Aprendizagem Competitiva com Consciência Aplicada ao Projeto de Dicionários para Quantização Vetorial de Voz

F. Madeiro, W. T. A. Lopes, M. S. Alencar e B. G. Aguiar Neto

***Resumo*—Este trabalho apresenta a avaliação de desempenho de um algoritmo de aprendizagem competitiva com consciência aplicado ao projeto de dicionários para quantização vetorial de forma de onda de voz. São apresentados resultados referentes à distribuição dos vetores de treino nas células de Voronoi, ressaltando a importância da introdução de consciência na aprendizagem. Resultados de simulação mostram que o algoritmo produz dicionários com qualidade próxima ou um pouco superior à obtida com o algoritmo LBG (Linde-Buzo-Gray), requerendo para isso um número menor de iterações do conjunto de treino.**

***Palavras-chave*—Aprendizagem competitiva, quantização vetorial, projeto de dicionário, codificação de voz.**

I. Introdução

A quantização vetorial (QV) [1,2] tem sido utilizada em diversos sistemas de compressão de sinais. O sinal a ser quantizado é dividido em blocos (que podem ser amostras consecutivas de um sinal de voz, parâmetros da codificação preditiva linear (LPC), blocos de *pixels* de uma imagem, etc.) e cada bloco é representado por um vetor. Um quantizador vetorial mapeia cada um dos vetores obtidos do sinal (voz, imagem, etc.) em um conjunto finito de vetores representativos $\boldsymbol{w}_i$, denominados vetores-código ou vetores de reconstrução. O conjunto $W = \{\boldsymbol{w}_i;\ i = 1,\ 2,\ \ldots,\ N\}$ de vetores-código é denominado dicionário, o número (N) de vetores-código do dicionário é denominado número de níveis ou tamanho do dicionário, e o número de componentes (denotado ao longo deste trabalho por K) de cada vetor $\boldsymbol{w}_i$ é denominado dimensão do quantizador vetorial.

De acordo com uma medida de distorção/distância $d(\boldsymbol{x}, \boldsymbol{w}_i)$, utilizada para medir a distorção introduzida ao se representar um determinado vetor de entrada $\boldsymbol{x}$ pelo correspondente vetor-código $\boldsymbol{w}_i$ (o vetor $\boldsymbol{w}_i$ é escolhido como representante de $\boldsymbol{x}$ se $d(\boldsymbol{x}, \boldsymbol{w}_i) < d(\boldsymbol{x}, \boldsymbol{w}_j), \forall j \neq i$), os vetores-código devem ser escolhidos (projetados) visando minimizar a distorção média introduzida ao se representarem os vetores do sinal a ser quantizado pelos correspondentes vetores-código. Tendo em vista que a função densidade de probabilidades da fonte a ser quantizada é, em geral, desconhecida, o dicionário geralmente é projetado utilizando um conjunto de treino, e a distorção média a ser minimizada é aproximada pela distorção introduzida dentro do conjunto de treino.

Dentre as técnicas utilizadas para projeto de dicionários, o algoritmo LBG (Linde-Buzo-Gray) [3] destaca-se por sua ampla utilização. O presente trabalho apresenta a avaliação de desempenho de um algoritmo competitivo com consciência (ACC). O algoritmo corresponde à incorporação de um princípio de consciência (motivado pelo trabalho de Krishnamurthy et al. [4]) no algoritmo competitivo (AC) considerado em [5]. São apresentados resultados referentes à avaliação comparativa de desempenho dos algoritmos AC, ACC e LBG em projeto de dicionários aplicados à quantização vetorial de forma de onda de voz.

O restante do artigo encontra-se organizado de acordo com as seções a seguir. A Seção II apresenta uma descrição do algoritmo AC. O algoritmo ACC é descrito na Seção III. Resultados e conclusões são apresentados, respectivamente, nas Seções IV e V.

II. Algoritmo AC

Seja n_{tot} o número total de iterações (número total de passagens do conjunto de treino) do algoritmo AC e M_{vet} o número total de vetores de treino. Após a inicialização do dicionário, o algoritmo AC pode ser descrito como

Algoritmo ACC:
Para $1 \leq n \leq n_{\text{tot}}$
  Para $1 \leq m \leq M_{\text{vet}}$
    Determine o vencedor $\boldsymbol{w}_{i^*}(n,m)$:
    $i^* = \arg\min d[\boldsymbol{x}(m), \boldsymbol{w}_i(n,m)]$
  Atualize o vencedor:
    $w_{i^*j}(n, m+1) = w_{i^*j}(n,m) + \Delta w_{i^*j}(n,m)$, com
    $\Delta w_{i^*j}(n,m) = \eta(n)[(x_j(m) - w_{i^*j}(n,m)]$.

