish limitations and gaps in knowledge related to this study.

2.2 Review of Fundamental Concept

This section contains the review of concepts and theories that are fundamental to this study, such as Voice, Speech production, VRS, Classification techniques, Feature extraction, Feature matching, CSA, Lèvy flight, Inertia weight, standard optimization test functions, amongst others.

2.2.1 A voice

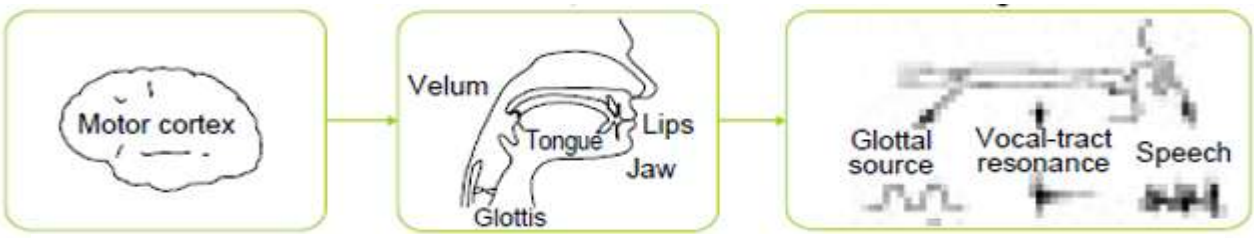
Voice as earlier defined, refers to sound produced in a person's larynx and articulated through the mouth, as speech or song (Das & Nahar, 2016). Speech convey levels of information to the listener, at the primary level it conveys message via words, while at the secondary level it conveys information about the language being spoken, emotion, gender and generally the identity of the speaker (Reynolds, 2002). Speech is the foundation of self-expression and primary means of communication with others. It is the dominant mode of human social bonding and information exchange, to understand speech, humans consider not only the specific information conveyed to the ear, but also the context in which the

information is being discussed (Huang *et al.*, 2001). From technological curiosity about the mechanisms for mechanical realization of human speech capabilities, to the desire to automate simple tasks requiring human-machine interactions, research in speech recognition has attracted attention over the past decades (Juang & Rabiner, 2005).

2.2.2 Speech production

The fundamental aspect of a voice recognition system is the speech signal produced as voice or sound. The production of speech among humans is a common phenomenon that is encountered in their day to day life which is also part of communication between them. Human beings can generate varieties of sounds whose loudness and frequency spectrum changes rapidly (Mahendru, 2014).

Speech production process is as shown in fig.2.1 (Honda, 2003).



(a) Motor Control (b) Articulatory Motion (c) Sound Generation

Fig.2.1: Speech Production Process (Honda, 2003)

Motor control function is energized by the human brain which generates a thought of what to speak and at the same time provides control signals through sensory nerves to the speech fabrication organs, then speech fabrication organs move and take proper form according to the words to be spoken or sound to be formed (Mahendru, 2014).

The articulation process is the most obvious one that takes place in the mouth and it is the process through which one can differentiate most speech sounds. In the mouth one can distinguish between the oral cavity, which acts as a resonator, and the articulators, which can be active or passive: upper and lower lips, upper and lower teeth, tongue (tip, blade, front, back) and roof of the mouth (alveolar ridge, palate and velum). So,

speech sounds are distinguished from one another in terms of the place and the manner they are articulated (Phonetics & Trujillo). Numerous organs are involved in the making of speech and sound, these organs are flexible in nature and their shape and size adjusts the command of motor control signals received from the brain, as to what type of speech and sound to be produced. The lungs provide the required air force for the generation of sound in form of acoustic wave. The air passes through the vocal tract (the pipe that links lungs and throat), vocal cords, glottis, epiglottis, and other organs in the mouth and finally comes out through mouth and nasal cavities in the form of sound wave (Mahendru, 2014). Various organs involved in the of production of speech and sound are as shown in Fig. 2.2 (Kinnunen & Li, 2010).

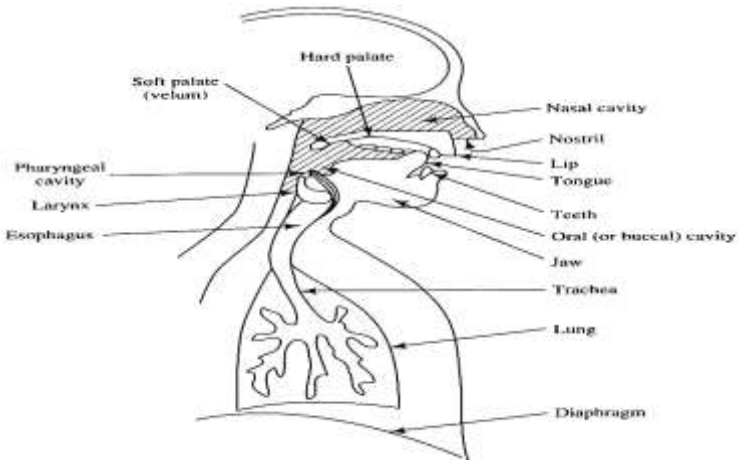


Fig.2.2: Speech production organs (Kinnunen & Li, 2010)

The human vocal machinery is driven by an excitation source, which also contains speaker-dependent data. The excitation can be categorized as whispering, vibration, phonation, compression, frication or a combination of these. Because of variations in the method of speech production, it is understandable to expect some speeches to be more accurate for certain categories of excitation than others (Campbell, 1997). Other features of speech production that could be employed in discriminating between speakers are learned features, including speaking rate, prosodic properties and dialect (Campbell, 1997)

Voiced speech can be generated as explained above, by modulating the air stream from the lungs, and the generation is performed by periodically opening and closing of vocal folds. The oscillation frequency of vocal folds is called the fundamental frequency, (F_0) and it depends on the physical characters of vocal folds. Hence, fundamental frequency is an important physical distinguishing factor, which has been found effective for automatic speech and speaker recognition (Feng & Hansen, 2005).

2.2.3 Voice recognition system

The ability of recognizing a person solely from his voice is known as speaker recognition (Atal, 1976). Voice recognition plays a pivotal role in the field of forensics, security and biometric authentication for verifying or identifying the voice of a speaker from a group of speakers (Das & Nahar, 2016). Generally speaking, human voice carries much information such as gender, emotion and identity of the speaker. The key purpose of voice recognition is to determine who is speaking based on the person's utterance (Muda *et al.,* 2010). Voice recognition has been an interesting research field for speech recognition, speaker recognition, speech synthesis, speech coding etc. (Bhalla *et al.,* 2012). Several techniques have been proposed in order to reduce the mismatch error between the testing and training data in voice recognition (Muda *et al.*, 2010).

2.2.3.1 Categories of voice recognition system

Generally, voice recognition systems can be broadly divided into two: speech recognition and speaker recognition. However, some researchers now include language recognition (Uddin *et al.*, 2016).

Speech recognition: Speech recognition is the ability of a machine to recognize what have been said, it covers the ability to match a voice pattern against an acquired or provided vocabulary. Normally, the vocabulary given is small and the user needs to record a new word to expand the vocabulary. In short, it is the ability of a machine to recognize some spoken words in a speech signal (Uddin *et al.,* 2016).

Speaker recognition: Speaker recognition is the process of automatically recognizing who is speaking on the basis of individual information included in speech signals. This is possible because different speakers have different spectra for similar sound. Spectra are the location and magnitude of peaks in spectrum (Uddin *et al.*, 2016).

2.2.3.2 Speaker recognition

Speaker recognition is a binary classification problem in which a person's identity is verified based on his/her voice (Zhang *et al.,* 2016). The general area of speaker recognition encompasses two more fundamental tasks: speaker verification and speaker identification.

1. Speaker verification is the process of determining whether the speaker identity is who the person claims to be, It performs a one-to-one comparison of determining whether a person is who he/she claims to be (a yes/no decision). Likewise, it is referred to as a binary decision amongst the features of an input voice and those of the claimed voice that is recorded in the system. Different terms for speaker verification were used in various works such as voice verification, voice authentication, speaker/talker authentication, talker verification (Campbell, 1997).
2. Speaker identification is the process of finding the identity of an unknown speaker by comparing his/her voice with voices of registered speakers in the database. The unknown person makes no identity claim and so the system must perform a l:N classification. Generally it is assumed the unknown voice must come from a fixed set of known speakers (Reynolds, 2002).

In different circumstances, speaker recognition is often categorised into closed-set recognition and open-set recognition, the closed-set are such cases where the unknown voice must come from a set of known or registered speakers; and the open-set means unknown voice may come from unregistered speakers, in such a case 'none of the above' option can be added to this recognition system (Feng & Hansen, 2005). Depending on the level of user cooperation and control in an application, the speech used for these tasks can be either text dependent or text-independent (Reynolds, 2002):

1. Text-dependent recognition is the process where speakers are only allowed to say some specific sentences or words (constraint on what is spoken), which are known to the system and the text must be the same for enrollment and verification.
2. Text-independent recognition is where the speaker is freely allowed to speak (no constraint on what is spoken), the system can process freely spoken speech, which is either user selected phrase or conversational speech. Text-independent systems are most often used for speaker identification as they require very little cooperation by the speaker. In this case the text during enrollment and test is different. However, the enrollment may happen without the user's knowledge, as in the case for many forensic applications (Das & Nahar, 2016).

2.2.3.3 Processes in speaker recognition system

Speaker recognition is the process of recognizing the speaker from a database, based on certain characteristics in his speech wave (Nijhawan & Soni, 2014). Speaker recognition systems have to function in two phases, the first one is the enrollment sessions or training phase while the second one is the operation sessions or testing phase (Bhalla *et al.,* 2012). In the training phase, each registered speaker has to provide samples of their speech so that the system can build or train a reference model for that speaker. It mainly consists of two main parts, first part processes each person's input voice sample to condense and summarize the characteristics of their vocal tracts and the second part involves pulling each person's data together into a single, easily manipulated matrix. In addition, a speaker-specific threshold is also computed from the training samples (Kumar & Rao, 2011). During the testing (operational) phase, the testing system mirrors the training system's architecture. Similar feature vectors are extracted from the test utterance, and the degree of their match with the reference is obtained using some matching technique. The level of match is used to arrive at a decision whether there is a recognition or not (Bhalla *et al.,* 2012). A block representation of the two phases involved is as shown in Fig. 2.3 (Muda *et al.,* 2010)

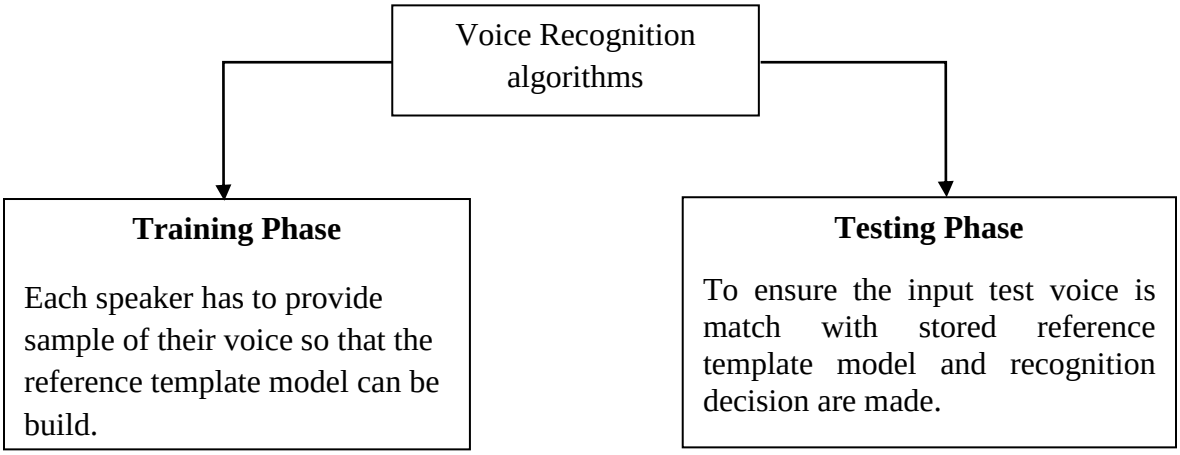


Fig. 2.3: Phases in Speaker Recognition (Muda *et al.*, 2010)

However, speaker recognition system involves certain processes that make sure that unique features were extracted from the person's speech signal, and these features are used for training and testing. The key components of speaker recognition system are signal acquisition, feature extraction, classification/pattern matching, decision logic then recognition. These processes are explained in the following subsections:

2.2.3.4 Speech signal acquisition process in ELSDSR voice database

Primarily, the acoustic wave is transformed into a digital signal to fit for voice processing. A microphone is used to convert the acoustic wave into an analogue electrical signal. This analogue signal is conditioned with antialiasing filtering (and possible additional filtering for compensation). The antialiasing filter limits the bandwidth of the signal to approximately the Nyquist rate (half the sampling rate) before sampling (Campbell, 1997).

The speech signal of the voice database of ELSDSR was created by the Ph. D and master students from the department of Informatics and Mathematical Modeling (IMM) of the Technical University of Denmark (DTU). The text language used was English, and contain voice messages from 22 speakers: 10 females, 12 males, with ages ranging from 24 years to 63 years old. The subjects (speakers) from this database were from different countries and different places of one country, the dialect of speaking English language in the database does not affect speaker recognition, since it is the voice features that are of interest. The file type--.wav was used to record the voice messages of the database, as MATLAB software recognizes only a .wav file audio format. The sampling frequency was chosen at 16 kHz with a bit rate of 16. The recording exercise lasted for a period of one month.

The text used in training part of the database was carried out with an attempt to capture all the possible pronunciations of English language at least once in a paragraph, this include the vowels, diphthongs, consonants and digraphs. Seven paragraphs of text with eleven sentences were constructed and recorded for training, each speaker read seven paragraphs one after the other, making seven records per speaker, which makes a total of one hundred and fifty-four (154) voice samples or utterances in the database.

Likewise, in the testing part, each speaker read two sentences curled from a passage of history of an Ancient Egypt and that of the History of Giza. The passages were dissected into forty-four sentences, making a total of forty-four (44) utterances or voice samples for the test.

2.2.3.5 Speech processing

In speech processing, the speech analysis helps in reducing the speech data (raw input data) to manageable quantity and to extract the information which represents all the acoustic properties that helps in representing or interpreting the speech signal (Prasad *et al.*, 2017). Speech processing removes the desired information from a speech signal and the analysis of the speech signals is based on short term spectral analysis. The signal is decomposed into short fixed-length speech frames, which form the

feature vectors (Kinnunen *et al.,* 2011). These vectors are then processed further to be used as models in both training and testing for recognition.

2.2.4 Speech Feature extraction

The extraction of the finest parametric representation of sound signals is an essential task to yield a better recognition process. The effectiveness of this stage is important for the next stage since it affects its behaviour (Muda *et al.,* 2010).

The most robust and commonly used method for extraction features of a speech signal is the Mel frequency cepstral coefficients (MFCC). It is a non-parametric technique for modelling the human hearing perception system, where speech signal samples are processed to get voice features.

Principally, MFCC is based on the known distinction of the human ear's critical bandwidths with frequency filters that were linearly spaced at low frequencies and also logarithmically spaced at high frequencies, in order to capture the phonetically important characteristics of a speech signal. This process was explicitly expressed in the mel-frequency scale, which has a linear frequency spacing below 1000 Hz and a logarithmic spacing above 1000 Hz (Kumar & Rao, 2011). Block representation of MFCC process is shown in Fig. 2.4 as presented by (Uddin *et al.,* 2016).

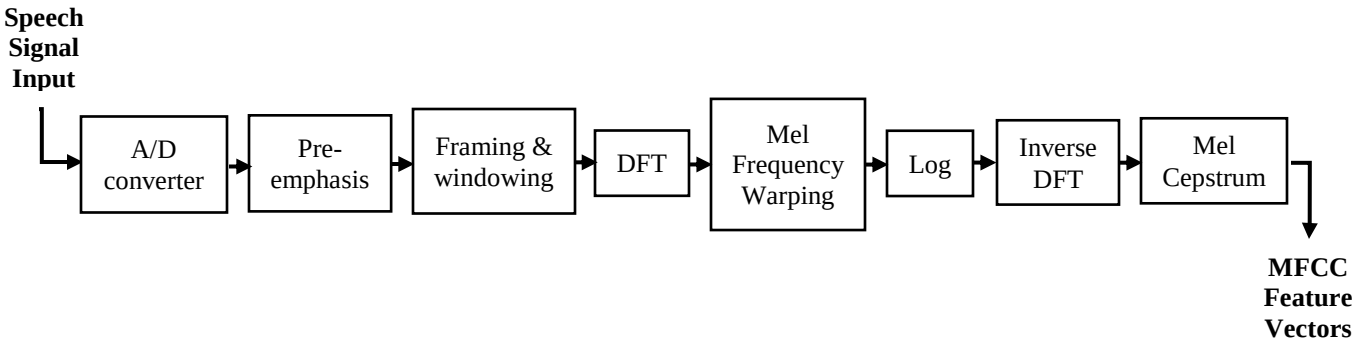


Fig. 2.4: Block Diagram Representation of MFCC Process (Uddin *et al.,* 2016).

2.2.5 Classification and feature matching

In speaker recognition system, the main problem lies in the pattern recognition, and in a much broader view, this problem belongs to a generic topic in science and engineering with the aim of minimizing mismatch error and improve recognition accuracy (Kumar & Rao, 2011).

The aim of pattern recognition is to classify objects of interest into one of a number of groups or classes. The objects of interest are commonly called patterns, while in this research they are the sequences of acoustic vectors extracted from the speech signal using the MFCC process described earlier. The groups or classes here refer to individual speakers (Kinnunen *et al.,* 2011).

Classification is the problem of identifying to which set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known (Tang *et al.,,* 2014). Classification is an unsupervised technique in data clustering that aims at grouping of similar samples into groups called clusters, each cluster has maximum within-cluster similarity and minimum between-cluster similarity based on certain similarity index (Aggarwal & Reddy, 2014).

Since the classification procedure in this case is applied on extracted features, it can be also referred to as feature matching. The extracted feature vectors are then further quantized, to locate clusters in the feature space and to reduce the amount of data after quantization. A codebook that contains cluster centroids, denoted as code vectors will be generated at the end of this process. The codebook represents the speaker model by approximating the distribution of the feature vectors in the feature space. The purpose of this vector quantization is to classify the extracted feature signals of various speakers (Kinnunen *et al.,,* 2011).

The most prevalent techniques used by researchers for feature matching in speaker recognition were;

1. Dynamic Time Warping (DTW): it uses template models, and the algorithm work based on Dynamic Programming techniques that measures similarity between two time series which may vary in time or speed (Salvador & Chan, 2007). It is a sequence of templates $(\overline{X_1}, \ldots\ldots, \bar{X}n)$ that must be matched to an input sequence $(X_1, \ldots\ldots, Xm)$, where n≠m, because of timing inconsistencies in human speech. The asymmetric match score $\mathcal{Z}$ is given by (Campbell, 1997)

$$\mathcal{Z} = \sum_{i=1}^{M} d\,(x_i, \bar{x}_{j(i)}) \qquad (2.1)$$

where the template indexes $j(i)$ are given by a DTW algorithm. Given reference and input signals, the DTW algorithm does a constrained, piece-wise linear mapping of one (or both) time axis(es) to align the two signals while minimizing $\mathcal{Z}$, at the end of the time warping, the accumulated distance is the basis of the match score. Principally, DTW is to compare two dynamic patterns and measure its similarity by calculating a minimum distance between them.

2. Hidden Markov Modeling (HMM): HMM, as in Fig. 2.5 (Das & Nahar, 2016), refers to double stochastic process in which the experimental data is assumed to be the result of having passed a concealed process through second process. Using a stochastic model, the pattern-matching problem can be expressed as measuring the probability of an observation (a feature vector of a collection of vectors from the unknown speaker) given the speaker model. The observation is an arbitrary vector with a conditional pdf (probability density function) that depends upon the speaker. The conditional pdf for the claimed speaker can be assessed from a set of training vectors, and, given the estimated density, the likelihood that the observation was generated by the claimed speaker can be determined (Campbell, 1997).

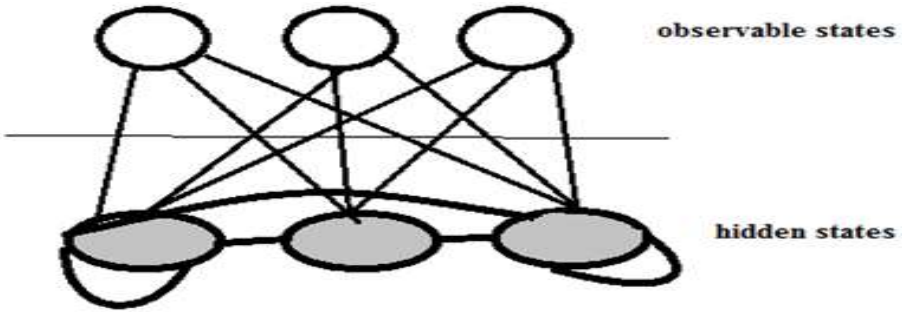


Fig. 2.5: Basic HMM Architecture (Das & Nahar, 2016)

3. Vector Quantization (VQ): VQ is a process of mapping vectors from a large vector space to a finite number of regions in that space, each region is called a cluster and can be represented by its centroid called a codeword. The collection of all codewords is referred to as codebook. An unknown voice after extracting voice feature vectors will be compared with the codebook of each speaker in the speaker database and distortion will be computed. Unknown voice will have minimum distortion with the true speaker (Kumar & Rao, 2011).

