

TEHNICI PREDICTIVE ÎN PROCESAREA SEMNALULUI VORBIT

MARINA CIDOTA

Universitatea București, Facultatea de Matematică și Informatică,
email cmarina@k.ro

Rezumat: Semnalul vorbit este nestaționar și din acest motiv nu poate fi modelat cu ajutorul modelelor lineare parametrice clasice (*AR*, *MA*, *ARMA*). Aplicarea rețelelor neurale recurente în domeniul predicției seriilor dinamice este potrivită pentru învățarea și recunoașterea structurii neliniare a semnalului vorbit. Prezentăm trei tipuri de algoritmi de antrenare a rețelelor recurente pentru predicție și rezultatele testelor efectuate pe două clase de semnale, rostite atât de femei cât și de bărbați în diferite dialecte ale limbii engleze.

Cuvinte cheie: tehnici predictive în procesarea semnalului vorbit, rețele recurente neurale

1. INTRODUCERE

Tradițional, predicția seriilor dinamice s-a realizat cu ajutorul modelelor *AutoRegressive*, *Moving Average* sau *ARMA* ai căror parametri sunt estimați fie în bloc fie secvențial cu algoritmul *Least Mean Square*. Problema este că aceste modele sunt liniare și nu pot procesa semnale neliniare sau nestaționare. Rețelele neurale sunt un instrument puternic de rezolvare a problemelor ale căror soluții necesită cunoștințe greu de specificat, dar pentru care există foarte multe exemple. Deoarece prin predicția seriilor dinamice înțelegem deducerea comportamentului viitor pe baza exemplului comportamentului trecut, înseamnă că aceasta este o aplicație potrivită pentru o rețea neurală. Putem privi sistemele adaptive de predicție ca fiind nucleul procesării semnalelor digitale pentru că de fapt orice filtru adaptiv încearcă să prezică răspunsul dorit. Un astfel de sistem constă din:

- o mulțime de ponderi ajustabile
- un bloc de calcul al erorii (diferența dintre răspunsul dorit și răspunsul sistemului)
- un algoritm de control (învățare) pentru ajustarea ponderilor

și poate fi reprezentat în figura de mai jos:

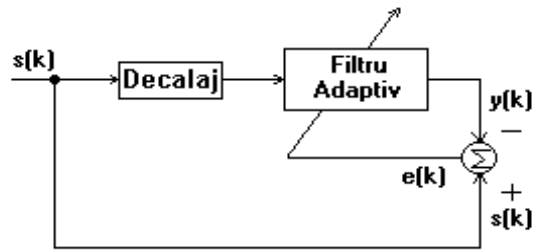


Fig.1 Configurația unui sistem de predicție

Sunt două clase de filtre adaptive:

- cele fără *feedback*, pentru care ieșirea depinde de intrarea curentă și de intrările de la p momente din trecut (*Finite Impulse Response* sau structuri *MA*)
- cele cu *feedback* pentru care ieșirea depinde atât de intrările cât și de ieșirile trecute (*Infinite Impulse Response* sau structuri *AR* sau *ARMA*)

Forma generală a unui filtru *IIR* neliniar este dată de ecuația

$$y(k) = \Theta \left(\sum_{i=1}^p a_i y(k-i) + \sum_{j=1}^q b_j e(k-j) \right) + e(k), \quad (1)$$

unde $y(k)$ reprezintă ieșirea, $e(k)$ intrarea iar $\Theta(\cdot)$ este o funcție diferențiabilă neliniară. Vom numi acest model *NARMA(p,q)* (*Nonlinear ARMA*) iar predictorul corespunzător este dat de

$$\hat{y}(k) = \Theta \left(\sum_{i=1}^p a_i y(k-i) + \sum_{j=1}^q b_j \hat{e}(k-j) \right), \quad (2)$$

rezidurile fiind $\hat{e}(k-j) = y(k-j) - \hat{y}(k-j)$, $j = 1, 2, \dots, q$.

Vom prezenta în continuare câteva implementări neurale ale unor filtre *IIR*.

