UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
ESCOLA POLITÉCNICA
DEPARTAMENTO DE ELETRÔNICA E DE COMPUTAÇÃO

ANÁLISE DE TAXA, DISTORÇÃO E COMPLEXIDADE PARA COMPRESSORES DE IMAGENS BASEADOS EM QUANTIZAÇÃO VETORIAL INTERPOLATIVA

ESTEVAN PEREIRA SERACO

Orientador:

Prof. José Gabriel R. C. Gomes, Ph.D.

Examinadores:

Prof. Antonio Petraglia, Ph.D.

Prof. Eduardo Antônio B. da Silva, Ph.D.

Projeto Final

Junho de 2008

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
ESCOLA POLITÉCNICA
DEPARTAMENTO DE ELETRÔNICA E DE COMPUTAÇÃO

ANÁLISE DE TAXA, DISTORÇÃO E COMPLEXIDADE PARA COMPRESSORES DE IMAGENS BASEADOS EM QUANTIZAÇÃO VETORIAL INTERPOLATIVA

Autor:
Estevan Pereira Seraco

Orientador:
Prof. José Gabriel R. C. Gomes, Ph.D.

Examinador:
Prof. Antonio Petraglia, Ph.D.

Examinador:
Prof. Eduardo Antônio B. da Silva, Ph.D.

DEL

Junho de 2008

Agradecimentos

Meus sinceros agradecimentos ao Prof. José Gabriel R. C. Gomes, pela sua excelente orientação técnica ao longo do desenvolvimento deste trabalho.

Sou grato aos examinadores deste texto, Prof. Antonio Petraglia e Prof. Eduardo Antônio B. da Silva, pelos construtivos comentários a respeito desta pesquisa.

À minha família, agradeço o incentivo e a motivação, que me guiaram ao longo dos inesquecíveis cinco anos deste curso de graduação.

Resumo

A análise de desempenho de compressores de imagens é crítica em dispositivos portáteis, devido à limitação de potência, refletindo em um compromisso entre taxa, distorção e complexidade para seu projeto. Neste trabalho, a preocupação com a economia de energia orienta à proposição de um esquema de compressão de imagens adequado para câmeras fotográficas, implementável no plano focal, em que a codificação é realizada antes da conversão analógico-digital do sinal.

Para o projeto de compressores de imagens no plano focal, é proposta a Quantização Vetorial Interpolativa (IVQ), técnica que possui sua implementação associada a circuitos de reduzida complexidade. É apresentado um aperfeiçoamento do algoritmo de projeto IVQ, para a determinação de codificadores com nível de desempenho equivalente a codificadores de Quantização Vetorial com Restrição de Entropia (ECVQ), consideráveis como estado-da-arte.

Uma métrica para a avaliação de complexidade dos esquemas de compressão é apresentada. Quando tal métrica é aplicada aos esquemas IVQ e ECVQ, constata-se que a complexidade de codificação IVQ é significativamente menor do que a codificação ECVQ, em troca de uma mínima perda de desempenho em termos de taxa e distorção.

Palavras-Chave

Compressão de Imagens, Quantização Vetorial Interpolativa, Redes Neurais, Análise RDC, Processamento de Imagens no Plano Focal.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

No universo das telecomunicações, o surgimento de sistemas de transmissão digital que possibilitam o envio de dados a altas taxas, através de dispositivos móveis, torna conveniente que recursos multimídia se integrem a sistemas portáteis. Nesse cenário, o projeto de câmeras fotográficas portáteis assume papel crítico, se for considerado o fato de que a complexidade do processamento para a captura, codificação e transmissão de imagens é associada a um consumo de energia elevado. Dentre soluções de engenharia propostas para o processamento da imagem nas câmeras, está a adoção de esquemas de compressão que realizem a codificação antes da conversão analógico-digital, usando métodos de processamento de sinais no *plano focal*. Espera-se que essa estratégia será capaz de prover considerável economia de energia, uma vez que concentraria a maior parte do processamento em circuitos CMOS.[1]

Para a implementação do processamento no sensor, é possível utilizar a técnica de quantização vetorial. Em termos de desempenho de esquemas de compressão de imagens, existe o clássico compromisso entre manter a fidelidade da informação após sua compressão e codificá-la de modo a reduzir as taxas de sua transmissão. Para projetos com esse compromisso, a solução proposta pela Quantização Vetorial com Restrição de Entropia (*Entropy-Constrained Vector Quantization* – ECVQ) provê desempenho considerável como estado-da-arte, embora associado a uma complexidade elevada.

Nesse contexto, o presente trabalho desenvolve a proposição de esquemas de compressão de imagens implementados por uma técnica alternativa à ECVQ, conhecida como Quantização Vetorial Interpolativa (*Interpolative Vector Quantization* – IVQ). Por levar

[1] Uma lista de siglas é oferecida no Apêndice A.

em conta o desempenho dos sistemas em termos de taxa, distorção (relacionada à qualidade da imagem) e *complexidade*, esse trabalho se enquadra na categoria de realização de análise RDC (*rate, distortion and complexity*) [6].

O **Capítulo 2** introduz os conceitos de quantização vetorial. São discorridas análises referentes à não-analiticidade da modelagem matemática do quantizador vetorial. As condições que definem um codificador ótimo são tratadas, o que termina na proposição de algoritmos de otimização para o cálculo de quantizadores. Por fim, para possibilitar a comparação entre esquemas de quantização ECVQ e IVQ, são apresentadas arquiteturas de redes neurais capazes de implementar esquemas ECVQ.

O **Capítulo 3** apresenta a quantização IVQ, destacando métodos para o cálculo analítico do extrator de características. São apresentadas as análises de componentes principais. Para a implementação de tais operações, é apresentada a rede neural MLP modelada para cada tipo de PCA. O algoritmo para projeto IVQ é analisado, definido e, em seguida, aprimorado.

O **Capítulo 4** foca na análise de distorção e entropia para o processamento de imagens. Um esquema completo para codificação de imagens é apresentado, desde a etapa de pré-processamento. São apresentados resultados de simulação pelos quais se avaliam os esquemas de compressão de imagens em termos de taxa e distorção.

O **Capítulo 5** apresenta uma métrica para avaliação de complexidade e compara os esquemas IVQ com a solução tradicional ECVQ, em termos de taxa, distorção e complexidade.

1.1 Notações Matemáticas

Ao longo deste trabalho, uma estratégia recorrente será o uso do formalismo da notação matemática para representar as idéias e fornecer suporte teórico de modo não-ambíguo. Uma das etapas importantes para se alcançarem essas metas é a adoção de uma notação metódica. Ao longo desta dissertação, foram distinguidas quatro importantes entidades matemáticas: *escalares*, *vetores*, *matrizes* e *conjuntos*. Reservam-se as seguintes convenções notacionais para esse texto:

- Letras em *itálico* para escalares;
- Letras minúsculas e em **negrito** para vetores;

- Letras maiúsculas e em **NEGRITO** para matrizes;
- Letras maiúsculas $\mathcal{CALIGRÁFICAS}$ para conjuntos.

O tamanho T de um vetor $\mathbf{x}$ especificará que sua dimensão é $T \times 1$, ou seja, os vetores serão do tipo coluna. A dependência de uma variável será identificada pelos parênteses. Assim, $\mathbf{x}(i)$ demonstra que o vetor $\mathbf{x}$ é função do escalar i. Já a notação com índice sub-escrito, como $\mathbf{x}_i$, identifica o i-ésimo vetor de um conjunto $\mathcal{X}$ de vetores ou a i-ésima coluna de uma matriz $\mathbf{X}$. O escalar x_i, por outro lado, representa a i-ésima entrada do vetor $\mathbf{x}$ e o escalar X_{ij} a variável de posição (i, j) da matriz $\mathbf{X}$. Se há necessidade de aninhamento de índices, então o índice mais representativo será aquele mais interno. Dessa forma, $[x_i]_j$ identifica a j-ésima entrada do vetor $\mathbf{x}_i$ e $[X_i]_{jk}$ se associa ao escalar que ocupa a posição (j, k) na matriz $\mathbf{X}_i$.

Finalmente, para a referência aos símbolos matemáticos utilizados, o leitor poderá consultar o Apêndice A.

Capítulo 2

Fundamentos de Quantização Vetorial

Um quantizador vetorial Q de dimensão M e tamanho K é definido como o mapeamento de vetores de dimensão M em vetores pertencentes a um conjunto finito $\mathcal{C}$ de tamanho K,

$$Q : \mathbb{R}^M \rightarrow \mathcal{C}. \tag{2.1}$$

O conjunto $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_k, \ldots, \mathbf{c}_K\}$ é chamado de "dicionário" e um vetor $\mathbf{c}_k \in \mathbb{R}^M$ é denominado "palavra". Como conseqüência direta da definição, a quantização vetorial permite que uma amostra de um sinal discreto com amplitude de precisão infinita seja quantizada para ser representada pelo respectivo índice k, ao invés de sua palavra no dicionário, $\mathbf{c}_k$. Para tanto, Q deve ser realizada em etapas, sendo inicializada pela codificação, α,

$$\alpha : \mathbb{R}^M \rightarrow \{1, \ldots, K\} \tag{2.2}$$

e seguida pela decodificação, β,

$$\beta : \{1, \ldots, K\} \rightarrow \mathcal{C}. \tag{2.3}$$

Dada uma realização do sinal de entrada $\mathbf{x}(n)$ para $n = 1, \ldots, N$, deseja-se que a sua versão quantizada $\hat{\mathbf{x}}(n) = Q(\mathbf{x}(n))$ seja de algum modo representativa do sinal original. Por ser uma função não-injetora, Q é não-invertível, de modo que há perda de informação, interpretável como uma inerente distorção. Em projetos de quantização vetorial, introduz-se como critério de avaliação de distorção o erro médio quadrático (*mean squared error* – MSE), definido por

$$D = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}(n) - \hat{\mathbf{x}}(n)\|^2, \tag{2.4}$$

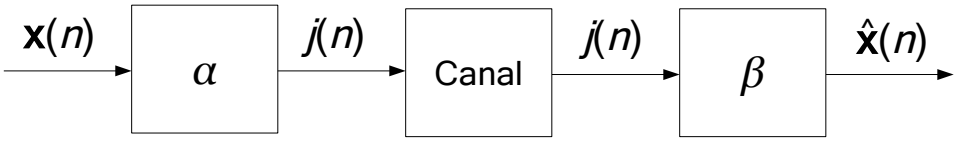


Figura 2.1: Esquema de quantização vetorial.

implicando que o problema de projeto de quantização vetorial pode ser formulado como

$$Q' = \arg\min_{Q} D. \tag{2.5}$$

2.1 Projeto de Quantizadores Vetoriais

A quantização vetorial definida pela Equação (2.1), quando subdividida em codificação e decodificação, é ilustrada pela Figura 2.1. Até o momento, nenhuma consideração a respeito da modelagem do mapeamento Q foi feita. Naturalmente, a fim de projetar o quantizador, faz-se necessária a análise matemática de tal operação. Para tanto, tomam-se como ponto de partida o critério de avaliação pela distorção D (Equação (2.5)) e o fato de que o quantizador deve ter o formato da Figura 2.1. Assim, é conhecido na literatura [8] que são condições necessárias para o quantizador ótimo:

Condição da Partição: Considere K fixo e que um dicionário fixo seja dado, $\mathcal{C}' = \{\mathbf{c}'_1, \ldots, \mathbf{c}'_k, \ldots, \mathbf{c}'_K\}$. O codificador ótimo para uma realização do sinal $\mathbf{x}(n)$, $n = 1, \ldots, N$, é dado por

$$j(n) = \arg\min_{k} \|\mathbf{x}(n) - \mathbf{c}'_k\|^2, \ n = 1, \ldots, N; \tag{2.6}$$

Condição do Centróide: Considere K fixo e que um codificador fixo α' seja dado. O decodificador ótimo é então dado por

$$\mathbf{c}_k = \frac{\sum_n \big(\mathbf{x}(n) \mid \alpha'(\mathbf{x}(n)) = k\big)}{\mathrm{card}\big(\mathbf{x}(n) \mid \alpha'(\mathbf{x}(n)) = k\big)}, \ k = 1, \ldots, K. \tag{2.7}$$

A principal conseqüência das Equações (2.6) e (2.7) é o fato de que a função Q é não-analítica. Não existe, portanto, uma solução analítica para a Equação (2.5), de modo que o projeto de quantização deve se guiar através de algoritmos de otimização. O algoritmo clássico para tal propósito é o Algoritmo de Lloyd Generalizado, cuja descrição é feita na Seção 2.1.1.

2.1.1 O Algoritmo de Lloyd

Embora exista uma infinidade de algoritmos de otimização empregados para o projeto de quantização vetorial, pode-se dizer que a grande maioria é uma variação do Algoritmo de Lloyd Generalizado (*Generalized Lloyd Algorithm* – GLA). Tome $\mathcal{X} = \{\mathbf{x}(1), \ldots, \mathbf{x}(n), \ldots, \mathbf{x}(N)\}$ como conjunto de dados para treino e considere que o tamanho do dicionário K seja uma especificação do projeto. Considere ainda que se dispõe de um dicionário inicial $\mathcal{C}^0$ e que ϵ seja um número real positivo arbitrariamente pequeno. Segue que a i-ésima iteração do GLA é definida pelos seguintes passos:

1. **Atualização do Codificador:** Aplique a Condição da Partição (Equação (2.6)) sobre $\mathcal{X}$ e $\mathcal{C}^{i-1}$, para calcular o codificador α^i.

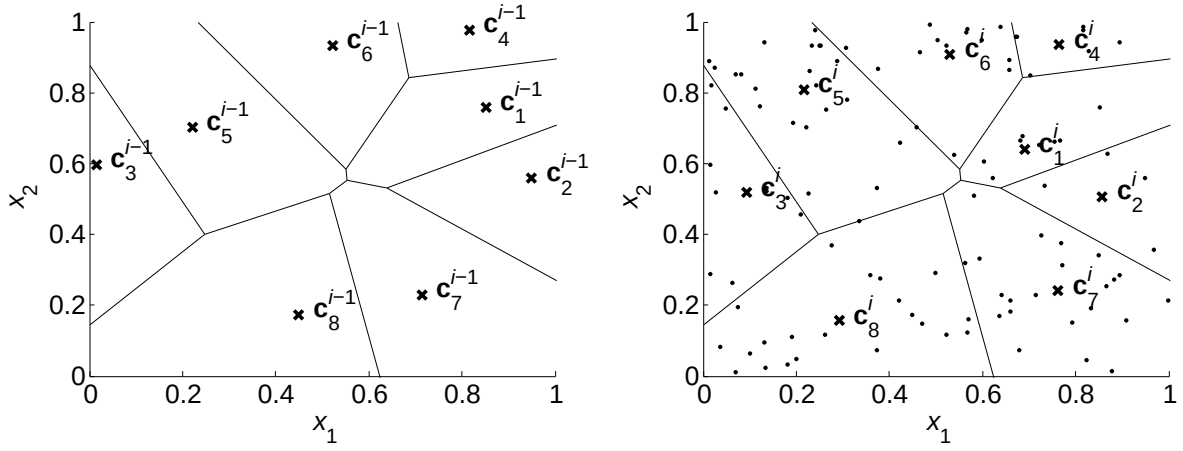
2. **Atualização do Decodificador (Dicionário):** Aplique a Condição do Centróide (Equação (2.7)) sobre $\mathcal{X}$ e α^i, para calcular o dicionário $\mathcal{C}^i$.

3. **Verificação de Convergência:** Calcule D_i (Equação (2.4)) e interrompa o algoritmo se
$$\frac{D_{i-1} - D_i}{D_{i-1}} < \epsilon. \tag{2.8}$$
Assuma como resultado do algoritmo o codificador $\alpha' = \alpha^i$ e o dicionário $\mathcal{C}' = \mathcal{C}^i$.

O GLA garante que a cada iteração a distorção será sempre menor ou igual à anterior, ou seja, há garantia de que o algoritmo sempre converge. No entanto, a solução é apenas localmente ótima, uma vez que o resultado depende da inicialização $\mathcal{C}^0$, que em casos práticos é arbitrada.

2.1.2 Interpretação Geométrica para o Algoritmo de Lloyd

É possível interpretar geometricamente as atualizações do codificador e do dicionário ocasionadas pelo GLA. A operação de codificação dada pela Equação (2.6) geometricamente consiste em particionar o espaço vetorial em K regiões, nas quais qualquer vetor $\mathbf{x}$ pertencente sofre a mesma indexação $j = \alpha(\mathbf{x})$, segundo o critério da menor distância euclidiana. Mais além, as fronteiras dos sub-espaços são polígonos de Voronoi [8], sendo funções exclusivas do dicionário, uma vez que a k-ésima região está definida unicamente pela palavra do dicionário $\mathbf{c}_k$ e suas vizinhas mais próximas.



(a) (b)

Figura 2.2: Exemplo de efeito do GLA. Considere o caso bidimensional ($M = 2$) no qual o conjunto de treino tenha tamanho $N = 100$ e o tamanho do dicionário seja $K = 8$. Na i-ésima iteração do GLA, dado o dicionário da iteração anterior, $\mathcal{C}^{i-1}$, a Condição da Partição é ilustrada pela Figura (a). Uma vez calculado o dicionário, α^i, a partição é mantida fixa (Figura (b)), e os dados $\mathbf{x}(n) = [x_1(n) \;\; x_2(n)]^\mathsf{T}$, $n = 1, \ldots, N$, definem os novos centróides que atualizam o dicionário. Particularmente, $i = 1$.