Different researchers developed various methods of classification using VQ approach, among which are Generalized Lloyd Algorithm (GLA) (Linde *et al.*, 1980), Self-Organizing Map (SOM) (Nasrabadi & Feng, 1988), Pairwise Nearest Neighbour (PNN) (Equitz, 1989), Randomized Local Search (RLS) (Fränti & Kivijärvi, 2000), etc. Some of the most recent variants are the Learning Vector Quantization (LVQ) neural networks (Haldar & Mishra, 2016); (Ge *et al.*, 2017).

However, nature inspired metaheuristic optimization algorithms are known to have a proven efficiency in solving many optimization problems (Yang, 2012). This research developed and used a modified nature inspired metaheuristic algorithm called Cuckoo Search Algorithm (CSA) for the classification of the extracted feature vectors of the speech signal. CSA is a swarm-intelligence-based algorithm that has been developed to address clustering related problems (Fister Jr *et al.*, 2013), but being it a metaheuristic process (that is problem independent) it is being used to solve various optimization problems. Therefore,

applying CSA in this context has greatly assisted in minimizing the mismatch error associated with voice recognition system, and has improved the recognition accuracy.

Cuckoo Search Algorithm (CSA) is a nature inspired metaheuristic algorithm, developed based on the natural brood parasitic behaviour of some Cuckoo species in combination with Lèvy flight behavior of some birds and fruit flies. An overview of Cuckoo bird, Lèvy flight and CSA will be relevant to this research work.

2.2.6 Cuckoo bird and its breeding behaviour

Cuckoos are solitary, skulking birds, more often heard than seen. The brood-parasitic cuckoos have no obvious long term pair bond, male and female lives separate and come together only to copulate (Payne *et al.,* 2005). Female cuckoos remove host egg to make room for their nestling cuckoo which then removes the others egg by evicting them from the nest. Many cuckoo hosts remove an egg from their nest if it looks unlike their own eggs, and this rejection behavior is the main line of defense against cuckoo parasitism. When the young cuckoo hatches, however, the foster parents accept the cuckoo and rear it at the cost of their own young (Davies & Cuckoos, 2000). Some host birds can engage direct conflict with the intruding cuckoos, if a host bird discovers the eggs are not their own, they will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere. Some cuckoo species such as the New World brood-parasitic Tapera have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry in colour and pattern of the eggs of a few chosen host species, this reduces the probability of their eggs being abandoned (Payne *et al.*, 2005). A typical image of a cuckoo bird is as shown in fig.2.6 as presented by (Fister Jr *et al.*, 2013)

Fig.2.6: Typical Cuckoo Bird (Fister Jr *et al.*, 2013)

2.2.7 Lèvy flight behaviour

Lévy flights or anomalous diffusion processes are known in the mathematical literature under the name of α-stable Lévy processes. They have infinite variance (except for the Gaussian case α = 2) and possess scale-invariance and self-similarity properties (Pantaleo *et al.,* 2009). Lèvy flights are a particular class of generalized random walk in which the step lengths during the walk are described by a 'heavy tailed' probability distribution (Barthelemy *et al.,* 2008).

Extensive investigations in diffusion processes have revealed that there exist some processes not obeying Brownian motion. One class of processes is enhanced diffusion, which has been shown to comply with Lèvy flights (Tran *et al.,* 2014). The Lèvy flights can be approximately characterized by following probability density function (Tran *et al.,* 2014):

$$P(x) \sim |x||x|^{-1-\beta}, \quad as\ x \to \infty \tag{2.2}$$

where $0 < \beta \leq 2$

In a Lévy flight, the steps of the random walk process have a power law distribution, meaning that extremely long jumps can occur. Consequently, the average step length diverges and the diffusion approximation breaks down for Lévy flights(Barthelemy *et al.,* 2008). A pattern of Lévy behavior is shown in fig. 2.7, as presented by (Tran *et al.,* 2014)

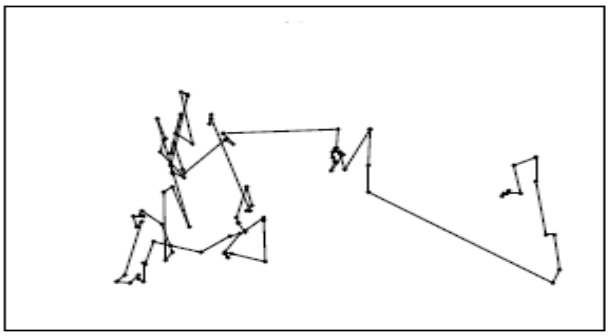


Fig. 2.7: Pattern of Lèvy Flight Behavior (Tran *et al.,* 2014)

The flight behaviour of many animals and insects has demonstrated the typical characteristics of Lèvy flights as described in various scholarly articles. Studies on human behaviour such as the Ju/'hoansi hunter-gatherer foraging patterns shows the typical feature of Lèvy flights (Brown *et al.,* 2007), fruit flies or *Drosophila melanogaster*, explore their landscape using a series of straight flight paths punctuated by a sudden 90^0 turn, leading to a Lèvy-flight-style intermittent scale free search pattern (Reynolds & Frye, 2007).

2.2.8 Cuckoo search algorithm (CSA)

Cuckoo search algorithm is a swarm intelligence algorithm inspired from the reproduction strategy of the cuckoo birds in combination with the Lèvy flight behavior of some birds and fruit flies. The cuckoo birds lay their eggs in a communal nest, and they may remove other's eggs to increase the probability of hatching their own eggs (El Aziz & Hassanien, 2016). It was developed by Yang and Deb in 2009 based on the brood parasitism of some cuckoo species (Fister Jr *et al.,* 2013), in combination with the Lèvy flight behavior of some birds and fruit flies (Yang & Deb, 2009).

To summarize the description of the CSA, three idealized rules are used (Yang & Deb, 2009):

1. Each cuckoo lays one egg at a time, and dump its egg in randomly chosen nest;
2. The best nests with high quality of eggs will carry over to the next generations;

3. The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability $pa \in [0, 1]$. In this case, the host bird can either throw the egg away or abandon the nest, and build a completely new nest. For simplicity, this assumption can be approximated by the fraction *pa* of the *n* nests, and are replaced by new nests (with new random solutions) (Yang & Deb, 2009).

From an implementation point of view, the simple representations of these idealized rules are used; that each egg in a nest represents a solution, and each cuckoo can lay only one egg (representing one solution), although each nest can have multiple eggs representing a set of solutions (Yang & Deb, 2014). The task of CSA is to generate new and potentially better solutions that will replace the worse solutions in the current nest population. The quality of solutions is evaluated with the objective function of the problem to be solved (which is the fitness function). The last rule is approximated by an additional parameter *pa* called the switching probability which determines when the worst of the *n* host nests is replaced by a new randomly generated nest. In fact, this parameter balances two components of the CSA process, i.e. exploration and exploitation (Fister Jr *et al.*, 2013).

The algorithm uses a balanced combination of a local random walk and the global explorative random walk, controlled by the switching parameter (*pa*). The local random walk is shown as follows (Yang & Deb, 2014):

$$x_i^{t+1} = x_i^t + \alpha s \otimes \mathrm{H}(p_a - \epsilon) \otimes \left(x_j^t - x_k^t\right) \tag{2.3}$$

where; $x_j^t \; and \; x_k^t$ are two different solutions selected randomly by random permutation, H(u) is a Heaviside function which is a unit step discontinuous function whose value is zero for negative

argument and one for positive argument (Weisstein, 2002), ∈ is a random number drawn from a uniform distribution, and *s* is the step size.

On the other hand, the global random walk intended for exploration of the search space is carried out using Lévy flights as expressed by (Yang & Deb, 2014):

$$x_i^{t+1} = x_i^t + \alpha \oplus \mathcal{L}(s, \lambda) \tag{2.4}$$

where Lèvy ($\mathcal{L}$) = $\mathcal{L}(s, \lambda) = \frac{\lambda \Gamma(\lambda) \sin(\pi \frac{\lambda}{2})}{\pi} \frac{1}{s^{1+\lambda}}$, $(s \gg s_0 > 0)$. (2.5)

Here, Γis gamma function extended from factorial function of positive real number, $\lambda = (1 + \beta)$, and β controls distribution by $0 \leq \beta \leq 2$.

In equation (2.3), when generating new solutions x_i^{t+1} for, say, a cuckoo *i*, a Lévy flight is performed where α > 0 is the step size which should be related to the scales of the problem of interests. The product ⊕ means entry-wise multiplications which is similar to those used in PSO, but here the random walk via Lévy flight is more efficient in exploring the search space as its step length is much longer in the long run (Yang & Deb, 2009). The Lévy flight essentially provides a random walk while the random step length is drawn from a Lévy distribution (equation 2.4), which has an infinite variance with an infinite mean. Here the steps essentially form a random walk process with a power-law step-length distribution with a heavy tail. Some of the new solutions should be generated by Lévy walk around the best solution obtained so far, this will speed up the local search. However, a substantial fraction of the new solutions should be generated by far field randomization and whose locations should be far enough from the current best solution, this will make sure the system will not be trapped in a local optimum (Yang & Deb, 2009).

The term $\mathcal{L}(s, \lambda)$ determines the characteristic scale and $\alpha > 0$ is a scaling factor of the step size s. The characteristic scale $\mathcal{L}$ depends on the problem to be solved. For instance, the $\alpha = O(L/10)$ is suitable when the dimensionality of the problem is small. In contrast, when the dimensionality of the problem is large, the $\alpha = O(L/100)$ is more appropriate. In this case, modifications are smaller and therefore, prevailing the cuckoos to move too far in the search space (Fister Jr *et al.,* 2013). A flowchart for the process of a standard cuckoo search algorithm is as shown in fig. 2.8.

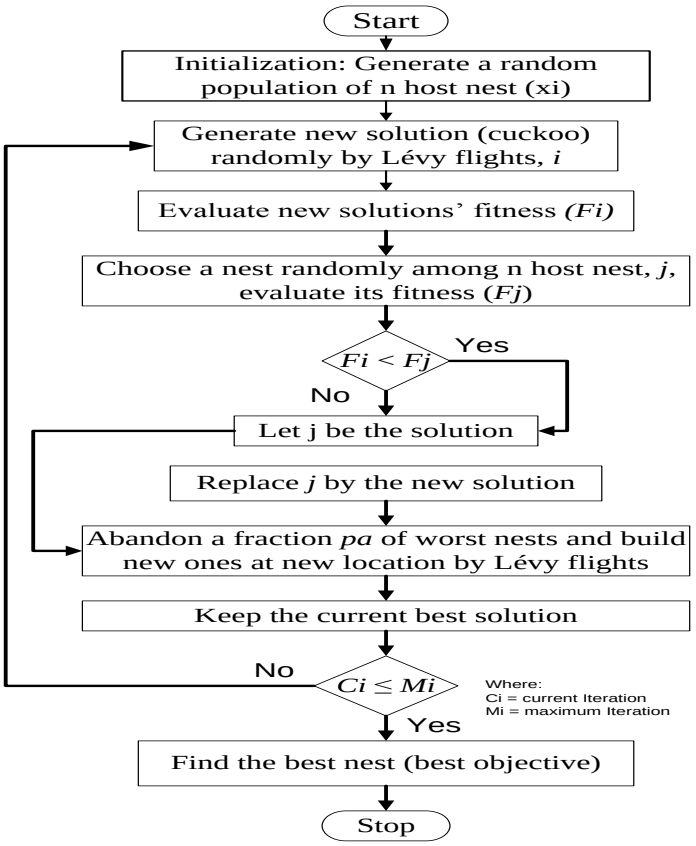


Fig. 2.8: Flowchart of Standard CSA

However, CSA has an inherent problem of having fixed control parameters (pa and α) which affects its accuracy and convergence ability. Hence, for the purpose of this research work, a dynamic CSA was

developed to address these problems, and it was applied in the VRS for the classification process to obtain an optimally extracted feature vectors. An inertia weight factor was introduced to the control parameters to make the developed algorithm become dynamic with respect to the position of Cuckoo to a solution in the solution search space.

2.2.9 Inertia weight factor

The concept on inertia weight was first introduced in 1998 by Shi and Ebahart for the purpose of tuning the parameters of PSO algorithm (Bansal *et al.,* 2011) and defined it as a decreasing function of time instead of a fixed constant. The inertia weight is also defined as a function of evolution speed factor and aggregation degree factor which changes dynamically based on the run and evolution (Chauhan *et al.*, 2013). Large inertia weight facilitates a global search while a small inertia weight facilitates a local search (Shi & Eberhart, 1998). Inertia weight plays a key role in the process of providing balance between exploration and exploitation process (Bansal *et al.,* 2011). Different inertia weight approaches could be categorized as linear strategy, nonlinear strategy, exponential strategy, adaptive or self-adaptive strategies, distribution based random adjustments, fuzzy rules based strategy and chaotic strategies (Chauhan *et al.,* 2013). Thus, the role of the inertia weight is considered crucial for convergence behavior because the inertia weight regulates the trade-off between the global (wide-ranging) and the local (nearby) exploration abilities of a swarm algorithm. A large inertia weight facilitates global exploration (searching new areas), while a small one tends to facilitate local exploration (i.e. fine tuning the current search area). A suitable value for the inertia weight provides balance between the global and local exploration ability of the swarm, resulting in better convergence rates (Ojha & Das, 2012).

Random inertia weight strategy introduced by (Eberhart & Shi, 2001) is used in this research in order to enhance the performance of the CSA, and the expression is as shown in equation (2.5) (Eberhart & Shi, 2001).

$$w = 0.5 + \frac{rand}{2} \tag{2.6}$$

The developed algorithm (dCSA) was then applied to optimally classify the extracted features vectors in the VRS. The flowchart of the process is as shown in Fig. 2.9.

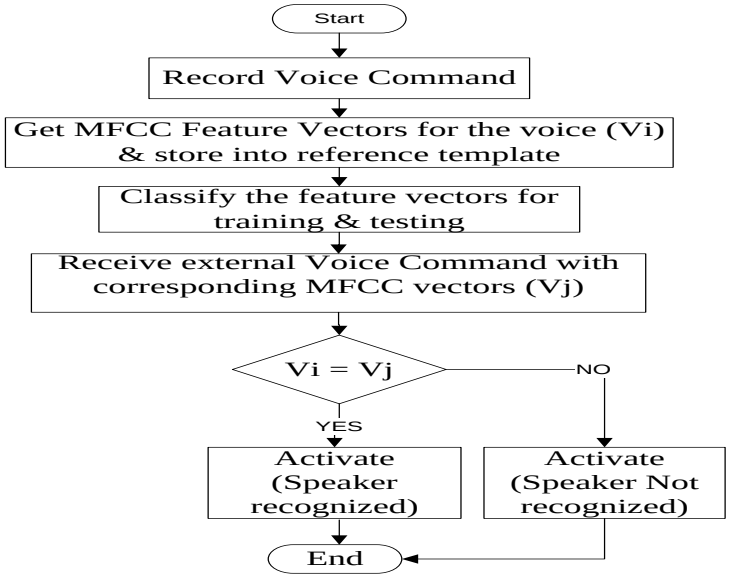


Fig. 2.9: Flow Chart for Speaker Recognition System.

2.2.10 CSA based classification

CSA is an effective swarm intelligence optimization technique for solving problems that works based on probability rules and population and has inherent clustering capabilities (Manikandan & Selvarajan, 2014). Thus, it is feasible to solve clustering problem using CSA (Zhao *et al.*, 2016). Clustering is essentially unsupervised learning technique for grouping a set of data objects into multiple groups or clusters so that objects within a cluster have high similarity, but are very dissimilar to objects in other clusters. Dissimilarities and similarities are assessed based on the attribute values describing the objects and often involve distance measures (Zhao *et al.*, 2014). It is worth noting that classification is basically a clustering problem.

CSA is used to evaluate the fitness of particles (in this case, feature vectors) with a distance measure, which is the objective function that needs to be minimized. The data (feature vectors) is divided into training set and testing set. The training set data is used to generate the cluster centres. The distance measure is defined as (Senthilnath *et al.*, 2013):

$$f_{(k)} = f_{(k)} = \sum_{k=1}^{k} \sum_{i=1}^{n_k} (x_i - c_k)^2 \quad (2.8)$$

where k=1,2,... K is the number of clusters, $x_i, i = 1,2, \ldots\ldots, n_k$ are the patterns in the k^{th} cluster, c_k is centre of the k^{th} cluster. Here the cluster centres are represented by

$$c_k = \frac{1}{n_k} \sum_{i=1}^{n_k} x_i \quad (2.9)$$

In this research, the nature-inspired dCSA algorithm was used to find the cluster centres from the training data set. This is done by placing each object to their respective cluster centres using the distance measure. The testing dataset is used to calculate percentage error using classification matrix (Senthilnath *et al.*, 2013).

2.2.11 Matching technique

Nearest neighbours (NN) method is a technique that keeps all the training data and can, therefore, use them as temporal information. The inter frame distance matrix is computed by measuring the distance between test-session frames (the input) and the claimant's enrolment-session frames (stored). The NN distance is the minimum distance between a test-session frame and the enrolment frames. The NN distances for all the test-session frames are then averaged to form a match score. Similarly, the test-session frames are also compared against a set of stored reference "cohort" speakers to form match scores. The match scores are then combined to form a probability ratio approximation (Campbell, 1997). Fig. 2.10 shows the matching technique of a Nearest Neighbour as illustrated by (Campbell, 1997).

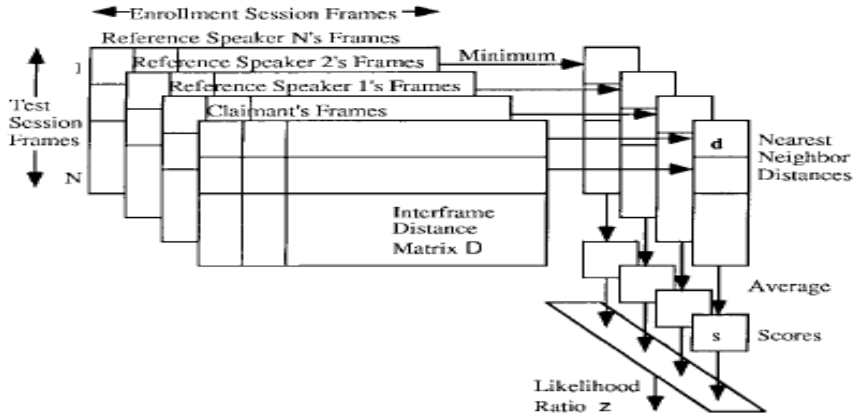


Fig. 2.10: Nearest Neighbour Technique (Campbell, 1997)

2.2.12 Decision theory

Having calculated a match score between the input speech-feature vector and a model of the claimed speaker's voice, a confirmation decision is to be made whether to accept or reject the speaker or to request for another utterance (or, without a claimed identity, a verification decision is made) (Campbell, 1997). Accept or reject decision process can be the form of, accept, continue, time-out, or reject problem. In this case, the decision-making procedure was a threshold level setting, if the test signal meets a threshold requirement, a decision will be taken as either accept/recognised or reject/not recognised.

2.2.13 Optimization test functions

Numerous metaheuristic algorithms have been successfully applied to many optimization problems (Li *et al.,* 2013). However, the test of reliability, efficiency and validation of optimization algorithms is frequently carried out by using a chosen set of common standard benchmarks or test functions from the literature (Jamil & Yang, 2013). A common practice followed by many researches is to compare different algorithms on a large test set, especially when the test involves function optimization (Jamil & Yang, 2013). There are many benchmark test functions in literature that are designed to test the

performance of optimization algorithm (Yang & Deb, 2009). Ten of such optimization test functions used in this research work are as follows:

1. **Ackleys' function**

   This is one of the classical benchmark functions used in many continuous optimization test suites. It has a lot of local minima around the global minima, with mathematical equation expressed as (Li *et al.*, 2013):

$$f_{(x)} = -20\exp\left[-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right] - \exp\left[\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right] + 20 + e \qquad (2.10)$$

   Where $n = 1, 2, \ldots$ and its' test area is typically limited to hyphercube $-32.768 \leq x_i \leq 32.768$ for i=1,2,…,n. The function has global minimum at $f_{(x)} = 0, at\ x_* = (0,0, \ldots\ldots,0)$.

2. **De Jong's first function**

   De Jong function is one of the simplest test benchmark function, which is continuous, convex and unimodal. It has the following mathematical expression (Molga & Smutnicki, 2005):

$$f_{(x)} = \sum_{i=1}^{n} x_i^2. \qquad (2.11)$$

   Test area is usually restricted to hypercube $-5.12 \leq x_i \leq 5.12, i = 1, \ldots . n.$ with global minimum at $`f_{(x)} = 0, for\ x_i = 0, i = 1, \ldots, n.$

3. **Easom's function**

   This is another unimodal test function, with the global minimum having a small area comparative to the search space. The function is upturned for minimization. Having only two variables and the resulting mathematical description (Molga & Smutnicki, 2005):

$$f_{(x,y)} = -\cos(x)\cos(y)\exp(-(x-\pi)^2 - (y-\pi)^2) \qquad (2.12)$$

   The function has a global minimum of $f_{\{x\}} = -1$ at (π, π) in a minute region.