2. ARHITECTURI NEURALE RECURENTE PENTRU PREDICȚIE

O rețea cu o bogată reprezentare a ieșirilor trecute este o rețea recurentă complet conectată, cunoscută sub numele de rețea *Williams-Zipser* (vezi Fig. 2). Constă din trei nivele: de intrare, de procesare și de ieșire. Pentru fiecare neuron i , $i=1, 2, \dots, N$, elementele u_j , $j=1, 2, \dots, p+1+N$ ale vectorului de intrare \mathbf{u} sunt ponderate și însumate pentru a produce activarea internă a fiecărui neuron, care apoi este trecută prin funcția de activare neliniară Φ pentru a obține ieșirea neuronului i , y_i . Vom considera în continuare funcția neliniară Φ ca fiind funcția logistică

$$\Phi(v) = \frac{1}{1 + e^{-\beta v}}, \beta > 0, \quad (3)$$

care este crescătoare și ia valori în intervalul (0,1).
 Ecuațiile următoare descriu rețeaua din figura 2

$$y_i(k) = \Phi(v_i(k)), \quad i = 1, 2, \dots, N,$$

$$v_i(k) = \sum_{l=1}^{p+1+N} w_{i,l}(k) u_l(k), \quad (4)$$

$$u_i^T(k) = [s(k-1), \dots, s(k-p), 1, y_1(k-1), y_2(k-1), \dots, y_N(k-1)].$$

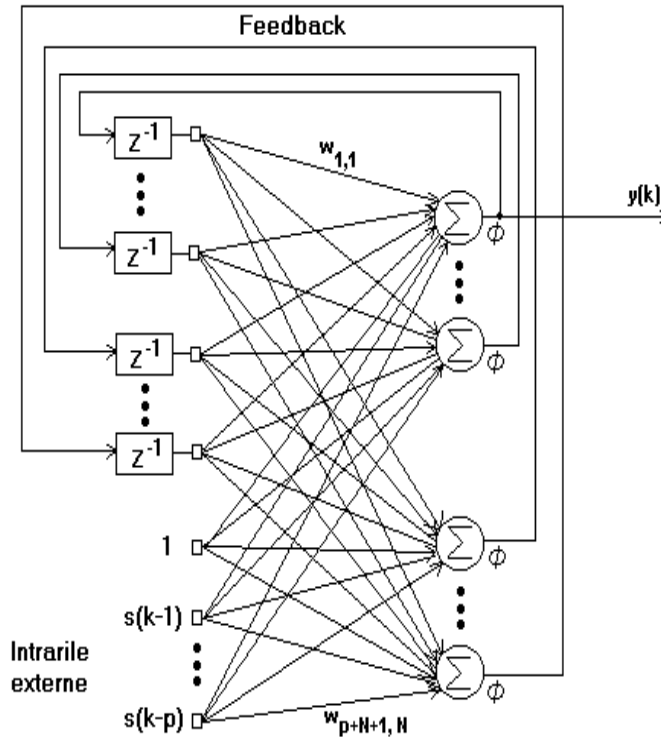


Fig.2 Rețea recurentă Williams-Zipser

Un proces general NARMA(p,q) poate fi exprimat prin

$$\hat{s}(k) = \Phi \left(\sum_{i=1}^p w_{1,i}(k) s(k-i) + w_{1,p+1}(k) + \sum_{j=p+2}^{p+q+1} w_{1,j}(k) \hat{e}(k+j-2-p-q) + \sum_{l=p+2+q}^{p+q+N} w_{1,l}(k) y_{l-p-q}(k-1) \right), \quad (5)$$

și poate fi implementat printr-o rețea recurentă *Williams-Zipser* pentru care intrarea este

$$s(k-1), \dots, s(k-p), 1, \hat{e}(k-1), \dots, \hat{e}(k-q), y_2(k-1), \dots, y_N(k-1). \quad (6)$$