Na descrição, $\boldsymbol{x}(m)$ é o m-ésimo vetor do conjunto de treino, enquanto $\boldsymbol{w}_i(n,m)$ e $\boldsymbol{w}_{i^*}(n,m)$ denotam, respectivamente, o i-ésimo vetor-código e o vencedor quando da apresentação do m-ésimo vetor de treino na n-ésima iteração. Por sua vez,

$$d[\boldsymbol{x}(m), \boldsymbol{w}_i(n,m)] = \sum_{j=1}^{K} [x_j(m) - w_{ij}(n,m)]^2 \qquad (1)$$

denota a distância euclidiana entre os vetores $\boldsymbol{x}(m)$ e $\boldsymbol{w}_i(n,m)$, em que $x_j(m)$ é a j-ésima componente do vetor $\boldsymbol{x}(m)$ e $w_{ij}(n,m)$ é a j-ésima componente do vetor $\boldsymbol{w}_i(n,m)$. Na expressão que descreve a atualização do vencedor, Δw_{i^*j} é a modificação introduzida na j-ésima componente do vencedor, $\eta(n)$ é a taxa de aprendizagem ou

Francisco Madeiro, Departamento de Estatística e Informática, Universidade Católica de Pernambuco, Recife, PE, e-mail: madeiro@dei.unicap.br

Waslon Terllizzie Araújo Lopes, Faculdade de Ciência e Tecnologia – ÁREA1, Salvador, BA, e-mail: waslon@area1.br.

Marcelo Sampaio de Alencar e Benedito Guimarães Aguiar Neto, Departamento de Engenharia Elétrica, Universidade Federal de Campina Grande, Campina Grande, PB, e-mails:{malencar,bganeto}@dee.ufcg.edu.br.

ganho de adaptação na n-ésima iteração e w_{i^*j} é a j-ésima componente do vencedor.

No algoritmo AC a taxa de aprendizagem decresce linearmente com a iteração n, mantendo-se constante ao longo de toda iteração, isto é, durante cada passagem completa dos M_{vet} vetores de treino. É expressa por

$$\eta(n) = \eta(1) + (n-1)\frac{\eta(n_{\text{tot}}) - \eta(1)}{n_{\text{tot}} - 1}, \qquad (2)$$

em que $\eta(1)$ e $\eta(n_{\text{tot}})$ denotam dois parâmetros do algoritmo AC: a taxa de aprendizagem inicial e a taxa de aprendizagem final, respectivamente.

III. Algoritmo ACC

O principal problema da aprendizagem competitiva simples é que alguns neurônios (vetores-código, no contexto de QV) podem ter pouca ou nenhuma chance de ganhar a competição, o que pode resultar um dicionário que contém vetores-código que não tenham sido suficientemente treinados (vetores-código sub-utilizados), podendo resultar, em casos extremos, um dicionário que contenha alguns vetores-código não treinados[1] (o que seria um problema equivalente ao da existência de células de Voronoi vazias no algoritmo LBG).

Uma das maneiras de contornar esse problema foi apresentada por Krishnamurthy et al. em [4]. A técnica proposta, denominada FSCL (*frequency sensitive competitive learning*), pode ser utilizada para melhorar o desempenho de redes neurais competitivas. Na técnica FSCL a distorção é também uma função da freqüência com que os vetores-código ganham a competição (número de vezes em que os vetores-código são eleitos vencedores). A abordagem FSCL procura treinar igualmente todos os vetores-código, isto é, procura fazer com que todos os vetores-código sejam treinados (tenham suas componentes ajustadas) aproximadamente o mesmo número de vezes. De acordo com [4], o algoritmo FSCL constitui uma implementação do princípio de consciência de Grossberg [6].

No presente trabalho, a idéia proposta na técnica FSCL é introduzida no contexto do algoritmo AC, resultando o algoritmo ACC (*algoritmo competitivo com consciência*).

No algoritmo ACC a freqüência com que cada vetor-código ganha a competição é monitorada. Esta informação é usada durante o treinamento para assegurar que todos os vetores-código tenham aproximadamente o mesma oportunidade de serem atualizados. Precisamente, o algoritmo ACC utiliza uma medida de distorção (distância) modificada, apresentada em [4], que incorpora a freqüência com que cada vetor-código é escolhido vencedor.