4. **Griewangk's function**

Griewangk's function contains many widespread local minima regularly distributed, but a single global minimum, the function is expressed mathematically as (Yang & Deb, 2009):

$$f_{(x)} = \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1. \quad (2.13)$$

Its' global minimum is $f_{(x)}$=0, at x_i=0, for i=1,…n.

The function analysis varies with the scale; the common overview recommends convex function, medium-scale view recommends existence of local extremum, and finally zoom on the details shows compound structure of several local extremum (Molga & Smutnicki, 2005).

5. Michalewicz's function

This function is a multimodal test function (owns ***d!*** local optima). The "steepness" of the valleys or edges is defined by a parameter 'm'. The more the size of m the more difficult the search become. When the size of m become very large, the function behaves like a needle in the haystack (values outside the narrow peaks of points in the space gives slight information on the position of the global optimum). The function is mathematically expressed as (Molga & Smutnicki, 2005):

$$f_{(x)} = -\sum_{i=1}^{d} \sin(x_i)\left[\sin(\frac{ix_i^2}{\pi})\right]^{2m}, (m=10), \quad (2.14)$$

$where\ 0 \le x_i \le \pi\ and\ i = 1,2,.,d.$ The global minimum is $f_{(x)} \approx -1.801\ for\ d = 2,$ $while\ f_{(x)} \approx -4.6877\ for\ d = 5.$

6. Rastrigin's function

Rastrigin's function is based on the function of De Jong with the addition of cosine modulation in order to produce frequent local minima. Thus, the test function is highly multimodal. However, the locations of the minima are regularly distributed. The function is defined mathematically as (Molga & Smutnicki, 2005):

$$f_{(x)} = 10d + \sum_{i=1}^{d}[x_i^2 - 10\cos(2\pi x_i]. \quad (2.15)$$

It has a global minimum $f_{(x)} = 0, at\ (0,0,..,0)\ for - 5.12 \leq x_i \leq 5.12$, where *i*=1,2,.,d.

7. Rosenbrock's fuction

Rosenbrock's function valley is a classic optimization problem, which is also known as Banana function (Tang *et al.,* 2007). The global optimum is inside a long, narrow, parabolic shaped flat valley. To find the valley is trivial, however convergence to the global optimum is difficult and hence this problem has been repeatedly used in assessing the performance of optimization algorithms. The function is mathematically represented as (Yang, 2010):

$$f(x) = \sum_{i=1}^{d-1}((x_i - 1)^2 + 100(x_{i+1} - x_i^2)^2) \quad (2.16)$$

This function has global minimum $f_{(x)} = 0$ occurs at $x_i = (1,1,...,1)$ in the domain of $-2.048 \leq x_i \leq 2.048$ where $i = 1, 2, ..., \mathrm{d}-1$. In the 2D case, it is often written as

$$f(x, y) = (x - 1)^2 + 100(y - x^2)^2 \quad (2.17)$$

8. Schwefel's function

Schwefel's function is deceptive in that the global minimum is geometrically distant over the parameter space, from the next best local minima. It is also a multimodal function with the following definition (Yang & Deb, 2009);

$$f_{(x)} = \sum_{i=1}^{d}\left[-x_i sin\left(\sqrt{|x_i|}\right)\right], \quad (2.18)$$

Test areas is usually restricted to hypercube$-500 \leq x_i \leq 500, for\ i = 1,..,d$. Its' global minimum $f_{(x)} = -418.9829n\ obtainable\ for\ x_i = 420.9687, for\ i = 1, \ldots, d.$

9. Shubert's bivariate function

This is also a multimodal test function which has only two variables and mathematically expressed as (Yang & Deb, 2009);

$$f_{(x,y)} = -\sum_{i=1}^{5} i\, cos[(i+1)x+1] \sum_{i=1}^{5} cos[(i+1)y+1], \quad (2.19)$$

It has 18 global minima in the region $(x, y) \in [-10,10] \times [-10,10]$. The value of its' global minima is $f_{(x)} = -186.7309$.

10. Sphere test function

This is the simplest form of De Jong function. Sphere function is mathematically represented as follows (Yang, 2010)

$$f(x) = \sum_{i=1}^{d} x_i^2 \quad (2.20)$$

This function is unimodal and convex with an obvious local minimum of $f_* = 0$ at $x_* = (0,0,...,0)$ in a domain of $-15 \le x_i \le 15$.

2.3 Review of Similar Works

The review of similar works performed in this research work is categorized into two, review of works done based on Voice Recognition System (VRS), and reviews done based on modifications of Cuckoo Search Algorithm (CSA).

2.3.1 Review of works based on voice recognition system

This section focuses on the review of works done based on VRS.

Zulfiqar *et al.* (2009) developed a speaker identification system using MFCC Features with VQ Technique, in their work they used MFCC as the feature extraction tool and Vector Quantization (VQ) technique was used for classification and matching, two purposeful speech databases of voice samples with added noise, recorded at sampling frequencies of 8000 Hz and 11025Hz from 44 different speakers for training and testing were used. They carried out two experiments and results obtained shows an

improvement from 2.27% to 13.64% when number of vectors were doubled in each VQ codebook. However, the computation and storage complexity in a Vector Quantization (VQ) technique is an exponential function of the number of bits used in quantizing each frame of spectral information. Likewise, the clustering method in VQ lacks effective cluster center assignment, hence, this affects accuracy of recognition of the VR system.

Muda *et al.* (2010) developed voice recognition algorithms using MFCC and DTW techniques, they used non-parametric method to model the human hearing perception system of MFCC as the extraction systems. Subsequently. DTW algorithm was used as the matching technique, which produced warping function that minimizes the total distance between the respective points of the signal. They also used the accumulated distance matrix to develop mapping paths which travel through the cells with smallest accumulated distances, then the total distance difference between these two signals was minimized. Thus, at the end optimal warping path was achieved where the test input matched with the reference template. However, DTW is a classical and deterministic method that lacks the ability to model stochastic signals. It also has less control of the possible routes of the warping path, and this can lead to lead to unintuitive alignments where a single point on one-time series maps onto a large subsection of another time series. Thus, recognition rate is affected by these limitations.

Yadav and Mandal (2011) developed speaker recognition system using Particle Swarm Optimization (PSO), in their work, they used two speaker databases in the simulation experiment, one for training and the other one for testing containing 11 speakers in each database, Fast Fourier Transform (FFT) was used to extract features, and later on feed the extracted features vectors to a trained classifier and return the identity of the speaker in a real-time fashion. The training was done using Artificial Neural Network (ANN) and then optimized with PSO as the classifier. Matlab 7.5 was the simulating tool, and result obtained shows a relative improvement, despite the fact that the speakers' society were different.

However, PSO algorithm has fast convergence rate which get it stuck into local optima, thus, optimal solution may not have been reached before its convergence, and this will affect recognition accuracy of the voice recognition system.

Nemati and Basiri (2011) presented text-independent speaker verification using Ant Colony Optimization (ACO) based selected features. At the top level, there is lexical and syntactic features, below that are prosodic features, further below these are phonetic features, and at the most basic level there is low-level acoustic features. So they used ACO to select and eliminate some low level features of the speech signal. However, their approach may reduce the recognition rate going by the kind of information that may be contained in those acoustic features if they were eliminated. Likewise, the algorithm has slow convergence and can get trapped into local optima, thus, the accuracy of recognition will also be affected in trying to get the minimum value to eliminate from acoustic features.

Sood and Kaur (2013) developed a speaker recognition system based on CSA. In their work, the CSA was used for feature extraction and reduction. The extracted features by CSA were compared with the inputted voice's features using correlation and the closest features to the stored features were then matched. To evade matching of voices in all situations, even in case of un- authenticated speaker, a limit of value was set in order to increase security and to appropriately validate or discard a speaker. This limit specified a possibility ratio, which signified the degree of match of speaker recognition. The method was tested for stored voices data and then for real time voice input. The simulation result showed that a speaker is recognized only if the voice sample was only present in the voice database. However, the extracted features may lack precision due to the inherent problem associated with the fixed control parameters of the CSA which gets it trapped into local optima and reduce its accuracy.

Shah *et al.* (2014) presented a biometric voice recognition in security system, in their work, they designed a voice recognition system in order to identify an administrator voice. They used MFCC for

the extraction process of the speech signal and vector quantization technique for clustering and feature matching process. Matlab and Arduino were used as the simulation software, Matlab was used for the voice recognition part while the Arduino served as an interface that communicate the speaker with the voice recognition system and the controls that activate a magnetically controlled door, if the administrator is recognized the door will unlock otherwise the door will remain locked, and an alarm will be triggered by the Arduino for 1 second. Ten people were experimented on the system and result shows significant improvement with Mean Square Error (MSE) difference between the administrator and the imposters ranging from 0.9038 to 19.0037. However, the clustering method in voice quantization lacks effective assignment of cluster centers, hence, this will affect the accuracy of recognition of the VR system. It also has high computational and storage complexity due to exponential rise of the number of bits used in quantizing each frame of spectral information.

Dash and Mohanty (2014) used CSA for feature extraction in finding and short listing the features from voice which can be uniquely identified while using a threshold to remove the undesirable signal or noise. The unique and best features were determined using a fitness function centered on mean of the individual sample, while discarding the remaining unwanted sample signals. The system was tested for stored voice dataset and for real time voice input. The simulation result showed that the maximum value of *9.*54 was obtained which matched with the third voice sample of the database. However, the process of the speaker recognition is optimized by matching of voices only on the extracted features produced by CSA, but CSA has a problem of slow convergence and can easily get stuck into local optima and so affect its accuracy, hence, recognition rate will be affected by this problem.

Harrag (2015) compared the performance of GA and PSO in the development of feature subset selection application to Arabic speaker recognition system, he used the algorithms to pick subsets of selected features of the speech signal after extraction process by MFCC. The subset selection is a pre-

classification process that further categorizes the extracted feature vectors for easier classification with the sole purpose of reducing the dimensionality of the extracted vectors. However, this process may lead to removal of some vectors that are vital for classification process and the recognition.

Bansal *et al.* (2015) presented an automatic speech recognition system using a CSA-based Artificial Neural Network (ANN) Classifier. In their work, a novel meta-heuristic algorithm CSA was proposed to train neural network to achieve fast convergence rate and to increase the recognition accuracy of the speech recognition system. The ANN trained two kinds of acoustic features extracted by MFCC and Linear Predictive Coding Coefficients (LPCC). The proposed method was implemented on MATLAB and the experimental results showed that the proposed CSA- based ANN classifier provided best results in terms of convergence rate, simplicity, and accuracy as compared with a normal ANN-based classifier. However, the CSA used had the problem of fixed control parameters that affect its accuracy and performance, hence, recognition rate will also be affected in return.

Das and Nahar (2016) developed a voice identification system using HMM, they worked on both speaker and speech recognition system. They used MFCC technique in feature extraction and a statistical approach of VQ and HMM for the classification in the speaker and speech recognition processes respectively. However, HMM process has some peculiar problems, first, in the determination of best sequence of model states, then, in the adjustment of model parameters to account for the observed signal, and in the evaluation of the probability (or likelihood) of a sequence of observations given a specific HMM, thus, these problems will affect the classification process which will later affect the recognition rate.

Uddin *et al.* (2016) developed a voice recognition system to cater for taking care of marking the student attendance register in the lecture hall. In their approach they use Euclidian distance at the trained data and test data, they also used the same Euclidian distance for scoring or at the decision logic. However,

the vectors extracted using Euclidian distance can be corrupted with noise as no specific filtering process was performed, and this can affect the classification and matching technique, and later affect the recognition process as a whole.

It is evident from these literatures, that research work in the field of VRS using various classification techniques have received tremendous attention from researchers, leading to variety of techniques aimed at improving accuracy of recognition and reducing error mismatch in the VRS. However, development of an optimal classification scheme has ensured a better performance of VRS with respect to accuracy and precision of recognition of the VRS it also reduced mismatch error.

2.3.2 Research works on the cuckoo search algorithm modification

Some literatures relevant to CSA and modifications in the algorithm are described below;

Yang and Deb (2009) formulated a new metaheuristic algorithm, called CSA, for solving optimization problems. This algorithm was based on the obligate brood parasitic behaviour of some cuckoo species in combination with the Lévy flight behaviour of some birds and fruit flies. In their work, exploration of the algorithm was controlled by a heavy tailed levy flight step size while the switching probability balances exploration and exploitation using a combination of a local random walk and the global explorative random walk. The algorithm was tested and validated with ten benchmark test functions using Matlab, and the performance of the algorithm was compared with GA and PSO. Simulation results showed that CSA was superior to these existing algorithms for multimodal objective functions. This was partly due to the fact that there were fewer parameters fine-tuned in CSA than in PSO and GA. When compared with other metaheuristic algorithms, CSA was more generic and robust for many optimization problems. However, the algorithm had a slow convergence rate due to the fixed values of the control parameters (*pa* and α), which also affects its accuracy.

Valian *et al.* (2011) developed an improved CSA for global optimization by introducing a tuning strategy to the CSA control parameters (*pa* and $\boldsymbol{\alpha}$) in order to increase convergence rate and performance accuracy. In the developed algorithm, the values of *pa* and α are dynamically changed with the number of generation. The algorithm was simulated using Matlab and tested with fifteen test functions. The result was compared with that of the CSA which showed that the proposed algorithm outperformed the CSA in terms of convergence rate and accuracy of the solutions found for most of the test functions. However, in their algorithm, bounds were set for the control parameters and this limited the operational range of the parameters. This meant that optimized results outside of the constrained limits were not captured.

Walton *et al.* (2011) developed a modified CSA called a new gradient free optimization algorithm. Their modification involved the addition of information exchange between the top eggs (or the best solutions) in order to speed up convergence to the true global minimum. Also the step size ($\boldsymbol{\alpha}$) was modified by making the value of $\boldsymbol{\alpha}$ to be decreasing as the number of generations increases so as to encourage more localized searching as the individuals or eggs gets closer to the solution. The proposed algorithm was simulated in Matlab, using seven standard optimization benchmark test functions to test the effects of these modifications and it was demonstrated that, in most cases, the modified cuckoo search outperformed the standard CSA in all the test functions, while it is significantly better than PSO and DE in some cases. However, in their algorithm only one parameter (i.e. the step size) was made to be changing with the change in iteration, and when the step size α became small relative to the switching probability *pa*, convergence speed of the algorithm increased but with no guarantee that an optimum solution will be obtained.

Zhao and Li (2012) introduced the Opposition-Based Learning (OBL) method to the CSA. The OBL increased the solution search space of the algorithm by simultaneously considering a solution with its

corresponding opposite solution which in-turn increased the accuracy of the algorithm. The main idea of OBL is to generate "the opposites" of the newly generated solution and which in-turn increased the coverage of the solution space leading to increased accuracy. This is because an optimal solution may lie in opposition to the newly generated solutions. The proposed algorithm was tested on four (4) standard benchmark test functions and the simulation result proved the algorithm to be correct and effective in converging to the global optimal solution. However, the opposite values generated further deepened the iterative process thereby increasing the convergence time of the algorithm.

Yang and Deb (2013) formulated a new Multi-Objective Cuckoo Search (MOCS) optimization algorithm, which is an extension of the original CSA to solve multi-objective problems. In their work, they maintained exploration by Lévy flight of the standard CSA, while incorporating genetic operators as mutation by a combination of Lévy flights and vectorized solution difference, crossover by selective random permutation, and selective elitism to the algorithm. The algorithm was validated using multi-objective test functions, where selected subset of these functions with convex, non-convex and discontinuous Pareto fronts were applied to solve structural design problems. However, the developed algorithm is only aimed at addressing multi-objective problem not issue of slow convergence of the standard CSA.

Wang *et al.* (2014) proposed a chaotic CSA-based method, by introducing chaos to the standard CSA. They introduced 12-dimensional non-invertible maps (labeled M1-M12) to generate chaotic sets and the chaotic maps were then used to tune the step size (α) of the standard CSA. In addition, elitism strategy was introduced in their modification to protect best cuckoos from being corrupted by the updating operator. In their experiment it was discovered that the best map that produced best result was the sinusoidal map. The algorithm was experimented on 27 standard benchmark test functions and simulation results showed that the algorithm out-performed 8 other algorithms (i.e. GA, PSO, ACO, DE,

HS, ES, PBIL, and CSA). However, their work ended up tuning only one parameter of the CSA (that is α, which is the step size) while keeping the control probability (*pa*) constant. This limited the algorithm in finding the best solution if in the process of tuning the step size the value of *pa* became larger than the step size.

(Sun *et al.* (2017)) developed an improved CSA for coverage optimization of visible light communication in smart home by introducing chaos and dimension cell mechanism into CSA, such that instead of generating initial solution randomly via Lévy flight, they use logistic map rule based on chaos theory at the initial solution formation, which produced uniformly distributed solutions. They also introduced dimension cell updating mechanism that divided the dimension of the solution into several cells, hence, updating method became cell by cell instead of the entire solution space. The algorithm was simulated in Matlab and the experimental result showed that the algorithm performed better than DE, PSO, and CSA. However, same constant values for the control parameters in the original CSA were maintained in their algorithm, thus restricting it to certain specific dimensions rather than wider explorative search produced by Lévy distribution.

Pande *et al.* (2017) developed a hybrid CSA for twitter sentiment analysis by hybridizing the CSA with K-means clustering algorithm. The hybrid algorithm was tested and compared with the standard CSA, PSO, DE and ICS and experimental result showed that, the algorithm out-performed the mentioned algorithms. However, the inherent problem of K-means algorithm of initial assignment of centroid head was not addressed, as the algorithm still assigned cluster head using K-means then subsequently Cuckoo Search (CS) is introduced to generate new solution. The problem of fixed control parameters of CSA was not addressed in this work.

It is evident from the literatures reviewed, that research work towards the modification and improvement of CSA in order to increase its accuracy and convergence speed has been given significant attention by various scholars. However, the developed dCSA which introduced random inertia weights at the control parameters (*pa* and α) of the standard CSA, has greatly increased the accuracy and convergence speed of CSA. This approach makes the control parameters become dynamic with respect to the position of each cuckoo, which in turn reduced the tendency of the algorithm falling into local optima.

CHAPTER THREE

MATERIALS AND METHODS

3.1 Introduction

This chapter gives details of the methods, materials and procedures employed for the successful completion of this research work. A standard dataset was obtained, and a voice recognition system using a newly formulated algorithm (dCSA) was developed. The algorithm was used to optimally classify extracted feature vectors from the voice signal. The steps of the methodology were as listed in section 1.6.

3.2 Development of Speakers' Database.

Voice samples are primary factors in the development of a voice recognition system, hence, the need to have a good database. Standard voice datasets were obtained from the English Language Speech Database for Speaker Recognition (ELSDSR) of the Technical University of Denmark (DTU). The voice data was developed in a controlled environment to minimize noise. Details about ELSDSR, the recording of the voice data and recording environment were explained in the following sub-sections.

3.2.1 Obtaining standard voice dataset from ELSDSR of DTU

English Language Speech Database for Speaker Recognition (ELSDSR) is a corpus of read speeches that has been designed to provide speech data for the development and evaluation of automatic speaker recognition system. Details about the acquisition of the speech signal for the development of the voice database was discussed in sub-section 2.2.3.4. Passages and paragraphs containing a breakdown of sentences were provided to each speaker, for reading both during the training session and testing session. The transcript of the audio message used during the training session is shown in appendix A_1, while the transcript of the audio message used during testing session is shown in appendix A_2

The average duration (in seconds) for reading the training message was 78.6s (for male speakers) and

88.3s (for female speakers) and a total average of 83.5s for the whole. While the duration for reading the test message was 16.1s (for male speakers) and 19.6s (for female speakers) and a total of 17.9s for all speakers. For the purpose of this research work, names of the speakers were customized to bear the local names of researcher's environment, this is because, only initials of the speakers were used to label each speaker in the database, with M or F at the beginning of each initials to signify a male or female speaker. Table 3.1 shows the speakers identity with the time taken by each speaker to read a training paragraph and a test sentence.