Uma outra interpretação para a Equação (2.6), para quando o codificador é dado, é associar a codificação à existência de uma partição geométrica, de modo que se definem subconjuntos de dados cuja indexação é a mesma. É possível recalcular o dicionário de modo que os respectivos centros de gravidade de cada subconjunto constituam suas palavras. Esse raciocínio é matematicamente formalizado pela condição do centróide (Equação (2.7)).

A ilustração para o raciocínio geométrico é apresentada na Figura 2.2.

2.2 Quantização Vetorial com Restrição de Entropia

Uma maneira de fazer com que a distorção de um quantizador vetorial seja tão pequena quanto se deseje é aumentar o tamanho do dicionário. No limite, um dicionário com tamanho igual ao conjunto de dados, N, teria distorção nula. Em situações práticas, o tamanho do dicionário é limitado, uma vez que existem restrições da taxa de dados que será enviada por um canal com recursos de potência limitados.

Após o sinal $\mathbf{x}(n)$ ser codificado, obtêm-se os correspondentes índices $j(n) = \alpha(\mathbf{x}(n))$, cuja entropia H em *bit* por vetor é dada por

$$p_k = \frac{1}{N}\text{card}\big(j(n) \mid \alpha(\mathbf{x}(n)) = k\big),\ k = 1, \ldots, K, \tag{2.9}$$

$$H = -\sum_{k=1}^{K} p_k \log_2 p_k. \tag{2.10}$$

Com a finalidade de transmissão por um canal a uma taxa próxima à entropia, os índices $j(n)$ devem ser mapeados pelo codificador de fonte discreta (*variable-length coder* – VLC), γ, em vetores de comprimento variável $\mathbf{v}(n)$,

$$\begin{aligned} \gamma : k \in \{1, \ldots, K\} &\rightarrow \{0,1\}^{\mathfrak{C}(k)},\ k = 1, \ldots, K \\ j(n) &\mapsto \mathbf{v}(n), \end{aligned} \tag{2.11}$$

$$\gamma^{-1} : j(n) = k \mid \mathbf{v}(n) = \gamma(k),$$

onde $\mathfrak{C}(k)$ depende do esquema VLC empregado e representa o comprimento do código utilizado para representar o k-ésimo índice do dicionário, ou seja, $\mathfrak{C}(k) = \mathfrak{l}(\gamma(k))$, para $\mathfrak{l}(\cdot)$ significando comprimento de vetor. Na quantização do sinal $\mathbf{x}(n)$, a taxa média de transmissão R, dada em *bit* por vetor, é igual a

$$R = \frac{1}{N}\sum_{n=1}^{N} \mathfrak{l}(\mathbf{v}(n)), \tag{2.12}$$

uma vez que o mapeamento γ possui vetores de símbolos binários como conjunto-imagem, conforme definido na Equação (2.11). Realizando o esquema VLC, é garantido que a taxa média de transmissão está confinada no intervalo [2]

$$R \in \left[H,\ H + \frac{1}{M}\right]. \tag{2.13}$$

A Equação (2.13) possui duas importantes implicações. A primeira versa a questão do valor de M: à medida que a dimensão do espaço vetorial aumenta, a taxa necessariamente se aproxima da entropia. Por esse motivo, a quantização vetorial possui desempenho em termos de entropia superior à quantização escalar, caso particular em que $M = 1$. A segunda conseqüência diz respeito à avaliação da taxa média transmitida através da codificação de entropia: o projeto de quantização vetorial pode se desenvolver sem que se conheça a operação realizada por γ em detalhes, uma vez que R estará necessariamente

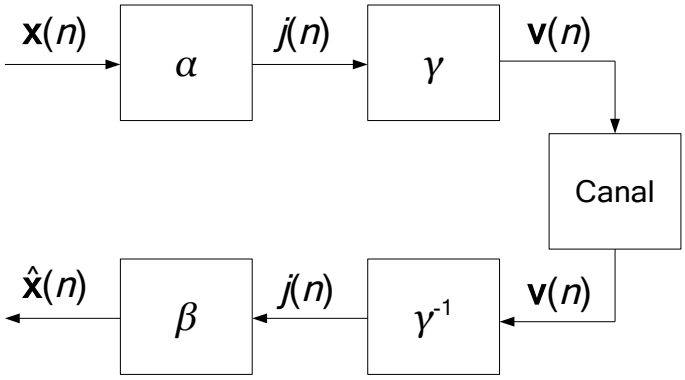


Figura 2.3: Quantização vetorial com restrição de entropia.

limitada por H. Além disso, pode-se assumir que, uma vez projetado o quantizador avaliado em termos de H, existirão esquemas de codificação de razoável simplicidade (como a de Huffman) que implementarão de forma adequada a VLC, ou seja, aproximarão o valor da taxa média R ao valor da entropia H.

O esquema para a compressão de dados incluindo a VLC é ilustrado pela Figura 2.3. O compromisso entre distorção e taxa pode ser modelado como uma função custo J ponderada por um multiplicador de Lagrange λ [4],

$$J = D + \lambda H, \tag{2.14}$$

requerendo uma adaptação do problema da Equação (2.5) a

$$Q' = \arg\min_{Q} J. \tag{2.15}$$

2.3 Projeto de Quantizadores Vetoriais com Restrição de Entropia

As condições necessárias para que a Quantização Vetorial com Restrição de Entropia (*Entropy-Constrained Vector Quantization* – ECVQ) seja ótima são extensões das Condições da Partição e do Centróide dadas pelas Equações (2.6) e (2.7):

Condição da Partição para ECVQ: Considere K fixo e que um dicionário fixo seja dado por $\mathcal{C}' = \{\mathbf{c}'_1, \ldots, \mathbf{c}'_k, \ldots, \mathbf{c}'_K\}$. O codificador ótimo para uma realização do sinal $\mathbf{x}(n)$, $n = 1, \ldots, N$, é dado por

$$j(n) = \arg\min_{k} \left(\|\mathbf{x}(n) - \mathbf{c}'_k\|^2 + \lambda \mathfrak{l}(\mathbf{v}(n)) \right), \ n = 1, \ldots, N; \tag{2.16}$$

Condição do Centróide para ECVQ: Considere K fixo e que um codificador fixo α' seja dado. O decodificador ótimo é então dado por

$$\mathbf{c}_k = \frac{\sum_n \big(\mathbf{x}(n) \mid \alpha'(\mathbf{x}(n)) = k\big)}{\mathrm{card}\big(\mathbf{x}(n) \mid \alpha'(\mathbf{x}(n)) = k\big)},\ k = 1, \ldots, K. \tag{2.17}$$

Embora as Condições do Centróide dadas pelas Equações (2.17) e (2.7) sejam idênticas, a Condição da Partição (Equação (2.16)) foi modificada pela inclusão do fator $\mathfrak{l}(\mathbf{v})$, ponderado pelo multiplicador de Lagrange. Essa operação é condizente com a nova expressão do custo, apresentada na Equação (2.14), que leva em conta a entropia. No entanto, como já justificado pela Equação (2.13), a avaliação de $\mathfrak{l}(\mathbf{v})$, que a princípio envolveria a determinação de γ, ocorrerá imediatamente pela seguinte estimativa:

$$\mathfrak{l}(\mathbf{v}(n)) \approx -\log_2 p_k \mid k = \alpha(\mathbf{x}(n)) \tag{2.18}$$

para p_k definido na Equação (2.9).

2.3.1 O Algoritmo de Lloyd para Codificação de Entropia

A inclusão da restrição de entropia da ECVQ não altera o fato de que a solução para o problema da Equação (2.15) é não-analítica. De forma análoga, o GLA é adaptado para o projeto ECVQ. Resgatando a notação, tome $\mathcal{X} = \{\mathbf{x}(1), \ldots, \mathbf{x}(n), \ldots, \mathbf{x}(N)\}$ como conjunto de dados para treino e considere que K seja dado e que o compromisso entre distorção e entropia, dado pelo multiplicador de Lagrange λ, seja uma especificação do projeto. Considere ainda que se dispõe de um dicionário inicial $\mathcal{C}^0$ e que ϵ seja um número real positivo arbitrariamente pequeno. Segue que a i-ésima iteração do GLA para ECVQ é redefinida pelos seguintes passos [17]:

1. **Atualização do Codificador:** Aplique a Condição da Partição para ECVQ (Equação (2.16)) sobre $\mathcal{X}$ e $\mathcal{C}^{i-1}$, para calcular o codificador α^i. Nesse ponto, o leitor terá utilizado a aproximação da Equação (2.18), aplicada ao codificador [1] α^{i-1} e ainda tomando como dicionário $\mathcal{C}^{i-1}$, para calcular $\mathfrak{l}(\mathbf{v}(n))$.

[1] O uso do dicionário anterior, α^{i-1}, requer a definição do codificador de inicialização

$$\alpha^0 : j(n) = \arg\min_k \|\mathbf{x}(n) - \mathbf{c}_k^0\|^2,\ n = 1, \ldots, N,$$

para $\{\mathbf{c}_1^0, \ldots, \mathbf{c}_k^0, \ldots, \mathbf{c}_K^0\} = \mathcal{C}^0$. O dicionário de inicialização $\mathcal{C}^0$, por sua vez, pode ser composto por K vetores escolhidos aleatoriamente ou por vetores gerados sistematicamente, através do centro de gravidade de $\mathcal{X}$.

2. **Atualização do Decodificador (Dicionário):** Aplique a Condição do Centróide (Equação (2.17)) sobre $\mathcal{X}$ e α^i, para calcular o dicionário $\mathcal{C}^i$.

3. **Verificação de Convergência:** Calcule J_i (Equação (2.14)) e interrompa o algoritmo se

$$\frac{J_{i-1} - J_i}{J_{i-1}} < \epsilon. \tag{2.19}$$

Assuma como resultado do algoritmo o codificador $\alpha' = \alpha^i$ e o dicionário $\mathcal{C}' = \mathcal{C}^i$.

2.4 Mapas Auto-Organizáveis de Kohonen

O projeto de quantização vetorial foi discutido nas Seções 2.1 e 2.3 com as apresentações, respectivamente, do GLA e do GLA para ECVQ. Foi dito que a solução encontrada por tais algoritmos é localmente ótima, dependente apenas de suas inicializações. Para demonstrar o poder desses algoritmos, considere um quantizador Q', projetado por algum método desconhecido de otimização para ECVQ. O quantizador Q' é especificado em termos de seu codificador, α', e de seu dicionário, $\mathcal{C}'$. Por melhor que seja Q', sempre será possível inicializar o algoritmo GLA com $\mathcal{C}'$ e obter como solução um quantizador Q'' otimizado. Na pior das hipóteses, Q'' terá o mesmo desempenho que Q'. Por essa razão, o GLA é considerado como o método de projeto estado-da-arte em termos do custo J, no caso em que os vetores $\mathbf{x}(n)$ são estatisticamente independentes. Qualquer novo método de projeto pode ser então comparado ao GLA em termos de desempenho.

A arquitetura resultante da ECVQ projetada pelo GLA é conhecida como Quantização Vetorial por Busca Completa (*Full-Search Vector Quantization*), uma vez que sua codificação requer o cálculo de K distâncias euclidianas entre o vetor de dados $\mathbf{x}$ e os centróides $\mathbf{c}_k$, seguido por comparação entre essas distâncias para determinar qual centróide é o mais próximo e, portanto, a palavra do dicionário mais representativa. Mapas Auto-Organizáveis de Kohonen, ou, simplesmente, Mapas Auto-Organizáveis (*Self-Organized Maps* – SOM) [15] são Redes Neurais capazes de implementar o codificador ECVQ projetado por GLA, como foi definido na Equação (2.16).[2]

Uma Rede Neural SOM que implementa a operação de codificação representada pela

[2]A teoria para SOM, além de descrever arquiteturas, também trata algoritmos para sua otimização. Nesse contexto, o algoritmo GLA figura como um caso específico, embora suficiente para a aplicação deste trabalho.

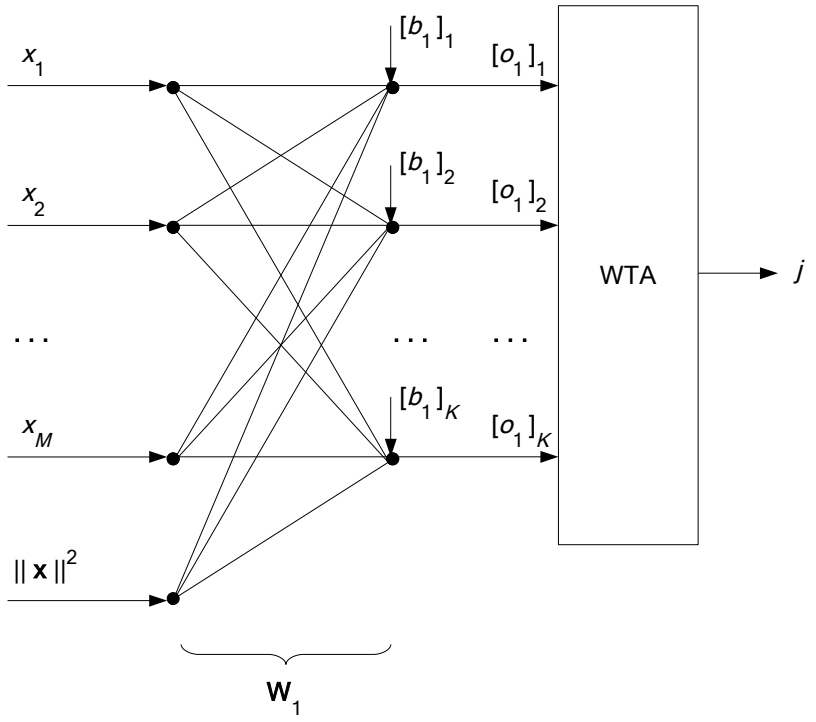


Figura 2.4: Rede SOM que implementa ECVQ.

Equação (2.16) tem a estrutura ilustrada pela Figura 2.4. Observando essa figura, percebe-se que a rede é composta por duas camadas. A primeira camada calcula as distâncias euclidianas e as pondera pelo comprimento do código VLC para cada respectivo centróide. O vetor de saída da camada 1, $\mathbf{o}_{1(K\times 1)}$, deve ser dado então por:

$$\begin{aligned}
[o_1]_k(n) &= \|\mathbf{x}(n) - \mathbf{c}_k\|^2 + \lambda\mathfrak{C}(k) \\
&= \sum_{m=1}^{M}(x_m(n) - [c_k]_m)^2 + \lambda\mathfrak{C}(k) \\
&= \|\mathbf{x}(n)\|^2 + \sum_{m=1}^{M}(-2[c_k]_m x_m(n)) + \|\mathbf{c}_k\|^2 + \lambda\mathfrak{C}(k),\ k = 1, \ldots, K \\
\Rightarrow \mathbf{o}_1(n) &= \mathbf{W}_1^{\mathsf{T}}\underline{\mathbf{x}}(n) + \mathbf{b}_1
\end{aligned} \tag{2.20}$$

o que considera as seguintes definições:

$$\mathbf{W}_{1(M+1\times K)} \triangleq \begin{bmatrix} -2[c_1]_1 & -2[c_2]_1 & \cdots & -2[c_K]_1 \\ -2[c_1]_2 & -2[c_2]_2 & \cdots & -2[c_K]_2 \\ \vdots & \vdots & & \vdots \\ -2[c_1]_M & -2[c_2]_M & \cdots & -2[c_K]_M \\ 1 & 1 & \cdots & 1 \end{bmatrix},$$

$$\underline{\mathbf{x}}_{(M+1\times 1)} \triangleq \begin{bmatrix} \mathbf{x} \\ \|\mathbf{x}\|^2 \end{bmatrix}, \ \mathbf{b}_{1(K\times 1)} \triangleq \begin{bmatrix} \|\mathbf{c}_1\|^2 + \lambda\mathfrak{C}(1) \\ \|\mathbf{c}_2\|^2 + \lambda\mathfrak{C}(2) \\ \vdots \\ \|\mathbf{c}_K\|^2 + \lambda\mathfrak{C}(K) \end{bmatrix}. \tag{2.21}$$

A segunda camada da rede SOM deve comparar os escalares $[o_1]_k$ e selecionar o índice correspondente ao argumento mínimo:

$$j = \arg\min_k [o_1]_k, \ k = 1, \ldots, K, \tag{2.22}$$

de modo que a operação de codificação fica completa. A operação da Equação (2.22) é conhecida em inglês por *Winner Takes All* (WTA).

Capítulo 3

Quantização Vetorial Interpolativa

O projeto de ECVQ foi discutido no Capítulo 2 com a apresentação do GLA e a implementação por redes SOM. Foi visto que esse esquema é conhecido como "quantização vetorial por busca completa", uma vez que a codificação requer o cálculo de K distâncias euclidianas mais a operação WTA para determinar a menor distância, lembrando que K representa o tamanho do dicionário. Esse paradigma está relacionado a uma complexidade não-desprezível [11], de modo que existem linhas de pesquisa que se comprometem a desenvolver quantização vetorial tendo como âmago operações matemáticas que culminem em uma menor complexidade. A Quantização Vetorial Interpolativa (*Interpolative Vector Quantization* – IVQ) [7] surge nesse contexto e pode ser pensada como uma alternativa à ECVQ, em cenários em que há restrição de complexidade.

