Prediction of Gain in LD-CELP Using Hybrid Genetic/PSO-Neural Models

Mansour Sheikhan and Sahar Garoucy
Department of Electrical Engineering,
Islamic Azad University, South Tehran Branch, Tehran, Iran
Telephone: +98 21 88605565
Fax: +98 21 33717140
E-mail address: msheikhn@azad.ac.ir

Abstract

Abstract: In this paper, the gain in LD-CELP speech coding algorithm is predicted using three neural models, that are equipped by genetic and particle swarm optimization (PSO) algorithms to optimize the structure and parameters of neural networks. Elman, multi-layer perceptron (MLP) and fuzzy ARTMAP are the candidate neural models. The optimized number of nodes in the first and second hidden layers of Elman and MLP and also the initial weights and biases of these nets are determined by genetic algorithm (GA) and PSO. In the fuzzy ARTMAP, the choice parameter, α, learning rate, β, and vigilance parameter, ρ, are selected by GA and PSO, as well. In this way, the performance of GA and PSO are compared when using different neural architectures in this application. Empirical results show that when gain is predicted by Elman and MLP neural networks with GA/PSO-optimized parameters, the segmental signal to noise ratio (SNRseg) and mean opinion score (MOS) are improved as compared to traditional implementation based on ITU-T G.728 recommendation. On the other hand, fuzzy ARTMAP-based gain predictor reduces the computational complexity noticeably, with no significant degradations in SNRseg and MOS.

Keywords: *Speech coding, neural networks, genetic algorithm, particle swarm optimization.*

Abbreviations:

LD-CELP: Low Delay Code Excited Linear Prediction;
AbS: Analysis by Synthesis;
LPC: Linear Prediction Coding;
ANN: Artificial Neural Network;
GA: Genetic Algorithm;
PSO: Particle Swarm Optimization;
SNR_{seg}: Segmental Signal to Noise Ratio;
MOS: Mean Opinion Score.

1. Introduction

In May 1992, Consultative Committee for International Telephony and Telegraphy (CCITT) approved a 16 kbps low delay code excited linear prediction (LD-CELP) coding algorithm with a delay less than 2 ms and recommended it as G.728 [1]. In 1994, fixed-point version of LD-CELP was introduced [2]. LD-CELP is a member of CELP coder family which is based on the analysis by synthesis (AbS) technique proposed by B.S. Atal and R. Remde in 1982 [3]. In LD-CELP, short-term and long-term predictor coefficients and also log-gain predictors are obtained adaptively backward and are updated by LPC analysis of the former quantized speech and excitation, respectively. Many researchers got interested to improve the performace of this codec [4-8].

On the other hand, artificial neural networks (ANNs) are increasingly used in the field of speech coding algorithms in the recent decades [9-17]. In our pervious work [18], three neural gain predictors were proposed for LD-CELP speech coding algorithm. Those ANNs were Elman, MLP and fuzzy ARTMAP. In that work, the parameters of networks were selected by experiments. In this paper, the structure and parameters of mentioned candidate neural networks (e.g. the number of nodes in hidden layers, initial weights and biases of Elman and MLP, and critical learning parameters of fuzzy ARTMAP) are optimized using genetic algorithm (GA) and particle swarm optimization (PSO) algorithm.

GA finds approximate solutions to optimization and search problems. Genetic algorithm is a particular class of evolutionary algorithms that uses techniques inspired by evolutionary biology such as inheritance, mutation, and recombination [19]. PSO is proposed by J. Kennedy and R.C. Eberhart in 1995, motivated by social behavior of organisms. PSO provides a population-based search procedure in which individuals, called particles, change their position (state) with time. In PSO, particles fly around in a multidimensional search space. During flight, each particle adjusts its position according to its own experience and neighboring particle, making use of the best position encountered by itself and its neighbor. Thus, as in modern GAs, a PSO system combines local search methods with global search methods, attempting to balance exploration and exploitation [20-24].

This paper is organized as follows. Section 2 gives a brief overview of the 16 kbps LD-CELP architecture. In section 3, proposed hybrid genetic-neural models are introduced. Proposed hybrid PSO-neural models are discussed in section 4. Empirical results are reported in section 5 and conclusions are drawn in section 6.

2. Architecture of LD-CELP Coder

LD-CELP is an analysis-by-synthesis codebook driven method for linear predictive speech coding [1]. In this coder, which is an encoding method based on a source filter model, speech is reproduced using excitation codevectors that are time-series signals and are stored in an excitation codebook to drive a linear predictive synthesis filter that represents the spectral envelope of input speech. The optimal excitation codevector is selected from the excitation codebook by using a closed-loop search according to the analysis-by-synthesis (AbS) method to find the one having the minimum perceptually-weighted waveform distortion of synthetic speech signal to the input speech signal. The basic structure of encoder with the proposed modification in its gain adaptation block is shown in Fig. 1.