Intrările rețelei conțin termenii erorii de predicție $\hat{e}(k-1), \dots, \hat{e}(k-q)$, ceea ce face ca învățarea să fie un proces dificil. Algoritmul de învățare recurentă în timp real (*Real Time Recurrent Learning*) a fost conceput pentru a minimiza eroarea pătratică instantanee de predicție $e(k)$ și din acest motiv nu poate fi aplicat direct acestei rețele deoarece intrarea conține termeni ai erorii de predicție. Din acest motiv trebuie găsită o altă reprezentare echivalentă a rețelei *NARMA(p,q)* astfel încât să putem aplica algoritmul de învățare *RTRL*. Dacă înlocuim $\hat{e}(k)$ cu $s(k) - y_1(k)$ și regroupăm termenii obținem

$$\hat{s}(k) = \Phi \left(\sum_{i=1}^p w_{1,i}(k) s(k-i) + w_{1,p+1}(k) + \sum_{j=p+2}^{p+q+1} w_{1,j}(k) y_1(k+j-2-p-q) + \sum_{l=p+2+q}^{p+q+N} w_{1,l}(k) y_{l-p-q}(k-1) \right), \quad (7)$$

și sub această formă poate fi implementat cu ajutorul rețelei *Williams-Zipser*. Cea mai simplă arhitectură neurală *NARMA* este perceptronul recurent prezentat în figura 3. Ieșirea perceptronului este dată de

$$y(k) = \Phi(\text{net}(k)),$$

$$\text{net}(k) = \sum_{j=1}^M w_j(k) s(k-j) + w_{M+1}(k) + \sum_{m=1}^N w_{m+M+1}(k) y(k-m). \quad (8)$$

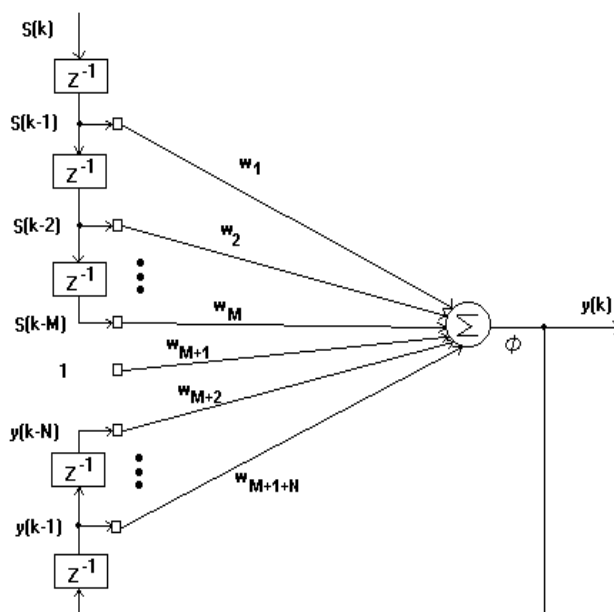


Fig.3 Percepton recurrent NARMA

3. ALGORITMI DE ÎNVĂȚARE PENTRU REȚELE NEURALE RECURENTE (RNN)

3.1 Algoritmul real time recurrent learning (RTRL)

Vom prezenta algoritmul *RTRL* de instruire a rețelei din figura 2, algoritmul pentru instruirea rețelei din figura 3 fiind un caz simplificat al acestuia. Învățarea *RTRL* are la bază ideea minimizării erorii pătratice instantanee, adică

$$\min(E(k)) = \min\left(\frac{1}{2}e^2(k)\right), \quad (9)$$

unde $e(k) = s(k) - y_1(k)$. Pentru a actualiza matricea ponderilor w , avem nevoie de

$$\Delta w_{n,l}(k) = -\frac{\eta}{2} \frac{\partial}{\partial w_{n,l}(k)} e^2(k) = -\eta e(k) \frac{\partial e(k)}{\partial w_{n,l}(k)}. \quad (10)$$

Deoarece intrarea $s(k)$ nu depinde de elementele matricei w , gradientul erorii devine

$$\frac{\partial e(k)}{\partial w_{n,l}(k)} = -\frac{\partial y_1(k)}{\partial w_{n,l}(k)}. \quad (11)$$

Dar $y_1(k) = \Phi(v_1(k))$, rezultă că

$$\begin{aligned} \frac{\partial y_1(k)}{\partial w_{n,l}(k)} &= \Phi'(v_1(k)) \frac{\partial v_1(k)}{\partial w_{n,l}(k)} \\ &= \Phi'(v_1(k)) \left(\sum_{\alpha=1}^N \frac{\partial y_\alpha(k-1)}{\partial w_{n,l}(k)} w_{1,\alpha+p+1}(k) + \delta_{n,l} u_l(k) \right). \end{aligned} \quad (12)$$

