Phoneme Generation with Elman-type Neural Networks

Marius Crisan
Department of Computers and Software Engineering, Polytechnic University of Timisoara, Timisoara, Romania
e-mail: marius.crisan@cs.upt.ro

***Abstract*—The paper continues some previous works and analyzes the possibility of phoneme generation with Elman-type neural networks. The vowels are composed of repetitions of the same elemental pattern with certain variations (assumed to be chaotic) of the signal parameters. The Elman-type network proved adequate to learn and generate independently in an open loop the elementals with signal parameters controlled by chaotic sources. The experimental results demonstrated that the technique can be further developed and used in synthesized speech.**

Keywords- Phoneme synthesis; Elman neural networks; Speech processing; Time series analysis; Dynamical systems

I. INTRODUCTION

Natural language processing along with its applications in synthetic speech generation and text-to-speech converters is still a challenging domain for artificial intelligence. Speech synthesis can be produced by varied techniques which can be mainly grouped in three classes: concatenative synthesis, formant/parametric synthesis, and articulatory synthesis [1-3]. These techniques have their own advantages and disadvantages, but in spite of the progress made in the field even the most refined speech synthesis systems are monotonous and cannot deal with the problem of emphasis. Even if it's not very clear what is missing from the synthesized sounds one possible clue is to take into account the dynamics of speech rather than the time/frequency domain representations. Recently, a new direction of research started towards developing models based on the nonlinear dynamics of speech. Different approaches have been suggested which are based on the identification of the nonlinear type of the vocal dynamics [4-6].

In aprevious work, we have started to investigate the possibility of synthesizing speech phonemes starting from a dynamic model based on chaotic control of the phonemes' elementals [7].A deeper investigation of the dynamic properties of phonemes revealed the fact that natural phonemes could be analyzed in terms of some elemental patterns. For instance, the vowels are composed of repetitions of the same elemental pattern with certain variations (assumed to be chaotic) of the signal parameters. It was assumed that the impression of naturalness in human voice is due to the chaotic nature of the speech phenomenon and is akin to a controlled chaos process. The suggested model for phoneme synthesis consisted of the generation of the elemental patterns by a harmonic series and the repetition of that pattern in time with the signal parameters controlled by a chaotic source. In another work, we investigated the possibility to generate the phoneme elementals through recurrent neural networks [8]. Neural networks may offer a good alternative for the dynamic approach of speech synthesis due to their inherent capability of capturing complex nonlinear relations in data. Neural networks were already used in different applications of speech synthesis, mostly in text-to-speech converters [9-10]. In these works, the role of the neural network was to map the phoneme symbols to the control parameters of the synthesizer, and not to generate directly the speech signal. The use of neural networks as generators was mostly studied for the case of time-series prediction. Different topologies have been proposed under the name of recurrent neural networks (RNNs) [11, 12]. The basic topology consisted of a feed-forward network in which feedback connections were added to previous layers. A good approach for time series prediction seems to be such a feed-forward network (perceptron), with or without one or more hidden layers, where the next input vector is determined by the previous output values in a process called sequence generation (moving the network over the time series) [13-15]. In general, the prediction performance is acceptable but only for one or a few instances in advance. In [8] we investigated the possibility of training a similar topology for the generation of three new periods of elemental pattern, which we considered to be a quite good long-run result. The phoneme sound was finally generated in a repetitive loop. The results were promising, and therefore the present work aimed to extend the research with other types of RNNs in order to obtain a better performance. Specifically, we have focused on a version of Elman network [16, 17]. The rest of the paper is organized as follows: The Elman-type network topology and training process are detailed in Section 2. Section 3 describes the implementation of the Elman-type network for phoneme generation. The simulation results are presented in Section 4 and the conclusions in the last section.