O presente capítulo inicia-se com o objetivo de introduzir IVQ (Seção 3.1). Ainda na Seção 3.1 e na conseguinte Seção 3.2, o codificador IVQ é especificado. A Seção 3.3 apresenta esquemas para implementação IVQ e a Seção 3.4 conclui o capítulo com a apresentação do método de projeto IVQ por meio de otimizações algorítmicas propostas.

3.1 O Paradigma da Quantização Vetorial Interpolativa

A operação de um quantizador Q que implementa IVQ difere-se da versão ECVQ pela definição de seu codificador. Na codificação IVQ, há a inclusão do extrator de características ψ, que desempenha a função de mapear os dados de entrada, $\mathbf{x}(n)$, em vetores $\breve{\mathbf{x}}(n)$, nos quais são realçadas características do sinal que de alguma forma permitam simplificar

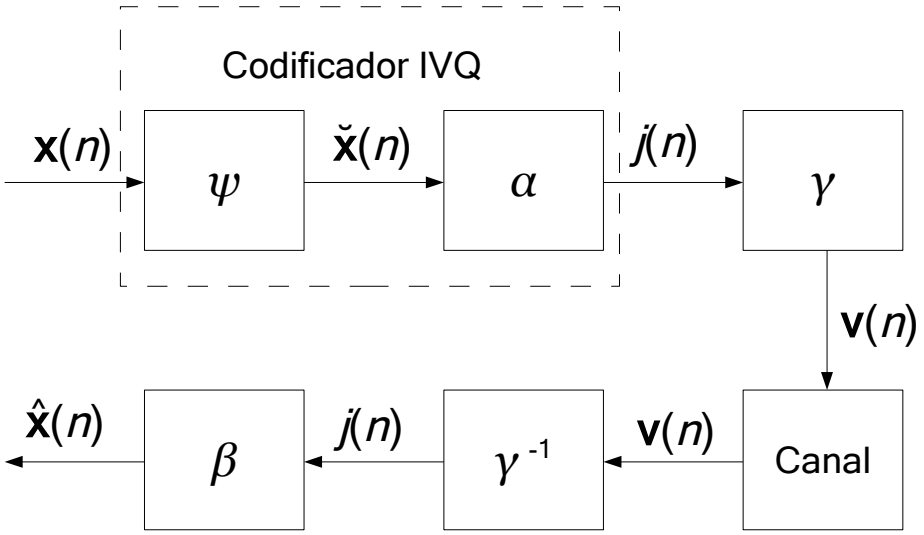


Figura 3.1: Esquema IVQ com restrição de entropia.

a partição do espaço vetorial que define o codificador. Neste trabalho, será suficiente definir o mapeamento ψ por:[1]

$$\psi : \mathbb{R}^M \rightarrow \mathbb{R}^M. \tag{3.1}$$

O sinal resultante, $\breve{\mathbf{x}}(n)$, é então submetido ao operador α, que mantém seu mapeamento definido pela Equação (2.2). Dessa forma, a codificação IVQ é definida pela operação serial

$$j(n) = \alpha \circ \psi(\mathbf{x}(n)) \triangleq \alpha\big(\psi(\mathbf{x}(n))\big). \tag{3.2}$$

O esquema de IVQ para o caso generalizado em que há restrição de entropia assume o formato ilustrado pela Figura 3.1.

As principais operações matemáticas empregadas no extrator de características ψ e conhecidas na literatura são: Análise de Componentes Principais (*Principal Component Analysis* – PCA), Interpolação Convencional (*Conventional Interpolation*) e Predição e Reconstrução por Momentos [7]. A PCA é tradicionalmente sub-dividida em suas versões, Linear (LPCA) e Não-Linear (*Kernel* PCA – KPCA). Neste trabalho serão analisadas as operações LPCA e KPCA. Considerando o cenário em que ψ é matematicamente imple-

[1]Uma abordagem mais genérica para IVQ prevê o mapeamento $\psi : \mathbb{R}^{M_1} \rightarrow \mathbb{R}^{M_2}$, em que M_2 pode ser maior ou menor do que M_1. Portanto, este trabalho utiliza o caso particular de $M_2 = M_1 = M$. A justificativa para tal estratégia refere-se à aplicação desta teoria em processamento de imagens, neste trabalho. Esse assunto será retomado na Seção 4.1.

mentado por PCA, a extração de características pode ser interpretada como a diagonalização da matriz de autocovariância do sinal $\mathbf{x}(n)$,

$$\mathbf{C} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}(n)-\bar{\mathbf{x}})(\mathbf{x}(n)-\bar{\mathbf{x}})^{\mathsf{T}},\ \text{para}\ \bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}(n), \tag{3.3}$$

significando que o sinal $\breve{\mathbf{x}}(n)$ pode ser encarado como M subsinais desacoplados entre si. Dessa forma, é admissível que cada sinal $\breve{x}_m(n)$ seja independentemente codificado, através de SQ. Essa simplificação diminui drasticamente a complexidade do operador α, quando comparado à versão ECVQ. A reconstrução, por outro lado, continua sendo realizada através de um dicionário, conforme a Equação (3.10).

Resgatando sua definição, SQ é o caso particular da Equação (2.1), no qual $M_{\text{SQ}} = 1$. Considere a m-ésima entrada do sinal, $\breve{x}_m(n)$. A codificação SQ associada é especificada pelo operador α_m, inicialmente função da quantidade de partições K_m. Uma vez que a codificação atuará na reta real, ela pode ser interpretada como o particionamento da reta em K_m regiões. Um esquema de partição que define o operador α_m pode ser formulado, sem perda de generalização, por

$$\mathbb{R} \leftrightarrow \big(-\infty,\ [t_m]_1\big]\ \cup\ \big([t_m]_1,\ [t_m]_2\big]\ \cup\ \ldots\ \cup\ \big([t_m]_{T_m},\ \infty\big), \tag{3.4}$$

que se associa ao vetor de limiares de decisão $\mathbf{t}_m$, de tamanho $T_m = K_m - 1$. Levando em conta as M dimensões, o vetor aumentado $\underline{\mathbf{t}}$ de tamanho T que representa todos os limiares de decisão é dado por

$$\begin{aligned} \underline{\mathbf{t}}_{T\times 1} &\triangleq \begin{bmatrix} \mathbf{t}_1^{\mathsf{T}} & \cdots & \mathbf{t}_M^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}}, \\ T &= \sum_{m=1}^{M} T_m. \end{aligned} \tag{3.5}$$

Com a finalidade de analisar a que intervalo da reta real pertence uma entrada $\breve{x}_m(n)$, introduz-se a função "degrau unitário", $\mathfrak{s}(\cdot)$. Para o caso escalar, $y \in \mathbb{R}$, $\mathfrak{s}(y)$ é definida como

$$\mathfrak{s}(y) \triangleq \begin{cases} 1, & y \geq 0 \\ 0, & y < 0, \end{cases} \tag{3.6}$$

que permite obter a saída do operador α_m, $j_m(n)$,

$$\begin{aligned} &\left[\underline{j}_m\right]_k (n) = \mathfrak{s}(\breve{x}_m(n) - [t_m]_k),\ k = 1, \ldots, T_m, \\ &\Rightarrow j_m(n) \triangleq \alpha_m(\breve{\mathbf{x}}(n)) = \sum_{k=1}^{T_m}\left[\underline{j}_m\right]_k (n). \end{aligned} \tag{3.7}$$

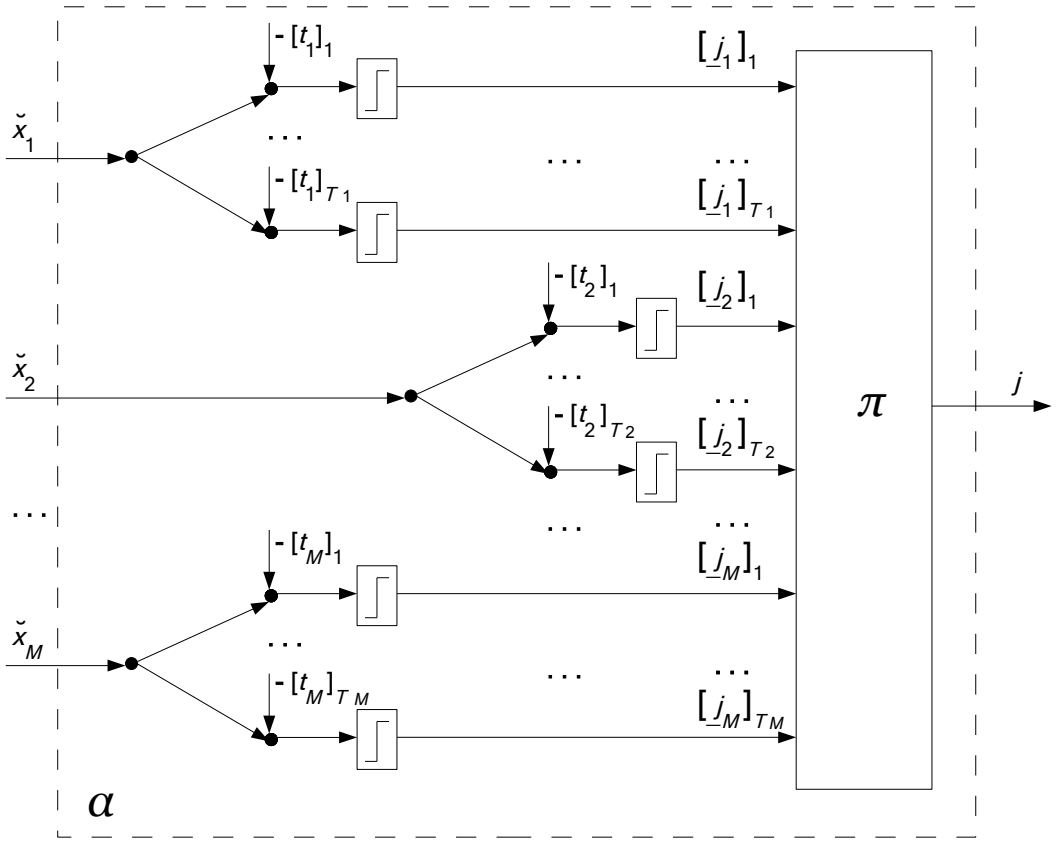


Figura 3.2: Operador α em IVQ.

O esquema de α para IVQ que leva em consideração a operação SQ formulada na Equação (3.7) é ilustrado pela Figura 3.2. Resgatando a definição do mapeamento α, dado pela Equação (2.2), observa-se que o seu sinal de saída $j(n)$ não requer em seu cálculo o uso dos sinais $j_m(n)$, bastando as suas versões aumentadas, dadas pelos vetores binários $\underline{\mathbf{j}}_m(n) \in \{0,1\}^{T_m}$. Definindo o vetor

$$\underline{\mathbf{j}}_{T\times 1} \triangleq \left[\begin{array}{ccc} \underline{\mathbf{j}}_1^{\mathsf{T}} & \cdots & \underline{\mathbf{j}}_M^{\mathsf{T}} \end{array} \right]^{\mathsf{T}}, \tag{3.8}$$

a operação $j(n) = \pi\left(\underline{\mathbf{j}}(n)\right)$ determina o sinal $j(n)$,

$$j(n) = \pi\left(\underline{\mathbf{j}}(n)\right) \triangleq \sum_{m=1}^{M} \sum_{k=1}^{T_m} \left(\left[\underline{j}_m\right]_k \prod_{m'=1}^{m-1} K_{m'} \right). \tag{3.9}$$

A operação π trata-se de uma função bijetora, que associa cada combinação possível para $\underline{\mathbf{j}}(n)$ a um índice do conjunto $\{1,\ldots,K\}$. Com isso, a descrição do codificador IVQ é concluída.

O dicionário $\mathcal{C}$ terá tamanho máximo $K = \prod_{m=1}^{M} K_m$,[2] o que permite submeter o sinal $j(n)$ a um esquema VLC (operador γ). Com isso, é garantido que a taxa transmitida

[2]Pode ocorrer que alguma palavra $\mathbf{c}_k$ do dicionário tenha probabilidade $p_k = 0$.

pelo canal obedece à relação da Equação (2.13), de modo que a operação IVQ é de fato quantização vetorial, embora em última análise seja implementada por concatenações de SQ. A decodificação IVQ mantém sua definição dada pela Condição do Centróide discutida na Seção 2.3, redefinida aqui devido à notação empregada para o codificador IVQ:

$$\mathbf{c}_k = \frac{\sum_n \big(\mathbf{x}(n) \mid \alpha \circ \psi(\mathbf{x}(n)) = k\big)}{\mathrm{card}\big(\mathbf{x}(n) \mid \alpha \circ \psi(\mathbf{x}(n)) = k\big)},\ k = 1, \ldots, K. \tag{3.10}$$

A operação de codificação definida pela Equação (3.10) possui uma importante implicação: a decodificação não precisa ser realizada no espaço de características, ou seja, a decodificação é efetuada por $\hat{\mathbf{x}}(n) = \beta(j(n))$, em detrimento de uma hipotética operação $\hat{\mathbf{x}}(n) = \psi^{-1} \circ \beta(j(n))$. Em outras palavras, não é necessário conhecer a operação ψ^{-1}, que pode até mesmo não existir. De fato, no caso em que ψ é especificado por KPCA, não convém determinar o mapeamento ψ^{-1}, que pode ser matematicamente complexo.

3.2 Análise de Componentes Principais

Neste trabalho, a implementação matemática de ψ será através de PCA. Através de PCA, o operador ψ é calculado analiticamente, objetivando a diagonalização da matriz de autocorrelação $\mathbf{C}$ (definida na Equação (3.3)). Primeiramente, será discutido o cálculo de ψ pelo caso mais simples, LPCA. Em seguida, a análise será voltada para o caso não-linear, KPCA.

3.2.1 O Caso Linear

Para implementação LPCA, o operador ψ realiza a operação linear $\breve{\mathbf{x}}(n) = \mathbf{V}^{\mathsf{T}}\mathbf{x}(n)$, em que $\mathbf{V}_{M\times M}$ é a matriz de transformação que diagonaliza $\mathbf{C}$:

$$\mathbf{C} = \mathbf{V}^{\mathsf{T}}\mathbf{R}\mathbf{V}. \tag{3.11}$$

A determinação de $\mathbf{V}$ se dá através da solução do problema de autovalores

$$\rho_m \mathbf{v}_m = \mathbf{C}\mathbf{v}_m \tag{3.12}$$

para $m = 1, \ldots, M$, $\rho_m \geq 0$ e $\mathbf{R} = \mathrm{diag}\{\rho_1, \ldots, \rho_M\}$. Os autovetores $\mathbf{v}_m \in \mathbb{R}^M - \mathbf{0}$ definem $\mathbf{V}$:

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_M \end{bmatrix}. \tag{3.13}$$

3.2.2 O Caso Não-Linear

Esquemas IVQ implementados por KPCA tendem a superar o desempenho de esquemas LPCA IVQ e se aproximam do caso ECVQ, uma vez que o operador ψ relacionado a KPCA gera um codificador IVQ que produz o efeito de partição não-linear do espaço $\mathbb{R}^M$ relacionado aos dados originais, $\mathbf{x}(n)$. Isso aproxima a partição do codificador IVQ ao formato ótimo de polígonos de Voronoi, como no caso do ECVQ.