Table 3.1: Speakers ID and Average Duration for Reading a Text per speaker

SN	MALES	TRAINING TIME (s)	TESTING TIME (s)	FEMALES	TRAINING TIME (s)	TESTING TIME (s)
1	ANGO	81.2	20.9	LAMI	99.1	18.7
2	AUDU	68.4	13.1	LADI	77.3	12.7
3	ANAS	91.6	15.8	JANE	92,8	24.0
4	UMAR	69.9	15.8	JOY	86.6	21.2
5	BALA	76.8	14.7	KATE	79.2	18.2

6	NURA	79.6	13.3	RABI	76.3	18.2
7	ADAM	73.1	10.9	FATI	99.1	24.1
8	JACK	82.9	20.3	BOSE	80.2	18.4
9	TJ	88.0	23.4	BUKY	102.9	15.8
10	MIKE	86.8	9.3	LUCY	89.5	25.1
11	SULE	79.1	21.8	-------	------	-----
12	JOHN	66.2	14.05	-------	------	-----

3.2.2 Recording environment

The recording of the voices was carried out in a computer room measuring 8.82m x 11.8m x 3.05m (width, length, and height of the room) with 22 monitors and 34 tables at the Technical University of Denmark (DTU). The recording was performed at the center of the room, using one microphone, with a table sizing 70cm x 120cm x 70cm in front of a speaker. Two deflection boards measuring 93cm x 211.5cm x 6cm, tilted at an angle, facing each other were placed in front of the table and the speaker. This was to ensure that reflection of the sound wave is deflected. Fig. 3.1(a) shows the plan view of the recording environment and Fig. 3.1(b) shows a 3D view of the recording environment (Feng & Hansen, 2005).

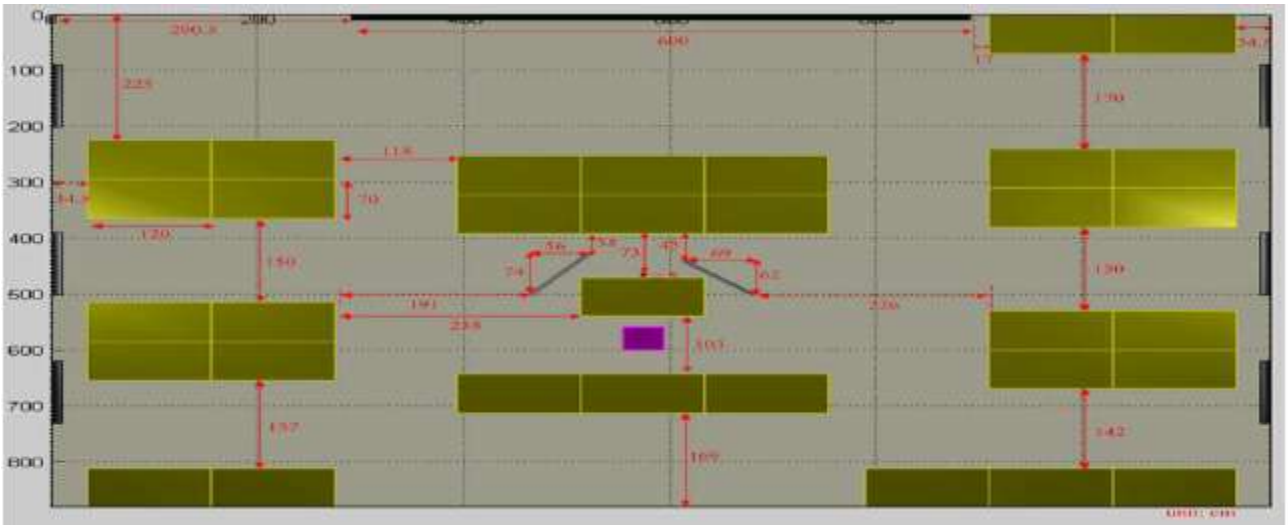


Fig. 3.1(a); Plan view of the Recording Environment (Feng & Hansen, 2005)

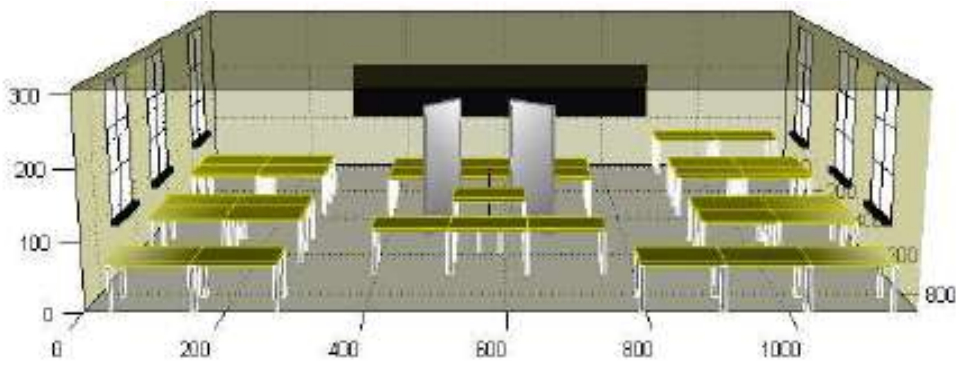


Fig.3.1 (b); 3D view of the Recording Environment (Feng & Hansen, 2005)

3.2.3 Recording equipment

The recording machine used was MARANTZ PMD670 portable solid state recorder. The machine supports recording in both compressed and uncompressed format, like MP2 and MP3 (for the compressed format) and linear Pulse Code Modulation (PCM) (for the uncompressed format). User can also choose the recording type (can either be stereo, mono or digital), and the file format can be recorded into .wav .bwf, .mpg or .mp3.

3.2.4 Extraction of voice features

The extraction of key voice features was carried out using Mel Frequency Ceptral Coefficient (MFCC), this process extracts the best parametric representation of the speech signal and present it in a vector form to be used for future referencing. The process ensures that a good representation of a speaker's identity was extracted from his voice. The implementation of this process was executed using MATLAB command shown in Fig. 3.2. The complete MATLAB file is shown in appendix C.

```
% Read speech samples, sampling rate and precision from file
[ speech, fs, nbits ] = wavread( wav_file );
% Feature extraction (feature vectors as columns)
[ MFCCs, FBEs, frames ] = ...
    mfcc( speech, fs, Tw, Ts, alpha, @hamming, [LF HF], M, C+1, L );
MFCCs(find(isnan(MFCCs))) = 0.0001;

mm = MFCCs;
```

Fig.3.2: Snippet of MATLAB Script for the Implementation of feature extraction with MFCC

3.2.5 Training of speakers extracted features

The training of the key extracted features was necessary in order to create a template for each speaker in the database. This training was conducted by making a speaker to read the provided text document for a certain number of times, seven paragraphs were provided to twenty-two speakers, when a speaker reads a paragraph, feature vectors of the speaker was extracted and was stored in a reference template of that speaker, this was done seven times which signifies the training session. Implementation of this command in MATLAB is shown in Fig. 3.3 overleaf. The complete MATLAB file for the whole process is shown in appendix C.

```
trainVoices = dir(fullfile('VOICE DATA/', '*.wav'));
for i =1:numel(trainVoices)
    persons = [persons; trainVoices(i).name(1:4)];
    wav_file = strcat('VOICE DATA/', trainVoices(i).name);
        mfcc( speech, fs, Tw, Ts, alpha, @hamming, [LF HF], M, C+1, L );
    MFCCs(find(isnan(MFCCs))) = 0.0001;
    mm = MFCCs;
    mf = mean(mm,2); cf = cov(mm');
    ff = mf;
    for i=0:(size(mm,1)-1)
        ff = [ff;diag(cf,i)];
    end
    fV = [fV; ff'];
end
```

Fig.3.3; Snippet of MATLAB Script for Training of Extracted Features

3.3 Development of dynamic Cuckoo Search Algorithm (dCSA)

The development of dynamic CSA was built upon the existing standard CSA, dynamic control parameters introduced in the standard CSA recorded an improvement that increased the accuracy and

convergence speed of the standard CSA. The processes were discussed in details in the next sub-sections.

3.3.1 Initialization of dCSA parameters

The performance of dCSA depends on the appropriate selection of its control parameters (population size, control probability, step size). For the purpose of replication and implementation of cuckoo search algorithm, some of the parameters like population size, control probability, step size reported in the literature (Yang & Deb, 2009) were adopted. The appropriate selected values of these parameters used for the development of dCSA were presented in Table 3.2.

Table 3.2: dCSA Simulation Parameters

SN	Simulation Parameter	Symbol	Value
1	Population Size (Nest)	N	15
2	Control Probability	Pa	$0 < pa \leq 0.25$.
3	Step size	A	$0.1 < \alpha \leq 1$

The simulation parameters presented in Table3.2 are dynamic, that is the parameters were ranged. The actual values used in the standard CSA were made to be fixed during iteration process (pa=0.25 and α=1)

(Yang & Deb, 2009). The complete MATLAB file for the algorithm can be found in Appendix B. The snippet of the MATLAB script for initializing random population of *n* host nest (which is the new solution) is shown in Fig. 3.4

```
% Random initial solutions
for i=1:n,
nest(i,:)=Lb+(Ub-Lb).*rand(size(Lb));
end
```

Fig. 3.4: Snippet of MATLAB Script for Initialization of Random Population of Host Nest

3.3.2 Introduction of inertia weight factor

In order to enhance the exploitation capability of CSA, the idea of inertia weight was introduced to the control parameters of CSA in the form of dynamic value of iteration weight and was implemented.

The dynamic value iteration weight given in equation (3.1) was introduced in order to improve the convergence speed and optimal performance of the standard CSA.

$$w = 0.5 + \frac{rand}{2} \tag{3.1}$$

Based on equation (3.1), the fixed control parameters of the standard CSA were made to be dynamic with respect to the position of the cuckoo as the iteration increases. The resulting control parameters i.e. control probability and step size are respectively modified as shown in equation 3.2 and 3.3.

$$pa = 0.25 \times (0.5 + \frac{rand}{2}) \tag{3.2}$$

$$\alpha = \mathrm{w} \times step \times (s - best) \tag{3.3}$$

where *rand* is a uniform random number, α is the step size, *w* is the inertia weight *s* is a randomly chosen nest and *best* is the current best solution.

The local search equation of the modified CSA is then written in equation (3.4)

$$x_i^{t+1} = x_i^t + \alpha w \oplus \mathcal{L}(s, \lambda) \tag{3.4}$$

Where the step size (α) controls the heavy tailed step size in generating new solutions. The implementation of this action in MATLAB can be seen as shown in Fig. 3.5

```
pa=0.25*(0.5+0.5*rand)
u=randn(size(s))*sigma;
v=randn(size(s));
step=u./abs(v).^(1/beta);
w=(0.5+(0.5*rand));
% w is the random inertia weight introduced
stepsize=w*step.*(s-best);
```

Fig. 3.5: Snippet of MATLAB Script for Inertia Weight Factor

3.3.3 Generation of new solution by lévy flight and updating cuckoo position

The generation of new solution was implemented by random walk through lévy flight using equation (2.4) and (2.5) respectively. A snippet of the MATLAB script for the generation of new solution is as

shown in Fig. 3.6. The MATLAB file for the algorithm can be found in Appendix B.

```
%% Get cuckoos by ramdom walk via Levy flight
function nest=get_cuckoos(nest,best,Lb,Ub)
% Levy flights
n=size(nest,1);
% Levy exponent and coefficient
beta=3/2;
sigma=(gamma(1+beta)*sin(pi*beta/2)/(gamma((1+beta)/2)*beta*2^((beta-1)/2)))^(1/beta);

for j=1:n,
    s=nest(j,:);
    %% Levy flights by Mantegna's algorithm
    u=randn(size(s))*sigma;
    v=randn(size(s));
    step=u./abs(v).^(1/beta);
    w=(0.5+(0.5*rand));
    % w is the random inertia weight introduced
    stepsize=w*step.*(s-best);
    s=s+stepsize.*randn(size(s));
   % Apply simple bounds/limits
   nest(j,:)=simplebounds(s,Lb,Ub);
end
```

Fig. 3.6: Snippet of MATLAB Script for Generation of new Solution in CSA

3.3.4 Evaluation and comparison of solutions.

To find the current best solution among all the generated solutions, the random initial solutions were compared with a randomly chosen solution (nest), they were then evaluated and compared, the current best solution was later then maintained for global search.

The implementation of the process is shown in Fig. 3.7, and the MATLAB file for the algorithm can be found in Appendix B.

```
%% Find the current best nest
function [fmin,best,nest,fitness]=get_best_nest(nest,newnest,fitness)
% Evaluating all new solutions
for j=1:size(nest,1),
    fnew=fobj(newnest(j,:));
    if fnew<=fitness(j),
       fitness(j)=fnew;
       nest(j,:)=newnest(j,:);
    end
end
% Find the current best
[fmin,K]=min(fitness) ;
best=nest(K,:);
```

Fig. 3.7: Snippet of MATLAB Script for obtaining current best solution

3.3.5 Replacement of worst solutions

In the real life, if a cuckoo's egg is very similar to a host's eggs, then the likelihood of the cuckoo egg to be discovered is less, and the fitness should then be related to the difference in solutions. On the other hand, if a cuckoo's egg is different with the host's egg, host will either eject the alien egg or abandon the nest (which is a worst nest). Thus, a fraction of worse nests are discovered with a probability (*pa*), thus, random walk in a biased way with some random step sizes was performed in order to replace the worst nest with new ones. As such, new sets of solutions were then generated and will be subjected to another round of fitness test and evaluation to get the best. The implementation of the process is shown in Fig. 3.8, and the complete MATLAB script is shown in Appendix B.

```
%% Replace some nests by constructing new solutions/nests
function new_nest=empty_nests(nest,Lb,Ub,pa)
% A fraction of worse nests are discovered with a probability pa
n=size(nest,1);
K=rand(size(nest))>pa;
%% New solution by biased/selective random walks
stepsize=rand*(nest(randperm(n),:)-nest(randperm(n),:));
new_nest=nest+stepsize.*K;
for j=1:size(new_nest,1)
    s=new_nest(j,:);
  new_nest(j,:)=simplebounds(s,Lb,Ub);
end
```

Fig. 3.8: Snippet of MATLAB Script for Replacement of Worst Solutions

3.4 Performance Evaluation of the Algorithms (CSA and dCSA)

The performance of the developed dynamic Cuckoo Search Algorithm and the standard Cuckoo search algorithm was evaluated using the ten optimization test functions listed in subsection 2.2.13 based on the closeness of the test function result to the global minimal value. The algorithms optimized each function 25 times and results were measured and recorded.

3.4.1 Visualization of the optimization test function

To enhance the understanding and visualization of the local minimal point of most of the optimization test functions described in section 2.2.13, Fig. 3.9 - 3.18 were generated in MATLAB program to show the surfaces and shapes of the test functions.

3.4.1.1 Ackley function

The 3D visualization of Ackley function is as shown in Fig. 3.9

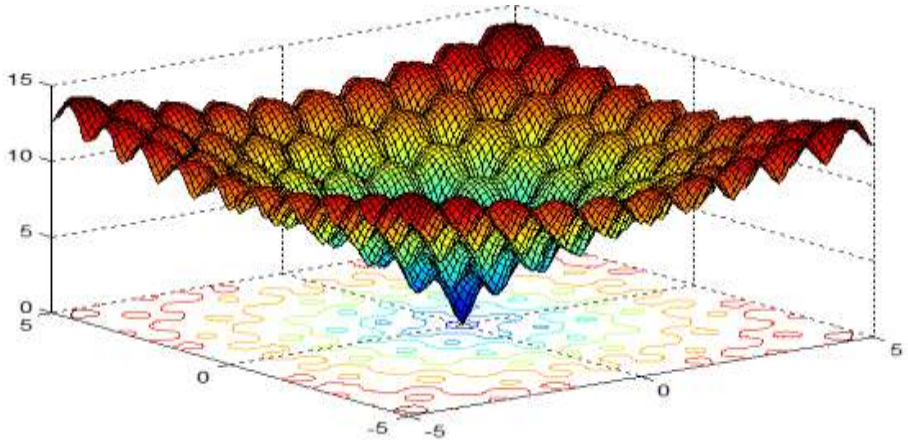


Fig. 3.9: 3D Visualization of Ackley Function

From Fig. 3.9, it is obvious that Ackley function has many local minima, but the global minima of this function can be observed at point (0, 0).

3.4.1.2 DeJong function

The global minimal of Dejong is 0 which can be observed in the 3D visualization of the function in Fig. 3.10. This function is described in section 2.2.13

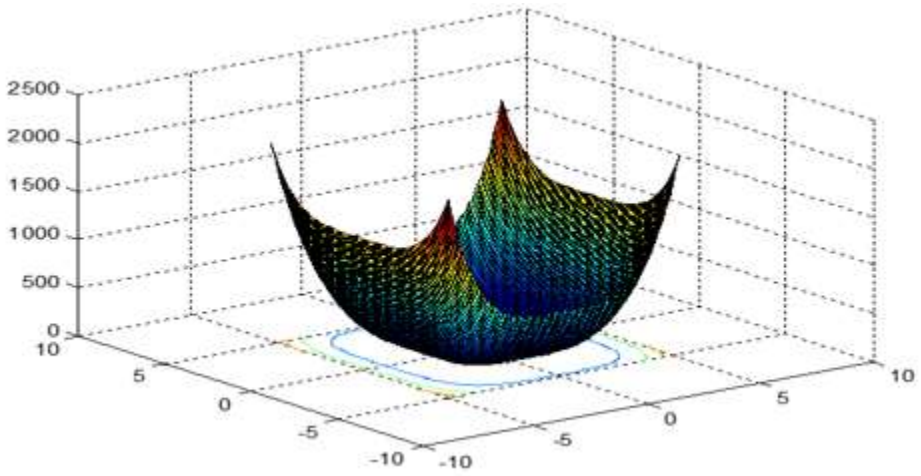


Fig. 3.10: 3D Visualization of Dejong Function

3.4.1.3 Easom function

The 3D visualisation of Easom function is as shown in Fig. 3.11

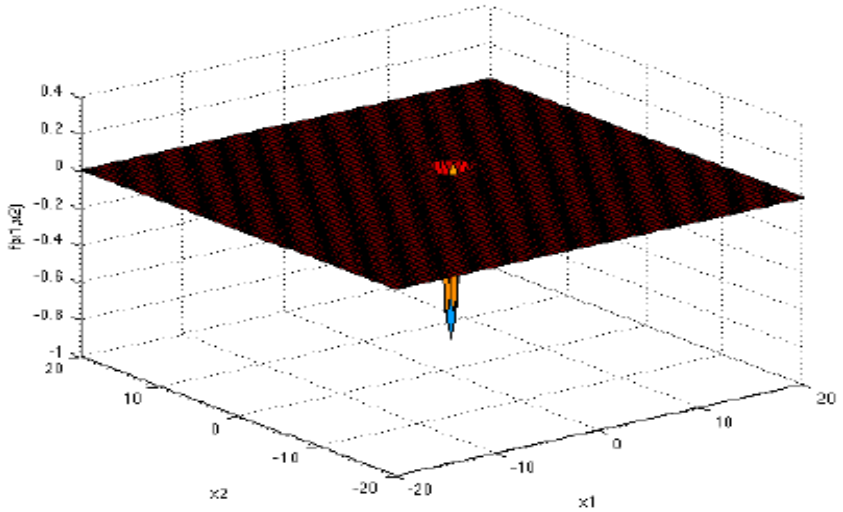


Fig. 3.11: 3D Visualization of Easom function

The Easom function has several local minima, and the global minimum has a small area relative to the search space. It is a unimodal function, details can be found in section 2.2.13.

3.4.1.4 Griewangk function

Griewangk's function is similar to Rastrigin's function. It has many widespread local minima as seen in the 3D visualization of Fig. 3.12.

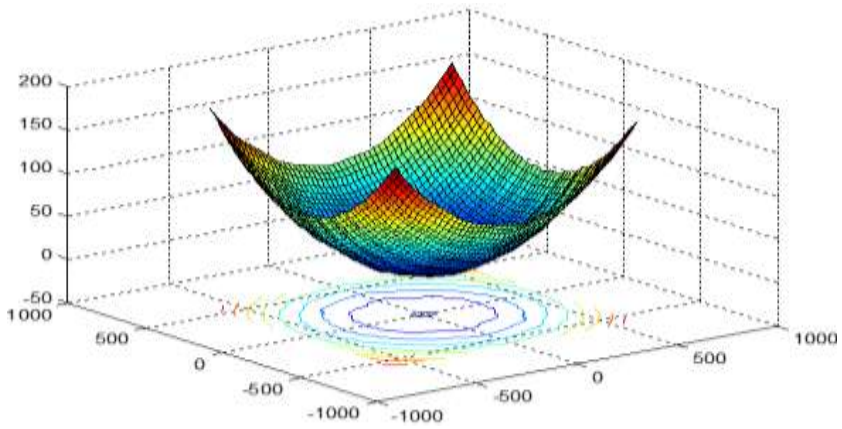


Fig. 3.12: 3D Visualization of Greiwangk Function

It can be observed that, the global minimal of this function is at point 0. The function is described in section 2.2.13

3.4.1.5 Michalewicz's function

The 3D visualization of Michalewicz function is as shown in Fig. 3.13.

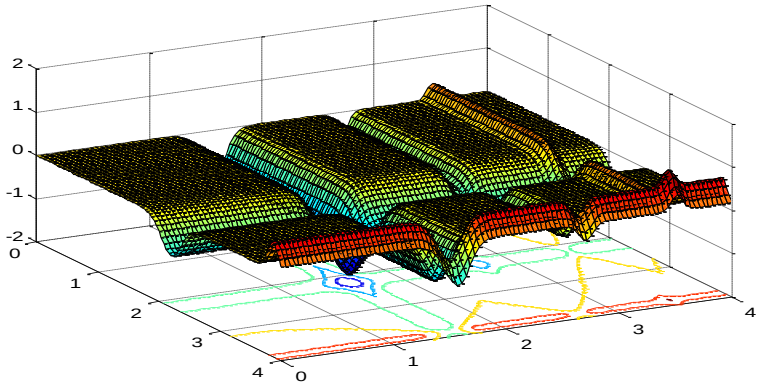


Fig. 3.13: 3D Visualization of Michalewicz's Function

The Michalewicz function has d! local minima, and it is multimodal. The parameter m defines the steepness of the valleys and ridges; a larger m leads to a more difficult search. The recommended value of m is m = 10. The function's three-dimensional form is shown in the plot above.