Seja $d[\boldsymbol{x}(m), \boldsymbol{w}_i(n,m)]$ a medida de distorção utilizada pelo algoritmo AC. A medida de distorção modificada, utilizada pelo algoritmo ACC, é dada por

$$\hat{d}[\boldsymbol{x}(m), \boldsymbol{w}_i(n,m)] = f_i \times d[\boldsymbol{x}(m), \boldsymbol{w}_i(n,m)], \qquad (3)$$

em que f_i denota o número de vezes em que o i-ésimo vetor-código foi até então escolhido como vencedor. A Equação (3), portanto, pode ser expressa como

$$\hat{d}[\boldsymbol{x}(m), \boldsymbol{w}_i(n,m)] = f_i \times \sum_{j=1}^{K} [x_j(m) - w_{ij}(n,m)]^2. \qquad (4)$$

A Equação (4) mostra que se um vetor-código é freqüentemente escolhido vencedor, sua distorção $\hat{d}$ aumentará. Conseqüentemente, sua chance de vencer a(s) próxima(s) competição(ões) diminuirá, dando, portanto, aos outros vetores-código com pequeno valor de f_i a oportunidade de serem vencedores (de serem atualizados) na(s) próxima(s) apresentação(ões) de vetores de treino. Essa natureza adaptativa do algoritmo ACC possibilita que os vetores-código tendam a ser atualizados aproximadamente o mesmo número de vezes durante o treinamento.

IV. Resultados

O primeiro conjunto de simulações teve como objetivo avaliar a freqüência com que os vetores-código são atualizados nos algoritmos AC e ACC. As Figuras 1 e 2 apresentam, respectivamente, para o algoritmo AC e para o algoritmo ACC, o número de vezes em que cada vetor-código de um dicionário com $K = 4$ e $N = 64$ é atualizado (selecionado como vencedor), ao final de 14560 atualizações, correspondentes a 2 iterações ou 2 passagens completas de um conjunto de treino constituído de 7280 vetores de dimensão $K = 4$. Nas Figuras 1 e 2, a linha horizontal corresponde ao número médio de seleções como vencedor. Comparando as figuras, observa-se que o algoritmo ACC leva a uma maior uniformidade (constatada também para outras combinações de K e N) quanto ao número de vezes em que cada vetor-código é selecionado como vencedor.

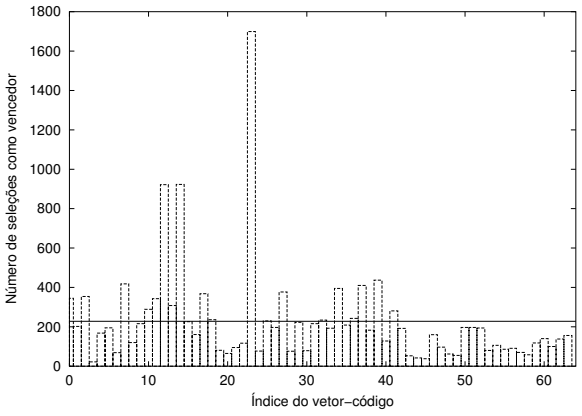

Figura 1. Número de vezes em que cada vetor-código de um dicionário AC com $K = 4$ e $N = 64$ é selecionado como vencedor.

O segundo conjunto de simulações procurou avaliar a homogeneidade da distribuição dos vetores de treino nas diversas células de Voronoi. Utilizou-se, para tanto, a entropia normalizada dos vetores-código (vide Apêndice para maiores detalhes). Conforme mostra a Tabela I, o algoritmo ACC produz dicionários que levam a uma distribuição mais homogênea (maiores valores de entropia

[1] No algoritmo de Kohonen, que utiliza vizinhanças para atualização dos vetores-código, a possibilidade de vetores-código não treinados ou pouco treinados pode ser eliminada por meio de uma sistemática adequada para escolha do tamanho de vizinhança.