II. ELMAN-TYPE NETWORK TOPOLOGY AND TRAINING

The main idea in RNNs is to add feedback connections to previous layers of a basic feed-forward structure. Of course, it's possible to have a fully connected or recurrent network where each node has input from all other nodes including the node itself. In this case the network does not have a distinct input layer. In time series applications, for practical reasons, we are interested in partial RNNs, where there are some nodes that provide the sequential context and receive feedback from other nodes. The feedback connections create some supplementary nodes to the input layer and they are often called context units. The role of these nodes is to create a long term memory that provides to the input the knowledge about the previous inputs. Different topologies of RNNs can be configured in this way, depending on the layer from where the feedback connections are taken. Two very popular, and in a way similar structures, were proposed by Jordan and Elman [21, 16]. In this work, we used a version of Elman network with a single output which is depicted in Fig. 1. The input layer is comprised of N units. The context layer and the hidden layer have both M units. The neurons in the hidden layer feed back to the neurons in the context layer. In this way, outputs from each of the hidden layer units at time t become additional inputs to the network at time $t + 1$. Also, there is feedback from each context unit to itself. The dynamics of the network is described by the following set of equations:

$$X(t) = (x_1(t), \dots, x_N(t)). \tag{1}$$

$$net_j^h(t) = \sum_{i=1}^{N} w_{ji}^h x_i(t) + \sum_{c=1}^{M} w_{cj}^h y_c^h(t-\tau) + \mu net_j^h(t-\tau) + \theta_j^h. \tag{2}$$

$$y_j^h(t) = f_j^h(net_j^h(t)). \tag{3}$$

$$net^o(t) = \sum_{j=1}^{M} w_j^o y_j^h(t) + \theta^o. \tag{4}$$

$$y^o(t) = f^o(net^o(t)). \tag{5}$$

$X(t)$: input function which supplies N temporal values from the time-series;

$net_j^h(t)$: net-input function of the j-th unit in the hidden layer;

w_{ji}^h: weight on the connection to the j-th unit in the hidden layer from the i-th unit in the input layer;

w_{cj}^h: weight on the connection to the c-th unit in the context layer from the j-th unit in the hidden layer;

$y_c^h(t-\tau)$: output of the c-th unit in the hidden layer at time $t - \tau$, where τ is the time delay;

μ: parameter which determines the amount of influence of previous time steps at the inputs of context layer;

θ_j^h: bias value of the j-th unit in the hidden layer;

$y_j^h(t)$: output of the j-th unit in the hidden layer at time t;

f_j^h: activation function of the neurons in the hidden layer;

$net^o(t)$: net-input function of the output unit;

w_j^o: weight on the connection to the output unit from the j-th unit in the hidden layer;

θ^o: bias value of the output unit;

$y^o(t)$: output of the Elman network;

f^o: activation function of the output neuron;

The basic structure of Elman network is a standard feed-forward layered network, and therefore the training of the connection weights follows the standard generalized delta rule. The activation function for the hidden and output layers used in the experiments was the hyperbolic tangent function.

First, the training procedure requires a set of input-output pairs that are instances of a functional mapping $\boldsymbol{y} = F(\boldsymbol{x})$, where $\boldsymbol{x}$ and $\boldsymbol{y}$ are real vectors, $\boldsymbol{x} \in R^N$, $\boldsymbol{y} \in R$. The neural network is given a number of N previous samples of the time series and has to generate the next sample value in the series. For a time series of S samples, the following input-output pairs or exemplars are formed:

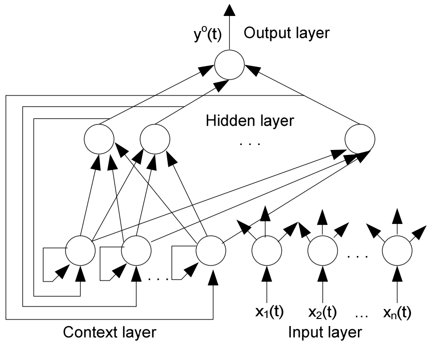


Figure 1. Structure of the Elman-type network.

$$IoPairs = \{[(x_1, x_2, \dots, x_N), (x_{N+1})], [(x_2, x_3, \dots, x_{N+1}), (x_{N+2})], \dots, [(x_i, x_{i+1}, \dots, x_{i+N-1}), (x_{i+N})], \dots, [(x_{S-N}, x_{S-N+1}, \dots, x_{S-1}), (x_S)]\}, i \in [1, S-N]. \tag{6}$$