3.4.1.6 Rastrigin function

As discussed in section 2.2.13, Rastrigin's function is based on ExpFun with the addition of cosine modulation to produce many local minima. The 3D visualization of Rastrigin function is as shown in Fig. 3.14 with a global minimal located at 0.

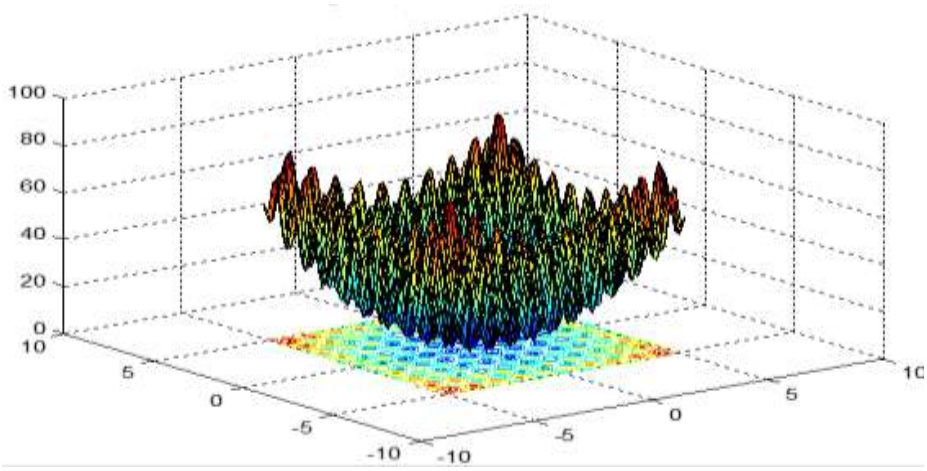


Fig. 3.14: 3D Visualization of Rastrigin Function

3.4.1.7 Rosenbrock function

Rosenbrock function has its global optimum inside a long, narrow, parabolic shaped flat valley which makes it very difficult for optimization algorithm to converge towards the global optimum as in Fig. 3.15.

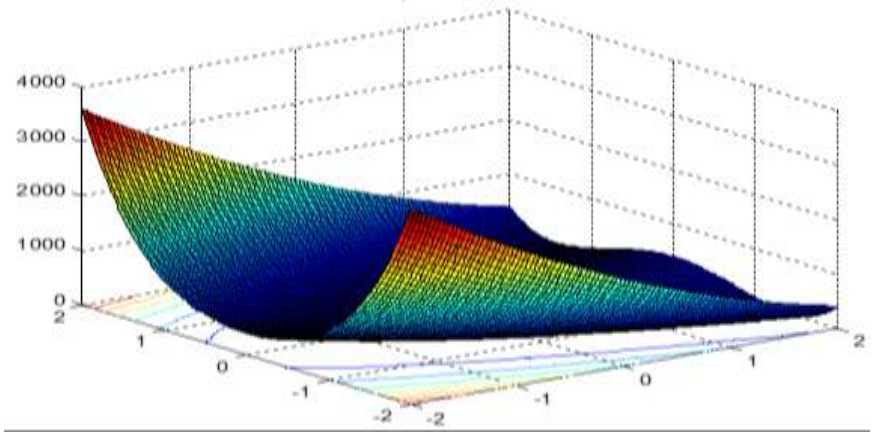


Fig. 3.15: 3D Visualization of Rosenbrock Function

As shown in Figure 3.15, global minimal of Rosenbrock function can be carefully observed at point 0.

3.4.1.8 Schwefel test function

This function is deceptive, with a lot of local minimal spread around the global minimal. The global point of this function is at 0, which is shown in the 3D visualization of Fig. 3.16. The detailed description of this function can be found in section 2.2.6.13

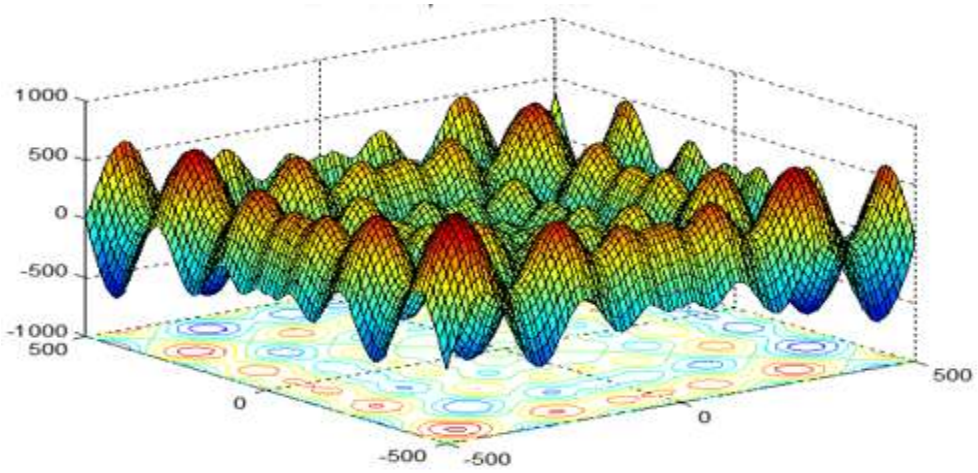


Fig. 3.16: 3D Visualization of Schwefel Function

3.4.1.9 Shubert function

Shubert function has several local minima and many global minima, but global minimum is found around $f_{(x^*)} = -186.7309$. Details of the function can be found in section 2.2.13. Fig. 3.17 shows a 3D visualization of Shubert function.

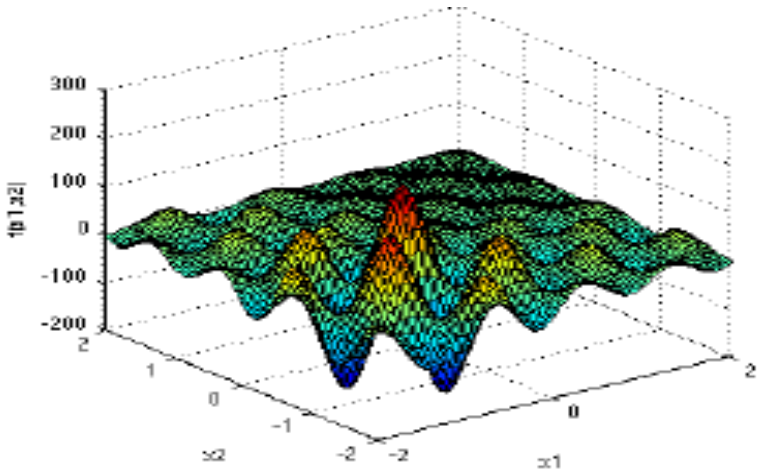


Fig. 3.17: 3D Visualization of Shubert Function

3.4.1.10 Sphere function

The sphere function is sometimes called the simplest form of De Jong function as described in section 2.2.13. It is a unimodal and non-convex. The local minimal of this function is 0 as can be seen in the 3D visualization of Fig. 3.18.

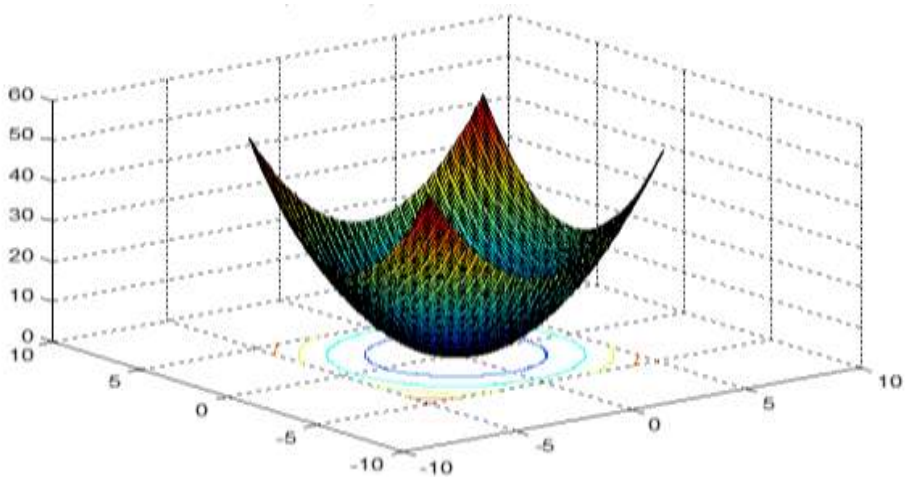


Fig. 3.18: 3D Visualization of Sphere Optimization Test Function

3.4.2 Percentage improvement

To show that dCSA improves the results of the standard CSA significantly, the percentage improvement was calculated using equation (3.5) and recorded.

$$Percentage\ Improvement = \frac{CSA - dCSA}{CSA} \times 100\% \quad (3.5)$$

3.5 Application of dCSA into Voice Recognition System (VRS)

The developed dCSA was used in the VRS for classification. Unique feature vectors extracted by MFCC process were optimally classified by dCSA. The classification process was in two phases, first at the training phase, when the database is being created, an instance where templates for the various speakers were created, and there is a need to record more samples from the speakers. Second is during the testing phase of the VRS. At each of these phases a new record is coming that needs to be classified to which set of category, group or template the new record belongs.

The optimal classification was done in such a way that the dCSA technique select an optimal value from the uniquely extracted feature vectors, and assign it to be the centroid or cluster head of these vectors. When a new record come, it will also have its optimal cluster head, then a distance measure was used to compare the distance between the new cluster head and all the optimal cluster heads of the other templates contained in the database, the one with closest distance to any of the templates will either be added to the template as additional record (for training) or be subjected to a decision logic for recognition.

The Euclidian distance measure was used for this purpose, as given by equation (3.6), and Fig.3.19 shows the snippet of how the clustering or classification process was implemented in the Matlab script.

$$Euclidean\ distance = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \quad (3.6)$$

where *pi* and *qi* are vectors indicating two points in Euclidean *n*-space.

```
cB = [];
for i=1:7:154
    data = fV(i:i+6,:);
    [fmin, ct, membership]=dCSA_clustering(data, 1);
    cB = [cB; ct];
end
```

Fig. 3.19: Snippet of MATLAB Script for Implementation of classification in VRS with dCSA

3.5.1 Testing of speakers for recognition

The total number of records in the database was one hundred and fifty-four (154) with twenty-two speakers, each having a voice sample of seven. A test was conducted, and improvement of recognition was achieved successfully. Snippet of the implementation of this process is shown in Fig.3.20.

```
p = [];
testResult = [];
testVoices = dir(fullfile('VOICE DATA/test/', '*.wav'));
test_names = [];
for v = 1: numel(testVoices)
    wav_file = strcat('VOICE DATA/test/', testVoices(v).name);
    n = testVoices(v).name(1:9);
    test_names = [test_names; strcat(n, '--  ')];
    % Read speech samples, sampling rate and precision from file
    [ speech, fs, nbits ] = wavread( wav_file );
```

Fig. 3.20: Snippet of MATLAB Script for the Implementation of Recognition Testing

3.6 Validation of Performance of CSA and dCSA Scheme in VRS

Performance evaluation method is the gauge or measure to investigate the efficiency of any recognition system, be it speaker, speech, or face recognition. The assessment is crucial for understanding the quality of the technique, for refining parameters in the iterative process of learning and for selecting the most adequate strategy or technique from a given set of models or techniques.

Standard performance metrics based on precision and accuracy were used for validation of this research work. The performance of the standard CSA-based scheme in a VRS was compared with the performance of the developed dCSA-based scheme in the VRS.

3.6.1 Accuracy

Accuracy can be defined as to how close a measured value is to the actual (true) value. Accuracy is the proportion of classifications over all the *n* examples that were correctly detected. Accuracy is mathematically defined as "the fraction of quantity of correct classification over the entire number of samples." The amount of predictions in classification techniques relies upon the counts of the test records properly or incorrectly predicted by the model. It is expressed as shown in equation (3.7).

$$Accuracy = \frac{Number\ of\ correctly\ recognised\ voice}{Total\ number\ of\ validation\ set} \tag{3.7}$$

CHAPTER FOUR

RESULTS AND DISCUSSIONS

4.1 Introduction

In this section, analysis on the behavior of some the selected voice data obtained was performed, this is to understand the signal behavior of the selected voice samples, the result of the extraction process was discussed. The performance of the developed dynamic Cuckoo search algorithm and that of the standard Cuckoo search algorithm were evaluated using the optimization test functions discussed in subsection 2.2.13 and relevant results reported, and percentage improvement of dCSA over CSA was also determined and recorded. Moreover, the effectiveness of the developed algorithm was also demonstrated by comparing the dCSA based classification scheme with that of CSA based scheme in a voice recognition system using precision and accuracy as measure of evaluation.

4.2 Speech Signal Representation and Analysis

Four samples of the voice signal (two from each gender) were collected from the database and were analyzed, the properties and behaviors of these signal were plotted as shown in Fig. 4.1 and Fig.4.2.

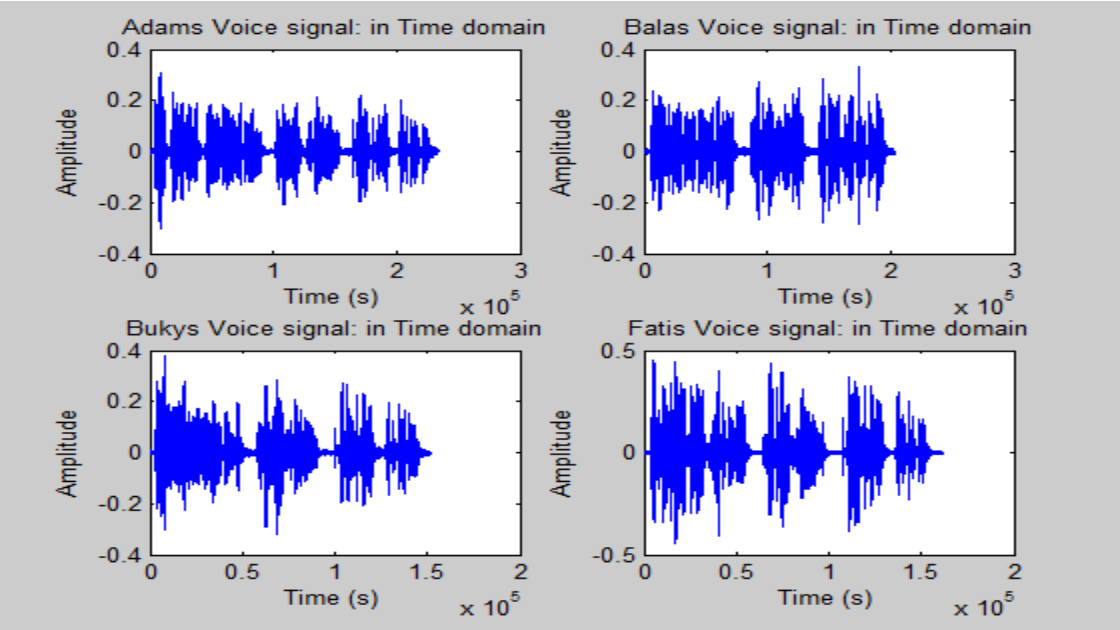


Figure 4.1: Graphs of Raw Speech Signal in Time Domain for Four Selected Speakers

In time domain, raw data contains great amount of information as such it is difficult to analyze the voice characteristic. Hence, the need to sample and extract speech feature. Fig.4.2 shows the signal converted to frequency domain using Fast Fourier Transform (FFT) for easier processing and analysis.

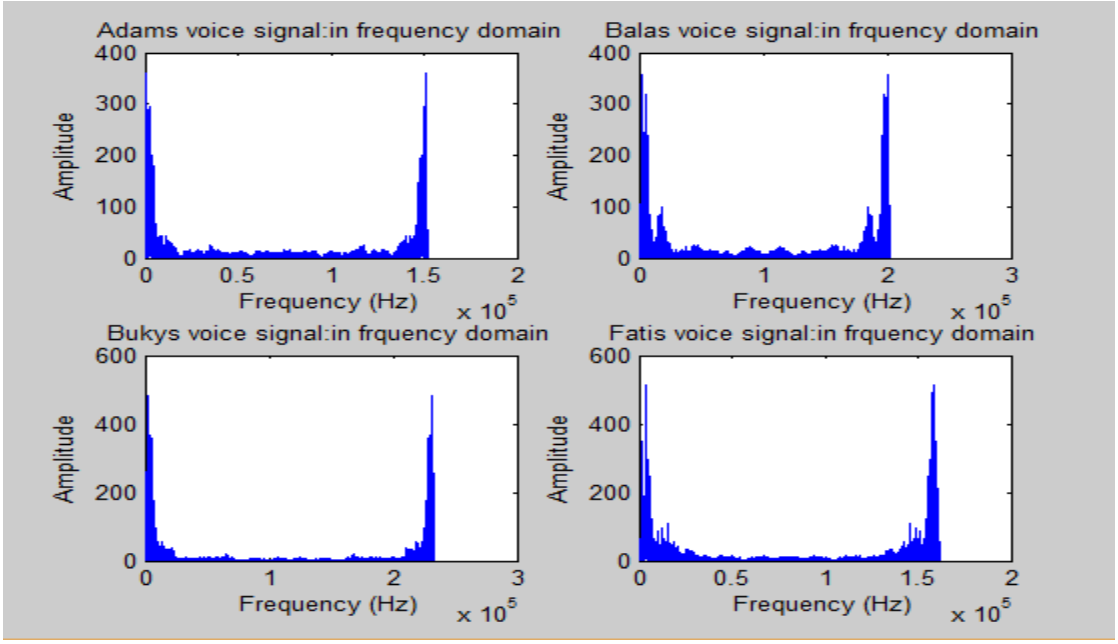


Fig. 4.2: Representation of the Sampled Speech Signals in Frequency Domain

4.2.1 Feature extraction with Mel Frequency Celptral Coefficients (MFCC)

A short-term power spectrum of a sound signal is normally presented as Mel-Frequency Ceptrum (MFC). This ceptrum was used to gain information of the speech signal, it separates the excitation signal (which contains words and pitch) and the transfer function (which contains the quality of voice). This ceptrum was produced as a result of taking Fourier transform of decibel spectrum as if it were a signal, and is represented as shown in equation (3.8)

$$Ceptrum\ signal = FT[\log\{FT(windowed\ signal)\}] \quad (3.8)$$

Furthermore, equally spaced frequency bands on Mel scale shown in equation (3.9), gives an approximated human auditory system's response used in the MFC. Thus, MFCC were coefficients made from MFC.

$$mel(f) = 2595 \times log_{10}(1 + f/700) \quad (3.9)$$

Mel frequency ceptral coefficient was used for feature extraction process, and four speakers were sampled in order to understand the signal behavior during the feature extraction process. The (Mel) filter bank was used to transform the spectrum of a signal into a representation which reflects more closely the behavior of the human ear. As the human ear favors low frequencies for analyzing speech, the filters are denser for the lower frequencies. To mimic the human ear, the filters are linearly distributed for low frequencies (below 1kHz). For higher frequencies (above 1 kHz), the filters were distributed logarithmically.

Fig. 4.3(a), 4.3(b), 4.3(c) and 4.3(d) shows the graphical and visual representation of voice signals from four selected speakers, namely; Adam, Buky Bala and Fati respectively.

The first plot for each speaker, is the speech waveforms in time domain, it is a plot of amplitude against time showing the actual energy concentration in the voice signal with frequency response, the thin black line indicates where a speaker paused for a while during his speech (i.e. a voiceless portion). It clearly distinguishes between a male and a female speaker from the nature of the frequency responses of the graph, with female speaker having a highest frequency response (i.e. the pitch) when compared with a male speaker.

The second plot is a spectrogram showing visual representation of the speech signal in terms of log (mel) filter bank energy with respect to channel index, the dark blue colour of the spectrogram indicates the right audio channel and the light blue colour indicates the left audio channel, the red colour signifies the energy content of the voice signal while the yellow colour represents the free space where there is no speech or sound (a voiceless portion).

The third plot is another spectrogram, but with filtered information where only the relevant signal representation was maintained. It contains the Mel frequency ceptrum extracted from the voice signal for each speaker.