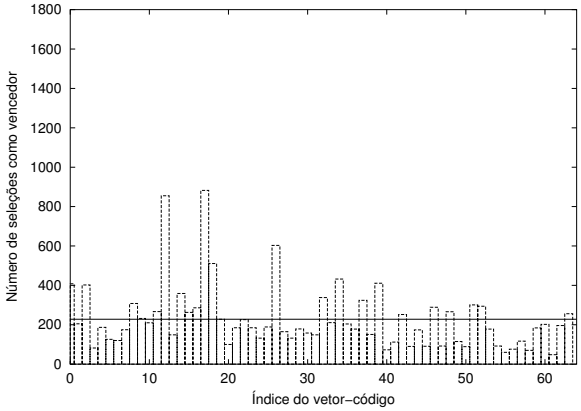

Figura 2. Número de vezes em que cada vetor-código de um dicionário ACC com $K = 4$ e $N = 64$ é selecionado como vencedor.

normalizada) dos vetores de entrada nas diversas células de Voronoi. Portanto, a utilização da medida de distorção modificada (Equação (4), usada pelo algoritmo ACC) em substituição à medida de distorção convencional (Equação (1), usada pelo algoritmo AC) contribui para a redução do número de células de Voronoi pequenas, isto é, contribui para a redução de vetores-código subutilizados.

Tabela I
Entropia normalizada ($\widetilde{H}$) dos vetores-código para diversos valores de número de níveis (N) fixada a dimensão $K = 2$ para os dicionários AC e ACC.

K	N	$\widetilde{H}$	
		AC	ACC
2	8	0,81	0,87
2	16	0,75	0,90
2	32	0,82	0,89
2	64	0,83	0,90
2	128	0,85	0,92
4	8	0,74	0,96
4	16	0,75	0,91
4	32	0,83	0,93
4	64	0,81	0,92
4	128	0,80	0,93

Observou-se, durante as simulações realizadas, que o algoritmo ACC, em geral, apresenta uma menor sensibilidade ao dicionário inicial quando comparado ao algoritmo AC.

É importante ressaltar que a velocidade de convergência (número de iterações) do algoritmo LBG depende fortemente do dicionário inicial, conforme mostra a Tabela II. Nos algoritmos ACC e AC não existe dependência entre número de iterações e dicionário inicial, tendo em vista que um dos parâmetros dos algoritmos ACC e AC é o número de iterações (n_{tot}). Comparando as Tabelas II e III, observa-se que os algoritmos AC e ACC necessitam de um número de iterações significativamente inferior ao requerido pelo algoritmo LBG para projetar dicionários. Ressalte-se que, mesmo com a utilização de um menor número de iterações, os algoritmos ACC e AC apresentam uma superioridade sobre o algoritmo LBG em termos de qualidade (avaliada via relação sinal-ruído segmental, SNRseg) do sinal de voz reconstruído para diversas taxas de codificação $R = \frac{1}{K}\log_2 N$, conforme mostram as Figuras 3 e 4. Em se tratando de QV a elevadas taxas de codificação observa-se nessas figuras que o algoritmo ACC apresenta-se como uma alternativa mais adequada que o algoritmo AC: para altas taxas de codificação, os sinais reconstruídos com dicionários ACC apresentam valores de SNRseg superiores aos apresentados pelos sinais reconstruídos usando dicionários AC.

Tabela II
Sensibilidade do algoritmo LBG a três dicionários iniciais diferentes (D_I, D_{II} e D_{III}) em termos de número total de iterações para diversos valores de tamanho do dicionário (N) e dimensão (K).

K	N	Número de iterações		
		D_I	D_{II}	D_{III}
2	32	52	75	52
2	64	74	130	118
2	128	77	198	95
4	32	61	61	51
4	64	84	94	58
4	128	81	68	38

Tabela III
Número total de iterações necessárias quando da aplicação dos algoritmos AC e ACC em projeto de dicionário para diversos valores de número de níveis (N) e dimensão (K).

K	N	Número de iterações
2	32	2
2	64	2
2	128	2
4	32	2
4	64	2
4	128	2

V. Conclusões

Neste trabalho foi avaliado um algoritmo competitivo com consciência (ACC), resultante da introdução de um princípio de consciência no processo de aprendizagem de um algoritmo competitivo (AC). Por meio da utilização de uma medida de distorção que leva em consideração o número de vezes em que os vetores-código são escolhidos vencedores, o algoritmo ACC procura dar a todos os vetores-código do dicionário aproximadamente a mesma oportunidade de serem treinados, isto é, de terem suas componentes ajustadas (adaptadas) ao longo da apresentação dos vetores de treino.