Dacă rata de învățare η este suficient de mică, atunci putem presupune că

$$\frac{\partial y_\alpha(k-1)}{\partial w_{n,l}(k)} \approx \frac{\partial y_\alpha(k-1)}{\partial w_{n,l}(k-1)}. \quad (13)$$

Făcând notațiile

$$\pi_{n,l}^j = \frac{\partial y_j(k)}{\partial w_{n,l}(k)} \quad 1 \leq j, n \leq N, 1 \leq l \leq p+1+N, \quad (14)$$

obținem următoare formulă recursivă

$$\pi_{n,l}^j(k+1) = \Phi'(v_j(k)) \left(\sum_{m=1}^N w_{j,m+p+1}(k) \pi_{n,l}^m(k) + \delta_{n,j} u_l(k) \right) \quad (15)$$

cu condițiile inițiale $\pi_{n,l}^j(0) = 0$, $1 \leq j, n \leq N, 1 \leq l \leq p+1+N$.

Deci ponderile se reactualizează după formula

$$w_{n,l}(k+1) = w_{n,l}(k) + \Delta w_{n,l}(k) \quad 1 \leq n \leq N, 1 \leq l \leq p+1+N. \quad (16)$$

3.2 Algoritmul extended recursive least squares (ERLS)

Algoritmul *ERLS* are la bază ideea filtrului *Kalman* extins. Filtrul *Kalman* constă dintr-un set de ecuații matematice care modelează un mecanism recursiv pentru estimarea stării unui proces prin minimizarea erorii medii pătratice. Filtrul este foarte puternic în câteva privințe: poate estima stările trecută, prezentă sau

viitoare chiar când natura exactă a sistemului modelat este necunoscută. Filtrul *Kalman* tratează problema generală a estimării stării unui proces discret guvernat de o ecuație diferențială liniară. Dar semnalul vorbirii nu este un proces liniar și pentru estimarea lui vom considera filtrul *Kalman* extins. Vom considera următoarele ecuații ale filtrului:

$$\begin{aligned} w(k) &= w(k-1) + q(k), \\ s(k) &= h(w(k)) + r(k), \end{aligned} \quad (17)$$

unde $w(k)$ este vectorul ponderilor de dimensiune $N^*(p+1+N)$, $s(k)$ este observația (semnalul) la momentul k , $q(k)$ este zgomot alb *Gaussian*, $q \sim N(0, Q)$, $r(k)$ reprezintă zgomotul (alb *Gaussian*) observației, $r \sim N(0, C)$ iar h este o funcție neliniară diferențiabilă.

Vom liniariza funcția neliniară $h(\cdot)$ folosind dezvoltarea *Taylor* în jurul estimației lui $w(k)$ bazată pe datele anterioare, notată cu $\hat{w}(k | k-1)$:

$$h(w(k)) \approx h(\hat{w}(k | k-1)) + \nabla h^T(w(k) - \hat{w}(k | k-1)), \quad (18)$$

unde prin ∇h^T înțelegem

$$\nabla h^T = \frac{\partial h(\hat{w}(k | k-1))}{\partial \hat{w}(k | k-1)} = H(k). \quad (19)$$

Ecuația observației va avea următoarea formă:

$$s(k) = H(k)w(k) + r(k) + [h(\hat{w}(k | k-1)) - H(k)\hat{w}(k | k-1)]. \quad (20)$$

Cu notațiile de mai sus, ecuațiile finale de reactualizare a vectorului ponderilor w pentru algoritmul *ERLS* devin:

$$\begin{aligned} K(k) &= Q(k-1)H^T(k)[C + H(k)Q(k-1)H^T(k)]^{-1}, \\ \hat{w}(k) &= \hat{w}(k-1) + K(k)[x(k) - h(\hat{w}(k-1))], \\ M(k) &= (I - K(k)H(k))Q(k-1). \end{aligned} \quad (21)$$

În cazul rețelei din figura 2 avem $h(w(k)) = y_I(k)$.