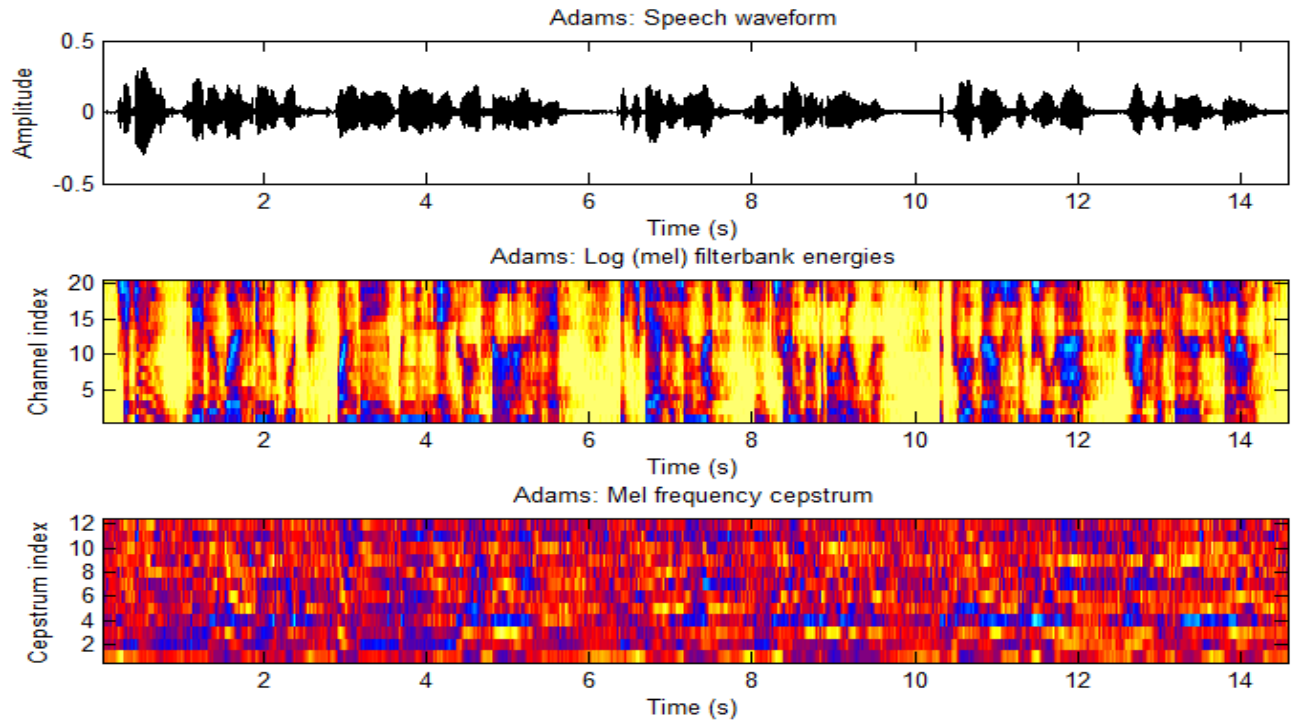


Fig.4.3(a): Speech Waveform and Spectrograms from Adam's Voice.

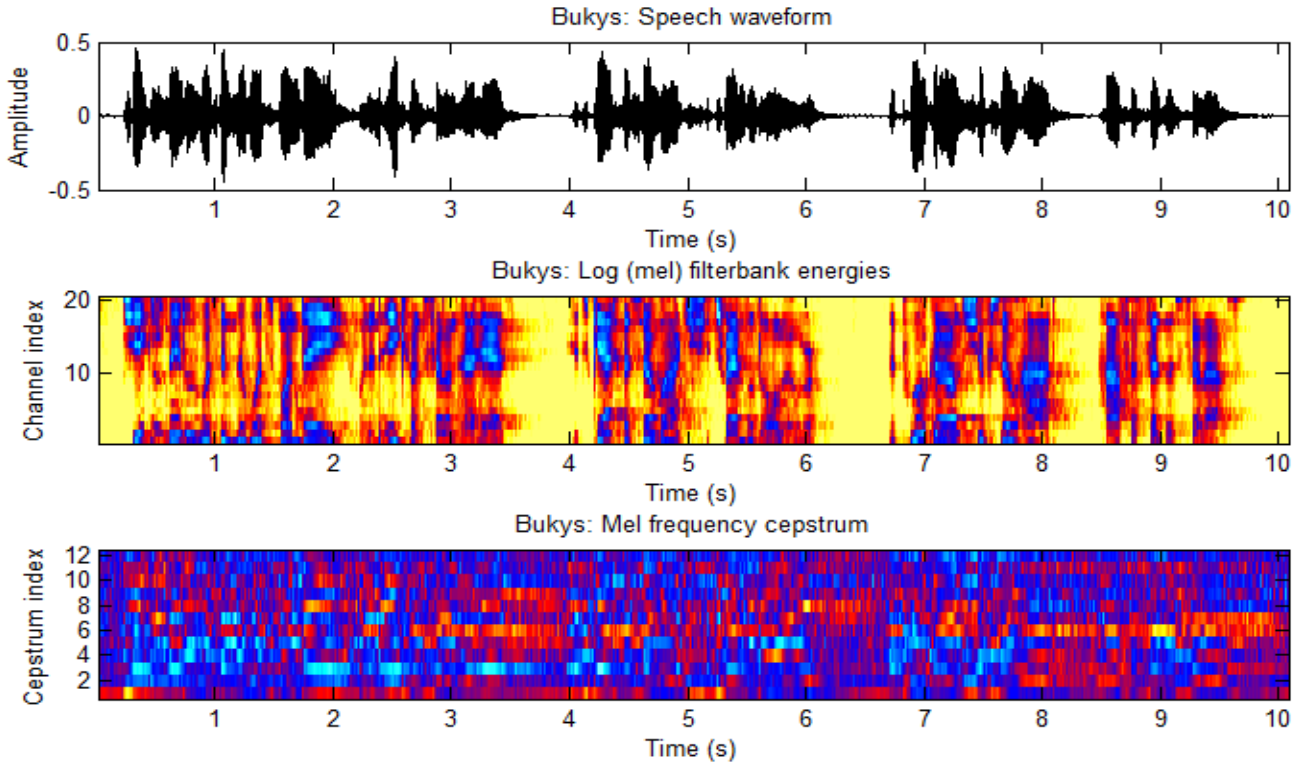


Fig.4.3(b): Speech Waveform and Spectrograms from Buky's Voice.

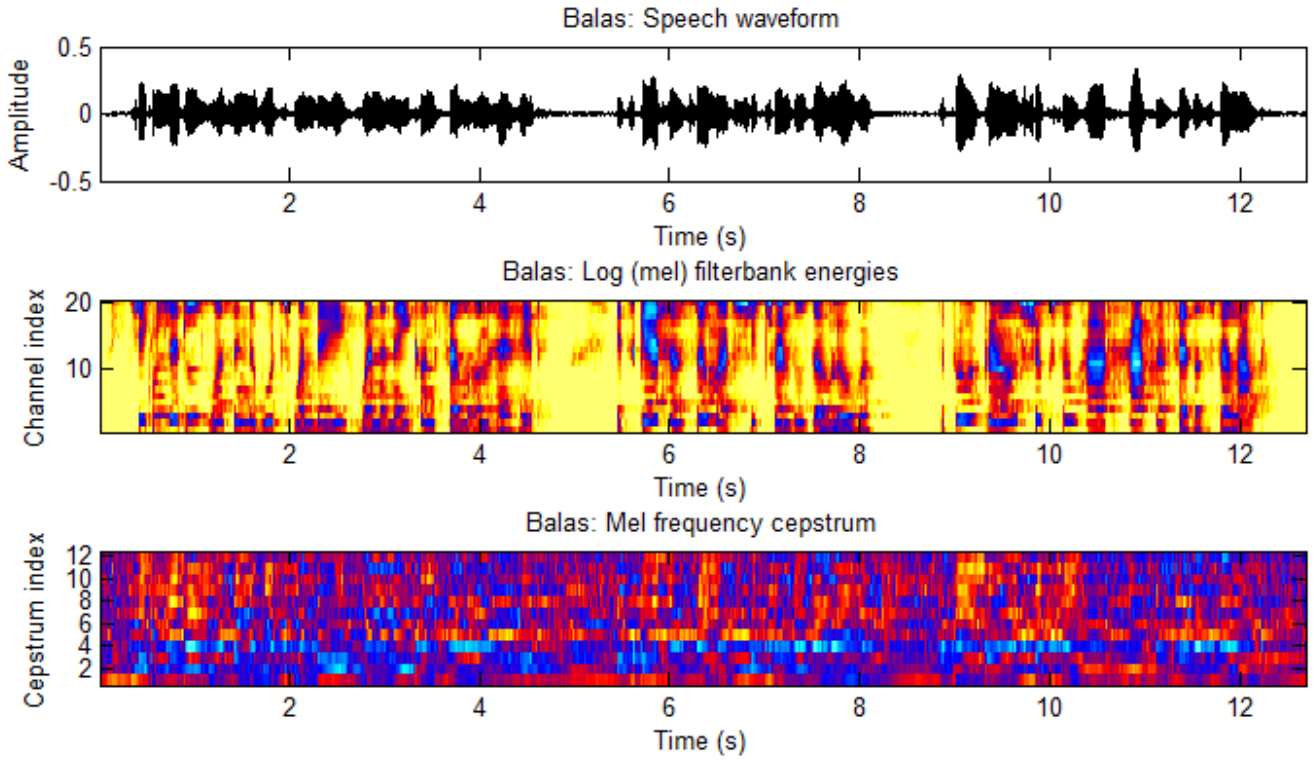


Fig.4.3(c): Speech Waveform and Spectrograms from Bala's Voice.

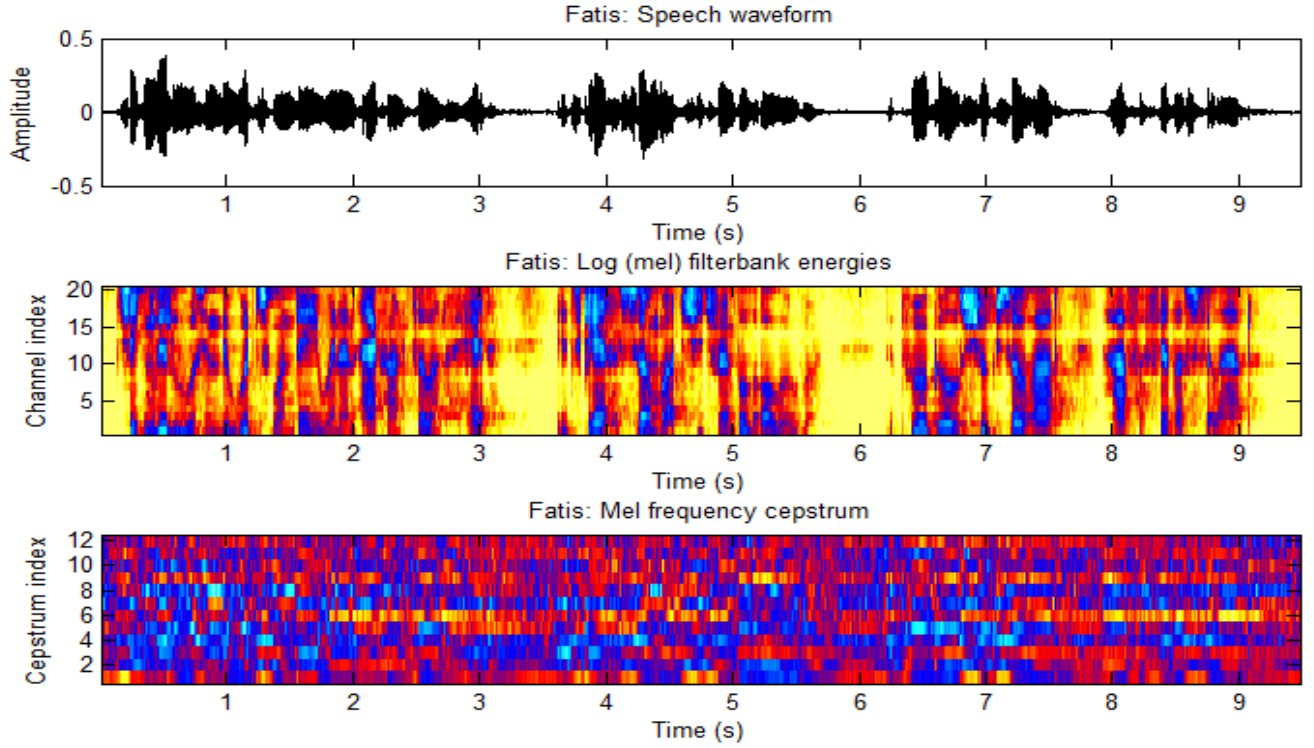
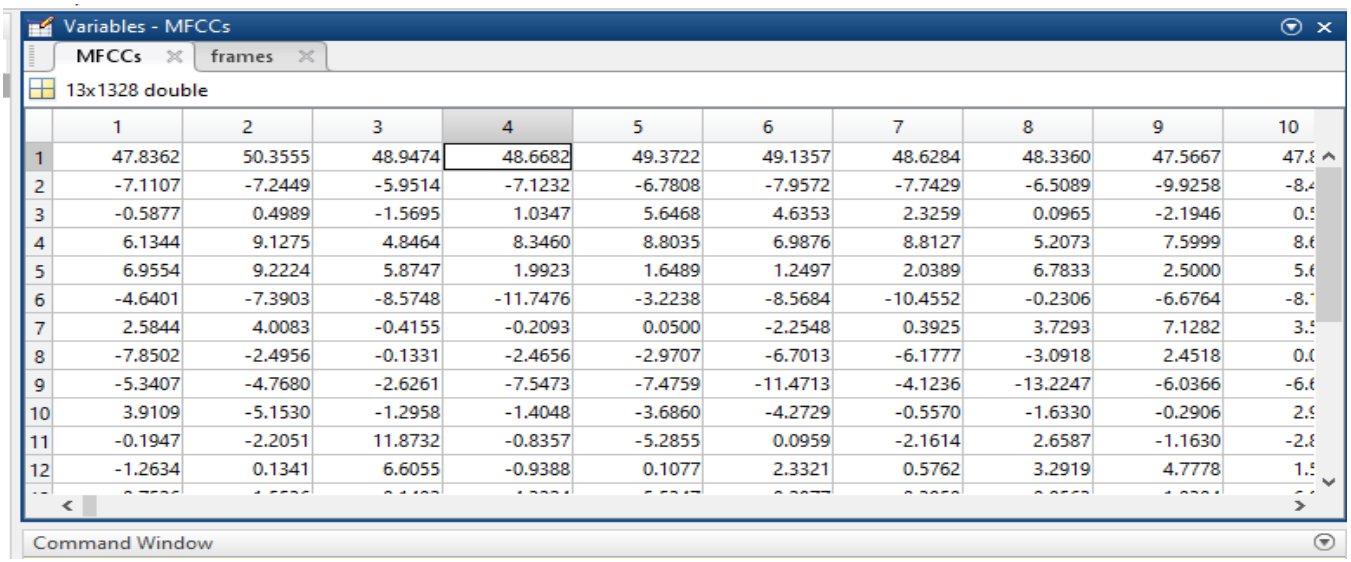


Fig.4.3(d): Speech Waveform and Spectrograms from Fati's Voice.

After getting the Mel frequency ceptrum of the speech signal as shown in the above graphical and visual representations, then the spectrum was converted back to time domain, such that equivalent values of real numbers for these spectrum were obtained for reference, and these are what is referred to as the coefficients of Mel Frequency Ceptrum, otherwise known as Mel Frequency Ceptral Coefficients (MFCC). These cepstral representation of the speech spectrum provides a good representation of the local spectral properties of a speech signal for a given frame analysis. The snippet of these values (extracted feature vectors) for a speech signal of one of the speakers are as shown in Fig. 4.4.



	1	2	3	4	5	6	7	8	9	10
1	47.8362	50.3555	48.9474	48.6682	49.3722	49.1357	48.6284	48.3360	47.5667	47.8
2	-7.1107	-7.2449	-5.9514	-7.1232	-6.7808	-7.9572	-7.7429	-6.5089	-9.9258	-8.4
3	-0.5877	0.4989	-1.5695	1.0347	5.6468	4.6353	2.3259	0.0965	-2.1946	0.5
4	6.1344	9.1275	4.8464	8.3460	8.8035	6.9876	8.8127	5.2073	7.5999	8.6
5	6.9554	9.2224	5.8747	1.9923	1.6489	1.2497	2.0389	6.7833	2.5000	5.6
6	-4.6401	-7.3903	-8.5748	-11.7476	-3.2238	-8.5684	-10.4552	-0.2306	-6.6764	-8.1
7	2.5844	4.0083	-0.4155	-0.2093	0.0500	-2.2548	0.3925	3.7293	7.1282	3.5
8	-7.8502	-2.4956	-0.1331	-2.4656	-2.9707	-6.7013	-6.1777	-3.0918	2.4518	0.0
9	-5.3407	-4.7680	-2.6261	-7.5473	-7.4759	-11.4713	-4.1236	-13.2247	-6.0366	-6.6
10	3.9109	-5.1530	-1.2958	-1.4048	-3.6860	-4.2729	-0.5570	-1.6330	-0.2906	2.9
11	-0.1947	-2.2051	11.8732	-0.8357	-5.2855	0.0959	-2.1614	2.6587	-1.1630	-2.8
12	-1.2634	0.1341	6.6055	-0.9388	0.1077	2.3321	0.5762	3.2919	4.7778	1.5

Fig. 4.4: Snapshot of Values for Extracted Feature Vectors of One Speaker

4.3 Results of the dCSA

The dynamic cuckoo search algorithm (dCSA) was developed. The performance of the developed dCSA was evaluated using ten optimization benchmark test functions and the optimum results obtained are shown in Table 4.1, It can be seen that the optimum value generated by the dCSA for optimizing Ackley was 8.8818E-16, 3.0160E-260 for Dejong, -1.0000E+00 for Easom, 0.0000E+00 for Griewangk, -10.9913E+00 for Michalwicz, 9.1745E+00 for Rsatrigin, 0.0000E+00 for Rosenbrock, -3.9337E+00 for Shwefel, -145.0460E+00 for Shubert and 1.1681E-21 for Sphere function. This indicates that the dCSA

obtained the exact optimum solution for Easom, Griewangk and Rosenbrock functions while obatining a near optimum solution for Ackley, Dejong, Michalwicz, Rastrigun, Shwefel, Shubert and Sphere functions.

4.3.1 Performance Evaluation of dCSA over CSA

The performance of the replicated standard Cuckoo Search (CSA) was evaluated using ten optimization benchmark test functions and the result is compared with the result obtained using the dynamic Cuckoo Search Algorithm (dCSA) with respect to global optimal of the benchmark test functions. Both algorithms were evaluated using the optimization benchmark test functions for 25 runs each. From the MATLAB codes of Appendix B and C, the results generated by each algorithm are as shown in Table 4.1.

Table 4.1: Performance Evaluation of CSA over dCSA

Test Functions	Global Minimal	CSA	dCSA	% Improvement
Ackley	0.0000E+00	9.1363E-06	**8.8818E-16**	100
Dejong	0.0000E+00	3.2228E-06	**3.016E-260**	100
Easom	-1.0000E+00	-0.5338E+00	**-1.0000E+00**	53.38
Greiwangk	0.0000E+00	3.0067E-06	**0.0000E+00**	100
Michalwicz	-9.6602+00	-2.9949E+00	**-10.9913E+00**	27.25
Rastrigin	0.0000E+00	**9.6917E-06**	9.1745E+00	-100
Rosenbrock	0.0000E+00	8.2910E-06	**0.0000E+00**	100
Shwefel	-4.1898E+02	-3.9020E+00	**-3.9337E+00**	0.81
Shubert	-1.8673E+02	-5.7704E+01	**-1.4505E+02**	39.78
Sphere	0.0000E+00	9.7799E-06	**1.1681E-21**	100

From the results, it is obvious that dCSA outperforms the CSA with respect to the global minimal result in almost all the optimization test function except in test case 6 (Rastrigin function) where CSA

outperformed the dCSA. However, CSA did well for this class of functions (low-dimensional and relatively easy functions), but perform fairly on others (high-dimensional and more complex functions). The superiority of dCSA over CSA is expected as inertia weight factor was incorporated into the control parameters of the CSA which makes them dynamic in the dCSA. The dynamic step size diversifies the solution search for sufficient exploration, while the dynamic control probability guides the evolution of dCSA towards obtaining the global optimal value of the optimization benchmark functions by ensuring proper balance between exploration and exploitation.

Note that algorithm with the best performances with respect to the global solution are shown in bold in the above table.

4.4 Application of dCSA in Voice Recognition System

The developed dCSA was applied for optimal classification of the extracted feature vectors of the speech signals in section 4.2.1 and the results obtained are as shown in fig. 4.5.

```
Total number of iterations=1000

fmin =

   4.2123e+03


centers =

   -5.1137   -3.7406    1.1791   -2.9166   -0.3770    2.9093   -2.0121
    3.4356   -6.2812   23.2838   -1.6533   33.7313    6.5686    0.6981
   17.6328   30.7320   10.0312   29.7527   32.5043   33.8699    3.9026
   35.0242   29.9181   34.6319   33.4464   17.7989   18.6204   28.9021
   27.4619    4.0244   25.7577   37.9220   21.9219   16.1040   16.0198
    2.7386   33.0969   27.0875    1.5769   28.9246   16.0080   14.3655
   12.4026    9.3394    1.1219   23.0281   -5.8415   -5.2391    5.5850
```

Fig. 4.5: Snapshot of Results for dCSA Classification Scheme

The optimally classified vectors are a representation of key and unique features of speech signal for each and every one of the speakers in the database, and a group of similar vectors makes a template for a particular speaker.