As shown in Fig. 1, backward gain adaptation block is replaced by a GA/PSO-optimized ANN model. As shown in the block diagram of backward adaptation of excitation gain (Fig. 2), input and output of this block are gain-scaled excitation, $e(n)$, and excitation gain, $\sigma(n)$, respectively. The 1-vector delay unit makes the previous gain-scaled excitation vector, $e(n-1)$, available. The root-mean-square (RMS) calculator then calculates the RMS value of the vector *e(n–1)*. Then, the logarithm calculator calculates the dB value of the RMS of *e(n–1)*. A log-gain offset value of 32 dB is stored in the log-gain offset value holder. The adder subtracts this log-gain offset value from the logarithmic gain produced by the logarithm calculator. The offset removed logarithmic gain, $\delta(n-1)$, is then used by the hybrid windowing module and the Levinson-Durbin recursion module. The output of Levinson-Durbin recursion module is the coefficients of tenth order LPC. The bandwidth expansion module then moves the roots of this polynomial radially toward the origin of z-plane. The predictor attempts to predict $\delta(n)$ based on a linear combination of $\delta(n-1)$, $\delta(n-2)$, ..., $\delta(n-10)$ [1]. The predicted version of $\delta(n)$ is denoted as $\hat{\delta}(n)$ and is given by:

$$\hat{\delta}(n) = -\sum_{i=1}^{10} a_i \delta(n-i) \tag{1}$$

In the next step, offset value adds to $\hat{\delta}(n)$ and then the log-gain limiter clips the level of it, if the log-gain value is below 0 dB or above 60 dB. Finally, in inverse logarithm calculator the value of log-gain in logarithmic domain is converted to linear domain.

To predict the gain by ANN, the scaled excitation vector, *e(n)*, is fed as the input pattern to network and the excitation gain, $\sigma(n)$, is assumed as the output of network. The codebook search module, searches through 1024 candidate codevectors in the excitation vector quantization (VQ) codebook and finds the index of the best codevector. Indeed, in excitation VQ codebook, the best shape codevector and the best gain value which are extracted from codebook module are multiplied by each other to get the quantized excitation vector *y(n)*. Then, this vector multiplies by gain and results the scaled excitation vector *e(n)*. Excitation gain is the output of backward gain adaptation block. The dimension of scaled excitation vector is 5.

3. Hybrid GA-Neural Models

The genetic algorithm is a method for solving optimization problems based on natural selection, the process that drives biological evolution. The genetic algorithm repeatedly modifies a population of individual solutions. At each step, the genetic algorithm selects individuals randomly from the current population to be parents and uses them to produce the children for the next generation. There are several methods for selecting parents such as stochastic uniform selection, remainder selection, roulette selection and tournament selection. Fig. 3 shows the flowchart of GA algorithm. The details of GA-neural models for gain prediction are reported in the following.

3.1 GA-Elman/MLP Gain Predictor

Elman NN is a type of partial recurrent networks with an additional feedback connection from the output of the first layer to its input layer [25-26]. In this study, Elman has tangent sigmoid ('tansig') neurons in its two hidden layers, and linear neurons in its output layer. In this work, the optimized number of neurons in hidden layers is selected by GA. The fitness function measures the quality of the solution in GA and is application-dependent. In this application, the fitness function in Elman and MLP neural model simulations is chosen as follows [27]:

$$F = \frac{1}{(MSE)^2} \tag{2}$$

At the beginning, the generated values by fitness function are not suitable for selection process of patterns. So, fitness scaling is necessary to map those raw values into a new suitable range for the selection function. The range of scaled values affects the performance of genetic algorithm. In this study, "Rank" fitness scaling function is used to remove the effects of raw scores spread. To create the next generation, GA uses elite children that are individuals with the best fitness values in the current generation. In our simulations, population size is assumed to be 40. Two elite children, 26 crossover children, and 12 mutation children are used. It is noted that the fraction of individuals that is used in crossover process is set to 0.7. The Gaussian function is used as the mutation function. The amount of mutation is decreased at each new generation (proportional to the standard deviation of Gaussian distribution). "Shrink" parameter determines the rate of this decrement. The standard deviation of Gaussian distribution is decreased linearly until its final value reaches to (1–Shrink) times of its initial value at the first generation. The value of "Shrink" parameter is set to 1 in our simulations. Several values of population size and different types of selection, crossover and mutation functions are used in our investigations. By using the "Rank" scaling function, some of the best results for the optimized number of nodes in hidden layers, in terms of fitness value *(F)*, are listed in Table 1.

