Optimization of Codebook Used in Speech Processing Using Artificial Bee Colony Algorithm

[1]Ch. Vinay Kumar
Student of RVR&JCCE,
Electronics and Communication Engineering,
Guntur,A.P India,
vinaychvinaykumar@gmail.com

[2]D. Surya Teja
Student of RVR&JCCE,
Electronics and Communication Engineering,
Guntur,A.P India,
davulurisurya03@gmail.com

[3]G. Vishnu
Student of RVR&JCCE,
Electronics and Communication Engineering,
Guntur,A.P India,
@vishnugonam9gmail.com

Abstract— The ABC algorithm, drawing inspiration from bee behavior, is applied to enhance codebook optimization in speech processing. This research aims to efficiently compress speech signals for minimized bandwidth consumption. Leveraging Vector Quantization, a block coding method, entails utilizing codebooks crafted through the Linde-Buzo-Gray (LBG) algorithm. The ABC algorithm is suggested as a means of optimizing these codebooks to mitigate spectral distortion. Findings indicate that ABC-optimized codebooks effectively decrease spectral distortion.

Keywords— *Vector quantization, Speech compression, Linde-Buzo-Gray, Codebook, ABC,.*

I. INTRODUCTION

Speech compression plays a vital role in contemporary communication systems, facilitating the streamlined transmission and storage of speech signals. Conventional compression techniques frequently rely on intricate algorithms, which may demand significant computational resources. Artificial Bee Colony (ABC) introduces a fresh perspective on optimizing the compression procedure, drawing on the principles of swarm intelligence inspired by the foraging patterns of bees.

Artificial Bee Colony (ABC) represents a swarm-based meta-heuristic optimization algorithm mimicking the intelligent foraging behavior of honey bees to seek optimal solutions for intricate problems. Within speech compression, ABC can be utilized to refine the codebook employed in vector quantization. Vector quantization entails segmenting the speech signal and representing each segment with a codeword from a codebook. The compression's quality hinges on the codebook's design, ideally comprising codewords closely mirroring the speech signal segments.

In this article, we introduce an innovative method for speech compression employing ABC to refine the codebook for vector quantization. Through experimental findings, we illustrate the efficacy of our approach, showcasing enhanced compression ratios and speech quality in contrast to conventional techniques. Our research makes a significant contribution to speech compression by offering a fresh angle on codebook optimization for proficient representation of speech signals.

II. VECTOR QUANTIZATION

Vector quantization serves as a lossy compression technique employed in signal processing to diminish the number of bits required for signal representation. This method involves segmenting the signal into small, non-overlapping vectors and encoding each vector using a codebook. The codebook comprises code vectors, and the essence lies in identifying the most appropriate code vector from the codebook to represent each vector of the signal. The aim of this process is to minimize distortion between the original signal and the reconstructed signal obtained by decoding the codebook. Typically, the Linde-Buzo-Gray (LBG) algorithm is employed to generate such a codebook, as it effectively partitions training data into clusters. Generally, vector quantization remains a prominent method for reducing data rates while maintaining acceptable quality levels for signals.

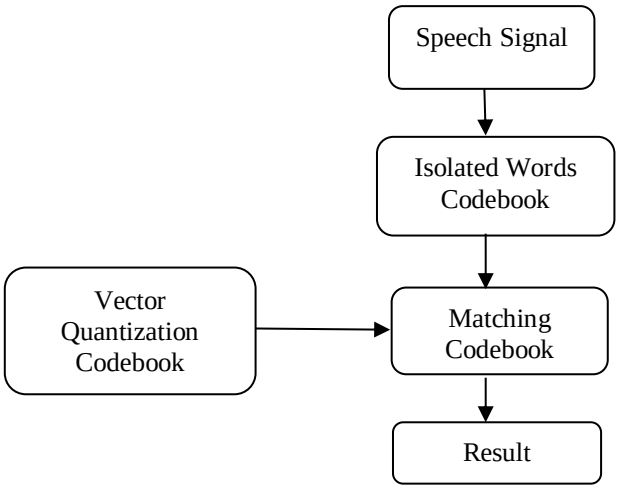

Figure-1. Block diagram of Vector Quantization

III. ABC ALGORITHM

The Artificial Bee Colony (ABC) algorithm, introduced by Karaboga in 2005, is a swarm-based meta-heuristic technique designed for optimizing challenges. In ABC, the search process is modeled after the foraging behavior of bees, where three main types of bees are simulated: employed bees, onlooker bees, and scout bees. Employed bees explore the search space by exploiting the information gathered from the current food sources, while onlooker bees choose food sources based on the employed bees' information. Scout bees are responsible for discovering new food sources by randomly exploring the search space.