The number of neurons in the input layer, N, matches the number of samples of previous data. The training algorithm tries to minimize for all the input-output pairs the mean square error

$$E_s = ½ \sum_{s=1}^{S-N} \delta_s^2, \tag{7}$$

where $\delta_s = (y_s^o - o_s)$, y_s^o is the generated output and o_s is the desired output for the s-th exemplar. The gradient descent is used down the global error surface $E = \Sigma E_s$. In order to derive the weight-update equations it is convenient to use the notations of the output layer delta, δ_s^o and the hidden layer delta, δ_{sj}^h :

$$\delta_s^o = \delta_s f^{o\prime}(net_s^o), \tag{8}$$

$$\delta_{sj}^h = f_j^{h\prime}(net_{sj}^h)\delta_s^o w_j^o. \tag{9}$$

Thus, the weight-update equations for the hidden and output layers are:

$$w_{ji}^h(t+1) = w_{ji}^h(t) + r\delta_{sj}^h x_i(t) + m\Delta w_{ji}^h(t), \tag{10}$$

$$w_j^o(t+1) = w_j^o(t) + r\delta_s^o y_j^h(t) + m\Delta w^o(t), \tag{11}$$

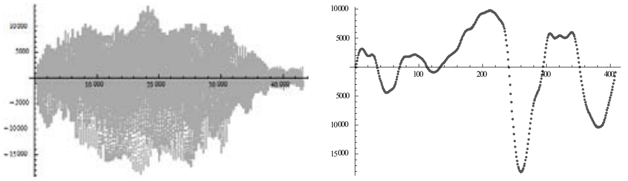
where$r$ is the learning rate parameter (usually, r<< 1), m is the momentum term (typically, 0 <m< 1), and

$$\Delta w_{ji}^h(t) = w_{ji}^h(t) - w_{ji}^h(t-1),$$

$$\Delta w^o(t) = w^o(t) - w^o(t-1).$$

III. Phoneme Generation Set-up

In this work we have focused upon vowels considering that the impression of artificial or mechanical sounds created by synthesizers is mostly due to their difficulty in reproducing sustained vowels. The proposed approach continues the main idea introduced in some previous works [7, 8], with the purpose to improve the results of elementals generation. Every sound of a phoneme is composed of a series of repetitive patterns that show slight variations of the signal parameters. However, among all the variations, a typical pattern or elemental can be identified. For illustration purpose, the signal of the vocal phonemes, /a/pronounced by a female person, along with an instance of the corresponding elemental pattern are shown in Fig. 2. The vocal sound data were sampled at 96 kHz with 16 bits. The elemental pattern may be considered like a seed in constructing the phoneme signal. If this pattern is repeated in time (concatenated) the phoneme sound can be reconstructed. An important observation should be made here. Even if in this process of concatenation the elemental sample was taken from the original phoneme signal, the resulted sound was not as natural as the original. Similar results were obtained with whatever elemental was chosen. Therefore it seems that the slight variations of the elemental shapes as they chain in time one after another may be the cause that produces the sensation of a natural sound.



/a/

Figure 2. The signal of vowel /a/ and a corresponding typical elemental pattern.

The present approach consisted of three phases: (i) we trained first an Elman-type network to learn as accurate as possible the shape of a typical elemental; (ii) the elementals are generated by the trained neural network using its own previous generated data as inputs in an open loop; (iii) the phoneme was synthesized using a series of elementals with modified parameters controlled by an external source.

In order to tune the parameters of the Elman-type network for obtaining the best performances, some information regarding the spectrum analysis were used [7]. We were interested to estimate the magnitude and phase of different frequency components. The spectrum analysis was performed based on discrete Fourier transform (DFT):

$$F_s = 1/n^{(1-a)/2} \sum_{k=1}^{n} x_k e^{2\pi i b(k-1)(s-1)/n}, \qquad (12)$$

where$a$ and b are parameters. From the sequence of complex numbers, of the same length as the experimental time-series, the power spectrum was obtained, and accordingly the periods, measured in number of samples, of the main frequencies could be computed (see Table I).