The initial weights and biases of Elman NN are optimized by GA, too. The optimized weights and biases of first hidden layer nodes for the 8-10-1 topology are reported in Eq. 3 and Eq. 4, respectively.

$$W_{opt\_GA} = \begin{bmatrix} 1.0877 & 1.3928 & 2.8673 & -0.1782 & -1.1239 \\ 0.9202 & -0.0769 & -0.3761 & 1.3426 & -2.7510 \\ 1.1511 & -1.3513 & 0.7900 & 1.2148 & 1.4329 \\ 0.6071 & -1.2600 & -0.0103 & 3.6416 & -1.5062 \\ 3.1232 & 3.5332 & -2.2861 & -2.0304 & 3.6926 \\ 1.9257 & -1.3547 & -3.7061 & -1.2756 & -2.9801 \\ -2.6184 & -1.8240 & -3.1049 & -2.5749 & 0.1191 \\ -1.4636 & -2.9232 & 0.2342 & -1.0640 & -3.2962 \end{bmatrix} \tag{3}$$

$$B_{opt\_GA} = [-2.3218 \; -2.8676 \; -3.0072 \; 2.7906 \; 2.4241 \; 1.2479 \; 0.7751 \; -3.8561] \tag{4}$$

The training parameters of Elman network in hybrid GA-neural model are listed in Table 2. It is noted that the Elman and MLP networks are simulated in Neural Networks Toolbox of MATLAB software. The training dataset includes 40,000 vectors of fifteen male and twenty female speakers with different accents. The test dataset includes 9,000 vectors, as well.

The optimized number of hidden layers nodes and initial weights and biases for MLP are determined by GA, too. The training parameters of MLP network in hybrid GA-neural model are listed in Table 3, as well.

3.2 GA-Fuzzy ARTMAP Gain Predictor

Until now, many types of adaptive resonance theory (ART) family networks have introduced and used. In general, this family of neural networks include ART_1, ART_2 [28], ART_3 [29], ARTMAP [30], Fuzzy ART [31], ART-EMAP [32], Distributed ARTMAP [33], Boosted ARTMAP [34], Fuzzy ARTVar [35], μ_ARTMAP [36] and Fuzzy ARTMAP [37].

Fuzzy ARTMAP is the neural network architecture for incremental supervised learning of recognition categories and multidimensional maps in response to arbitrary sequences of analog or binary input vectors. It achieves a synthesis of fuzzy logic and ART neural networks by exploiting a close formal similarity between the computations of fuzzy method and ART category choice, resonance and learning. ARTMAP networks consist of two ART_1 networks, ART_a and ART_b, bridged via an inter-ART module. An ART_1 module has three layers: input layer *(F_0)*, the comparison layer *(F_1)*, and the recognition layer *(F_2)*. Fuzzy ARTMAP is a natural extension to ARTMAP that uses fuzzy ART instead of ART_1 modules.

The operation of fuzzy ARTMAP is affected by two network parameters, the choice parameter, α, and the baseline vigilance parameter, ρ. Both of these parameters take values in the interval [0,1] and affect the number of nodes created in the category representation layer of fuzzy ARTMAP. Another important parameter is the leaning rate, β. In this study, the optimized values of these three parameters are determined by GA to have the best correct identification rate.

The dataset which is used to train the Elman and MLP networks is not suitable for fuzzy ARTMAP and some preprocessing is needed. Fuzzy ARTMAP requires input patterns to be presented as vectors of floating point numbers in the range [0, 1]. Therefore, the training and test datasets need normalization. The value of excitation gain in G.728 recommendation is in the range of [0 dB,60 dB]. In our fuzzy ARTMAP structure, this interval is divided to 540 classes. So, the resolution of this classification is about 0.1 dB. The fitness function in fuzzy ARTMAP simulation is chosen as follows:

$$F = (pc)^2 \tag{5}$$

where pc is the correct classification rate. The population size and selection/crossover/mutation functions that result the best fitness value are reported in Table 4. The specifications of fuzzy ARTMAP-based gain predictor in our simulations are reported in Table 5.