Em KPCA, ao invés de diagonalizar $\mathbf{C}$, deseja-se diagonalizar uma matriz $\mathbf{C}'$ que é a autocorrelação dos dados $\mathbf{q}(n) = \phi(\mathbf{x}(n))$,

$$\mathbf{C}' = \frac{1}{N}\sum_{n=1}^{N}\mathbf{q}(n)\mathbf{q}(n)^{\mathsf{T}}. \tag{3.14}$$

O novo mapa $\phi : \mathbb{R}^M \to \mathbb{R}^I$ é uma operação complexa, que realça características que o mapeamento LPCA não é capaz de separar. Essa propriedade advém do fato de I ser um número elevado, possivelmente infinito. O sinal $\breve{\mathbf{x}}(n) = \psi(\mathbf{x}(n))$ passa então a ser dado por $\breve{\mathbf{x}}(n) = \mathbf{V}^{\mathsf{T}}\mathbf{q}(n)$, onde $\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_I \end{bmatrix}$ é a matriz que diagonaliza $\mathbf{C}'$, o que define o problema de autovalores

$$\mathbf{C}'\mathbf{v}_i = \rho_i\mathbf{v}_i,\ i = 1, \ldots, I. \tag{3.15}$$

A tarefa de solucionar a Equação (3.15) pode ser difícil, sobretudo se I tender ao infinito. Ao invés de calcular $\mathbf{v}_i$, recorre-se ao artifício de associar o vetor $\mathbf{v}_i$ a uma combinação linear dos vetores $\mathbf{q}(n)$,

$$\mathbf{v}_i = \frac{1}{N}\mathbf{Q}\mathbf{a}_i, \tag{3.16}$$

em que $\mathbf{Q}_{I\times N} = \begin{bmatrix} \mathbf{q}(1) & \cdots & \mathbf{q}(N) \end{bmatrix}$ e $\mathbf{a}_{i(N\times 1)}$ é o vetor que especifica a combinação linear. Através da substituição da Equação (3.16) em (3.15), chega-se a

$$\mathbf{C}'\mathbf{Q}\mathbf{a}_i = \rho_i\mathbf{Q}\mathbf{a}_i. \tag{3.17}$$

Multiplicando Equação (3.17) por $\mathbf{Q}^{\mathsf{T}}$ em ambos os lados da igualdade,

$$\begin{aligned} \mathbf{Q}^{\mathsf{T}} \cdot \frac{1}{N}\mathbf{Q}\mathbf{Q}^{\mathsf{T}} \cdot \mathbf{Q}\mathbf{a}_i &= \rho_i\mathbf{Q}^{\mathsf{T}}\mathbf{Q}\mathbf{a}_i \\ \Rightarrow \mathbf{G}^2\mathbf{a}_i &= N\rho_i\mathbf{G}\mathbf{a}_i \end{aligned} \tag{3.18}$$

para $\mathbf{G}_{N\times N} \triangleq \mathbf{Q}^{\mathsf{T}}\mathbf{Q}$. Assim, é possível obter $\mathbf{a}_i$ através da resolução do problema de autovalores

$$\mathbf{G}\mathbf{a}_i = N\rho_i\mathbf{a}_i, \tag{3.19}$$

para $i = 1, \ldots, N$. Matematicamente, isso significa que somente N componentes principais do espaço de características dado por ϕ serão consideradas. Embora a dimensão de $\mathbf{G}$ seja relativamente reduzida, observa-se que a especificação do elemento $G_{i_1 i_2} = \mathbf{q}(i_1)^\mathsf{T}\mathbf{q}(i_2)$ não permite determinar a matriz, uma vez que o sinal $\mathbf{q}(n)$ é desconhecido e pode ter dimensão I infinita. Para o cálculo de $\mathbf{G}$, introduz-se o conceito do operador *kernel*, $\kappa : (\mathbb{R}^M, \mathbb{R}^M) \rightarrow \mathbb{R}$, que avalia o produto interno em questão através de uma função não-linear κ:

$$G_{i_1 i_2} = \mathbf{q}(i_1)^\mathsf{T}\mathbf{q}(i_2) = \kappa(\mathbf{x}(i_1), \mathbf{x}(i_2)). \tag{3.20}$$

Definindo a matriz $\mathbf{A} = \left[\begin{array}{ccc} \mathbf{a}_1/\sqrt{\rho_1} & \cdots & \mathbf{a}_N/\sqrt{\rho_N} \end{array} \right]$, já é possível determinar o mapa KPCA $\breve{\mathbf{x}}(n) = \psi(\mathbf{x}(n))$ como:

$$\begin{aligned} \breve{\mathbf{x}}(n) &= \mathbf{V}^\mathsf{T}\mathbf{q}(n) \\ &= (\mathbf{q}^\mathsf{T}(n)\mathbf{V})^\mathsf{T} \\ &= (\mathbf{q}^\mathsf{T}(n)\mathbf{Q}\mathbf{A})^\mathsf{T} \\ &= \Big(\left[\begin{array}{ccc} \kappa(\mathbf{x}(n), \mathbf{x}(1)) & \cdots & \kappa(\mathbf{x}(n), \mathbf{x}(N)) \end{array} \right] \mathbf{A} \Big)^\mathsf{T}. \end{aligned} \tag{3.21}$$

Definindo o vetor de produtos internos

$$\mathbf{z}(n) = \left[\begin{array}{ccc} \kappa(\mathbf{x}(n), \mathbf{x}(1)) & \cdots & \kappa(\mathbf{x}(n), \mathbf{x}(N)) \end{array} \right]^\mathsf{T}, \tag{3.22}$$

a Equação (3.21) se resume a

$$\breve{\mathbf{x}}(n) = \psi(\mathbf{x}(n)) = \mathbf{A}^\mathsf{T}\mathbf{z}(n). \tag{3.23}$$

Projeção em Treliça

Da maneira como o operador ψ está definido pela Equação (3.23), sua implementação prática é difícil, em termos de complexidade. A matriz $\mathbf{A}$ tem tamanho $N \times N$, onde N é o número de realizações do sinal, que pode ser elevado. Com isso, o operador ψ se torna complexo, além de produzir um sinal $\breve{\mathbf{x}}(n)$ de dimensão $N \times 1$. Isso influenciaria também no operador α, que teria sua complexidade aumentada, uma vez que o número de esquemas SQ passaria a ser N. Como se não bastasse, para o cálculo, por exemplo, de $\mathbf{z}(1)$ (confira pela Equação (3.22)), seria necessária desde a primeira amostra do sinal, $\mathbf{x}(1)$, até a última, $\mathbf{x}(N)$, o que provocaria um atraso de grupo de tamanho N ao sistema de quantização.

Por tudo isso, faz-se necessário reduzir o tamanho da matriz $\mathbf{A}$. Resgatando o fato de que $\mathbf{A}$ é formada pelos vetores $\mathbf{a}_i$, que especificam o autovetor $\mathbf{v}_i$ como combinação linear de $\mathbf{q}(n)$ (Equação (3.16)), nota-se que é possível redefinir $\mathbf{A}$ levando em consideração que há a possibilidade de expressar $\mathbf{v}_i$ como combinação linear de vetores que não sejam necessariamente $\mathbf{q}(n)$, mas que de alguma forma sejam representativos do espaço de características relacionados a $\mathbf{q}(n)$. Uma vez que o espaço de características é definido pela matriz $\mathbf{G}$, que é função apenas de $\mathbf{x}(n)$ (confira pela Equação (3.20)), o problema se resume a determinar uma matriz $\mathbf{L} = \left[\begin{array}{ccc} \mathbf{l}_1 & \cdots & \mathbf{l}_L \end{array} \right]$ de vetores que representem o espaço de $\mathbf{x}(n)$, $\mathbb{R}^M$.

Na literatura [9], a matriz $\mathbf{L}$ proposta é a menor treliça para a dimensão M, de tamanho $M+1$:

$$\mathbf{L}_{M\times(M+1)} = \left[\begin{array}{cccc} \mathbf{l}_1 & \mathbf{l}_2 & \cdots & \mathbf{l}_{M+1} \end{array} \right] = \left[\begin{array}{cc} \mathbf{0}_{M\times 1} & \mathbf{I}_M \end{array} \right]. \tag{3.24}$$

Dessa forma, a matriz $\mathbf{G}$ passa a ter tamanho $(M+1)\times(M+1)$ e seus elementos passam a ser dados por $G_{i_1 i_2} = \kappa(\mathbf{l}(i_1), \mathbf{l}(i_2))$. Nesse ponto, a matriz $\mathbf{G}$ é substituída por sua versão $\mathbf{G}'$, que centraliza em zero [21] o espaço de características,

$$\mathbf{G}' = \mathbf{G} - \mathbf{1}'\mathbf{G} - \mathbf{G}\mathbf{1}' + \mathbf{1}'\mathbf{G}\mathbf{1}', \tag{3.25}$$

onde $\mathbf{1}' = \mathbf{1}_{M+1}/(M+1)$. Como conseqüência imediata, o problema de autovalores da Equação (3.19) aplicado a $\mathbf{G}'$ passa a ter como solução $M+1$ autovalores, sendo $\rho_{M+1} = 0$, e $M+1$ autovetores $\mathbf{a}_i$. Pelo fato de $\mathbf{a}_{M+1}$ estar relacionado ao autovalor nulo ρ_{M+1}, a matriz de autovetores $\mathbf{A}' = \left[\begin{array}{ccc} \mathbf{a}_1 & \cdots & \mathbf{a}_M \end{array} \right]$ contém os primeiros M autovetores e assume a estrutura $(M+1)\times M$. O vetor de produtos internos $\mathbf{z}(n)$ da Equação (3.22) é redefinido para

$$\mathbf{z}(n) = \left[\begin{array}{ccc} \kappa(\mathbf{x}(n), \mathbf{l}_1) & \cdots & \kappa(\mathbf{x}(n), \mathbf{l}_{M+1}) \end{array} \right]^{\mathsf{T}} \tag{3.26}$$

e, devido à operação de centralização em zero [21], assume o formato definitivo

$$\mathbf{z}'(n) = (\mathbf{I}_{M+1} - \mathbf{1}')\mathbf{z}(n) + \mathbf{1}'\mathbf{G}^{\mathsf{T}}\mathbf{1}'' - \mathbf{G}^{\mathsf{T}}\mathbf{1}'', \tag{3.27}$$

onde $\mathbf{1}'' = 1/(M+1)\cdot\mathbf{1}_{(M+1)\times 1}$.

Finalmente, o operador ψ passa a realizar o mapeamento padrão $\mathbb{R}^M \rightarrow \mathbb{R}^M$ e sua operação se resume a

$$\breve{\mathbf{x}}(n) = \psi(\mathbf{x}(n)) = [\mathbf{A}']^{\mathsf{T}}\mathbf{z}'(n). \tag{3.28}$$

3.3 Percéptrons Multicamadas

Enquando o codificador ECVQ é implementado por uma Rede Neural do tipo SOM, a codificação IVQ, definida pela operação $j(n) = \alpha \circ \psi(\mathbf{x}(n))$, é implementada pela estrutura de Redes Neurais conhecida como Percéptrons Multicamadas (*Multilayer Perceptrons* – MLP). A operação da l-ésima camada de uma rede MLP de L camadas é definida por [13]

$$\begin{aligned} \mathbf{u}_l(n) &= \mathbf{W}_l^{\mathsf{T}} \mathbf{o}_{l-1}(n) + \mathbf{b}_l, \\ \mathbf{o}_l(n) &= f_l(\mathbf{u}_l(n)), \end{aligned} \tag{3.29}$$

onde a função de ativação $f_l(\cdot)$ pode ser linear ou não-linear. Os vetores $\mathbf{o}_0$ e $\mathbf{o}_L$ são, respectivamente, os sinais de entrada e de saída da rede MLP. Desde que respeitada a compatibilidade dimensional, os tamanhos da matriz de sinapses $\mathbf{W}_l$ e do vetor de polarização $\mathbf{b}_l$ são livres e possivelmente diferentes para cada camada da rede. Como conseqüência, os sinais de saída e de entrada da rede MLP podem ter dimensões diferentes. De maneira análoga, a função de ativação não requer homogeneidade ao longo das camadas.

Devido a essa flexibilidade, uma infinidade de operações matemáticas podem ser implementadas por estruturas MLP. No caso particular deste trabalho, redes MLP serão utilizadas para implementar todas as variações da operação de codificação IVQ, que a princípio se referem às diferentes análises PCA empregadas para a determinação de ψ e se referem também ao operador α. Nesse ponto, vale salientar que a operação VLC definida pelo mapeamento γ da Equação (2.11) não requer como entrada necessariamente $j(n)$. Na verdade, em casos práticos, o operador π (Equação (3.9)) é dispensável, pois se trata de uma função biunívoca cujo sinal $j(n) = \pi\left(\underline{\mathbf{j}}(n)\right)$ é submetido a uma nova operação biunívoca, que é a relacionada ao VLC. Dessa forma, um esquema VLC simplificado $\mathbf{v}(n) = \gamma \circ \pi\left(\underline{\mathbf{j}}(n)\right)$ é suficiente para implementações práticas e requer que a saída da rede MLP, $\mathbf{o}_L(n)$, seja dada simplesmente por $\underline{\mathbf{j}}(n)$.

Nas próximas Seções, redes MLP para a implementação de LPCA e KPCA são especificadas.

3.3.1 Rede LPCA MLP

Para a implementação do codificador LPCA, é suficiente uma rede MLP de tamanho $L = 2$ camadas. A operação de codificação IVQ é definida pela Equação (3.2). No caso LPCA, o operador ψ é definido pela operação $\breve{\mathbf{x}}(n) = \mathbf{V}^{\mathsf{T}}\mathbf{x}(n)$, com $\mathbf{V}^{\mathsf{T}}$ definido pela Equação (3.13), e implementado na primeira camada.

Na segunda camada, o operador α definido pelas Equações (3.7) e (3.9) é parcialmente implementado, obtendo como resultado o sinal $\underline{\mathbf{j}}(n)$ (Equação (3.7)). A especificação das entidades MLP da segunda camada, $\mathbf{W}_2$, $\mathbf{b}_2$ e $f_2(\cdot)$, são calculadas lembrando que o sinal de saída da primeira camada $\mathbf{o}_1(n) = \breve{\mathbf{x}}(n)$, o que implica em

$$\begin{aligned}
\mathbf{o}_2(n) = \underline{\mathbf{j}}(n) &= \mathfrak{s}\left(\begin{bmatrix} \mathbf{1}_{1\times T_1} & & \\ & \ddots & \\ & & \mathbf{1}_{1\times T_M} \end{bmatrix}^{\mathsf{T}} \breve{\mathbf{x}}(n) - \underline{\mathbf{t}}\right) \\
&= f_2(\mathbf{u}_2(n)) \\
\Rightarrow \mathbf{u}_2(n) &= \begin{bmatrix} \mathbf{1}_{1\times T_1} & & \\ & \ddots & \\ & & \mathbf{1}_{1\times T_M} \end{bmatrix}^{\mathsf{T}} \breve{\mathbf{x}}(n) - \underline{\mathbf{t}} \\
&= \mathbf{W}_2^{\mathsf{T}}\mathbf{o}_1(n) + \mathbf{b}_2,
\end{aligned} \tag{3.30}$$

para $f_2(\cdot) = \mathfrak{s}(\cdot)$. Dessa forma, a especificação da rede LPCA MLP é apresentada na Tabela 3.1 e assume o formato ilustrado pela Figura 3.3.

3.3.2 Redes KPCA MLP

A implementação do codificador KPCA por rede MLP exige $L = 3$ camadas para sua operacionalização. Em suma, a primeira camada realiza a projeção em treliça da entrada $\mathbf{x}(n)$ na matriz $\mathbf{L}$, para então calcular o produto interno $\mathbf{z}(n)$ no espaço de características, conforme a Equação (3.26). A segunda camada realiza a centralização em zero, conforme a Equação (3.27), e a projeção da Equação (3.28) para a obtenção do sinal $\breve{\mathbf{x}}(n)$, que é, em seguida, submetido à terceira camada, onde a operação α é realizada.

Em particular, a primeira camada calcula os produtos internos no espaço de características através do operador κ, conforme a Equação (3.26) e a sua definição, vista na Equação (3.20). No entanto, até o momento, não se demonstrou qual é de fato a operação realizada

Tabela 3.1: Especificação da Rede LPCA MLP

Número de camadas $L = 2$

Sinal de entrada $\mathbf{o}_0(n) = \mathbf{x}(n)$

Sinal de saída $\mathbf{o}_L(n) = \underline{\mathbf{j}}(n)$

Especificação das camadas

Entidade MLP	Descrição	Tamanho
$\mathbf{W}_1$	$\mathbf{V}$	$M \times M$
$\mathbf{b}_1$	$\mathbf{0}$	$M \times 1$
$\mathbf{W}_2$	$\begin{bmatrix} \mathbf{1}_{1\times T_1} & & \\ & \ddots & \\ & & \mathbf{1}_{1\times T_M} \end{bmatrix}$	$M \times T$
$\mathbf{b}_2$	$-\underline{\mathbf{t}}$	$T \times 1$

Funções de ativação

$$f_1(\mathbf{u}_1) = \mathbf{u}_1$$
$$f_2(\mathbf{u}_2) = \mathfrak{s}(\mathbf{u}_2)$$

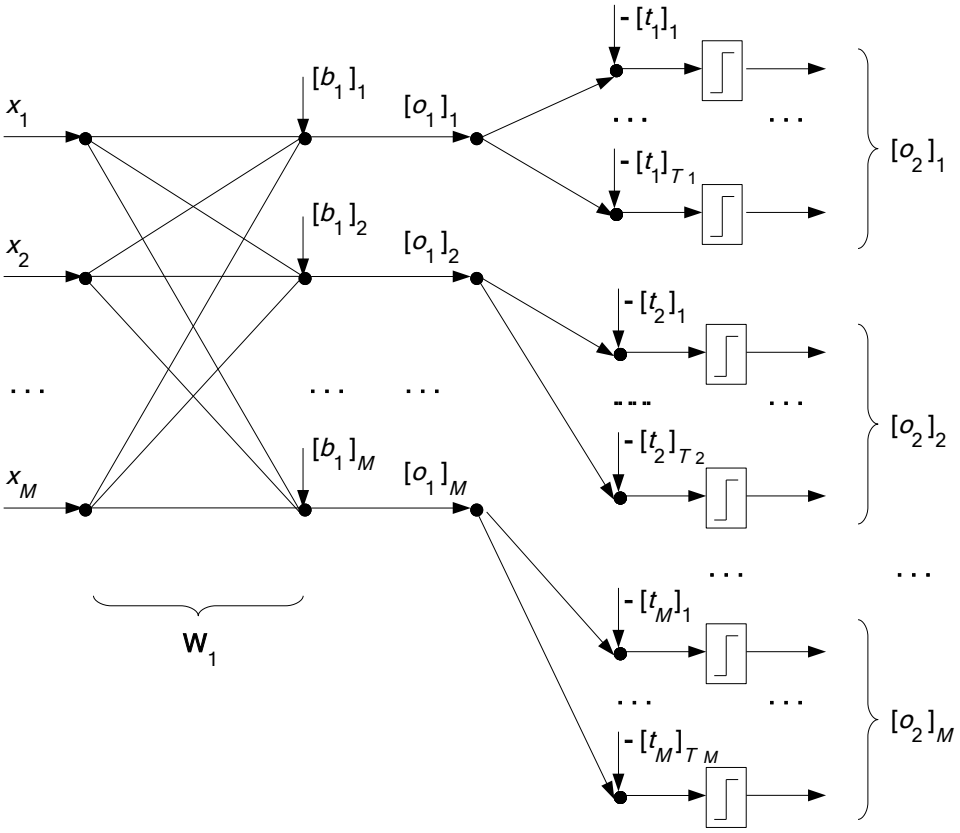


Figura 3.3: Rede MLP que implementa LPCA.

por κ. Existem diferentes operações κ, capazes de realçar características diferenciadas do sinal. Neste trabalho, serão abordadas *três* diferentes operações de *kernel*. Considere dois vetores quaisquer $\{\mathbf{x}, \mathbf{l}\} \in \mathbb{R}^M$. A operação $\kappa(\mathbf{x}, \mathbf{l})$ poderá assumir as seguintes fórmulas:

1. **Tangente Hiperbólica:** (*Hiperbolic Tangent* – HT)

$$\kappa(\mathbf{x}, \mathbf{l}) = \tanh(g\mathbf{x}^\mathsf{T}\mathbf{l} + \theta) \tag{3.31}$$

   onde g e θ são, respectivamente, as constantes reais de ganho e polarização.