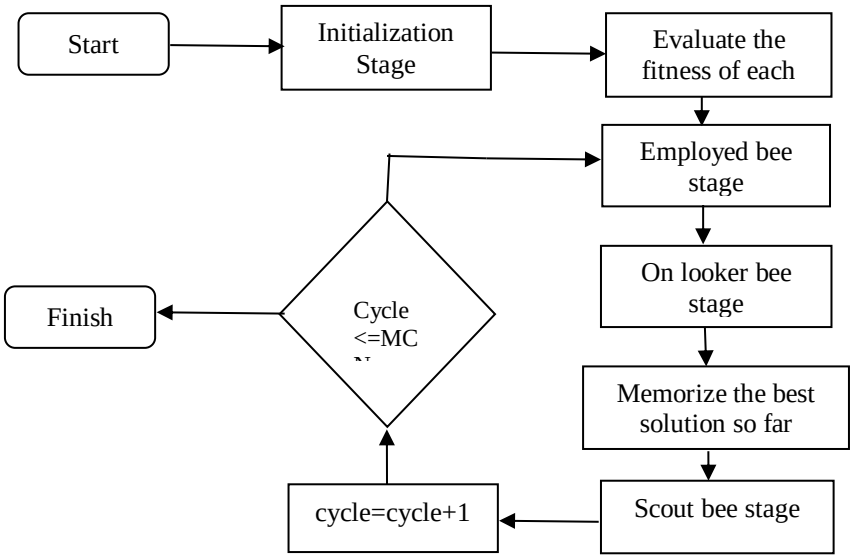

Figure-2. Flow Chart of ABC Algorithm

The ABC algorithm operates iteratively, where each iteration is called a cycle. During each cycle, employed bees explore the neighborhood of the current solutions and determine the fitness of the new solutions. Onlooker bees select food sources based on the employed bees' information and determine their fitness. The best solution found by the employed and onlooker bees is then updated. If a solution is not improved for a certain number of cycles, scout bees are activated to search for new food sources by randomly sampling the search space.

Steps to implement the Artificial Bee Colony Optimization Algorithm are:

Step 1: Generate the initial solution and initial best solution. Each bee represent one codebook.
In this stage, the result of LBG algorithm is first set to the best solution Q, and then the set of initial trivial solutions, x_i, (i=1,2,...,SN) are randomly generated, meanwhile, the each $trail_i$ is assigned to be 0.

Step 2: Place the employed bees to their solutions and each employed bees produces new solution x_i.Calculate the fitness using the fitness equation given below. If the fitness of the new one is higher than that of the previous one, the employed bee memorizes the new position and forgets the old one; otherwise the employed bee keeps the old solution.

$$x_{i,j} = x_{i,j} + \phi_{i,j}(x_{i,j} - x_{k,j})$$

Where k ∈{1,2,.....,SN} but k≠i and j ∈{1,2,....,N_c} are randomly selected indexes.$\emptyset_{ij}$ is a random number between [-1, 1] and the $x_{i,j}$, is the j-th codeword of codebook x_i .

Step 3: Send the onlooker bees to the food sources depending on their amount of nectar.
In Step 3, we first calculate the probability value p_i of the solution x_i by means of their fitness values using below equation. The probability p_i of selecting a food source x_i is determined using the following equation.

$$p_i = \frac{fit(x_i)}{\sum_{i=1}^{SN} fit(x_i)}$$

An onlooker bee selects a solution to update its solution depending on the probabilities and determines a neighbor solution around the chosen one. In the selection procedure for the first onlooker, a random number is produced between [0, 1] and if this number is less than p_1 , the solution is updated. Otherwise, the random number is compared with p_2 and if less than that, the second solution is chosen. Otherwise, the third probability of third solution is checked. This process is repeated until all onlookers have been distributed to solutions. The distributed onlooker bee updates its own solution just as the employed bees do.