4. Hybrid PSO-Neural Models

In PSO, particles move in a multidimensional search space. In this algorithm, each particle has a velocity and a position as follow:

$$v_i(k+1) = v_i(k) + \gamma_{1i}(P_i - x_i(k)) + \gamma_{2i}(G - x_i(k)) \tag{6}$$

$$x_i(k+1) = x_i(k) + v_i(k+1) \tag{7}$$

where i is particle index, k is discrete time index, v_i is velocity of ith particle, x_i is position of ith particle, P_i is the best position found by ith particle (personal best), G is the best position found by swarm (global best). γ_{1i} and γ_{2i} are random numbers in the interval [0,1] applied to ith particle. In our simulations, the following equation is used for velocity [20]:

$$v_i(k+1) = \phi(k)v_i(k) + \alpha_1\left[\gamma_{1i}(P_i - x_i(k))\right] + \alpha_2\left[\gamma_{2i}(G_i - x_i(k))\right] \tag{8}$$

in which ϕ is inertia function, α_1 and α_2 are acceleration constants.

4.1 PSO-Elman/MLP Gain Predictor

The number of nodes in hidden layers, and the initial weights and biases of Elman and MLP neural models are selected by PSO algorithm in this section. In our simulations, the maximum particle velocity is set to 2, population size is 20, and acceleration constants are set to 2. The inertia is taken as a decreasing linear function from 0.9 to 0.2. So, the influence of past velocity becomes smaller. The training parameters of Elman and MLP networks in hybrid PSO-neural models are listed in Table 6 and Table 7, respectively.

4.2 PSO-Fuzzy ARTMAP Gain Predictor

The optimized values of learning rate, choice and vigilance parameters, as three important parameters in fuzzy ARTMAP, are determined by PSO algorithm to have the best correct identification rate in classifier. The specifications of this neural gain predictor in our simulation are listed in Table 8.

5. Experimental Results

A 16 kbps LD-CELP coder, based on the ITU-T G.728 recommendation, is implemented in this work [1]. Farsi speech data files of FARSDAT [38] are used as dataset in this study. The performance comparison of Elman-based and MLP-based gain predictors, with optimized parameters by GA or PSO, shows that MSE in Elman is lower than MLP. The number of training epochs in Elman is lower than MLP, too. On the other hand, the number of epochs and training time of fuzzy ARTMAP are the lowest ones. The execution time of proposed hybrid models, calculated for 1000 frames of speech, is also compared to traditional backward gain adaptation, based on G.728, as the reference. This comparison shows a reduction in execution time, when using each of GA/PSO-neural hybrid models. However, GA-fuzzy ARTMAP hybrid model has the lowest execution time (Fig. 4).

The performance of proposed optimized-neural gain predictors, in terms of segmental signal-to-noise ratio (SNR_{seg}) and mean opinion score (MOS), is also compared with a traditional G.728 [4, 5]. In this way, the SNR_{seg} and MOS of traditional implementation are 18.45 dB and 3.91, respectively [5, 39]. The comparison of SNR_{seg} and MOS of proposed hybrid models with traditional G.728, in terms of relative values, is shown in Fig. 5. This comparison shows that an average of 0.6 dB improvement in SNR_{seg} and also an improvement of 0.3 in MOS are achieved, when using GA/PSO-Elman/MLP hybrid models. However, an average of 0.18 dB reduction in SNR_{seg} and 0.4 reduction in MOS are experienced when using GA/PSO-fuzzy ARTMAP hybrid models.

6. Conclusions

In this paper, backward gain adaptation module of G.728 speech coder was replaced by three candidate neural gain predictors with optimized structures and parameters by employing GA and PSO algorithms. Elman, MLP and fuzzy ARTMAP were the candidate neural models in this work. Empirical results showed that gain prediction by optimized-GA/PSO Elman and MLP neural networks, improved the SNR_{seg} and MOS as compared to traditional implementation of G.728. On the other hand, fuzzy ARTMAP-based gain predictor reduced the computational complexity noticeably, with no significant degradations in SNR_{seg} and MOS.

References:

1. CCITT: Recommendation G.728, Coding of speech at 16 kb/s using low delay code excited linear prediction. Geneva (1992)
2. ITU-T: G.728 Annex G, 16 kbps Fixed point specification. (1994)
3. Atal, B.S., Remde, R.: A new model of LPC excited for producing natural-sounding speech at low bit rates. Proc. ICASSP, 614-617 (1982)
4. ITU-T: G.728 Implementer's Guide, Coding of speech at 16kb/s using low delay code excited linear prediction. (2006)
5. Zahang, G., Xie, K.M., Huangfu, L.Y.: Optimization gain codebook of LD-CELP. Proc. ICASSP. 2, 149-152 (2003)
6. Zahang, G., Xie, K.M., Huangfu, L.Y.: Improving G.728's hybrid window and excitation gain. Proc. IEEE Asia-Pacific Circuits and System Conf. 1, 185-188 (2004)
7. Zahang, G., Xie, K.M., Zhang, X.Y., Huangfu, L.Y.: Improvement on G.728 by normalized shape codebook with exact gain. J. China Institute of Communications. 24, 87-92 (2003)
8. Xueying, Z., Qun, Z.Q., Yanys, M.Z.: Reducing the complexity of LD-CELP speech coding algorithm using direct vector quantization. Proc. Int. Conf. Communications, Circuits and Systems. 10, 811-815 (2008)
9. Faundez, M.: Adaptive hybrid speech coding with a MLP/LPC structure. Proc. Int. Work-Conf. Artificial and Natural Neural Networks. 11, 814-823 (1999)
10. Birgmeier, M.: Nonlinear prediction of speech signals using radial basis function networks. Proc. European Signal Processing Conf. 1, 459-462 (1996)
11. Faúndez-Zanuy, M.: Nonlinear speech coding with MLP, RBF and Elman based prediction. Lecture Notes in Computer Science. 2687, 671-678 (2003)
12. Sassi, S.B., Braham, R., Belghith, A.: Neural speech synthesis system for Arabic language using CELP algorithm. Proc. Int. Conf. Computer Systems and Applications, 119-121 (2001)
13. Hernandez-Gomez, L.A., Lopez-Gonzalo, E.: Phonetically-driven CELP coding using self-organizing maps. Proc. ICASSP. 2, 628-631 (1993)
14. Huong, V., Min, B-J., Park, D-C., Woo, D-M.: A new vocoder based on AMR 7.4kbit/s mode in speaker dependent coding system. Proc. Int. Conf. Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing. 6, 163-167 (2008)
15. Indrayanto, A., Langi, A., Kinsner, W.: A neural network mapper for stochastic codebook parameter encoding in code excited linear predictive speech processing. Proc. IEEE Western Canada Conf. Computer, Power and Communications Systems. 6, 221-224 (1991)
16. Easton, M.G., Goodyear, C.C.: A CELP codebook and search technique using a Hopfield net. Proc. ICASSP, 685-688 (1991)
17. Wu, S., Zhang, G., Zhang, X., Zhao, Q.L.: A LD-aCELP speech coding algorithm based on modified SOFM vector quantizer. Proc. Int. Symp. Intelligent Information Technology Applications, 408-411 (2008)
18. Sheikhan, M., Tabataba Vakili V., Garoucy, S.: Complexity reduction of LD-CELP speech coding in prediction of gain using neural networks. World Applied Sciences Journal. 7 (Special Issue of Computer & IT). 7, 38-44 (2009)
19. Goldberg, D.E.: Genetic Algorithms in Search Optimization and Learning. Addison Wesley, New York (1989)
20. Shi, Y., Eberhart, R.C.: Parameter selection in particle swarm optimization. Proc. Int. Conf. *Evolutionary Programming*, 591-600 (1998)
21. Shi, Y., Eberhart, R.C.: Empirical study of particle swarm optimization. Proc. IEEE Int. Conf. *Evolutionary Computation*, 1945-1950 (1999)