2. **Função de Base Radial:** (*Radial Basis Function* – RBF)

$$\kappa(\mathbf{x}, \mathbf{l}) = \exp\Big(-\frac{\|\mathbf{x} - \mathbf{l}\|^2}{2\sigma^2}\Big) \tag{3.32}$$

   onde σ^2 é a constante real positiva de variância.

3. **Polinomial:**

$$\kappa(\mathbf{x}, \mathbf{l}) = (\mathbf{x}^\mathsf{T}\mathbf{l})^r \tag{3.33}$$

   onde $r \in \mathbb{R}$ é a ordem, arbitrável.

Dessa forma, definem-se três codificadores e, por extensão, três redes MLP para KPCA distintas: HT MLP, RBF MLP e MLP Polinomial. Uma vez que a operação κ é realizada somente na primeira camada MLP, esta será tratada separadamente para cada caso MLP: HT MLP (Página 27), RBF MLP (Página 28) e MLP Polinomial (Página 30).

Independente da variação MLP, as estruturas das camadas 2 e 3 são calculadas do mesmo modo. A camada 2 deve realizar a operação de centralização em zero (Equação (3.27)) e a projeção da Equação (3.28), para assim implementar o operador ψ e produzir o sinal $\breve{\mathbf{x}}(n)$. Lembrando que o sinal de saída da camada 1, $\mathbf{o}_1(n)$, é o vetor de produtos internos $\mathbf{z}(n)$, o vetor de saída dessa camada, $\mathbf{o}_2(n)$, é calculado por

$$\begin{aligned}\mathbf{o}_2(n) = \breve{\mathbf{x}}(n) &= [\mathbf{A}']^\mathsf{T}\mathbf{z}' = [\mathbf{A}']^\mathsf{T}\big((\mathbf{I}_{M+1} - \mathbf{1}')\mathbf{o}_1(n) + \mathbf{1}'\mathbf{G}^\mathsf{T}\mathbf{1}'' - \mathbf{G}^\mathsf{T}\mathbf{1}''\big) \\ &= f_2(\mathbf{u}_2(n)) = \mathbf{u}_2(n)\end{aligned} \tag{3.34}$$

onde

$$\begin{aligned}\mathbf{u}_2(n) &= [\mathbf{A}']^\mathsf{T}\big((\mathbf{I}_{M+1} - \mathbf{1}')\mathbf{o}_1(n) + \mathbf{1}'\mathbf{G}^\mathsf{T}\mathbf{1}' - \mathbf{G}^\mathsf{T}\mathbf{1}'\big) \\ &= [\mathbf{A}']^\mathsf{T}(\mathbf{I}_{M+1} - \mathbf{1}')\mathbf{o}_1(n) + [\mathbf{A}']^\mathsf{T}(\mathbf{1}'\mathbf{G}^\mathsf{T}\mathbf{1}'' - \mathbf{G}^\mathsf{T}\mathbf{1}'') \\ &= \mathbf{W}_2^\mathsf{T}\mathbf{o}_1(n) + \mathbf{b}_2\end{aligned} \tag{3.35}$$

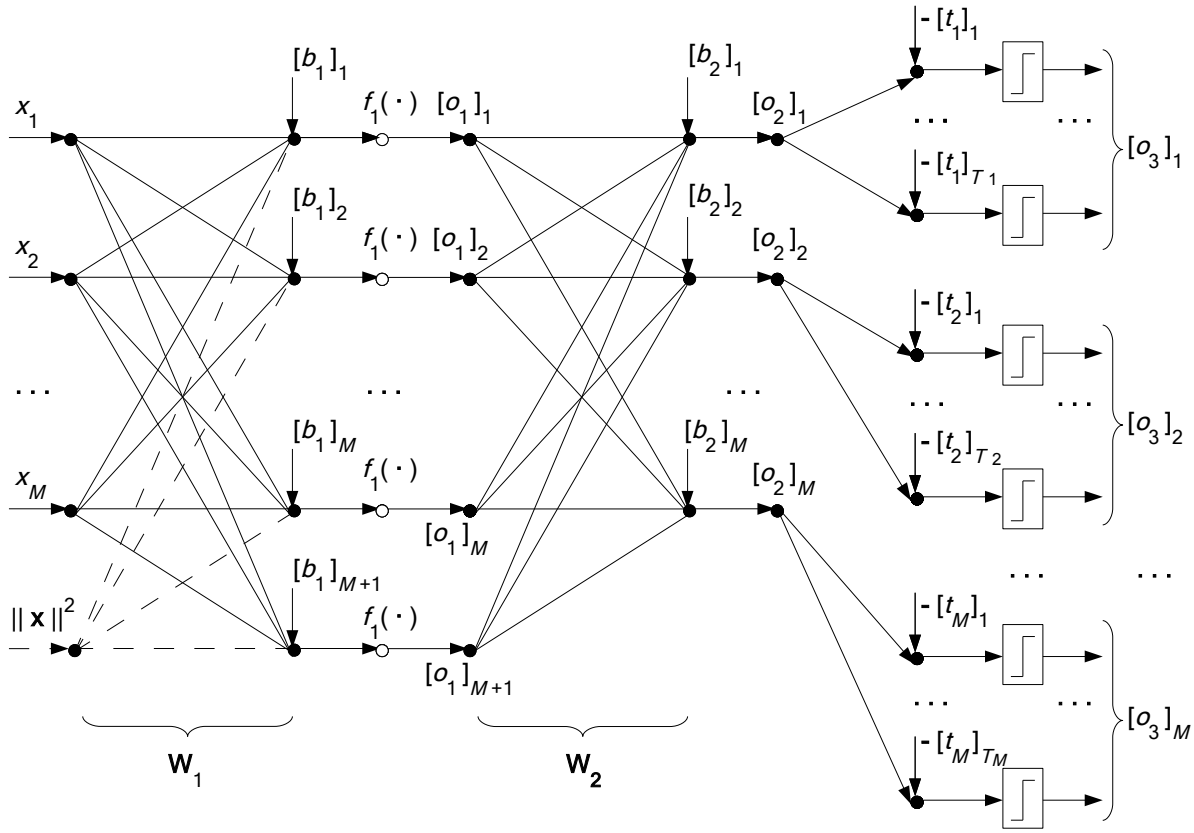


Figura 3.4: Rede MLP que implementa KPCA. A estrutura tracejada é encontrada somente no caso de implementação RBF.

para $\mathbf{W}_2 \triangleq (\mathbf{I}_{M+1} - \mathbf{1}')\mathbf{A}'$ e $\mathbf{b}_2 \triangleq [\mathbf{A}']^\mathsf{T}(\mathbf{1}'\mathbf{G}^\mathsf{T}\mathbf{1}'' - \mathbf{G}^\mathsf{T}\mathbf{1}'')$, lembrando que $\mathbf{1}' = \mathbf{1}_{M+1}/(M+1)$ e $\mathbf{1}'' = 1/(M+1) \cdot \mathbf{1}_{(M+1)\times 1}$.

A camada 3 para rede KPCA MLP é análoga à camada 2 do caso LPCA. Dessa forma, para sua especificação basta aplicar a Equação (3.30) para $\mathbf{o}_3(n)$. O esquema para redes KPCA MLP é ilustrado na Figura 3.4. Nessa figura, é possível notar variações na estrutura da camada 1, devido às diferentes operações κ empregadas. Nas próximas páginas, as peculiaridades de especificação da camada 1 para os diferentes casos KPCA serão analisadas.

Rede HT MLP

Resgatando o fato de que a operação da camada 1 deve projetar os dados $\mathbf{x}(n)$ em $\mathbf{L}$, para então calcular os *kernels*, deve-se, portanto, implementar a operação da Equação (3.26)

com o formato MLP da Equação (3.29). O vetor de saída $\mathbf{o}_{1(M+1\times 1)}$ deve ser dado por:

$$\begin{aligned}\mathbf{o}_1(n) = \mathbf{z}(n) &= \left[\begin{array}{ccc} \kappa(\mathbf{x},\mathbf{l}_1) & \cdots & \kappa(\mathbf{x},\mathbf{l}_{M+1}) \end{array}\right]^{\mathsf{T}} \\ &= f_1(\mathbf{u}_1(n))\end{aligned} \tag{3.36}$$

onde a função de ativação é definida por $f_1(\mathbf{u_1}) \triangleq \tanh(\mathbf{u_1})$ e é considerado que κ realiza a operação HT definida na Equação (3.31). Dessa forma, o vetor interno $\mathbf{u}_1(n)$ deve ser tal que

$$\begin{aligned} f(\mathbf{u}_1(n)) &= \left[\begin{array}{ccc} \kappa(\mathbf{x},\mathbf{l}_1) & \cdots & \kappa(\mathbf{x},\mathbf{l}_{M+1}) \end{array}\right]^{\mathsf{T}} \\ &= \left[\begin{array}{ccc} \tanh(g\mathbf{x}^{\mathsf{T}}(n)\mathbf{l}_1+\theta) & \cdots & \tanh(g\mathbf{x}^{\mathsf{T}}\mathbf{l}_{M+1}+\theta) \end{array}\right]^{\mathsf{T}} \\ &= \tanh\left[\begin{array}{ccc} g\mathbf{x}^{\mathsf{T}}(n)\mathbf{l}_1+\theta & \cdots & g\mathbf{x}^{\mathsf{T}}\mathbf{l}_{M+1}+\theta \end{array}\right]^{\mathsf{T}} \\ \Rightarrow \mathbf{u}_1(n) &= \left[\begin{array}{ccc} g\mathbf{x}^{\mathsf{T}}(n)\mathbf{l}_1+\theta & \cdots & g\mathbf{x}^{\mathsf{T}}(n)\mathbf{l}_{M+1}+\theta \end{array}\right]^{\mathsf{T}} \\ &= g\mathbf{L}^{\mathsf{T}}\mathbf{x}(n) + \theta\mathbf{1}_{(M+1)\times 1} \\ &= \mathbf{W}_1^{\mathsf{T}}\mathbf{x}(n) + \mathbf{b}_1 \end{aligned} \tag{3.37}$$

onde há as definições $\mathbf{W}_1 \triangleq g\mathbf{L}$ e $\mathbf{b}_1 \triangleq \theta\mathbf{1}_{(M+1)\times 1}$, que terminam por especificar completamente a camada 1. A tabela 3.2 descreve a rede HT MLP em termos de número de camadas; sinais de entrada e de saída; matrizes de sinapses; vetores de polarizações e funções de ativação.

Rede RBF MLP

No caso RBF, a operação κ é função da distância euclidiana entre $\mathbf{x}$ e $\mathbf{l}$. Dessa maneira, a camada 1 deverá implementar tal operação, de uma forma muito similar à camada 1 da implementação SOM para o codificador ECVQ (Seção 2.4). A exemplo do caso HT, o vetor de saída da camada 1 para o caso RBF, $\mathbf{o}_1(n)$, continua sendo dado dado pela Equação (3.36). A diferença está na definição da função de ativação $f_1(\mathbf{u}_1) \triangleq \exp(\mathbf{u}_1)$, implicando que o vetor interno $\mathbf{u}_1(n)$ é tal que

$$\begin{aligned} f(\mathbf{u}_1(n)) &= \left[\begin{array}{ccc} \kappa(\mathbf{x}(n),\mathbf{l}_1) & \cdots & \kappa(\mathbf{x}(n),\mathbf{l}_{M+1}) \end{array}\right]^{\mathsf{T}} \\ &= \exp\left(\left[\begin{array}{ccc} -\frac{\|\mathbf{x}(n)-\mathbf{l}_1\|^2}{2\sigma^2} & \cdots & -\frac{\|\mathbf{x}(n)-\mathbf{l}_{M+1}\|^2}{2\sigma^2} \end{array}\right]^{\mathsf{T}}\right) \\ \Rightarrow \mathbf{u}_1(n) &= \left[\begin{array}{ccc} -\frac{\|\mathbf{x}(n)-\mathbf{l}_1\|^2}{2\sigma^2} & \cdots & -\frac{\|\mathbf{x}(n)-\mathbf{l}_{M+1}\|^2}{2\sigma^2} \end{array}\right]^{\mathsf{T}} \end{aligned} \tag{3.38}$$

Tabela 3.2: Especificação da Rede HT MLP

Número de camadas $L = 3$

Sinal de entrada $\mathbf{o}_0(n) = \mathbf{x}(n)$

Sinal de saída $\mathbf{o}_L(n) = \underline{\mathbf{j}}(n)$

Especificação das camadas

Entidade MLP	Descrição	Tamanho
$\mathbf{W}_1$	$g\mathbf{L}$	$M \times (M+1)$
$\mathbf{b}_1$	$\theta\mathbf{1}$	$(M+1) \times 1$
$\mathbf{W}_2$	$(\mathbf{I}_{M+1} - \mathbf{1}')\mathbf{A}'$	$(M+1) \times M$
$\mathbf{b}_2$	$[\mathbf{A}']^\mathsf{T}(\mathbf{1}'\mathbf{G}^\mathsf{T}\mathbf{1}'' - \mathbf{G}^\mathsf{T}\mathbf{1}'')$	$M \times 1$
$\mathbf{W}_3$	$\begin{bmatrix} \mathbf{1}_{1\times T_1} & & \\ & \ddots & \\ & & \mathbf{1}_{1\times T_M} \end{bmatrix}$	$M \times T$
$\mathbf{b}_3$	$-\underline{\mathbf{t}}$	$T \times 1$

Funções de ativação

$$f_1(\mathbf{u}_1) = \tanh(\mathbf{u}_1)$$
$$f_2(\mathbf{u}_2) = \mathbf{u}_2$$
$$f_3(\mathbf{u}_3) = \mathfrak{s}(\mathbf{u}_3)$$

A descrição da m-ésima entrada de $\mathbf{u}_1(n)$, $[u_1]_m(n)$, se procede de maneira análoga à Equação (2.20) do caso SOM,

$$\begin{aligned}[u_1]_m(n) &= -\frac{1}{2\sigma^2}\|\mathbf{x}(n)-\mathbf{l}_m\|^2 \\ &= -\frac{1}{2\sigma^2}\left(\|\mathbf{x}(n)\|^2+\sum_{m'=1}^{M+1}\big(-2[l_m]_{m'}x_m(n)\big)+\|\mathbf{l}_{m'}\|^2\right),\end{aligned} \tag{3.39}$$

para $m=1,\ldots,M+1$. Definindo:

$$\begin{aligned}\mathbf{W}_{1(M+1)} &\triangleq -\frac{1}{2\sigma^2}\begin{bmatrix} -2[l_1]_1 & -2[l_2]_1 & \cdots & -2[l_{M+1}]_1 \\ -2[l_1]_2 & -2[l_2]_2 & \cdots & -2[l_{M+1}]_2 \\ \vdots & \vdots & & \vdots \\ -2[l_1]_M & -2[l_2]_M & \cdots & -2[l_{M+1}]_M \\ 1 & 1 & \cdots & 1 \end{bmatrix} \\ &= \frac{1}{\sigma^2}\begin{bmatrix} \mathbf{L} \\ -\frac{1}{2}\mathbf{1}_{1\times(M+1)} \end{bmatrix},\end{aligned}$$

$$\underline{\mathbf{x}}_{(M+1\times 1)} \triangleq \begin{bmatrix} \mathbf{x} \\ \|\mathbf{x}\|^2 \end{bmatrix},\ \mathbf{b}_{1((M+1)\times 1)} \triangleq -\frac{1}{2\sigma^2}\begin{bmatrix} \|\mathbf{l}_1\|^2 \\ \vdots \\ \|\mathbf{l}_{M+1}\|^2 \end{bmatrix}, \tag{3.40}$$

tem-se finalmente

$$\mathbf{u}_1(n) = \mathbf{W}_1^{\mathsf{T}}\underline{\mathbf{x}}(n)+\mathbf{b}_1. \tag{3.41}$$

A especificação da rede RBF MLP é apresentada na Tabela 3.3. O esquema RBF MLP é ilustrado pela Figura 3.4, onde as linhas tracejadas são consideradas e representam a inclusão da norma $\|\mathbf{x}(n)\|^2$, para a geração do vetor aumentado $\underline{\mathbf{x}}(n)$.