Step 4: Send the scouts to the search area to discover new food sources.
If the solution z_i is not improved through the Step 2 and Step 3, the $trail_i$ value of solution x_i will be increased by 1. If the $trail_i$ of solution is more than the predetermined “limit” the solution x_i is considered to be an abandoned solution, meanwhile, the employed bee will be changed into a scout. The scout randomly produces the new solution by Step 2 equation and then compares the fitness of new solution with that of its old one. If the new solution is better than the old solution, it is replaced with the old one and set its own $trail_i$ into 0. This scout will be changed into an employed bee. Otherwise, the old one is retained in the memory.

$$v_{i,j}^t = x_{i,j}^{\min} + rand(0,1)(x_{i,j}^{\max} - x_{i,j}^{\min}), j = 1,\ldots,N_c, \quad t = 1,2,\ldots,L.$$

where the min $x_{i,j}^{min}$ and max $x_{i,j}^{max}$ are the minimum and maximum of t-th component of all codewords $x_{i,j}$, in the solution x_i , the rand(0,1) is a random number generating function that generates the random number between [0, 1].

Step 5: Record the best solution.
If the fitness of x_{best} is larger than the fitness of the best solution Q, the best solution Q is replaced by x_{best} and increases the cycle by 1.

Step 6: Check the termination criterion.
If the cycle is equal to the maximum cycle number (MCN) then the algorithm is finished; otherwise go to Step 2.

IV. SPECTRAL DISTORTION MEASURE

In In the realm of speech coding, maintaining a superior quality of the speech signal holds paramount importance. A key method for evaluating this quality is by gauging spectral distortion, typically expressed in decibels (dB). To attain transparency in coding, where any quantization remains imperceptible to the listener, it's crucial to limit the average spectral distortion to less than 1 dB. This distortion is assessed by examining the LPC power spectra of both the quantized and original speech signals frame by frame. The total spectral distortion value is subsequently derived by averaging these distortions across all frames.

$$SD_i = \sqrt{\frac{1}{(f_2-f_1)}\int_{f_1}^{f_2}[10\log_{10} s_i(f) - 10\log_{10} \hat{s}_i(f)]df(db)}$$

Where $S_i(f)$ and $\hat{S}_i(f)$the LPC power spectra of the unquantized and quantized ith frame respectively. The frequency "f" is expressed in Hz, while "f1" indicates the frequency range. For narrowband speech coding, the frequency range in use is 0 to4000 Hz. The average or mean of the spectral distortion SD is given by equation

$$SD = \frac{1}{N}\sum_{i=1}^{N} SD_i$$

The conditions for transparent speech coding are:
1. Average spectral distortion (SD) ≤ 1dB.
2. No outlier frames with distortion > 4dB.
3. Percentage of frames with 2-4dB distortion < 2%.

V. RESULTS

Transparent speech coding requires that the average spectral distortion (SD) remains below 1dB, with no outlier frames showing a distortion exceeding 4dB. Additionally, the percentage of frames with distortions between 2 and 4dB should be less than 2%.

Bits / frame	SD (dB)	Percentage of outliers	
		2-4 dB	>4dB
24(8+8+8)	1.411	0.22	0.03
23(7+8+8)	1.900	0.23	0.03
22(7+7+8)	1.907	0.24	0.03
21(7+7+7)	1.915	0.27	0.10
20(6+7+7)	2.481	0.28	0.10

Table-1. Spectral distortion of LBG Vector quantization

Bits / frame	SD (dB)	Percentage of outliers	
		2-4 dB	>4dB
24(8+8+8)	1.204	0.15	0.02
23(7+8+8)	1.542	0.20	0.05
22(7+7+8)	1.518	0.23	0.06
21(7+7+7)	1.498	0.17	0.07
20(6+7+7)	2.412	0.26	0.19