4.5 Testing of Speakers for Recognition

Training of the extracted feature vectors was initially carried out, such that the templates needed or required for each speaker was developed and stored in a database. Building the database ensured that whenever a recognition is required or during a testing period, the template is referenced for matching before a decision could be made as to weather a speaker is recognized or otherwise. Fig. 4.6 shows a snapshot of Graphic User Interface (GUI) showing Number of Trained Samples.

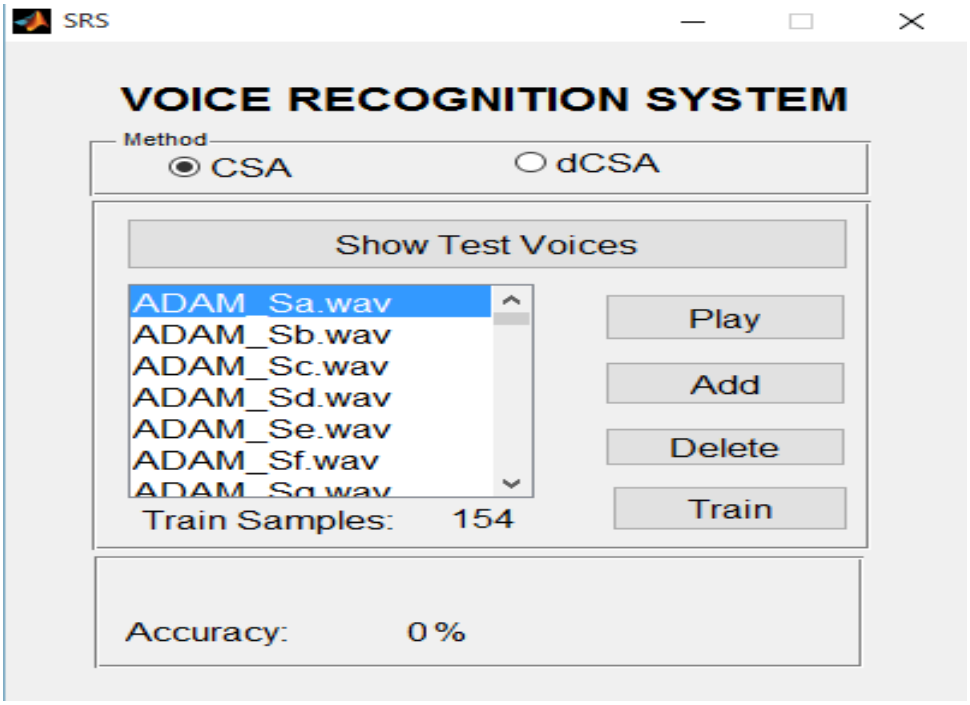
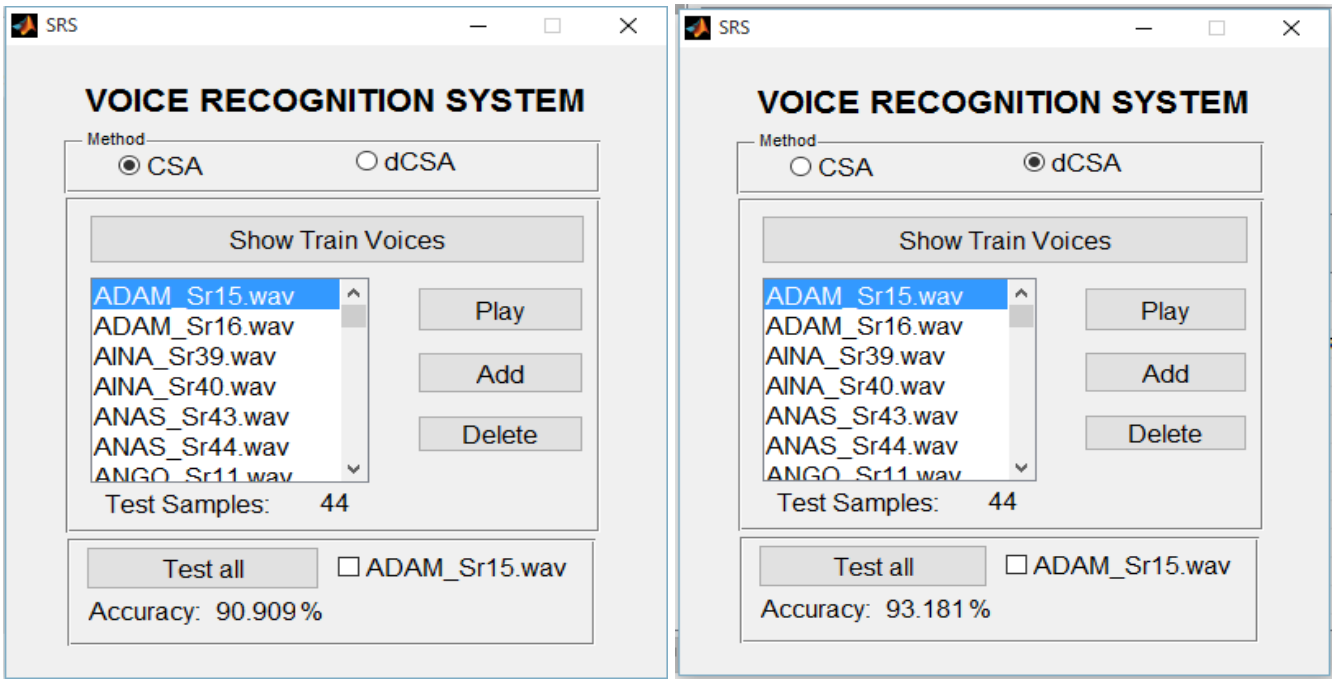


Fig.4.6: Snapshot of GUI Window Showing Number of Trained Samples.

Test was carried out on the available data using standard CSA and the dCSA in the voice recognition system and the output result indicates that the accuracy of the dCSA was higher than that of the standard

CSA. A GUI snapshot of the system displaying the performance accuracy of both the standard CSA and the dCSA is as shown in fig. 4.7(a) and 4.7(b).



(a) Accuracy Level for CSA (b) Accuracy Level for dCSA

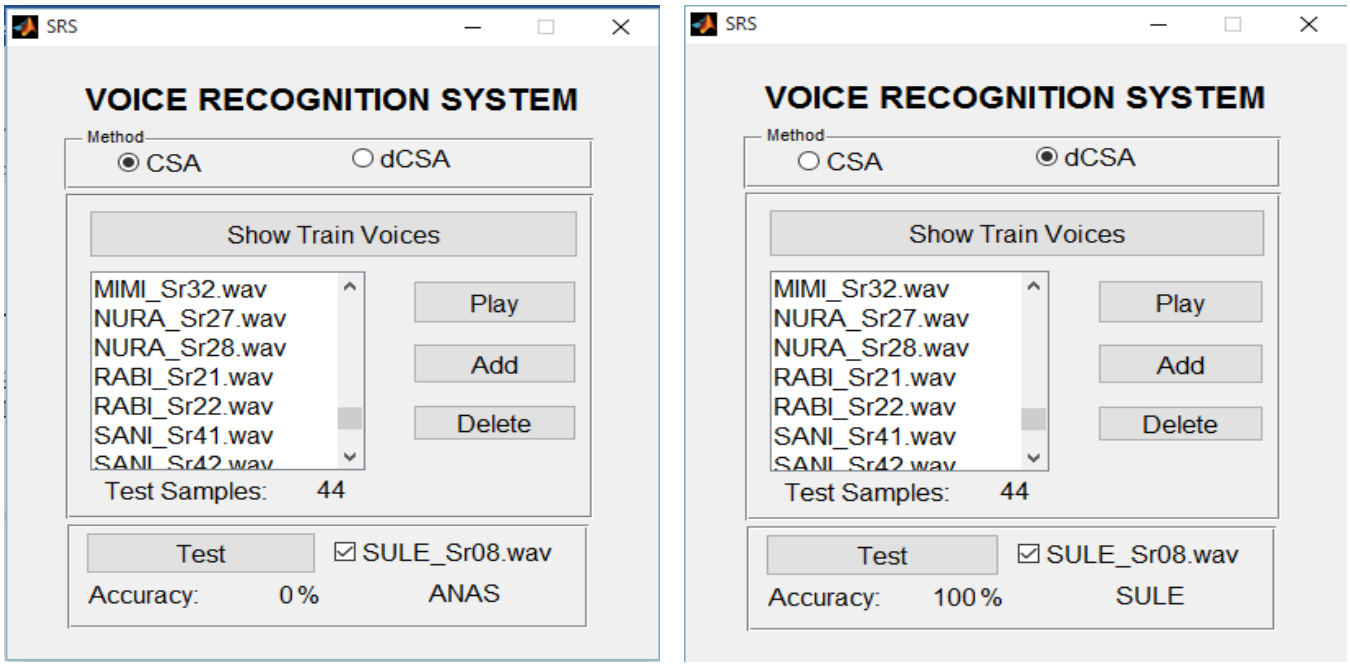
Fig.4.7(a) and (b): Snapshot of VRS GUI for Accuracy Level

4.5.1 VRS GUI Usage Procedures

After loading the data, a selection for the classification based scheme was chosen, then train button was pressed for the computer to train the data using the specified scheme. The data has to be trained under any of the chosen scheme before the test could be carried out.

Testing was carried out after the training has finished, selecting test all on the GUI menu, the algorithm will test the whole dataset without any exception, then the number of the tested data samples will be displayed together with the performance accuracy of any of the scheme specified. Likewise, if a name of a particular speaker was chosen for testing and after running the test, if features of that speaker exist in

the database a matching accuracy and the name of the speaker will be displayed on GUI, and if a chosen scheme has recognition error even if the speaker's data was captured in the database, it will still have erred the speaker. Fig. 4.8(a) and (b) shows a snapshot of recognition error from CSA-based scheme and the accuracy level of the dCSA-based scheme. The complete MATLAB script for the VRS GUI is shown in appendix D.



(a) Recognition Error in CSA (b) Accuracy level in dCSA

Fig.4.8 Snapshot of VRS GUI for Recognition Level of CSA and dCSA Based Schemes

CHAPTER FIVE

SUMMARY AND CONCLUSION

5.1 Summary

This research is aimed at the development of an optimal extracted feature classification scheme in voice recognition system using dynamic cuckoo search algorithm. Standard dataset was obtained from English Language Speech Database for Speaker Recognition (ELSDSR) from the Technical University of Denmark (DTU), processed and trained the data for voice recognition.

A swarm intelligent metaheuristic algorithm referred to as dynamic Cuckoo Search algorithm (dCSA) was developed to optimally classify the extracted feature vectors that were used in the Voice Recognition System (VRS). Performance of the VRS was compared using a CSA-based classification scheme and that of the developed dCSA-based classification scheme, and the results obtained showed that dCSA-based scheme produced more accurate result in the VRS than the CSA-based scheme.

5.2 Conclusion

Classification in any voice recognition system is an active and integral part that determines the accuracy of recognition. Classical and other traditional method were the predominant techniques used in VRS, hence, recognition level can be low. In order to overcome this common problem and increase the recognition accuracy, a metaheuristic algorithm was developed to optimally choose and classify the feature vectors of voice signals being used in voice recognition systems. The optimal extracted feature classification scheme using dynamic Cuckoo Search Algorithm in a VRS was developed in MATLAB R2013b. Moreover, the performance of the developed algorithm (dCSA) that was used for the classification technique was evaluated using ten unimodal and multimodal applied mathematical optimization test functions (Ackley, De jong, Easom, Rosenbrock, Griewangk, , Michalewicz, Rastrigin

Rosenbrock, Schwefel, Shubert and Sphere). The simulation results obtained shows that dCSA performed better when compared with the standard CSA with an increase of 52% performance accuracy. The dCSA was then used in a voice recognition system to optimally select and classify extracted feature vectors of speech signal, and results obtained demonstrated that dCSA-based scheme has high recognition rate or accuracy than the CSA-based scheme with 3.2% accuracy increase. Likewise, it clearly demonstrated that the future of optimal classification techniques will be brighter as it may likely replace the traditional and classical methods being used.

5.3 Significant Contributions

The significant contributions of this research work are as follows:

1. A Graphic User Interface based optimal classification scheme for extracted feature vectors in a voice recognition system using dynamic cuckoo search algorithm was developed.
2. The developed dynamic cuckoo search algorithm based optimal extracted feature classification scheme produced a recognition accuracy of 93.18% compared to 90% produced by standard CSA-based classification scheme.
3. The dynamic cuckoo search algorithm produced an improvement of 52% when compared with the standard cuckoo search algorithm.

5.4 Recommendations for Further Work

The following possible areas of further work are recommended for consideration for future research:

1. Extension of the work for taking University class attendance to check students' impersonation.
2. Implementation of the work for the control of automated vehicles
3. Modification of the algorithm by hybridization for enhanced exploitation capabilities
4. Extension of areas of application of the dCSA to other constrained and/or unconstrained optimization problems

REFERENCES

Aggarwal, C. C., & Reddy, C. K. (2014). Data clustering. *Algorithms and Applications, Chapman & Halls*.

Amarasinghe, A., & Wimalaratne, P. (2017). An Assistive Technology Framework for Communication with Hearing Impaired Persons. *GSTF Journal on Computing (JoC), 5*(2).

Atal, B. S. (1976). Automatic recognition of speakers from their voices. *Proceedings of the IEEE, 64*(4), 460-475.

Bansal, D., Turk, N., & Mendiratta, S. (2015). *Automatic speech recognition by cuckoo search optimization based artificial neural network classifier*. Paper presented at the Soft Computing Techniques and Implementations (ICSCTI), 2015 International Conference on.

Bansal, J. C., Singh, P., Saraswat, M., Verma, A., Jadon, S. S., & Abraham, A. (2011). *Inertia weight strategies in particle swarm optimization.* Paper presented at the Nature and Biologically Inspired Computing (NaBIC), 2011 Third World Congress on.

Barthelemy, P., Bertolotti, J., & Wiersma, D. S. (2008). A Lévy flight for light. *Nature, 453*(7194), 495-498.

Bhalla, A. V., Khaparkar, S., & Bhalla, M. R. (2012). Performance improvement of speaker recognition system. *International Journal of Advanced Research in Computer Science and Software Engineering, 2*(3).

Brown, C. T., Liebovitch, L. S., & Glendon, R. (2007). Lévy flights in Dobe Ju/'hoansi foraging patterns. *Human Ecology, 35*(1), 129-138.

Campbell, J. P. (1997). Speaker recognition: A tutorial. *Proceedings of the IEEE, 85*(9), 1437-1462.

Chauhan, P., Deep, K., & Pant, M. (2013). Novel inertia weight strategies for particle swarm optimization. *Memetic computing, 5*(3), 229-251.

Das, T., & Nahar, K. M. (2016). A Voice Identification System using Hidden Markov Model. *Indian Journal of Science and Technology, 9*(4).

Dash, M., & Mohanty, R. (2014). Cuckoo search algorithm for speech recognition. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), 3*(10).

Davies, N., & Cuckoos, C. (2000). Other Cheats. *T. & AD Poyser, London*.

Eberhart, R. C., & Shi, Y. (2001). *Tracking and optimizing dynamic systems with particle swarms.* Paper presented at the Evolutionary Computation, 2001. Proceedings of the 2001 Congress on.

El Aziz, M. A., & Hassanien, A. E. (2016). Modified cuckoo search algorithm with rough sets for feature selection. *Neural Computing and Applications*, 1-10.

Equitz, W. H. (1989). A new vector quantization clustering algorithm. *IEEE transactions on acoustics, speech, and signal processing, 37*(10), 1568-1575.

Feng, L., & Hansen, L. K. (2005). A new database for speaker recognition.

Fister Jr, I., Fister, D., & Fister, I. (2013). A comprehensive review of cuckoo search: variants and hybrids. *International Journal of Mathematical Modelling and Numerical Optimisation, 4*(4), 387-409.

Fränti, P., & Kivijärvi, J. (2000). Randomised local search algorithm for the clustering problem. *Pattern Analysis & Applications, 3*(4), 358-369.

Ge, Z., Iyer, A. N., Cheluvaraja, S., Sundaram, R., & Ganapathiraju, A. (2017). Neural Network Based Speaker Classification and Verification Systems with Enhanced Features. *arXiv preprint arXiv:1702.02289*.

Haldar, R., & Mishra, P. K. (2016). Learning Vector Quantization (LVQ) Neural Network Approach for Multilingual Speech Recognition.

Harrag, A. (2015). Nature-inspired feature subset selection application to arabic speaker recognition system. *International Journal of Speech Technology, 18*(2), 245-255.

Honda, M. (2003). Human speech production mechanisms. *NTT Technical Review, 1*(2), 24-29.

Huang, X., Acero, A., Hon, H.-W., & Foreword By-Reddy, R. (2001). *Spoken language processing: A guide to theory, algorithm, and system development*: Prentice hall PTR.

Jamil, M., & Yang, X.-S. (2013). A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation, 4*(2), 150-194.

Juang, B.-H., & Rabiner, L. R. (2005). Automatic speech recognition–a brief history of the technology development. *Georgia Institute of Technology. Atlanta Rutgers University and the University of California. Santa Barbara, 1*, 67.

Kamat, S., & Karegowda, A. G. (2014). A brief survey on cuckoo search applications. *International Journal of Innovative Research in Computer and Communication Engineering, 2*(2), 7-14.

Kinnunen, T., Kilpelainen, T., & Franti, P. (2011). Comparison of clustering algorithms in speaker identification. *dim, 1*, 2.

Kinnunen, T., & Li, H. (2010). An overview of text-independent speaker recognition: From features to supervectors. *Speech communication, 52*(1), 12-40.

Kumar, C. S., & Rao, P. M. (2011). Design of an automatic speaker recognition system using MFCC, Vector Quantization and LBG algorithm. *International Journal on Computer Science and Engineering, 3*(8), 2942.

Li, X., Tang, K., Omidvar, M. N., Yang, Z., Qin, K., & China, H. (2013). Benchmark functions for the CEC'2013 special session and competition on large-scale global optimization. *gene, 7*(33), 8.

Linde, Y., Buzo, A., & Gray, R. (1980). An algorithm for vector quantizer design. *IEEE Transactions on communications, 28*(1), 84-95.

Mahendru, H. C. (2014). Quick review of human speech production mechanism. *International Journal of Engineering Research and Development, 9*, 48-54.

Manikandan, P., & Selvarajan, S. (2014). *Data clustering using cuckoo search algorithm (CSA).* Paper presented at the Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28-30, 2012.

Molga, M., & Smutnicki, C. (2005). Test functions for optimization needs. *Test functions for optimization needs*.

Muda, L., Begam, M., & Elamvazuthi, I. (2010). Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques. *arXiv preprint arXiv:1003.4083*.

Nasrabadi, N. M., & Feng, Y. (1988). *Vector quantization of images based upon the Kohonen self-organizing feature maps.* Paper presented at the Proc. IEEE Int. Conf. Neural Networks.

Nemati, S., & Basiri, M. E. (2011). Text-independent speaker verification using ant colony optimization-based selected features. *Expert Systems with Applications, 38*(1), 620-630.

Nijhawan, G., & Soni, M. (2014). Speaker Recognition Using MFCC and Vector Quantisation. *International Journal on Recent Trends in Engineering and Technology, 11*(1), 211-218.

Ojha, R., & Das, M. (2012). An Adaptive Approach for Modifying Inertia Weight using Particle Swarm Optimisation. *IJCSI International Journal of Computer Science Issues, 9*(5), 105-112.

Pandey, A. C., Rajpoot, D. S., & Saraswat, M. (2017). Twitter sentiment analysis using hybrid cuckoo search method. *Information Processing & Management, 53*(4), 764-779.

Pantaleo, E., Facchi, P., & Pascazio, S. (2009). Simulations of Lévy flights. *Physica Scripta, 2009*(T135), 014036.

Payne, R., Sorenson, M., & Klitz, K. (2005). The cuckoos, vol. 15: Oxford University Press, Oxford.

Prasad, K. S., Ramaiah, G. K., & Manjunatha, M. (2017). Speech Features Extraction Techniques for Robust Emotional Speech Analysis/Recognition. *Indian Journal of Science and Technology, 8*(1).

Price, J., & Eydgahi, A. (2006). *Design of matlab®-based automatic speaker recognition systems.* Paper presented at the 9th International Conference on Engineering Education T4J-1.

Reynolds, A. M., & Frye, M. A. (2007). Free-flight odor tracking in Drosophila is consistent with an optimal intermittent scale-free search. *PloS one, 2*(4), e354.

Reynolds, D. A. (2002). *An overview of automatic speaker recognition technology.* Paper presented at the Acoustics, speech, and signal processing (ICASSP), 2002 IEEE international conference on.

Salvador, S., & Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis, 11*(5), 561-580.

Senthilnath, J., Das, V., Omkar, S., & Mani, V. (2013). *Clustering using levy flight cuckoo search.* Paper presented at the Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012).

Shah, H. N. M., Ab Rashid, M. Z., Abdollah, M. F., Kamarudin, M. N., Chow, K. L., & Kamis, Z. (2014). Biometric voice recognition in security system. *Indian Journal of Science and Technology, 7*(2), 104-112.