Rede MLP Polinomial

Devido à relativa simplicidade da operação κ para o caso KPCA Polinomial (Equação (3.33)), a camada 1 da rede MLP Polinomial deve simplesmente projetar o sinal $\mathbf{x}(n)$ na treliça $\mathbf{L}$ para então calcular κ. Por essa razão, a matriz de sinapses $\mathbf{W}_1=\mathbf{L}$, o vetor de polarização $\mathbf{b}_1=\mathbf{0}$ e a função de ativação é $f(\mathbf{u}_1)=(\mathbf{u}_1)^r$. Para uma visão geral, consultar Tabela 3.4.

Tabela 3.3: Especificação da Rede RBF MLP

Número de camadas $L = 3$

Sinal de entrada $\mathbf{o}_0(n) = \underline{\mathbf{x}}(n)$

Sinal de saída $\mathbf{o}_L(n) = \underline{\mathbf{j}}(n)$

Especificação das camadas

Entidade MLP	Descrição	Tamanho
$\mathbf{W}_1$	$\frac{1}{\sigma^2}\begin{bmatrix} \mathbf{L} \\ -\frac{1}{2}\mathbf{1}_{1\times(M+1)} \end{bmatrix}$	$(M+1)\times(M+1)$
$\mathbf{b}_1$	$-\frac{1}{2\sigma^2}\begin{bmatrix} \|\mathbf{l}_1\|^2 \\ \vdots \\ \|\mathbf{l}_{M+1}\|^2 \end{bmatrix}$	$(M+1)\times 1$
$\mathbf{W}_2$	$(\mathbf{I}_{M+1} - \mathbf{1}')\mathbf{A}'$	$(M+1)\times M$
$\mathbf{b}_2$	$[\mathbf{A}']^\mathsf{T}(\mathbf{1}'\mathbf{G}^\mathsf{T}\mathbf{1}'' - \mathbf{G}^\mathsf{T}\mathbf{1}'')$	$M\times 1$
$\mathbf{W}_3$	$\begin{bmatrix} \mathbf{1}_{1\times T_1} & & \\ & \ddots & \\ & & \mathbf{1}_{1\times T_M} \end{bmatrix}$	$M\times T$
$\mathbf{b}_3$	$-\underline{\mathbf{t}}$	$T\times 1$

Funções de ativação

$f_1(\mathbf{u}_1) = \exp(\mathbf{u}_1)$

$f_2(\mathbf{u}_2) = \mathbf{u}_2$

$f_3(\mathbf{u}_3) = \mathfrak{s}(\mathbf{u}_3)$

Tabela 3.4: Especificação da Rede MLP Polinomial

Número de camadas $L = 3$

Sinal de entrada $\mathbf{o}_0(n) = \mathbf{x}(n)$

Sinal de saída $\mathbf{o}_L(n) = \underline{\mathbf{j}}(n)$

Especificação das camadas

Entidade MLP	Descrição	Tamanho
$\mathbf{W}_1$	$\mathbf{L}$	$M \times (M+1)$
$\mathbf{b}_1$	$\mathbf{0}$	$(M+1) \times 1$
$\mathbf{W}_2$	$(\mathbf{I}_{M+1} - \mathbf{1}')\mathbf{A}'$	$(M+1) \times M$
$\mathbf{b}_2$	$[\mathbf{A}']^{\mathsf{T}}(\mathbf{1}'\mathbf{G}^{\mathsf{T}}\mathbf{1}'' - \mathbf{G}^{\mathsf{T}}\mathbf{1}'')$	$M \times 1$
$\mathbf{W}_3$	$\begin{bmatrix} \mathbf{1}_{1\times T_1} & & \\ & \ddots & \\ & & \mathbf{1}_{1\times T_M} \end{bmatrix}$	$M \times T$
$\mathbf{b}_3$	$-\underline{\mathbf{t}}$	$T \times 1$

Funções de ativação

$$f_1(\mathbf{u}_1) = (\mathbf{u}_1)^r$$
$$f_2(\mathbf{u}_2) = \mathbf{u}_2$$
$$f_3(\mathbf{u}_3) = \mathfrak{s}(\mathbf{u}_3)$$

3.4 O Algoritmo de Quantização Vetorial Interpolativa

Nesta seção será discutido o projeto do codificador IVQ. Na Seção 3.1, foi discutido o esquema de codificação $\alpha \circ \psi$ e o decodificador ideal β, que é a Condição do Centróide definida na Equação (3.10). Na Seção 3.2, foi introduzida PCA com a finalidade de calcular analiticamente o extrator de características, ψ, deixando o projeto do codificador agora função apenas do operador α. A análise da definição do operador α, através da Equação (3.7) fornece pistas de como seu projeto deve se proceder: trata-se novamente de uma função não-analítica, cuja especificação, assumindo a estrutura apresentada, é dada de forma suficiente pelo vetor aumentado de limiares $\underline{\mathbf{t}}$. A análise da definição de $\underline{\mathbf{t}}$ revela que não somente os limiares são incógnitas, como também o número total de limiares T e como é a distribuição de tais limiares ao longo das M dimensões (especificada por T_m, $m = 1, \ldots, M$).

Lembrando que o projeto de um quantizador vetorial genérico tem como objetivo determinar como argumento o quantizador Q' que minimiza a função custo,

$$Q' = \arg\min_{Q} J, \tag{3.42}$$

e lembrando também que a otimização da Equação (3.42) deve ser realizada sob o codificador, é possível desenvolver um algoritmo de otimização que aja no operador α, uma vez que ψ a princípio é dado pelo cálculo analítico, já descrito.

O algoritmo que otimiza o operador α IVQ com a finalidade de determinar limiares e suas distribuições ao longo das M dimensões é proposto em [10] e será chamado neste trabalho pelo termo "Algoritmo de Quantização Vetorial Interpolativa" (*Interpolative Vector Quantization Algorithm* – IVQA). Utilizando a notação proposta, considere o conjunto de treino disponível com tamanho N e dado por $\mathcal{X} = \{\mathbf{x}(1), \ldots, \mathbf{x}(n), \ldots, \mathbf{x}(N)\}$, $\mathbf{x}(n) \in \mathbb{R}^M$. Uma vez que não se conhecem as alocações T_m e tampouco a quantidade total de limiares T, o parâmetro para projeto relacionado ao tamanho do dicionário será um valor máximo, K_{MAX}. A restrição de entropia será considerada e formalmente representada pelo multiplicador de Lagrange λ, incluso na função custo J. Considere, ainda, que o esquema de mapeamento no espaço de características, ψ, já seja conhecido, através de um determinado cálculo analítico, como os descritos na Seção 3.2.

Uma notação conveniente para a descrição do algoritmo precisa ser aqui introduzida. Assuma uma nova representação para o agrupamento dos limiares, passando do vetor de

limiares $\mathbf{t}_m$ a um conjunto $\mathcal{T}_m \triangleq \{[t_m]_1, \ldots, [t_m]_{T_m}\}$ de tamanho T_m e passando do vetor aumentado de limiares da dimensão, $\underline{\mathbf{t}}$, a um conjunto $\underline{\mathcal{T}} \triangleq \bigcup_{m=1}^{M} \mathcal{T}_m$. Considere também que uma lista de esquemas de quantização $\{Q_1, Q_2, \ldots\}$ define o conjunto de esquemas de quantização $\mathcal{Q}$. O IVQA tem a finalidade de determinar algum $\mathcal{Q}$ localmente ótimo, partindo da inicialização $\mathcal{Q}^0 = \varnothing$.

Uma vez que não se conhece T_m e T, a inicialização para o conjunto de limiares, $\mathcal{T}^0$, é obtida como

$$\underline{\mathcal{T}}^0 = \bigcup_{m=1}^{M} \mathcal{T}_m^0, \quad \underline{\mathcal{T}}_m^0 = \varnothing, \ m = 1, \ldots, M. \tag{3.43}$$

Dessa forma, a i-ésima iteração para o IVQA terá como objetivo atualizar o conjunto $\underline{\mathcal{T}}^i$ e é definida pelos seguintes passos:

1. **Alocação de Novos Limiares:** Para $m = 1, \ldots, M$, aloque novos limiares para originar o conjunto $\mathcal{T}_m^i$ pela expressão

$$\mathcal{T}_m^i = \begin{cases} \bar{x}_m \ , \text{ se } \mathcal{T}_m^{i-1} = \varnothing, \\ \\ \mathcal{T}_m^{i-1} \ \cup \ [t_m]_1^{i-1} - \epsilon_m \ \cup \ \left(\bigcup_{k=2}^{T_m^{i-1}} [t_m]_k^{i-1} - \frac{\Delta(k)}{2} \right) \cup \\ \cup \, [t_m]_{T_m^{i-1}} + \epsilon_m, \text{ caso contrário,} \end{cases} \tag{3.44}$$

   onde $\Delta(k)$ e ϵ_m são, respectivamente, o intervalo entre limiares adjacentes e uma constante incremental positiva e real, definidos como

$$\begin{aligned} \Delta(k) &\triangleq [t_m]_k^{i-1} - [t_m]_{k-1}^{i-1}, \\ \epsilon_m &\triangleq 0,1 \, \mathrm{std}(x_m(n)). \end{aligned} \tag{3.45}$$

2. **Otimização do Conjunto de Limiares:** Para $m = 1, \ldots, M$, fixe $\mathcal{T}_{m'}^i = \mathcal{T}_{m'}^{i-1}$ para $m' \neq m$ de modo que

$$\underline{\mathcal{T}}^i(m) = \mathcal{T}_m^i \cup \left(\bigcup_{\substack{m'=1, \\ m' \neq m}}^{M} \mathcal{T}_{m'}^{i-1} \right), \tag{3.46}$$

   e calcule o conjunto de limiares otimizado

$$\left[\underline{\mathcal{T}}'\right]^i (m) = \arg \min_{\underline{\mathcal{T}}^i(m)} J \tag{3.47}$$

   usando o Algoritmo de Nelder-Mead [16]. O conjunto $\left[\underline{\mathcal{T}}'\right]^i (m)$ especifica um esquema de quantização $[Q']^i(m)$.

3. **Seleção dos Melhores Codificadores:** Atualize o conjunto de quantizadores projetados $\mathcal{Q}^i$

$$\mathcal{Q}^i = \mathcal{Q}^{i-1} \cup \left(\bigcup_{m=1}^{M} [Q']^i(m) \right) \tag{3.48}$$

e selecione os *dois* melhores quantizadores $\{Q_1, Q_2\} \in \mathcal{Q}^i$ em termos de seus respectivos custos J_1 e J_2. Os quantizadores Q_1 e Q_2 definem os conjuntos $\underline{\mathcal{T}}_1$ e $\underline{\mathcal{T}}_2$ que serão utilizados para inicializar *duas* iterações paralelas $(i+1)$.[3] O objetivo da utilização de duas iterações em paralelo, e não somente uma, é obter uma estimativa da susceptibilidade do algoritmo de Nealder-Mead aos múltiplos mínimos locais da função de custo J.

O IVQA deve realizar sua iteração para $i = 1, \ldots, I$, com $I = \lceil \log_2 K_{\text{MAX}} \rceil$, para garantir que o máximo tamanho do dicionário seja K_{MAX}. Uma vez que $\mathcal{Q}^I$ é determinado, o resultado final do algoritmo será o subconjunto $\mathcal{Q}' \subset \mathcal{Q}^I$ tal que $\mathcal{Q}' = \mathfrak{h}(\mathcal{Q}^I)$. O operador $\mathfrak{h}(\cdot)$ é o critério da casca convexa inferior (*lower convex hull*), cuja definição é discutida no Apêndice B.

3.4.1 O Aprimoramento Algorítmico Proposto

Uma evolução natural para o IVQA é incluir na otimização do codificador os elementos que caracterizam o extrator de características ψ. Essa extensão é justificada pelo fato de que o cálculo analítico de ψ tem como objetivo diagonalizar a matriz de autocorrelação do sinal, $\mathbf{C}$, o que não garante que o codificador $\alpha \circ \psi$ resultante será ótimo em termos de custo.[4]

Considere que se disponha de um extrator de características ψ, calculado analiticamente, e de um operador α, otimizado pelo IVQA. Considere também que o esquema de codificação resultante tenha sua implementação MLP conhecida e, portanto especificada por L matrizes de sinapses e seus respectivos vetores de polarização e funções de ativação.[5] Dessa forma, é possível propor um algoritmo que resolva o problema relacionado à Equação (3.42) através da otimização conjunta de todos os parâmetros MLP. Até o final desse

[3] No caso de iterações paralelas i, a seleção dos dois melhores codificadores se dá após o conjunto $\mathcal{Q}^i$ ser atualizado por todas. O conjunto $\mathcal{Q}^i$ é, portanto, o mesmo para ambas as iterações paralelas.

[4] Notar que o codificador ótimo é definido pela Condição da Partição, Equação (2.6).

[5] No caso de rede MLP Polinomial, a especificação MLP também inclui a ordem polinomial r, que especifica a função de ativação $f_1(\cdot)$ correlata.

texto, este algoritmo será referido pelas siglas MLP OA, da expressão inglesa "*Multilayer Perceptron Optimization Algorithm*".

Considere um vetor $\mathbf{z} \in \mathbb{R}^Z$ que agrupe todos os parâmetros MLP passíveis de otimização,

$$\mathbf{z} = \begin{bmatrix} \text{vec}\{\mathbf{W}_1\} \\ \vdots \\ \text{vec}\{\mathbf{W}_{L-1}\} \\ \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_L \end{bmatrix}. \tag{3.49}$$

A decisão de não incluir $\mathbf{W}_L$ em $\mathbf{z}$ é tomada com o objetivo de diminuir a dimensão Z e é apoiada pelo fato de que $\mathbf{W}_L$ é a matriz relacionada ao operador α, cuja estrutura foi proposta tendo como base a Equação (3.7), que caracteriza a definição de IVQ. O tamanho Z é função da operação PCA empregada e do operador α, que varia com T e T_m. Dessa forma, o objetivo do MLP OA é resolver o problema

$$\mathbf{z}' = \arg\min_{\mathbf{z}} J, \tag{3.50}$$

através do algoritmo de otimização proposto, que é o de Nelder-Mead. Uma vez que o Algoritmo de Nelder-Mead é sujeito a mínimos locais [16], a inicialização do MLP OA é crítica. Portanto, é natural que se utilize como sua inicialização quantizadores otimizados pelo IVQA. Essa prática é enfatizada se for considerado o fato de que não se conhece um esquema de alocação de limiares T_m *a priori*.

Capítulo 4

Quantização Vetorial Aplicada à Compressão de Imagens

As técnicas de quantização vetorial apresentadas nos Capítulos 2 e 3 terão seus desempenhos avaliados para a aplicação em compressão de imagens.

4.1 Esquemas de Compressão de Imagens

Um sistema de compressão de imagens é composto por três etapas ao longo da codificação: *transformação* dos dados, *quantização* e *codificação de fonte discreta*, como ilustrado pela Figura 4.1. Os esquemas de quantização ECVQ (Figura 2.3) e IVQ (Figura 3.1) se contextualizam a esse paradigma pelo fato de implementarem a etapa de quantização dos dados, através das operações α (no caso ECVQ) e $\alpha \circ \psi$ (no caso IVQ). O sinal para quantização, $\mathbf{x}(n)$, deve conter de forma representativa as informações das imagens e é gerado na etapa de transformação, também conhecida como pré-processamento.

Imagens são sinais bidimensionais, que podem ser associados a um mapa de coordenadas discretas, no qual cada ponto (denominado *pixel*) possui um valor específico de luminância ou, alternativamente, de valores de componentes de cor. Nesse trabalho, será considerado somente a informação de luminância, uma vez que toda a análise pode ser estendida para cada componente de cor, sem perda de generalidade. Imagens são processos estocásticos e, portanto, devem ser analisadas probabilisticamente com a finalidade de que os esquemas de compressão sejam representativos para um número abrangente de imagens.

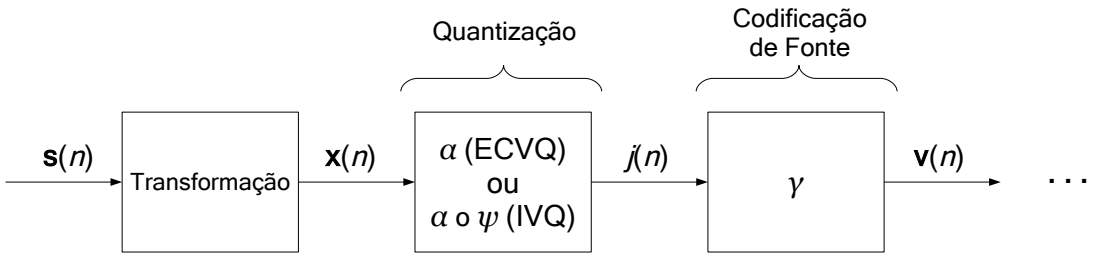


Figura 4.1: Codificador para um esquema generalizado de compressão de imagens.