22. Zhang Y., Huang, S.: A novel multiobjective particle swarm optimization for buoys-arrangement design. Proc. *IEEE/WIC/ACM* Int. Conf. *Intelligent Agent Technology*, 24-30 (2004)
23. Parsopoulos, K.E., Vrahatis, M.N.: Particle swarm optimization method in multiobjective problems. Proc. ACM Int. Conf. *Applied Computing,* 603-607 (2002)
24. Clerc, M., Kennedy, J.: The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evolutionary Computation.* 6, 58-73 (2002)
25. Elman, J.L.: Finding structure in time. Cognitive Science. 14, 179-211 (1990)
26. Hertz, J., Krogh, A., Palmer, R.G.: Recurrent neural networks: Introduction to the theory of neural computations. In: Wesley, A. (eds.), *Neural Networks*, pp. 776–778. Massachusetts (1991)
27. Chen, M., Yao, Z.: Classification techniques of neural networks using improved genetic algorithm. Proc. IEEE Int. Conf. Genetic and Evolutionary Computing, 115-119 (2008)
28. Carpenter, G.A., Grossberg S.: ART_2: Self-organization of stable category recognition codes for analog input patterns. Applied Optics. 26, 4919-4930 (1987)
29. Carpenter, G.A., Grossberg S.: ART_3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. Neural Networks. 3, 129-152 (1990)
30. Carpenter, G.A., Grossberg, S., Reynolds, J.H.: ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. Neural Networks. 4, 565-588 (1991)
31. Carpenter, G.A., Grossberg, S., Rosen, D.B.: Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. Neural Networks. 4, 759-771 (1991)
32. Carpenter, G.A., Ross, W.: ART-EMAP: A neural network architecture for learning and prediction by evidence accumulation. IEEE Trans. Neural Networks. 6, 805-818 (1995)
33. Carpenter, G.A., Milenova, B.L., Noeske, B.W.: Distributed ARTMAP: A neural network for fast distributed supervised learning. Neural Networks. 11, 793-813 (1998)
34. Verzi, S.J., Heileman, G.L., Georgiopoulos, M., Healy, M.J.: Boosted ARTMAP. Proc. IEEE Int. Conf. Computational Intelligence. 1, 396-400 (1998)
35. Dagher, I., Georgiopoulos, M., Heileman, G.L., Bebis, G.: Fuzzy ARTVar: An improved fuzzy ARTMAP algorithm. Proc. IEEE Int. Conf. Computational Intelligence. 3, 1688-1693 (1998)
36. Gomez-Sanchez, E., Dimitriadis, Y.A., Cano-Izquierdo, J.M., Lopez-Coronado, J.: μ_ARTMAP: Use of mutual information for category reduction in fuzzy ARTMAP. IEEE Trans. Neural Networks. 13, 58-69 (2002)
37. Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H., Rosen, D.B.: Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. IEEE Trans. Neural Networks. 3, 698-713 (1992)
38. Bijankhan, M., Sheikhzadegan, J., Roohani, M.R, Smareh, Y., Lucas, C., Tebiani, M.: The speech database of Farsi spoken language. Proc. Int. Conf. Speech Science and Technology, 826-831 (1994)
39. Chen, J.-H., Cox, R.V., Chun Lin, Y., and Jayant, N., Melchner, M.J.: A low delay CELP coder for CCITT 16 kb/s speech coding standard. IEEE Journal Selected Areas on Communications. 10, 830-849 (1992)