Shi, Y., & Eberhart, R. (1998). *A modified particle swarm optimizer.* Paper presented at the Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on.

Sood, M., & Kaur, G. (2013). Speaker recognition based on cuckoo search algorithm. *International Journal of Innovative Technology and Exploring Engineering (IJITEE), 2*(5), 311-313.

Sun, G., Liu, Y., Yang, M., Wang, A., Liang, S., & Zhang, Y. (2017). Coverage optimization of VLC in smart homes based on improved cuckoo search algorithm. *Computer Networks, 116*, 63-78.

Tang, J., Alelyani, S., & Liu, H. (2014). Feature selection for classification: A review. *Data Classification: Algorithms and Applications*, 37.

Tang, K., Yáo, X., Suganthan, P. N., MacNish, C., Chen, Y.-P., Chen, C.-M., & Yang, Z. (2007). Benchmark functions for the CEC'2008 special session and competition on large scale global optimization. *Nature Inspired Computation and Applications Laboratory, USTC, China*.

Tran, T., Nguyen, T. T., & Nguyen, H. L. (2014). Global optimization using L\'evy flights. *arXiv preprint arXiv:1407.5739*.

Uddin, N., Rashid, M. M., & Mostafa, M. G. (2016). Development of voice recognition for student attendance. *Global Journal of Human-Social Science Research, 16*(1).

Vaijayanthi, P., Yang, X.-S., Natarajan, A., & Murugadoss, R. High Dimensional Data Clustering Using Cuckoo Search Optimization Algorithm. *International Journal of Advanced Computer Engineering and Communication Technology, 3*, 31-35.

Valian, E., Mohanna, S., & Tavakoli, S. (2011). Improved Cuckoo Search Algorithm for Global Optimization. *International Journal of Communications and Information Technology, IJCIT, 1*, 31-44.

Walton, S., Hassan, O., Morgan, K., & Brown, M. R. (2011). Modified cuckoo search: A new gradient free optimisation algorithm. *Chaos, Solitons & Fractals, 44*, 710-718. doi: 10.1016

Wang, G. G., Deb, S., Gandomi, A. H., Zhang, Z., & Alavi, A. H. (2014). *A novel cuckoo search with chaos theory and elitism scheme.* Paper presented at the Soft Computing and Machine Intelligence (ISCMI), 2014 International Conference on.

Yadav, R., & Mandal, D. (2011). Optimization of artificial neural network for speaker recognition using particle swarm optimization. *International Journal of Soft Computing and Engineering, 1*(3).

Yang, X.-S. (2010). A new metaheuristic bat-inspired algorithm *Nature inspired cooperative strategies for optimization (NICSO 2010)* (pp. 65-74): Springer.

Yang, X.-S. (2012). Nature-inspired mateheuristic algorithms: Success and new challenges. *arXiv preprint arXiv:1211.6658*.

Yang, X.-S., & Deb, S. (2009). *Cuckoo search via Lévy flights.* Paper presented at the Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on.

Yang, X.-S., & Deb, S. (2013). Multiobjective cuckoo search for design optimization. *Computers & Operations Research, 40*(6), 1616-1624.

Yang, X.-S., & Deb, S. (2014). Cuckoo search: recent advances and applications. *Neural Computing and Applications, 24*(1), 169-174.

Yılmaz, S., & Küçüksille, E. U. (2015). A new modification approach on bat algorithm for solving optimization problems. *Applied Soft Computing, 28*, 259-275.

Zhang, S.-X., Chen, Z., Zhao, Y., Li, J., & Gong, Y. (2016). *End-to-End attention based text-dependent speaker verification.* Paper presented at the Spoken Language Technology Workshop (SLT), 2016 IEEE.

Zhang, S.-X., Chen, Z., Zhao, Y., Li, J., & Gong, Y. (2017). End-to-End Attention based Text-Dependent Speaker Verification. *arXiv preprint arXiv:1701.00562*.

Zhao, J., Lei, X., Wu, Z., & Tan, Y. (2014). *Clustering using improved cuckoo search algorithm.* Paper presented at the International Conference in Swarm Intelligence.

Zhao, M., Tang, H., Guo, J., & Sun, Y. (2016). A Data Clustering Algorithm Using Cuckoo Search *Frontier Computing* (pp. 225-230): Springer.

Zhao, P., & Li, H. (2012). *Opposition-based Cuckoo search algorithm for optimization problems.* Paper presented at the Computational Intelligence and Design (ISCID), 2012 Fifth International Symposium on.

Zulfiqar, A., Muhammad, A., & AM, M. E. (2009). *A speaker identification system using MFCC features with VQ technique.* Paper presented at the Intelligent Information Technology Application, 2009. IITA 2009. Third International Symposium on.

APPENDIX A_1

TRANSCRIPT OF AUDIO MESSAGE USED DURING TRAINING SESSION

paragraph	Content
A	Chicken Little was in the woods one day when an acorn fell on her head. It scared her so much she trembled all over.**(1)** The poor girl shook so hard, half her feathers fell out.**(2)**
B	Billions of black, shrimp-size bugs with transparent wings and beady red eyes are beginning to carpet trees, buildings, poles, and just about anything else vertical in the U.S. from the eastern seaboard west through Indiana and south to Tennessee.**(3)**
C	Oymyakon, in Siberia, is the coldest permanently inhabited place on Earth.**(4)** Now geographer and adventurer Nick Middleton reveals the locals' secrets for coping with the cold.**(5)**
D	Few shores are immune from the tide of plastic soda bottles, bags, cartons, and other trash floating on the ocean today.**(6)** Now a new study suggests the problem runs deeper: Microscopic bits of plastic permeate the world's beaches and marine environment.**(7)**
E	One hundred years later, the life of the Negro is still sadly crippled by the manacles of segregation and the chains of discrimination.**(8)**
F	People are finding medieval toys in Britain's Thames River—and these toys have been changing how historians view the lives of medieval kids.**(9)**
G	My friend Tricia suggests me to drive to the woods to watch the poor bear being hunted

for pleasure.[10] And I say yes.[11]

APPENDIX A_2

TRANSCRIPT OF AUDIO MESSAGE USED DURING TESTING SESSION

Passage	Content
Ancient Egypt	There are days when the sand blows ceaselessly, blanketing the remains of a powerful dynasty that ruled Egypt 5,000 years ago.[1]When the wind dies down and the sands are still, a long shadow casts a wedge of darkness across the Sahara, creeping ever longer as the north African sun sinks beyond the horizon.[2]Five thousand years ago, the fourth dynasty of Egypt's Old Kingdom was a highly advanced civilization where the kings, known as pharaohs, were believed to be gods.[3] They lived amidst palaces and temples built to honor them and their deified ancestors.[4] "Pharaoh" originally meant "great house," but later came to mean king.[5] This web site will show you science in action -- bringing you face to face with the evidence archaeologists use to understand the meaning of Giza's pyramids, and to the process of evaluating the finds they will uncover beneath the sands of the plateau.[6] Before looking closely at pharaonic society and the beginning of the Pyramid Age, one first has to step into Egypt's landscape and take a look around.[7] Ancient Egyptians called their land "Kemet," which meant "black," after the black fertile silt-layered soil that was left behind each year during the annual innundation, when the Nile flooded the fields.[8] The most prevalent color of the desert, however, is a decidedly reddish-yellow ochre.[9] The Egyptians called the desert "deshret," meaning "red," and this endless carpet of sand covers an estimated 95 % of Egypt, interrupted only by the narrow band of green carved by the waters of the Nile.[10] It was at this time that hieroglyphic writing made its first appearance, in the tombs and treasures of the pharaohs.[11] To seal the unification of Upper and Lower Egypt, Menes founded the capital city of the kingdom at the place where the two met: at the apex of the Nile, where it fans out onto the fertile silt plain.[12] The fortress city was named "White Walls" by Menes, but it is known today by its Greek name, Memphis.[13] For much of the 3,000 years of ancient Egypt, it remained the capital seat of the pharaohs.[14] Only 20 miles to the north of Memphis is the modern capitol, Cairo, still situated near the juncture of the Nile valley and the delta.[15] How does the pyramid fit into early Egyptian life?[16] In this society, each individual's eternal life was dependent on the continued existence of their king, a belief that made the pharaoh's tomb the

concern of the entire kingdom.[17] Pictures on the walls of tombs tell us about the lives of the Kings and their families.[18] We know pyramids were built during a king's lifetime because heiroglyphs on tomb walls have been found depicting the names of the gangs who built the pyramids for their kings.[19] Furniture and riches were buried with the king so he would have the familiar comforts of his lifetime buried near him.[20] Whole subdivisions of tombs of those in high positions in the court of a king can be found surrounding the pyramids of Giza.[21] These are primarily mastabas, or covered rectangular tombs that consist of a deep burial shaft, made of mud brick and half-buried by the drifts of sand on the plateau.[22] The first pyramid was the Step Pyramid at Saqqara, built for King Zoser in 2750 BC.[23] This first application of large scale technology, however, is often attributed to Imhotep, the architect of the Step Pyramid.[24] He was not a pharaoh, but was the Director of Works of Upper and Lower Egypt.[25] The superstructure of the pyramid was made of small limestone blocks and desert clay.[26] Inside, the burial chamber and storage spaces for Zoser's grave goods were carved out of the earth and rock beneath the structure.[27] Imhotep's intent was to mimic the basic structure of King Zoser's palatial home in the burial chamber.[28] The tomb, like those that followed, was meant to be a replica of the royal palace.[29] In early tombs, the central area was always the burial place.[30] It is thought that in 816 AD Caliph al-Mamun first ordered workers to blast through the blocked stone entrance in order to explore within Khufu's pyramid.[31] But looters, probably from dynastic Egyptian times, had already absconded with King Khufu's burial treasures and his body.[32] This is true of all of the pyramids at Giza, so very little is known about Khufu or any of his successors who were buried at Giza.[33] Archaeologists, nonetheless, continue to look for pieces of this puzzle to further our understanding of the Pyramid Age and the pharaohs that ruled Egypt.[34]

History of Giza

Standing at the base of the Great Pyramid, it is hard to imagine that this monument -- which remained the tallest building in the world until early in this century -- was built in just under 30 years.[35] It presides over the plateau of Giza, on the outskirts of Cairo, and is the last survivor of the Seven Wonders of the World.[36] Today, Giza is a suburb of rapidly growing Cairo, the largest city in Africa and the fifth largest in the world.[37] About 2,550 B.C., King Khufu, the second pharaoh of the fourth dynasty, commissioned the building of his tomb at Giza.[38]

Some Egyptologists believe it took 10 years just to build the ramp that leads from the Nile valley floor to the pyramid, and 20 years to construct the pyramid itself.[39] On average, the over two million blocks of stone used to build Khufu's pyramid weigh 2.5 tons, and the heaviest blocks, used as the ceiling of Khufu's burial chamber, weigh in at an estimated 40 to 60 tons.[40] This question has long been debated, but many Egyptologists agree the stones were hauled up ramps using ropes

of papyrus twine.[41] The popular belief is that the gradually sloping ramps, built out of mud, stone, and wood were used as transportation causeways for moving the large stones to their positions up and around the four sides of the pyramids.[42] Giza, however, is more than just three pyramids and the Sphinx.[43] Each pyramid has a mortuary temple and a valley temple linked by long causeways that were roofed and walled.[44]

APPENDIX B

COMPLETE MATLAB FILE FOR DYNAMIC CUCKOO SEARCH ALGORITHM (dCSA)

```
%%=====================================================================%%
  %  %%%%%%%  DYNAMIC_CUCKOO_SEARCH_ALGORITHM (dCSA)   %%%%%%%  %
%%=====================================================================%%

function [bestnest,fmin]=cuckoo_search_new(n)
if nargin<1,
% Number of nests (or different solutions)
n=25;
end
% Discovery rate of alien eggs/solutions
% Introducion of Random Inertia Weight Factor at the (pa)
pa=0.25*(0.5+0.5*rand)
% You change this if you like
N_IterTotal=1000;
% Simple bounds of the search domain
% Lower bounds
nd=15;
Lb=-5*ones(1,nd);
% Upper bounds
Ub=5*ones(1,nd);
% Random initial solutions
for i=1:n,
nest(i,:)=Lb+(Ub-Lb).*rand(size(Lb));
end
% Get the current best
fitness=10^10*ones(n,1);
[fmin,bestnest,nest,fitness]=get_best_nest(nest,nest,fitness);
N_iter=0;
% Starting iterations
for iter=1:N_IterTotal,
    % Generate new solutions (but keep the current best)
     new_nest=get_cuckoos(nest,bestnest,Lb,Ub);
     [fnew,best,nest,fitness]=get_best_nest(nest,new_nest,fitness);
    % Update the counter
      N_iter=N_iter+n;
    % Discovery and randomization
      new_nest=empty_nests(nest,Lb,Ub,pa) ;
    % Evaluate this set of solutions
```

```
      [fnew,best,nest,fitness]=get_best_nest(nest,new_nest,fitness);
    % Update the counter again
      N_iter=N_iter+n;
    % Find the best objective so far
    if fnew<fmin,
        fmin=fnew;
        bestnest=best;
    end
end %% End of iterations
% Post-optimization processing
% Display all the nests
disp(strcat('Total number of iterations=',num2str(N_iter)));
fmin
bestnest;
% --------------- Other Sub-functions used ------------------
% Get cuckoos by ramdom walk
function nest=get_cuckoos(nest,best,Lb,Ub)
% Levy flights
n=size(nest,1);
% Levy exponent and coefficient
beta=3/2;
sigma=(gamma(1+beta)*sin(pi*beta/2)/(gamma((1+beta)/2)*beta*2^((beta-1)/2)))^(1/beta);

for j=1:n,
    s=nest(j,:);
        % Levy flights by Mantegna's algorithm
    u=randn(size(s))*sigma;
    v=randn(size(s));
    step=u./abs(v).^(1/beta);
 % Introducion of Random Inertia Weight Factor at the (stepsize)
    w=(0.5+(0.5*rand));
 %(s-best) means when the solution is the best solution, it remains unchanged.
    stepsize=w*step.*(s-best);
    % Now the actual random walks or flights
    s=s+stepsize.*randn(size(s));
   % Apply simple bounds/limits
   nest(j,:)=simplebounds(s,Lb,Ub);
end
% Find the current best nest
function [fmin,best,nest,fitness]=get_best_nest(nest,newnest,fitness)
% Evaluating all new solutions
for j=1:size(nest,1),
    fnew=fobj(newnest(j,:));
    if fnew<=fitness(j),
       fitness(j)=fnew;
       nest(j,:)=newnest(j,:);
    end
end
% Find the current best
[fmin,K]=min(fitness) ;
best=nest(K,:);
% Replace some nests by constructing new solutions/nests
function new_nest=empty_nests(nest,Lb,Ub,pa)
```

```
% A fraction of worse nests are discovered with a probability pa
n=size(nest,1);
% Discovered or not -- a status vector
K=rand(size(nest))>pa;
% New solution by biased/selective random walks
stepsize=rand*(nest(randperm(n),:)-nest(randperm(n),:));
new_nest=nest+stepsize.*K;
for j=1:size(new_nest,1)
    s=new_nest(j,:);
  new_nest(j,:)=simplebounds(s,Lb,Ub);
end
% Application of simple constraints
function s=simplebounds(s,Lb,Ub)
  % Apply the lower bound
  ns_tmp=s;
  I=ns_tmp<Lb;
  ns_tmp(I)=Lb(I);
  % Apply the upper bounds
  J=ns_tmp>Ub;
  ns_tmp(J)=Ub(J);
  % Update this new move
  s=ns_tmp;
%  Objective function
function z=fobj(x)
% (1) %%%%%%%%%% The d-dimensional sphere function  %%%%%%%%%%
%       z=sum((x-1).^2);
```

Published with MATLAB® R2013b

APPENDIX C

COMPLETE MATLAB FILE FOR dCSA-BASE CLASSIFICATION SCHEME IN VRS

```
### dCSA-based classification scheme in VRS  ###
```

```
addpath('mfcc/');

% Clean-up MATLAB's environment
clear all; close all; clc;

% Define variables
Tw = 25;                % analysis frame duration (ms)
Ts = 10;                % analysis frame shift (ms)
alpha = 0.97;           % preemphasis coefficient
M = 20;                 % number of filterbank channels
C = 12;                 % number of cepstral coefficients
L = 22;                 % cepstral sine lifter parameter
LF = 300;               % lower frequency limit (Hz)
HF = 3700;              % upper frequency limit (Hz)

fV = [];
persons = [];
trainVoices = dir(fullfile('VOICE DATA/train/', '*.wav'));
for i = 1:numel(trainVoices)
    persons = [persons; trainVoices(i).name(1:4)];
    wav_file = strcat('VOICE DATA/train/', trainVoices(i).name);
    % Read speech samples, sampling rate and precision from file
    [ speech, fs, nbits ] = wavread( wav_file );
    % Feature extraction (feature vectors as columns)
    [ MFCCs, FBEs, frames ] = ...
        mfcc( speech, fs, Tw, Ts, alpha, @hamming, [LF HF], M, C+1, L );
    MFCCs(find(isnan(MFCCs))) = 0.0001;

    mm = MFCCs;
    mf = mean(mm,2); cf = cov(mm');
    ff = mf;
    for i=0:(size(mm,1)-1)
        ff = [ff;diag(cf,i)];
    end
```

```
    fV = [fV; ff'];
end

persons = unique(persons, 'rows');
cB = [];
for i=1:7:154
    data = fV(i:i+6,:);
    [fmin, ct, membership]=dCSA_clustering(data, 1);
    cB = [cB; ct];
end


p = [];
testResult = [];
testVoices = dir(fullfile('VOICE DATA/test/', '*.wav'));
test_names = [];
for v = 8(testVoices)
    wav_file = strcat('VOICE DATA/test/', testVoices(v).name);
    n = testVoices(v).name(1:9);
    test_names = [test_names; strcat(n, '--  ')];
    % Read speech samples, sampling rate and precision from file
    [ speech, fs, nbits ] = wavread( wav_file );
    % Feature extraction (feature vectors as columns)
    [ MFCCs, FBEs, frames ] = ...
        mfcc( speech, fs, Tw, Ts, alpha, @hamming, [LF HF], M, C+1, L );
    MFCCs(find(isnan(MFCCs))) = 0.0001;

    mm = MFCCs;
    mf = mean(mm,2); cf = cov(mm');
    ff = mf;
    for i=0:(size(mm,1)-1)
        ff = [ff;diag(cf,i)];
    end
    fV = ff';

    d = [];
    for i =1:size(cB,1)
        d = [d; norm(cB(i,:)- fV)];
    end
    [val ind] = min(d);

    p = [p; ind];
    if strcmp(persons(ind, :), testVoices(v).name(1:4))
        testResult = [testResult; 1];
    else
        testResult = [testResult; 0];
    end

end

[test_names num2str([testResult p]) repmat('--', size(p)), persons(p,:)]
TP = sum(testResult)
N = numel(testResult)
```

Published with MATLAB® R2013b

APPENDIX D

COMPLETE MATLAB FILE FOR VOICE RECOGNITION SYSTEM GRAPHIC USER INTERFACE

```
==========================================================================
                    Voice Recognition System GUI
 ==========================================================================
function varargout = SRS(varargin)
% SRS MATLAB code for SRS.fig
%      SRS, by itself, creates a new SRS or raises the existing
%      singleton*.
%
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @SRS_OpeningFcn, ...
                   'gui_OutputFcn',  @SRS_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in play_pushbutton.
function play_pushbutton_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to play_pushbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
ind = get(handles.listbox1, 'Value');
list = get(handles.listbox1, 'String');

if strcmp(get(handles.show_pushbutton, 'String'), 'Show Train Voices')
    audioFile = strcat('VOICE DATA/test/', list(ind,:));
else
    audioFile = strcat('VOICE DATA/train/', list(ind,:));
end

[y, Fs] = audioread(audioFile);

sound(y,Fs)
enable(handles, 'off')
drawnow
pause(length(y)/Fs)
enable(handles, 'on')
drawnow


% --- Executes on button press in add_pushbutton.
% set(hObject, 'String', 'Please wait ...')

% --- Executes on button press in checkbox1.
% Hint: get(hObject,'Value') returns toggle state of checkbox1
val = get(hObject,'Value');
if (val)
    set(handles.test_pushbutton, 'String', 'Test')
else
    set(handles.test_pushbutton, 'String', 'Test all')
end

% --- Executes on button press in train_pushbutton.
function train_pushbutton_Callback(hObject, eventdata, handles)
enable(handles, 'off')
drawnow

method = get(handles.radiobutton3, 'Value');

if method == 1
    [persons, cB] = CSA_train();
else
   [persons, cB] = dCSA_train();
end

dat.persons = persons;
dat.cB = cB;
set(handles.train_pushbutton, 'userdata', dat)

enable(handles, 'on')
drawnow
```

```
% --- Executes when selected object is changed in uipanel2.
function uipanel2_SelectionChangeFcn(hObject, eventdata, handles)
dat = [];
set(handles.train_pushbutton, 'userdata', dat)
```

Published with MATLAB® R2013b