The maximum mode in the spectrum and the period were considered when the number of neurons in the input layer and in the hidden layer was selected.The topology of the Elman network with feedback connections from each context unit to itself has the advantage of providing supplementary control of the context for the next output vector in the sequence. In this regard, the parameter μ in (2) was used in the generating phase to provide a slight modification in the network output elementals, simulating the variations of elemental as in the original time series. For each output the value of the parameter μ was controlled by a chaotic source given by the quadratic map:

TABLE I. PHONEMES SIGNAL ANALYSIS

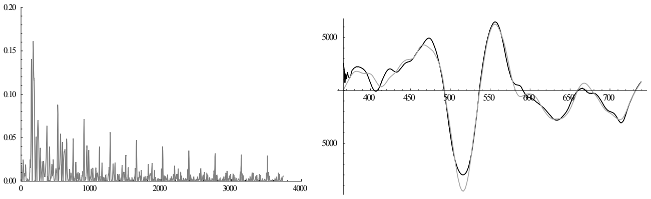
Vocals elementals	**Signal Analysis**			
	Sample length	***Optimal embedding dimension***	***Spectrum max. mode***	***Period***
/a/	365	18	297	117.8
/e/	399	20	149	229.1
/i/	360	9	141	462.8
/o/	371	8	182	124.6
/u/	365	18	156	429.7

$$x_{t+1} = a_1 + a_2 x_t + a_3 x_t^2. \qquad (13)$$

Depending on the values of a_1, a_2, and a_3, the initial conditions may be drawn to a chaotic attractor. During the learning phase the value of μ was kept constant.

IV. EXPERIMENTAL RESULTS

The network was trained to learn the elemental patterns for each phoneme. The training data was formed out of a series of three samples of elemental data according to (6). The best results were obtained when the number of neurons in the input and hidden layer were adjusted for each phoneme in part. Detailed results are presented here for the phoneme /a/. The number of neurons in the input layer was chosen that it may cover at least the period according to Table I. The maximum value tested was for the number of samples of an elemental. The number of neurons in the hidden layer (and consequently in the context layer) determines the amount of influence of previous steps. If this number approaches the number of input layer an over fitting phenomenon was manifested. However, this effect is reduced when the input layer and hidden layer have the same number of neurons, around the value of the period. A series of tests were performed for the following values: input layer, 365 neurons; hidden (context) layer, 220 neurons; r = 0.001; m = 0.05; μ = 0.01. Convergence can be noticed after several epochs as shown in Fig. 3a. In Fig. 3b the capability of the network to learn the original elemental fluctuations can be observed.



(a) (b)

Figure 3. (a) The mean squared error. (b) The generated elemental pattern (black) along with the input data (gray) for elemental /a/.

The network is more dynamic than the previous tested topology in [8]. The negative spike which was difficult to follow in the previous approach can be traced much easier with the present configuration. The network started to generate the future signal using at each step its own output which is shifted one unit to the right in the input vector. In this way, the network was completely independent from the training data and predicted the continuation of the input signal with 365 new samples.

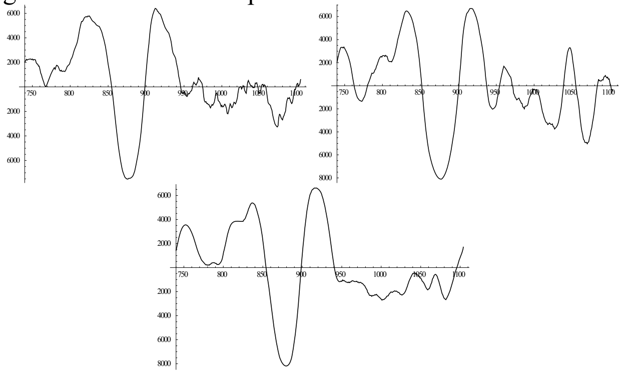


Figure 4. Three elemental samples generated when μ was controlled by a chaotic source.

The final phase of simulation involved the generation of elementals with slight variations of the signal shape. A source of chaos was used in order to alter the value of μ in the iterated process. These chaotic changes at the input of the context layer had a visible influence in the shape of the generated signal. In Fig. 4 three different elementals are depicted. Slight variations of the signal shape can be observed which look similar with the variations of the elementals in the original waveform. The dynamics of the original elemental is well preserved, even more accurate than in the training phase when the negative spike was not fully covered. In a recursive process the entire phoneme was generated as shown in Fig. 5.

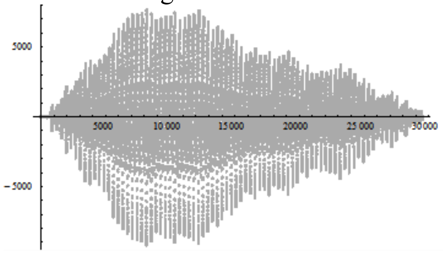


Figure 5. The phoneme /a/ generated in a recursive process.