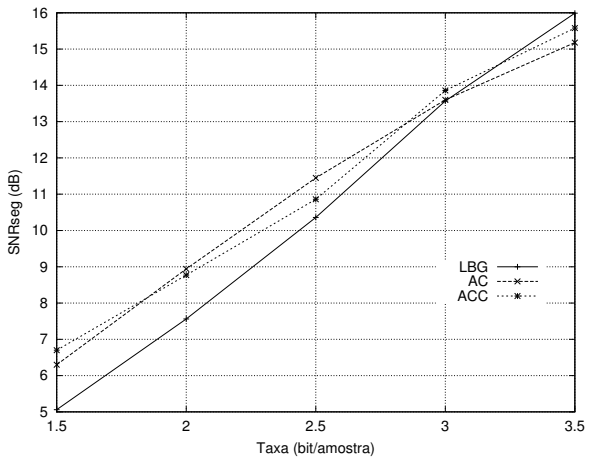

Figura 3. Desempenho dos algoritmos AC, ACC e LBG em QV de forma de onda de voz: SNRseg do sinal reconstruído versus taxa de codificação para o quantizador vetorial com dimensão $K = 2$.

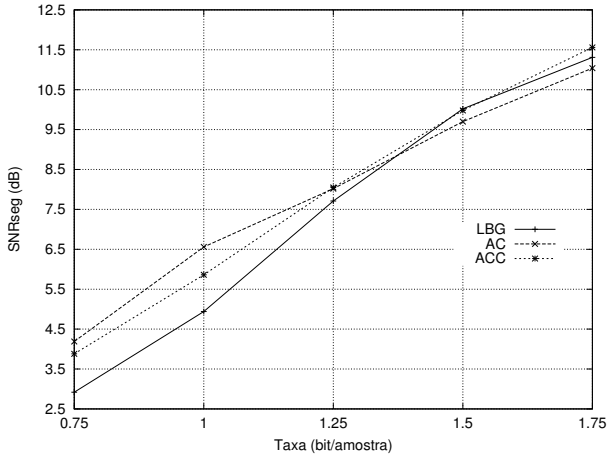

Figura 4. Desempenho dos algoritmos AC, ACC e LBG em QV de forma de onda de voz: SNRseg do sinal reconstruído versus taxa de codificação para o quantizador vetorial com dimensão $K = 4$.

Resultados referentes à aplicação dos algoritmos ACC e AC em projeto de dicionários destinados à quantização vetorial de forma de onda de voz mostraram que o algoritmo ACC produz dicionários com entropia normalizada dos vetores-código superior à apresentada pelos dicionários AC. O algoritmo ACC, portanto, leva a uma melhor distribuição (isto é, uma distribuição mais homogênea) dos vetores de treino nas diversas células de Voronoi. Em outras palavras, quando comparados aos dicionários AC, os dicionários ACC têm um menor número de vetores-código sub-utilizados, ou seja, têm uma menor quantidade de vetores-código com as correspondentes células de Voronoi pequenas.

Em diversas taxas de codificação de voz, o algoritmo ACC apresentou-se como uma alternativa adequada para projeto de dicionário. De fato, para várias taxas de codificação avaliadas, observou-se que os dicionários ACC (como também AC) levam a sinais de voz reconstruídos com qualidade superior (em termos de SNRseg) à obtida com uso de dicionários LBG. Além disso, ao contrário do que ocorre com o algoritmo LBG (em que o número de iterações depende fortemente do dicionário inicial), o número de iterações do algoritmo ACC (a exemplo do algoritmo AC) é especificado *a priori*, como um parâmetro do algoritmo. Os resultados de simulação mostraram que os algoritmos ACC e AC produzem dicionários com qualidade próxima ou levemente superior à apresentada pelos dicionários LBG usando um número de iterações significativamente menor que o requerido pelo algoritmo LBG.

No tocante à codificação de forma de onda de voz utilizando QV a elevadas taxas, observou-se que o algoritmo ACC apresenta um desempenho superior ao apresentado pelo algoritmo AC: os sinais reconstruídos com o uso de dicionários ACC apresentam valores SNRseg superiores aos apresentados pelos sinais reconstruídos utilizando dicionários AC para altas taxas de codificação.

Apêndice – Entropia Normalizada dos Vetores-Código

Este apêndice descreve a entropia normalizada dos vetores-código, que serve como uma medida para avaliar o grau de homogeneidade da distribuição dos vetores de entrada ao longo das células de Voronoi [7].