Do ponto de vista probababilístico, imagens são sinais *não-estacionários*, ou seja, não possuem um comportamento estatístico fixo ao longo do tempo. Com isso, o projeto de esquemas de compressão de imagens, que requer um conjunto de treino, se tornaria inviável, uma vez que não existiria um conjunto de treino suficientemente abrangente para a representação do sinal. Dessa forma, faz-se necessário adaptar o sinal, de modo a aproximá-lo por um processo estocástico que seja pelo menos aproximadamente estacionário.

Na literatura [14], a solução clássica para esse fim é dividir as imagens em blocos de tamanho fixo e assumir como sinal de processamento os vetores cujos valores são cada luminância pertencente ao bloco. O tamanho do bloco deve ser tal que a estacionariedade possa ser assumida. Uma maneira de se avaliar o grau de estacionariedade é calcular a matriz de autocorrelação do sinal. Em termos práticos, é suficiente existir estacionariedade no sentido amplo, o que implica que tal matriz deve ser (aproximadamente) *toeplitz*. Considere um conjunto de imagens $\mathcal{S} = \{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_I\}$, de tamanho I. A i-ésima imagem é representada por uma matriz $\mathbf{S}_i$, que pode ser expressa em função de blocos de tamanho $P \times Q$:

$$\mathbf{S}_i = \begin{bmatrix} [\mathbf{B}_i]_{11} & [\mathbf{B}_i]_{12} & \cdots & [\mathbf{B}_i]_{1q_i} \\ [\mathbf{B}_i]_{21} & [\mathbf{B}_i]_{22} & \cdots & [\mathbf{B}_i]_{2q_i} \\ \vdots & \vdots & & \vdots \\ [\mathbf{B}_i]_{p_i1} & [\mathbf{B}_i]_{p_i2} & \cdots & [\mathbf{B}_i]_{p_iq_i} \end{bmatrix} \tag{4.1}$$

de modo que o tamanho da imagem é p_iPq_iQ. Notando-se que os blocos podem ser expressos em termos do vetor $\mathbf{b}'$ de tamanho PQ:

$$\begin{aligned} \mathbf{B} &= \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_Q \end{bmatrix} \\ \Rightarrow \mathbf{b}' &= \begin{bmatrix} \mathbf{b}_1^\mathsf{T} & \mathbf{b}_2^\mathsf{T} & \cdots & \mathbf{b}_Q^\mathsf{T} \end{bmatrix}^\mathsf{T} \end{aligned} \tag{4.2}$$

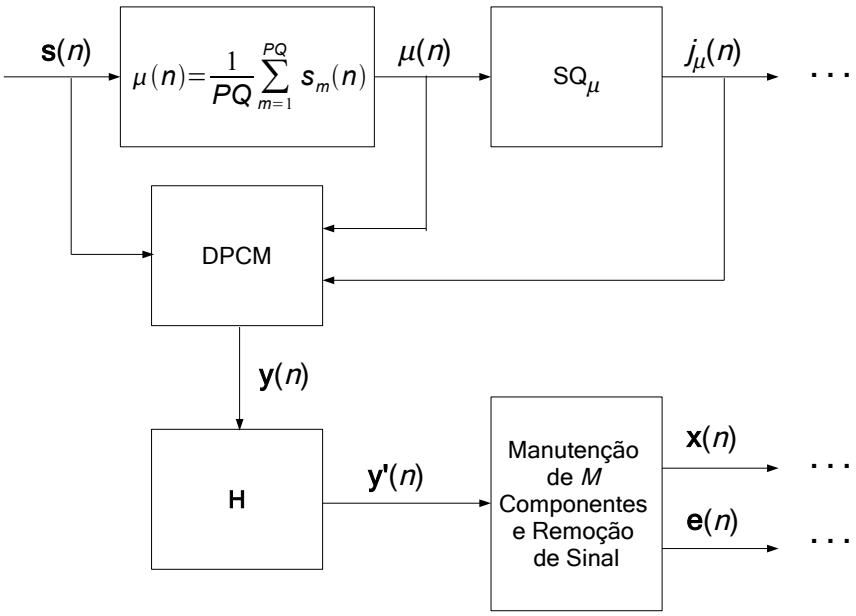


Figura 4.2: Pré-processamento de imagens, para extração de componentes $\mathbf{x}(n)$ para quantização vetorial.

é possível associar ao conjunto de imagens $\mathcal{S}$ o sinal $\mathbf{s}(n) \in \mathbb{R}^{PQ}$, tal que

$$\left[\begin{array}{cccc} \mathbf{s}(1) & \mathbf{s}(2) & \cdots & \mathbf{s}(N) \end{array} \right] = \left[\begin{array}{cccc} [\mathbf{b}'_1]_1 & [\mathbf{b}'_1]_2 & \cdots & [\mathbf{b}_I]_{p_I q_I} \end{array} \right] \tag{4.3}$$

para

$$N = I \cdot \sum_{i=1}^{I} p_i q_i. \tag{4.4}$$

O sinal $\mathbf{s}(n)$ é apenas uma nova representação do conjunto $\mathcal{S}$, embora permita que as imagens sejam submetidas a processamentos em que são modeladas como vetores (realizações de um processo estocástico aproximadamente estacionário). Dessa maneira, adota-se $\mathbf{s}(n)$ como sinal de entrada do pré-processamento, esquematizado pela Figura 4.2. O pré-processamento tem a finalidade de tornar a informação mais concisa em termos do número de dimensões M do sinal $\mathbf{x}(n)$, entrada do bloco de quantização.

Para esse objetivo, assumem-se premissas referentes às características inerentes do sinal em questão, ou seja, de imagens. Imagens de cenas naturais são em geral compostas por baixas freqüências. Isso permite afirmar que os valores de luminância de *pixels* adjacentes são muito próximos, o que fornece uma primeira chave para o processamento. Através da modulação por codificação diferencial de pulso (*differential pulse-codem modulation* – DPCM), é possível eliminar a média do sinal para manter como informação apenas sua

diferença. Naturalmente, a remoção da média requer que ela seja transmitida em paralelo pelo canal, razão pela qual se requer uma operação de quantização Q_μ exclusiva, pois o sinal de média $\mu(n)$ ainda possui precisão infinita ($\mu(n) \in \mathbb{R}$). Uma vez que $\mu(n)$ é um sinal unidimensional ($M_\mu = 1$), a operação Q_μ é quantização escalar, podendo ser projetada, por exemplo, pelo GLA, produzindo um codificador α_μ e um decodificador β_μ.

Para que o erro de quantização da média não se propague, de posse do sinal $j_\mu(n)$, a DPCM é realizada localmente – no próprio codificador – com a finalidade de se determinar o sinal sem média $\mathbf{y}(n)$:

$$\begin{aligned}
\mu(n) &= \frac{1}{PQ}\sum_{m=1}^{PQ} s_m(n) \\
\hat{\mu}(n) &= \beta_\mu(j_\mu(n)) \\
\delta(1) &= \hat{\mu}(1) \\
\delta(n) &= \mu(n) - \hat{\mu}(n-1),\ n = 2, \ldots, N \\
\mathbf{y}(n) &= \mathbf{s}(n) - \delta(n) \cdot \mathbf{1}_{PQ\times 1}
\end{aligned} \tag{4.5}$$

A matriz de transformação $\mathbf{H}$ tem a finalidade de concentrar a energia do sinal em menor número de componentes. A matriz, por si, não reduz a informação, uma vez que o sinal $\mathbf{y}'(n)$ mantém-se com PQ componentes. Com a finalidade de se manter somente as $M < PQ$ componentes principais, deve-se escolher aquelas que sejam representativas do sinal. Nesse ponto, duas considerações práticas devem ser assumidas. A primeira é o tamanho do bloco: $P = Q$ e $PQ = 16$. Assim, o tamanho dos blocos será 4×4. A segunda, é que serão escolhidas as $M = 4$ componentes principais. A matriz de transformação $\mathbf{H}$ utilizada é proposta por Malvar para a utilização no padrão de codificação de vídeo H.264/AVC, que também utiliza a partição da imagem em blocos 4×4. A trasformada de Malvar [19] foi desenvolvida como uma adaptação à transformada discreta dos cossenos (*discrete cossine transform* – DCT), para trabalhar com números inteiros e fazer com que o número de *bits* médio necessário para representar o vetor transformado $\mathbf{y}' = \mathbf{H}\mathbf{y}$ seja aproximadamente o mesmo que o número médio de *bits* para $\mathbf{y}$, ou seja, o ganho que $\mathbf{H}$ atribui ao sinal transformado é próximo da unidade. A exemplo da DCT, a informação contida no sinal $\mathbf{y}'(n)$ concentra a energia nos coeficientes de menor valor, o que permite, finalmente, que a informação residual presente nos demais coeficientes seja descartada.[1] Os coeficientes mantidos são os de índices 2, 3, 5 e 9. O primeiro coeficiente

[1]Notar que a supressão de coeficientes no sinal $\mathbf{y}'(n)$ é um processo de compressão em que a informação

é desconsiderado, pois se trata da média, já retirada pela DPCM.

Dessa forma, já na fase de pré-processamento há supressão de informação. Por essa razão, as M componentes resultantes serão consideradas para efeitos de tamanho dos dicionários de decodificação da quantização. Esse fato ajuda a explicar a definição apresentada no Capítulo 3 do codificador IVQ, com relação ao mapeamento do extrator de características ser $\mathbb{R}^M \rightarrow \mathbb{R}^M$, que mantém como tamanho do dicionário o valor M.

Finalmente, para a determinação do sinal de entrada da quantização vetorial, é removida de $\mathbf{y}(n)$ a informação de sinal:

$$\begin{aligned} \mathbf{e}(n) &= \text{sign}(\mathbf{y}'(n)) \\ \mathbf{x}(n) &= |\mathbf{y}'(n)| \end{aligned} \tag{4.6}$$

4.2 Simulação de Compressão de Imagens

Nesta seção, a proposição de esquemas IVQ para compressão de imagens é validada através do cálculo do desempenho de distorção e entropia para sistemas projetados. O objetivo é verificar a evolução entre métodos de projeto que levem em conta os algoritmos IVQA e MLP OA e avaliar a diferença de desempenho entre sistemas IVQ e ECVQ. A análise é iniciada pelas considerações práticas necessárias para o projeto ECVQ e IVQ.

Um conjunto de treino $\mathcal{S}_\text{R}$ composto por $I_\text{R} = 21$ imagens será utilizado para a simulação. Sua especificação é encontrada na Tabela 4.1.

4.2.1 Considerações Práticas para o Projeto de Compressores de Imagens

Tanto para o projeto de compressores com mapeamento Q do tipo ECVQ quanto IVQ, foi utilizado o pré-processamento descrito na Seção 4.1, de modo que o sinal de entrada $\mathbf{x}(n)$ relativo ao conjunto de treino e aplicado a qualquer Q é único. A quantização do sinal de média $\mu(n)$ foi projetada através do GLA para um tamanho de dicionário $K = 2^4$.

Projeto ECVQ

O projeto ECVQ foi realizado através do algoritmo GLA (Seção 2.3), com tamanho de dicionário $K = 2^{11}$. Foram projetados 18 mapeamentos Q diferentes, associados a diferentes

descartada é irrecuperável.

Tabela 4.1: O Conjunto de Imagens de Treino

	Imagem	Resolução
1	Aircarrier	480×640
2	Airplane-1	512×512
3	Astro-1	512×640
4	Bicycle	1152×1440
5	Couple	256×256
6	Crownmilk	512×512
7	Diver	384×576
8	F15	384×576
9	F22	576×768
10	F117	576×768
11	Flowers	512×512
12	Girl	256×256
13	House	256×256
14	Jellybeans-1	256×256
15	Jellybeans-2	256×256
16	MIG29B	576×768
17	Sailboat	512×512
18	Teeny	1152×1440
19	Tiffany	512×512
20	Tomcat	512×512
21	Tree	256×256

restrições de entropia λ, conforme a Tabela 4.2.

Projeto IVQ

Devido às diferentes proposições para a implementação do extrator de características ψ, foram avaliados esquemas IVQ implementados por LPCA e KPCA. Todos os sistemas foram projetados através do IVQA, apresentado na Seção 3.4. Foi submetido ao IVQA o conjunto de treino $\mathcal{X}_{\mathrm{R}}$, para otimização inicial somente em termos de distorção ($\lambda = 0$) e para um tamanho máximo do dicionário $K_{\mathrm{MAX}} = 2^{11}$. A solução do algoritmo, após ser aplicado o critério da casca convexa inferior (Apêndice B), é um conjunto com

Tabela 4.2: Especificação para Projeto ECVQ

Tamanho máximo do dicionário $K_{\mathrm{MAX}} = 2^{11}$

Multiplicadores de Lagrange λ:

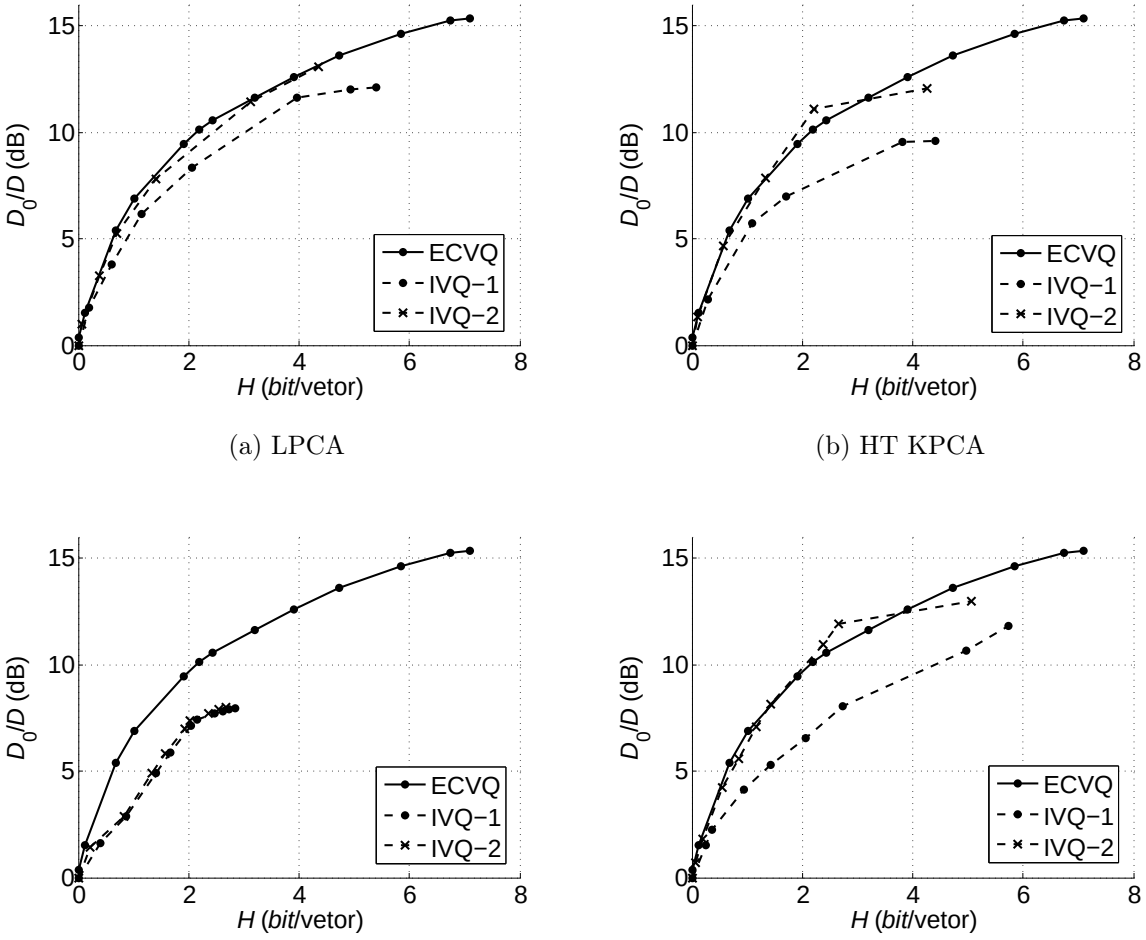
ECVQ	λ	ECVQ	λ	ECVQ	λ
1	0	7	$5 \cdot 10^{-3}$	13	0,08
2	$1 \cdot 10^{-4}$	8	0,01	14	0,1
3	$2 \cdot 10^{-4}$	9	0,02	15	0,5
4	$5 \cdot 10^{-4}$	10	0,05	16	1
5	$1 \cdot 10^{-3}$	11	0,06	17	5
6	$2 \cdot 10^{-3}$	12	0,07	18	10

I' quantizadores IVQ, $\mathcal{Q}'$. Lembrando que, além de selecionar os melhores sistemas, o operador $\mathfrak{h}(\cdot)$ também associa a cada $Q'_i \in \mathcal{Q}'$ um λ'_i *local*, é possível reiniciar I' vezes o algoritmo IVQA, substituindo a inicialização da Equação (3.43), que atribui como vazio o conjunto de limiares $\underline{\mathcal{T}}^0$, pelo conjunto $\underline{\mathcal{T}}'_i$, associado ao i-ésimo quantizador Q'_i e tomando como multiplicador de Lagrange o valor de λ'_i. O tamanho máximo do dicionário é mantido o mesmo, $K_{\mathrm{MAX}} = 2^{11}$. A solução dessa segunda rodada, após nova submissão ao operador $\mathfrak{h}(\cdot)$, é o conjunto de quantizadores $\mathcal{Q}''$. Sistemas IVQ projetados por esse algoritmo serão identificados pela sigla IVQ-1.