Table-2. Spectral distortion of ABC Vector quantization

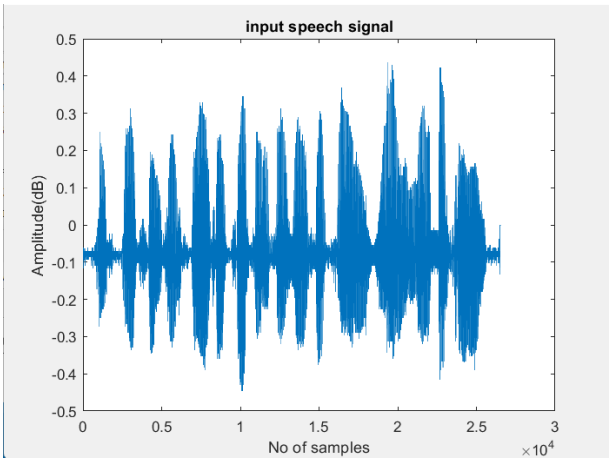

Figure-2(a). Input Speech Signal

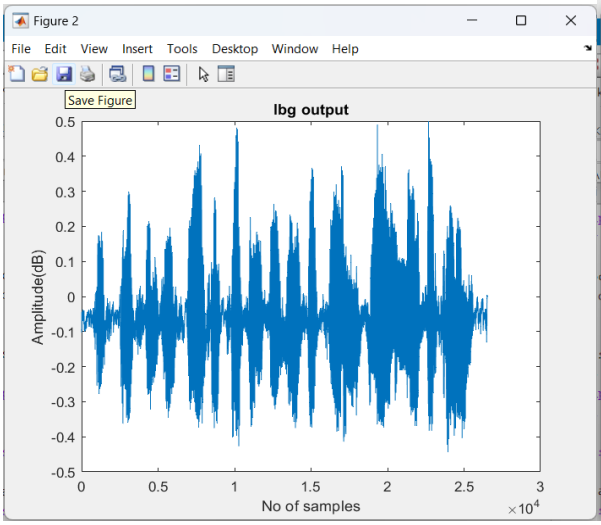

Figure-2(b). LBG Speech Signal (8-bit)

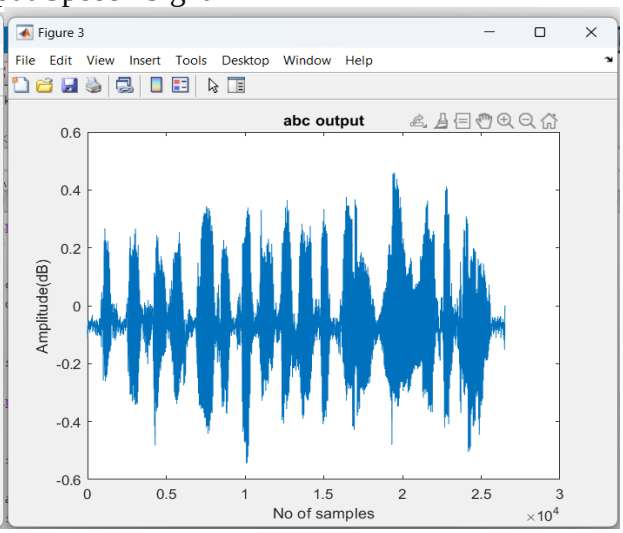

Figure-2(c). ABC Speech Signal (8-bit)

VI. CONCLUSION

In summary, the ABC algorithm presents a hopeful avenue for refining codebooks in speech signal compression, leading to a notable decrease in spectral distortion. Through its utilization of bee-inspired behavior, ABC improves the efficacy of codebook generation, working alongside the established LBG algorithm. This research underscores the potential of ABC in enhancing speech processing capabilities and optimizing bandwidth utilization within communication systems.

VII. REFERENCES

- C. Chuang, Y. C. Hu, C. C. Lo, and W. L. Chen, "Grayscale image tamper detection and recovery based on vector quantization," International Journal of Security and Its Applications, Vol. 7, 2013, pp. 209-228
- D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," Journal of Global Optimization, Vol. 39, 2007, pp. 459-471.
- Linde, A. Buzo, and R. –M. Gray, An Algorithm for Vector Quantizer design. IEEE Transaction on Communications, no. 1, vol. 28, pp. 84-95, 1980
- B. Karayiannis and Z. Liu, "Split and merge codebook design algorithms for image compression," Journal of Electronic Imaging, Vol. 9, 2000, pp. 509-520
- D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," Journal of Global Optimization, Vol. 39, 2007, pp. 459-471.
- D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," Applied Soft Computing, Vol. 8, 2008, pp. 687-697.
- Z. C. Lai, Y. C. Liaw, and J. Liu, "A fast VQ codebook generation algorithm using codeword displacement," Pattern Recognition, Vol. 41, 2008, pp. 315-319