Seja $W = \{\boldsymbol{w}_i;\ i = 1,\ 2,\ \ldots,\ N\}$ um dicionário de tamanho N, em que $\boldsymbol{w}_i$ denota o i-ésimo vetor-código K-dimensional. Seja p_i a probabilidade de que um dado vetor de entrada pertença à região ou célula de Voronoi correspondente a $\boldsymbol{w}_i$. Em outras palavras, p_i representa a probabilidade de que $\boldsymbol{w}_i$ seja o vizinho mais próximo de um dado vetor de entrada $\boldsymbol{x}$ (probabilidade de que $\boldsymbol{w}_i$ seja o vetor-código mais semelhante a $\boldsymbol{x}$ dentre todos os vetores-código do dicionário).

Seja $S = \{\boldsymbol{x}(m),\ m = 1, ..., M_{\text{vet}}\}$ um longo conjunto de treino (isto é, $M_{\text{vet}} >> N$), ou seja, S é um longo conjunto de vetores que são utilizados no projeto de dicionário. Devido ao mapeamento promovido pela quantização vetorial, o conjunto S é particionado em N conjuntos (disjuntos) de Voronoi $S_i,\ i = 1, ..., N$, em que cada célula S_i coleciona todos os vetores de treino que são mapeados no i-ésimo vetor-código: $S_i = \{\boldsymbol{x}(m) :\ Q(\boldsymbol{x}(m)) = \boldsymbol{w}_i\}$. Em outras palavaras, $\boldsymbol{w}_i$ é o vetor-código mais próximo de todos $\boldsymbol{x}(m) \in S_i$.

Seja M_i o tamanho do subconjunto S_i, isto é, M_i é o número de vetores de entrada mapeados no i-ésimo vetor-código. Uma estimativa para a probabilidade de que $\boldsymbol{w}_i$ seja o vetor-código mais próximo de qualquer $\boldsymbol{x}(m)$ (que corresponde à probabilidade de que S_i seja a célula de Voronoi de $\boldsymbol{x}(m)$) pode ser obtida como

$$p_i = \frac{M_i}{M_{\text{vet}}}. \tag{5}$$

A entropia H dos vetores-código é definida como

$$H = \sum_{i=1}^{N} p_i \log_2(\frac{1}{p_i}). \tag{6}$$

A entropia normalizada $\widetilde{H}$ dos vetores-código é dada por

$$\widetilde{H} = \frac{H}{\log_2 N}, \tag{7}$$

isto é,

$$\widetilde{H} = \frac{\sum_{i=1}^{N} p_i \log_2(\frac{1}{p_i})}{\log_2 N}. \tag{8}$$

A máxima entropia normalizada ocorre para vetores-código equiprováveis. De fato, equiprobabilidade implica $H = \log_2 N$. Como conseqüência, $\widetilde{H} = 1$. É importante observar que $\widetilde{H} \to 1$ à medida que aumenta a homogeneidade da distribuição dos vetores de entrada ao longo das células de Voronoi, isto é, $\widetilde{H} \to 1$ à medida que $H \to \log_2 N$. Por outro lado, para N determinado, a entropia normalizada decresce à medida que aumenta o número de células de Voronoi pequenas.

Referências

[1] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, Boston, MA, 1992.

[2] R. M. Gray. "Vector Quantization". *IEEE ASSP Magazine*, pp. 4–29, April 1984.

[3] Y. Linde, A. Buzo and R. M. Gray. "An Algorithm for Vector Quantizer Design". *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95, January 1980.

[4] A. K. Krishnamurthy, S. C. Ahalt, D. E. Melton and P. Chen. "Neural Networks for Vector Quantization of Speech and Images". *IEEE Journal on Selected Areas in Communications*, vol. 8, no. 8, pp. 1449–1457, October 1990.

[5] F. Madeiro, W. T. A. Lopes, B. G. Aguiar Neto and M. S. Alencar. "Complexidade Computacional de um Algoritmo Competitivo Aplicado ao Projeto de Quantizadores Vetoriais". *Learning and Nonlinear Models – Revista da Sociedade Brasileira de Redes Neurais*, vol. 1, no. 3, pp. 172–186, 2004.

[6] S. Grossberg. "Adaptive Pattern Classification and Universal Recording: I. Parallel Development and Coding of Neural Feature Detectors". *Biological Cybernetics*, vol. 23, pp. 121–134, 1976.

[7] K. K. Paliwal and V. Ramasubramanian. "Effect of Ordering the Codebook on the Efficiency of the Partial Distance Search Algorithm for Vector Quantization". *IEEE Transactions on Communications*, vol. 35, no. 5, pp. 538–540, May 1989.