Pentru calcularea gradientului vom presupune că $\hat{w}(k | k-1) \approx \hat{w}(k-1 | k-1) = \hat{w}(k-1)$ și în aceste condiții

$H(k) = \frac{\partial h(\hat{w}(k-1))}{\partial \hat{w}(k-1)}$. Elementele matricei gradient se calculează recurent astfel:

$$\frac{\partial y_j(k)}{\partial w_{n,l}(k)} \approx \Phi'(v_j(k)) \left[\sum_{\alpha=1}^N \frac{\partial y_\alpha(k-1)}{\partial w_{n,l}(k-1)} w_{j,\alpha+p+1}(k) + \delta_{n,j} u_l(k) \right]. \quad (22)$$

Algoritmul *ERLS* prezentat diferă de algoritmul *RTRL* doar prin formula de reactualizare a ponderilor, structura rețelei neurale fiind aceeași.

3.3 Algoritmi de învățare a posteriori

Pentru a obține un predictor neliniar *a posteriori*, să considerăm pentru simplitate perceptronul recurent *NARMA* a cărui ieșire este:

$$y(k) = \Phi(u^T(k)w(k)), \quad (23)$$

unde $u^T(k) = [s(k-1), \dots, s(k-M), 1, y(k-1), y(k-2), \dots, y(k-N)]$. Deoarece vectorul ponderilor actualizat $w(k+1)$ este disponibil înainte de introducerea vectorului de intrare $u(k+1)$, putem calcula ieșirea *a posteriori*:

$$\bar{y}(k) = \Phi(u^T(k)w(k+1)). \quad (24)$$

Erorile corespunzătoare la ieșire sunt:

$$\begin{aligned} e(k) &= s(k) - y(k) && \text{eroarea a priori,} \\ \bar{e}(k) &= s(k) - \bar{y}(k) && \text{eroarea a posteriori,} \end{aligned} \quad (25)$$

unde $s(k)$ este semnalul dorit. Ieșirea *a posteriori* poate fi folosită pentru a obține vectorul de intrare *a posteriori*

$$\bar{u}^T(k) = [s(k-1), \dots, s(k-M), 1, \bar{y}(k-1), \bar{y}(k-2), \dots, \bar{y}(k-N)] \quad (26)$$

care poate înlocui vectorul de intrare *a priori* pentru calcularea ieșirii și actualizarea ponderilor.

Vom prezenta o variantă a metodei coborârii pe gradient astfel încât eroarea *a posteriori* să fie mai mică decât eroarea *a priori*:

$$\begin{aligned}
w(k+1) &= w(k) - \eta \nabla_w E(k), \\
e(k) &= s(k) - \Phi(u^T(k)w(k)), \\
\bar{e}(k) &= s(k) - \Phi(u^T(k)w(k+1)), \\
\text{cu condiția ca } |\bar{e}(k)| &\leq \gamma |e(k)|, \quad 0 < \gamma < 1.
\end{aligned} \tag{27}$$

iar $E(k) = \frac{1}{2} e^2(k)$.

În cazul *NARMA*, ecuația de reactualizare a ponderilor este:

$$w(k+1) = w(k) + \eta(k)e(k)\pi(k), \tag{28}$$

unde $\pi^T(k) = [\pi_1(k), \pi_2(k), \dots, \pi_{M+N+1}(k)]$ iar $\pi_i(k) = \frac{\partial y(k)}{\partial w_i(k)}$.

Putem scrie eroarea *a posteriori* sub forma:

$$\bar{e}(k) = s(k) - \Phi(u^T(k)w(k)) - [\Phi(u^T(k)w(k+1)) - \Phi(u^T(k)w(k))]. \tag{29}$$

Dacă pentru funcția logistică $\Phi(x) = \frac{1}{1 + e^{-\beta x}}$ luăm $0 < \beta < 4$ atunci $\Phi'(x) \leq 1 \quad \forall x \in \mathfrak{R}$ și, revenind la expresia erorii *a posteriori*, avem conform teoremei *Lagrange*:

$$\Phi(u^T(k)w(k+1)) - \Phi(u^T(k)w(k)) = \alpha(k)u^T(k)\Delta w(k), \tag{30}$$

unde $\alpha(k) = \Phi'(\xi) < 1$, $\xi \in (u^T(k)w(k), u^T(k)w(k+1))$.