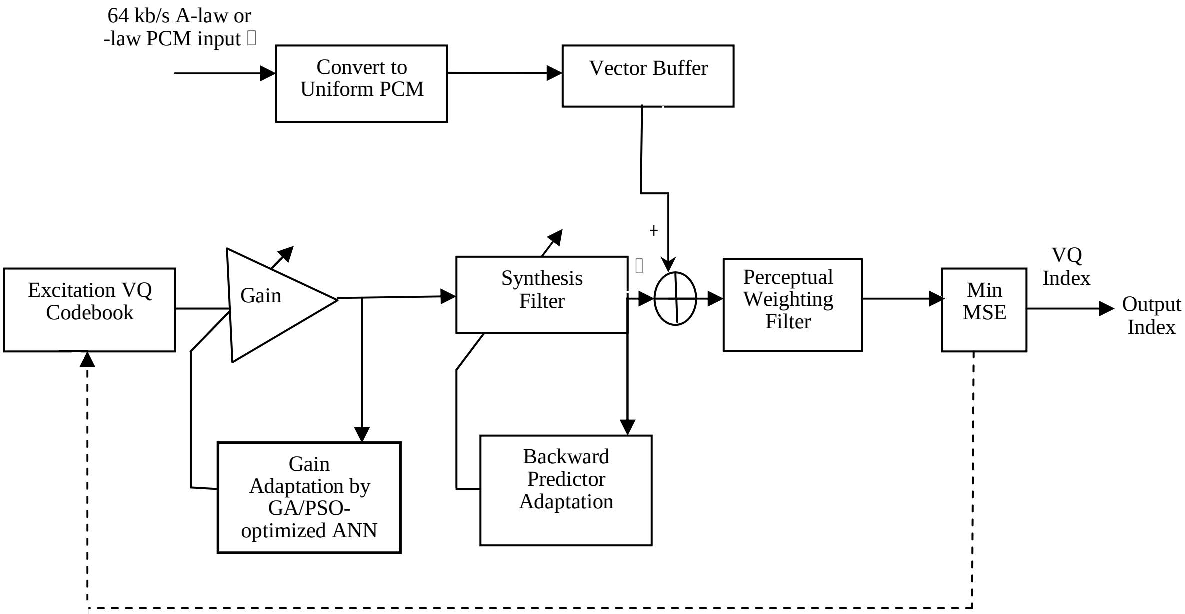

Fig. 1 *Block diagram of LD-CELP encoder [1] and the proposed modification*

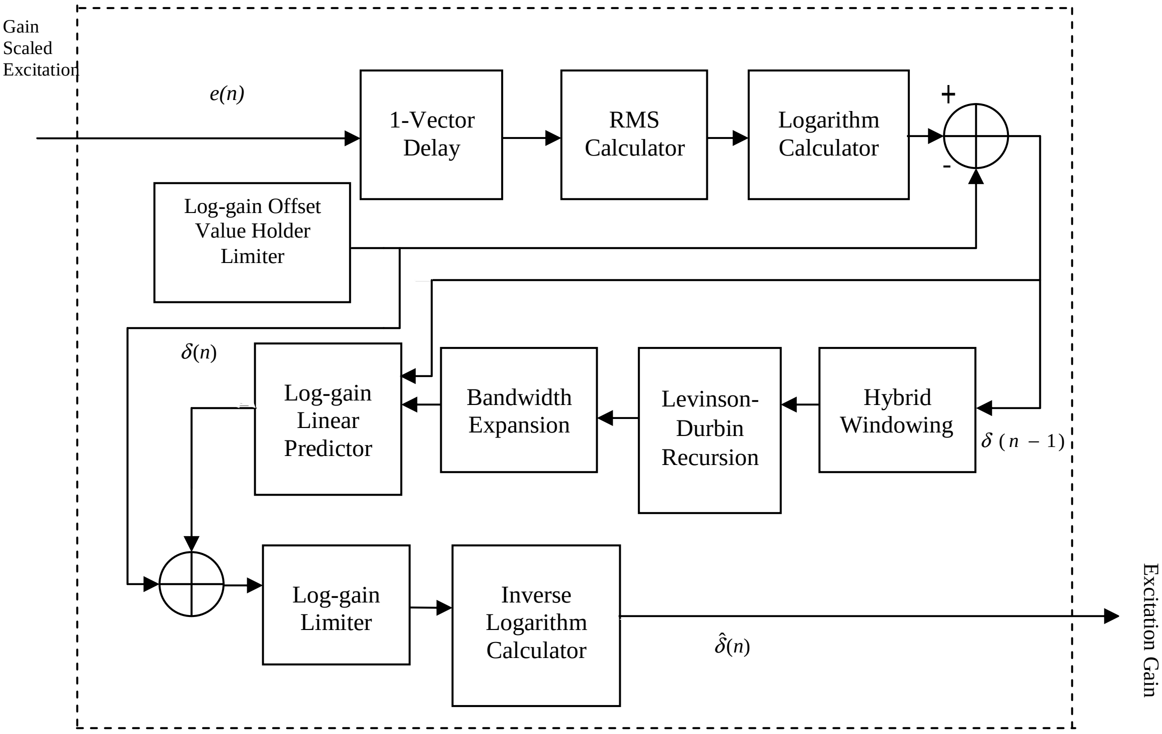

Fig. 2 *Block diagram of backward gain adaptation in LD-CELP [1]*

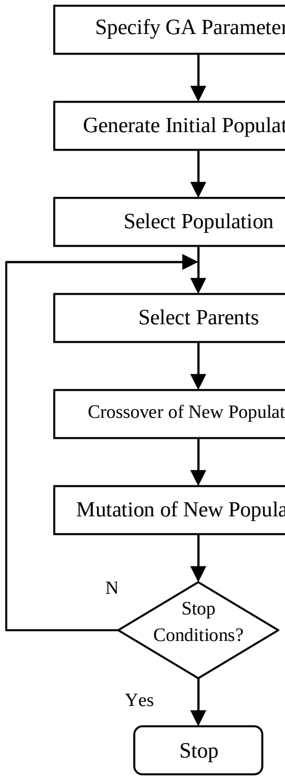

Fig. 3 *GA flowchart*

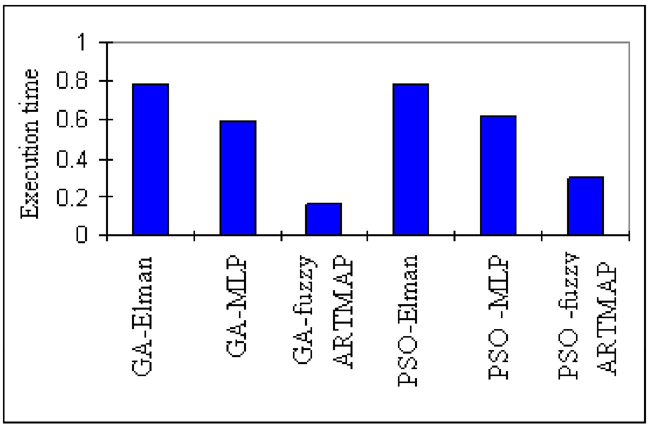

Fig. 4 *Relative execution time of proposed models as compared to traditional implementation of gain predictor*