The results proved the possibility of phoneme generation by Elman-type networks and also the capability of inserting chaos and modifying the signal form in the process of generation.

V. Conclusions

In conclusion, the purpose was to study the possibility of phoneme synthesis using an Elman-type neural network. The proposed model for phoneme synthesis consists of generating the elemental patterns of the phonemes. The patterns are repeatedly generated in time with the signal parameters controlled by a chaotic source applied to the context layer. The slight variations of the signal shape resemble the natural phonemes waveforms. The obtained final results gave confidence that the researches can be continued in this direction.

References

[1] T. Styger and E. Keller, Formant synthesis.In E. Keller (ed.), Fundamentals of Speech Synthesis and Speech Recognition: Basic Concepts, State of the Art, and Future Challenges (pp. 109–128).Chichester: John Wiley, 1994.

[2] J. Holmes and W. Holmes, Speech Synthesis and Recognition, 2nd Edition, Taylor & Francis, N.Y. 2001.

[3] D. Jurafsky and J.H. Martin,Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, Second Edition, Pearson Prentice Hall, 2008.

[4] M. Banbrook, S. McLaughlin, and I. Mann, "Speech characterization and syntheis by nonlinear methods,"*IEEE Trans. Speech Audio Process.*, vol. 7, no. 1, pp. 1–17, Jan. 1999.

[5] S. McLaughlin andP. Maragos,"Nonlinear methods for speech analysis and synthesis." In: Marshall S, Sicuranza G, editor. Advances in nonlinear signal and image processing, Vol. 6. Hindawi Publishing Corporation 2007, p.103.

[6] V. Pitsikalis and P. Maragos, "Analysis and classification of speech signals by generalized fractal dimension features," Speech Communication, Vol.51, no.12, 2009, pp. 1206–1223.

[7] M. Crisan, "Upon Phoneme Synthesis Based on Chaotic Modeling," Proc.The 5th International Conference on New Trends in Information Science and Service Science, Oct. 2011, Macao, pp. 134–139.

[8] M. Crisan, "A Neural Network Model for Phoneme Generation," in Applied Mechanics and Materials, vol. 367, 2013, pp. 478-483.

[9] M. Malcangi and D. Frontini: A Language-Independent Neural Network-Based Speech Synthesizer. In: *Neurocomputing*, 73:1–3 2009 Dec, pp. 87-96.

[10] E.V. Raghavendra, P. Vijayaditya and K. Prahallad,"Speech synthesis using artificial neural networks," National Conference on Communications (NCC), Chennai, India, 2010, pp. 1–5.

[11] S.D. Balkin,"Using Recurrent Neural Networks for Time Series Forecasting," Technical Report 97–11, Pennsylvania State University, 1997.

[12] L. R. Medsker and L. C. Jain, "Recurrent Neural Networks: Design and Applications," CRC Press, 2001.

[13] A. Priel and I. Kanter: Time series generation by recurrent neural networks, In: Annals of Mathematics and Artificial Intelligence 39 2003, pp. 315–332.

[14] W. Kinzel: Predicting and generating time series by neural networks: An investigation using statistical physics. In: Computational Statistical Physics 2002, pp. 97–111.

[15] R.J.Frank, N.Davey and S.P.Hunt: Time Series Prediction and Neural Networks. In: Journal of Intelligent and Robotic Systems 31, (2001), pp. 91-103.

[16] J. L. Elman, "Finding structure in time," Cognitive Science, Vol. 14, 1990, pp. 179-211.

[17] J.A. Freeman, "Simulating Neural Networks with Mathematica," Addison-Wesley, 1994.

[18] A. Kalinli and S. Sagiroglu, "Elman Network with Embedded Memory for System Identification," Journal of Information Science and Engineering 22, 2006, pp. 1555–1568.

[19] X.Z. Gao, "A modified Elman neural network model with application to dynamical systems identification," IEEE International Conference on Systems, Man, and Cybernetics, Vol. 2, 1996, pp. 1376 –1381.

[20] P. Stagge and B. Sendhoff, "An extended Elman net for modeling time series," Artificial Neural Networks — ICANN'97, Lecture Notes in Computer Science Volume 1327, 1997, pp 427–432.