Rezultă că:

$$\begin{aligned}
\bar{e}(k) &= e(k) - \alpha(k)u^T(k)\Delta w(k) \\
&= e(k) - \alpha(k)\eta(k)u^T(k)\pi(k)e(k) \\
&= (1 - \alpha(k)\eta(k)u^T(k)\pi(k))e(k).
\end{aligned} \tag{31}$$

iar rata de învățare $\eta(k) = \frac{1}{\alpha(k)u^T(k)\pi(k)}$ minimizează expresia de mai sus.

Problema este că nu cunoaștem $\alpha(k)$ și din acest motiv vom încerca să găsim o limită inferioară a erorii $\bar{e}(k)$. Plecând de la egalitatea

$$\Phi(u^T(k)w(k+1)) = \Phi(u^T(k)w(k) + \eta(k)e(k)u^T(k)\pi(k)) \quad (32)$$

și folosind proprietatea funcției logistice $\Phi(a+b) \leq \Phi(a) + \Phi(b) \forall a, b \in \mathfrak{R}$ deducem că

$$\bar{e}(k) \geq e(k) - \Phi(\eta(k)e(k)u^T(k)\pi(k)). \quad (33)$$

Dacă $0 < \beta < 4$ avem $|\Phi(x)| < |x|, \forall x \in \mathfrak{R}$ și obținem în final

$$\bar{e}(k) \geq (1 - \eta(k)|u^T(k)\pi(k)|)e(k), \quad (34)$$

iar domeniul permis pentru rata de învățare este:

$$0 < \eta(k) < \frac{1}{|u^T(k)\pi(k)|}. \quad (35)$$

4. APLICAȚII

Pentru testarea algoritmilor prezentați am folosit baza de date *TIMIT* ce conține semnale rostite de 630 de vorbitori în 8 dialecte ale limbii engleze, fiecare înregistrând câte 10 propoziții. Am considerat două clase de semnale, fiecare reprezentând câte o propoziție rostită atât de femei cât și de bărbați. Prima propoziție "*Those who teach values first abolish cheating*" este rostită de trei femei și patru bărbați iar a doua propoziție "*The eastern coast is a place for pure pleasure and excitement*" este rostită de o femeie și șase bărbați. Nu toate aceste persoane vorbesc același dialect. Fișierele le-am obținut în formatul *WAV* și le-am prelucrat cu ajutorul soft-ului *HTK (Hidden Markov ToolKit)* specializat în procesarea semnalului vorbit. Fișierele al căror nume începe cu "*f*" reprezintă semnale rostite de femei, celelalte (care încep cu "*m*") sunt rostite de bărbați. Ca fișier de antrenare am folosit "*abc0.txt*" pentru toți algoritmi (vezi fig. 4). Deoarece funcția logistică ia valori în intervalul (0,1), am ajustat amplitudinea semnalelor astfel încât să fie în același interval (0,1).

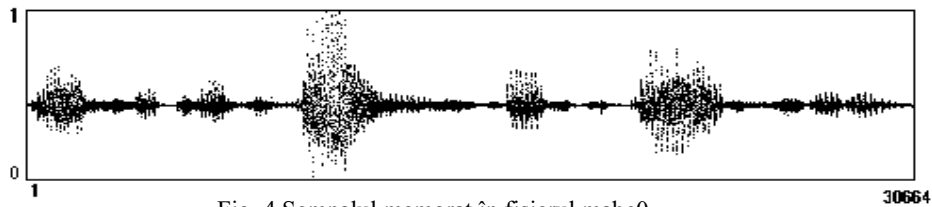


Fig. 4 Semnalul memorat în fișierul mabc0

Pentru a măsura calitatea predictorilor am folosit “câștigul predicției” definit astfel:

$$R_p = 10 \log_{10} \left(\frac{\hat{\sigma}_s^2}{\hat{\sigma}_e^2} \right) \text{dB}, \quad (36)$$

unde $\{\hat{\sigma}_s^2\}$ reprezintă varianța estimată a semnalului $\{s(k)\}$ în timp ce $\{\hat{\sigma}_e^2\}$ este varianța estimată a erorii de predicție $\{e(k)\}$.