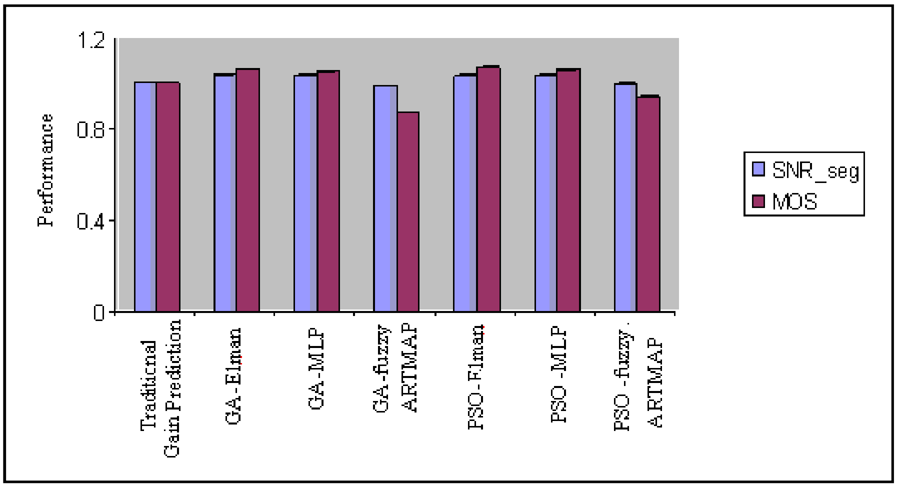

Fig. 5 *Relative SNR_{seg} and MOS of proposed models as compared to traditional G.728 implementation*

Table 1 *Optimized number of hidden layer nodes in Elman NN*

Population size	**Selection function**	**Crossover function**	**Mutation function**	**Number of hidden layer nodes**	**F(×10^2)**
40	Uniform	Heuristic	Gaussian	8-10	49.43
40	Roulette	Scattered	Gaussian	9-10	47.50
40	Stochastic uniform	Scattered	Gaussian	10-9	42.90
20	Stochastic uniform	Intermediate	Uniform	10-8	37.68

Table 2 *Training parameters of Elman network in hybrid GA-neural model*

Parameter	***Value or type***
Train function	'trainlm'
net.trainParam.goal	*0.01*
Number of nodes in layers	*8-10-1*
Transfer function of layers	*'tansig', 'tansig', 'purelin'*
Number of epochs	*1500*
MSE on the test data	*0.01155*

Table 3 *Training parameters of MLP network in hybrid GA-neural model*

Parameter	***Value or type***
Train function	'trainlm'
net.trainParam.goal	*0.01*
Number of nodes in layers	*11-9-1*
Transfer function of layers	*'tansig', 'tansig', 'purelin'*
Number of epochs	*2000*
MSE on the test data	*0.01308*

Table 4 *GA specifications in GA-fuzzy ARTMAP hybrid gain prediction*

Population size	**Selection function**	**Crossover function**	**Mutation function**	**F(×10^2)**
40	Stochastic uniform	Scattered	Gaussian	81

Table 5 *Fuzzy ARTMAP gain predictor specifications, GA-optimized parameters*

Specification	**Value**
Learning rate β	*0.9846*
Vigilance parameter ρ_a	*0.9738*
Vigilance parameter ρ_{ab}	*0.3802*
Choice parameter α	*0.9889*
Number of F_0 nodes	*10*
Number of F_1 nodes	*301*
Number of F_2 nodes	*301*
Number of epochs	*1*
Correct identification rate	*90%*

Table 6 *Training parameters of Elman network in hybrid PSO-neural model*

Parameter	***Value or type***
Train function	'trainlm'
net.trainParam.goal	*0.01*
Number of nodes in layers	*11-11-1*
Transfer function of layers	*'tansig', 'tansig', 'purelin'*
Number of epochs	*1200*
MSE on the test data	*0. 01115*

Table 7 *Training parameters of MLP network in hybrid PSO-neural model*

Parameter	***Value or type***
Train function	'trainlm'
net.trainParam.goal	*0.01*
Number of nodes in layers	*11-10-1*
Transfer function of layers	*'tansig', 'tansig', 'purelin'*
Number of epochs	*2000*
MSE on the test data	*0.01174*

Table 8 *Fuzzy ARTMAP gain predictor specifications, PSO-optimized parameters*

Specification	**Value**
Learning rate β	1
Vigilance parameter ρ_a	0.8008
Vigilance parameter ρ_{ab}	0.3001
Choice parameter α	1
Number of F_0 nodes	10
Number of F_1 nodes	621
Number of F_2 nodes	621
Number of epochs	1
Correct identification rate	95%