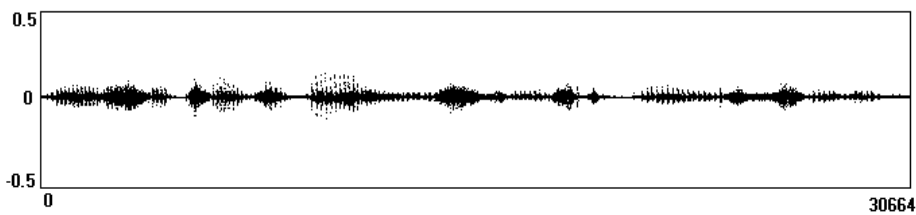


Fig. 5 Eroarea rezultată din testarea fișierului mabc0

În tabelele de mai jos am afișat indicele R_p rezultat din testarea algoritmilor *RTRL*, *NARMA*, *ERLS* și *NARMA a posteriori* antrenați cu parametrii specificați în tabele.

| | RTRL p=12 N=3 β=2 | NARMA M=12 N=3 β=2 | ERLS p=10 N=2 β=2 | NARMA posteriori p=12 N=2 β=2 | a |
|-------|----------------------|--------------------------|----------------------|-------------------------------------|---|
| fjlr0 | 17.46 | 17.15 | 18.25 | 17.04 | |
| fkde0 | 20.59 | 20.33 | 21.96 | 20.26 | |
| fleh0 | 21.62 | 21.25 | 22.28 | 21.23 | |
| mabc0 | 29.85 | 29.88 | 28.63 | 29.88 | |
| mgag0 | 25.96 | 26.07 | 25.61 | 26.09 | |
| mpmb0 | 24.80 | 24.93 | 25.26 | 25.04 | |
| mrem0 | 28.80 | 28.70 | 27.19 | 28.63 | |

Tabelul 1 Indicele R_p obținut prin testarea semnalelor din prima clasă (“Those who teach values first abolish cheating”)

| | RTRL p=12 N=3 β=2 | NARMA M=12 N=3 | ERLS p=10 N=2 β=2 | NARMA posteriori | a |
|--|----------------------|-------------------|----------------------|---------------------|---|
|--|----------------------|-------------------|----------------------|---------------------|---|

| | | $\beta=2$ | | $p=12$ $N=2$ $\beta=2$ |
|-------|-------|-----------|-------|------------------------|
| fsmm0 | 15.89 | 15.69 | 18.00 | 15.66 |
| mbma1 | 20.48 | 20.60 | 20.81 | 20.60 |
| mfxs0 | 25.10 | 25.24 | 24.90 | 25.25 |
| mmab1 | 27.66 | 27.49 | 27.82 | 27.50 |
| mmp0 | 24.50 | 24.73 | 24.92 | 24.82 |
| mrab0 | 16.97 | 16.87 | 18.82 | 16.84 |
| mtlb0 | 18.74 | 18.65 | 18.92 | 18.55 |

Tabelul 2 Indicele R_p obținut prin testarea semnalelor din a doua clasă (“*The eastern coast is a place for pure pleasure and excitement*”)

BIBLIOGRAFIE

1. Mandic, D.P., Chambers, J.A., *Recurrent Neural Networks for Prediction –Learning Algorithms, Architectures and Stability*, John Wiley & Sons, 2001
2. Welch, G., Bishop, G., *An Introduction to the Kalman Filter*, (internet)
3. Catlin, D., *Estimation, Control and the Discrete Kalman Filter*, Springer Verlag, 1989
4. Young, S.,Evermann, G., *The HTK Book* (internet)

Abstract: Speech signal has statistically non-stationary properties and cannot be processed properly by the use of classical linear parametric models (*AR*, *MA*, *ARMA*). The neural network approach to time series prediction is suitable for learning and recognizing the nonlinear nature of the speech signal. We present three types of learning algorithms for recurrent neural networks for prediction and test them on two classes of speech signal spoken by both men and women in different dialects of English language.