Para a avaliação do ganho de desempenho promovido pela otimização conjunta dos operadores α e ψ, implementada pelo MLP OA, o conjunto $\mathcal{Q}''$ fornece I'' quantizadores para a inicialização do algoritmo. Novamente, foi utilizado como valor para o multiplicador de Lagrange o valor λ''_i, calculado pelo operador $\mathfrak{h}(\cdot)$ e associado ao i-ésimo quantizador Q''_i. Finalmente, os sistemas resultantes do MLP OA definem uma lista de quantizadores $\mathcal{Q}'''$ e são identificados pela sigla IVQ-2.

4.2.2 Resultados

As curvas da Figura 4.3 apresentam o desempenho dos sistemas IVQ para as variantes de implementação de ψ: (a) LPCA, (b) HT KPCA, (c) RBF KPCA, e (d) KPCA Polinomial. Adicionalmente, foi incluída em cada gráfico a curva de desempenho associada ao projeto



(a) LPCA

(b) HT KPCA

(c) RBF KPCA

(d) KPCA Polinomial

Figura 4.3: Desempenho dos sistemas ECVQ e dos sistemas IVQ, de acordo com a implementação de ψ.

ECVQ, com a finalidade de permitir a avaliação de diferença de desempenho entre sistemas projetados por IVQ e ECVQ. Nesses gráficos, a qualidade da quantização é medida pela razão sinal-ruído de quantização (*signal-to-quantization noise ratio* – SQNR), igual à razão entre a distorção associada à taxa zero, D_0, e a distorção D do sistema. A distorção D_0 é conceitualmente igual à variância do conjunto de treino $\mathcal{X}_{\mathrm{R}}$:

$$D_0 = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}(n) - \bar{\mathbf{x}}\|^2. \tag{4.7}$$

No gráfico da Figura 4.3b, são apresentados os conjuntos $\mathcal{Q}'''$ e $\mathcal{Q}''$ associados à melhor configuração HT KPCA. Diferentes configurações KPCA foram obtidas através da escolha arbitrada dos parâmetros livres para o operador κ (confira pelas Equações (3.31), (3.32) e (3.33)). Em particular, para HT KPCA, o desempenho apresentado ocorreu para valores de ganho e polarização, respectivamente, iguais a $g = -0,005$ e $\theta = 0$. De modo análogo, a curva RBF KPCA (Figura 4.3c) é associada ao parâmetro $\sigma^2 = 1$ e a curva KPCA Polinomial (Figura 4.3d) à ordem $r = 0,5$.

A Figura 4.3 demonstra a diferença de desempenho entre quantizadores ECVQ e IVQA. Exceto os sistemas do caso RBF KPCA, que obtiveram subdesempenho devido a problemas de mínimos locais do IVQA, os esquemas IVQ alcançaram o desempenho ECVQ, após submissão ao algoritmo MLP OA (curvas IVQ-2). Isso demonstra que a proposição MLP OA para otimização conjunta consegue superar o desempenho IVQ-1 e se aproximar do ECVQ. Nesse ponto, ressalva-se o desempenho superior de alguns sistemas IVQ em relação ao ECVQ, como por exemplo os dois sistemas KPCA Polinomial IVQ-2 com entropia próxima a $2,5$ *bits*/vetor (confira pela Figura 4.3d). Como já discutido no Capítulo 2, o projetista poderia inicializar um projeto ECVQ, através do GLA, com o dicionário resultante dos 2 sistemas IVQ em questão. O resultado seria quantizadores ECVQ de melhor desempenho ou no mínimo igual. No entanto, esse fato não desvaloriza o desempenho IVQ alcançado, uma vez que demonstra que a solução algorítmica proposta consegue escapar bem dos problemas de mínimos locais de baixa qualidade.

4.3 Codificação de Imagens

As imagens do conjunto de treino, $\mathcal{S}_{\mathrm{R}}$, foram submetidas aos sistemas projetados para a avaliação do efeito da compressão. A razão sinal-ruído de pico (*peak signal-to-noise*

Tabela 4.3: O Conjunto de Imagens de Teste

	Imagem	Resolução
1	Baboon	512×512
2	Bike	2560×2048
3	Goldhill	576×720
4	Lena	512×512
5	Pepper	512×512
6	Woman	2560×2048

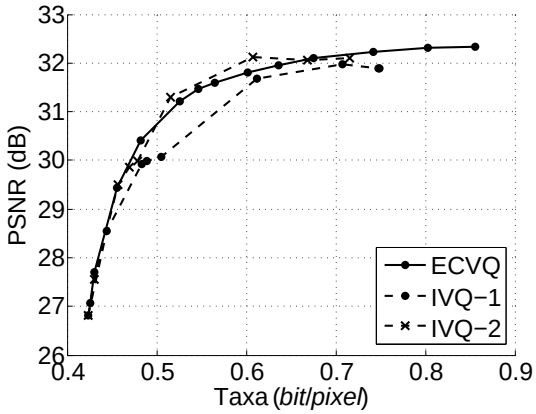
ratio – PSNR) foi calculada levando em conta a reconstituição do sinal codificado $\hat{\mathbf{s}}(n)$, resultante da operação de decodificação ECVQ ou IVQ, mais a adição do sinal $\mathbf{e}(n)$ e do mapeamento inverso dado pela matriz $\mathbf{H}$, para $\hat{\mathbf{y}}(n) = \mathbf{H}^{-1}\hat{\mathbf{y}}'(n)$, com $\hat{\mathbf{y}}'(n)$ resultante da interpolação do sinal $\hat{\mathbf{x}}$ com adição de zeros nas dimensões descartadas durante o pré-processamento.

Adicionalmente, um conjunto de teste foi submetido aos esquemas de quantização projetados. O conjunto de 6 imagens $\mathcal{S}_{\mathrm{T}}$, especificado pela Tabela 4.3, foi submetido ao pré-processamento, para a obtenção do conjunto de vetores residuais $\mathcal{X}_{\mathrm{T}}$.

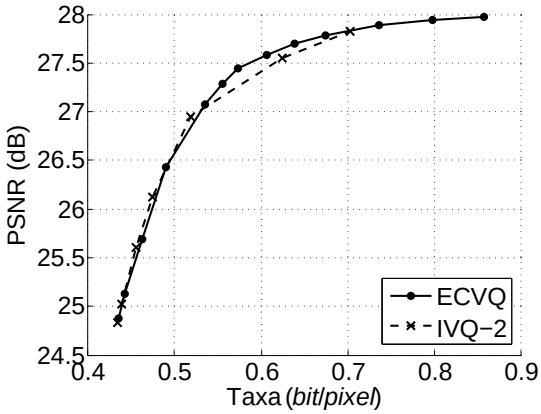
A Figura 4.4 apresenta os valores de PSNR para as imagens codificadas. Nos gráficos das Figuras 4.4a, 4.4c e 4.4e são apresentados os desempenhos para o conjunto de imagens de treino, $\mathcal{S}_{\mathrm{R}}$. Já nas Figuras 4.4b, 4.4d e 4.4f são apresentados os desempenhos quando as imagens do conjunto de teste, $\mathcal{S}_{\mathrm{T}}$, são submetidas ao esquema de compressão.

A análise da Figura 4.4 confirma o desempenho equivalente entre ECVQ e IVQ-2. O ganho de PSNR entre os sistemas IVQ-2 e IVQ-1 chegou a $0,5$ dB para taxas aproximadas de $0,61$ *bit/pixel* no caso LPCA IVQ e para taxas de $0,59$ para o caso KPCA Polinomial IVQ.

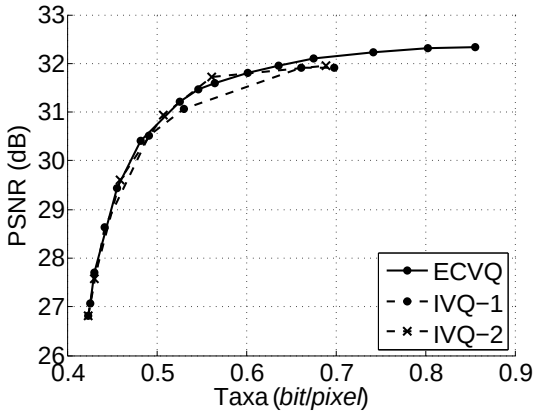
Finalmente, a Figura 4.5 mostra o resultado de codificação para a imagem "Bikesmall", que é uma sub-imagem de resolução 512×512 extraída da imagem "Bike".



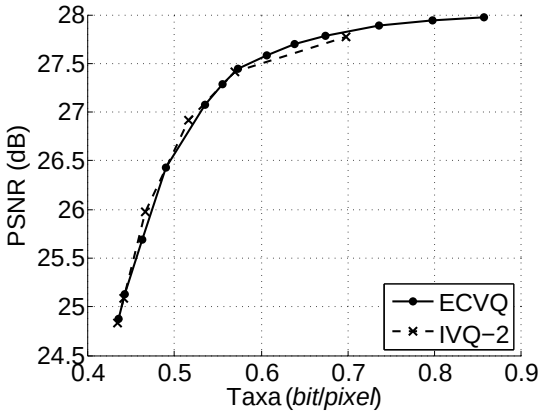
(a) LPCA, $\mathcal{S}_{\mathrm{R}}$



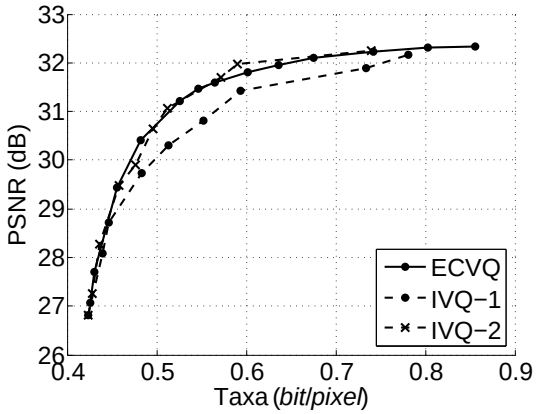
(b) LPCA, $\mathcal{S}_{\mathrm{T}}$



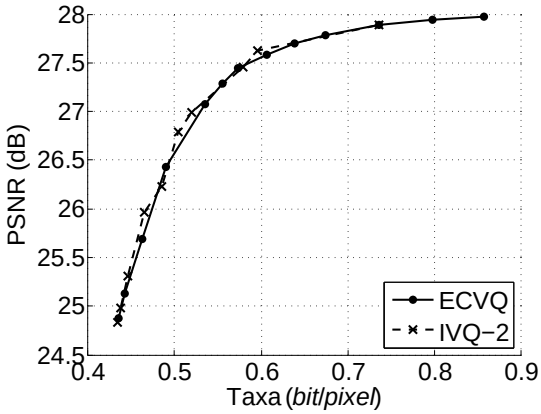
(c) HT KPCA, $\mathcal{S}_{\mathrm{R}}$



(d) HT KPCA, $\mathcal{S}_{\mathrm{T}}$

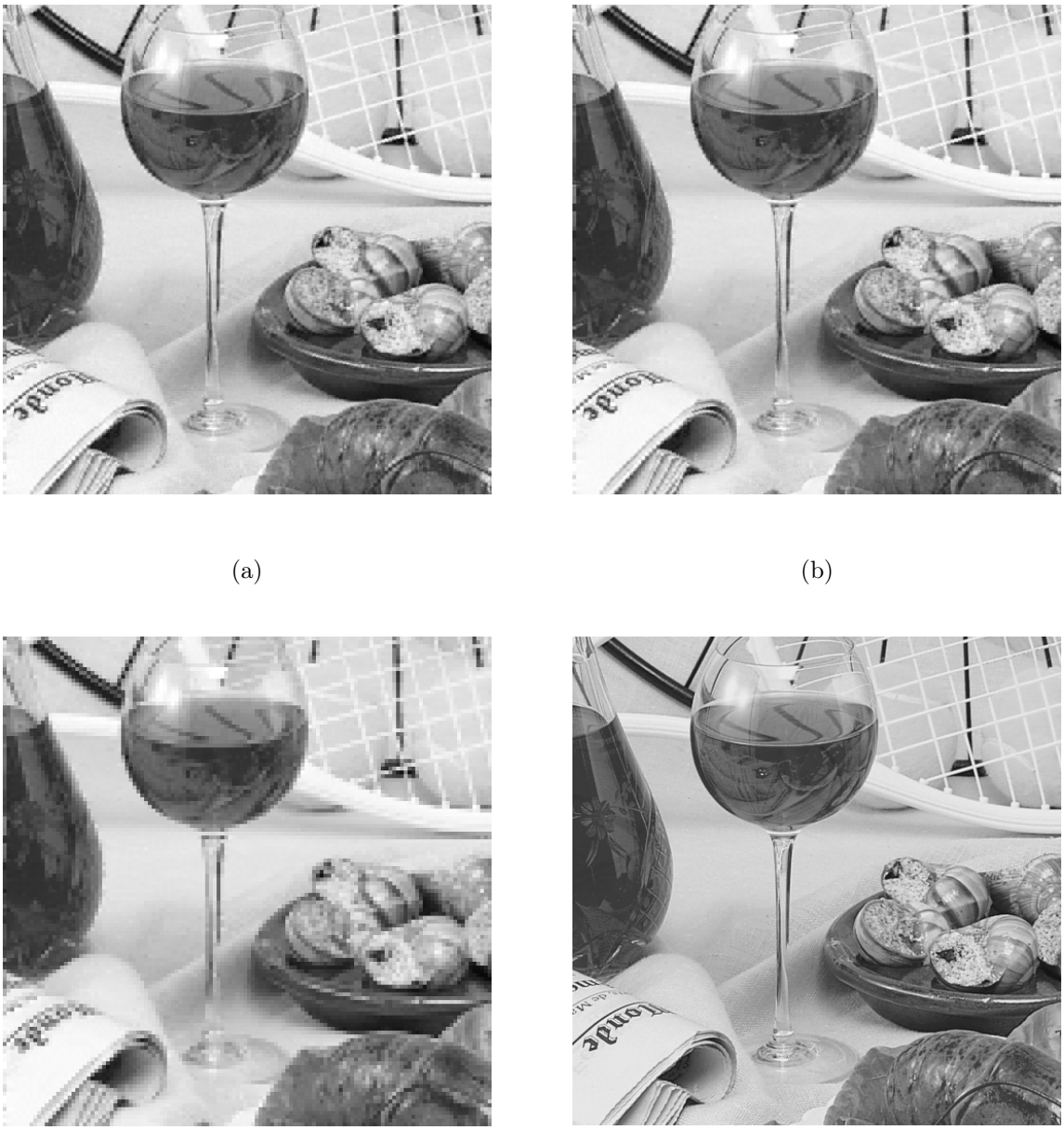


(e) KPCA Polinomial, $\mathcal{S}_{\mathrm{R}}$



(f) KPCA Polinomial, $\mathcal{S}_{\mathrm{T}}$

Figura 4.4: Avaliação dos esquemas de compressão de imagens em termos de PSNR e taxa.



(a) (b)

(c) (d)

Figura 4.5: Bikesmall, comprimida por esquemas (a) LPCA IVQ-2, à taxa de 0,74 *bit/pixel* e PSNR = 26, 2 dB e (b) ECVQ, à taxa de 0,76 *bit/pixel* e PSNR = 26, 3 dB. (c) Imagem reconstituída somente por $\hat{\mu}(n)$. (d) Imagem original.

Capítulo 5

Análise de Complexidade

A adoção de esquemas de quantização com arquiteturas de redes neurais, como os apresentados nos Capítulos 2 e 3, permite sua utilização para a implementação de sistemas de compressão de imagens cujas operações de compressão sejam implementáveis ainda em domínio analógico, antes da conversão do sinal de analógico para digital [23] [10]. Isso é particularmente conveniente em sensores de câmeras fotográficas portáteis, uma vez que aumenta o nível de integração do circuito e permite o uso de microprocessadores com menor gasto de energia, pois as operações digitais se tornam mais simples.

Um compromisso natural de projetos para esse contexto é o que se relaciona à complexidade do esquema analógico, que não deve ser grande ao ponto de inviabilizar o projeto. Isso justifica a adoção de esquemas IVQ, como os discutidos no Capítulo 3, em comparação aos ECVQ, que requerem uma rede neural mais complexa. O objetivo deste capítulo é apresentar uma avaliação quantitativa de quão menos custosos são os esquemas IVQ em relação aos ECVQ em termos de complexidade.

Tendo como finalidade o projeto de esquemas de compressão de imagens, somente a etapa de codificação será considerada na análise de complexidade do presente trabalho. As operações de pré-processamento e decodificação são as mesmas tanto para IVQ quanto para ECVQ, o que direciona a atenção para o esquema de codificação por quantização vetorial.

Ao longo desse trabalho, foram apresentadas redes neurais do tipo SOM e MLP para a implementação de codificadores ECVQ e IVQ, respectivamente. A modelagem dessas redes foi realizada através da análise das operações matemáticas necessárias para a implementação de cada codificação. De modo análogo, o alicerce para a avaliação de com-

plexidade de um circuito que implementa tais arquiteturas será a heurística que envolve as operações matemáticas correlatas. Considere um codificador implementado por uma quantidade P de diferentes operações matemáticas. A p-ésima operação matemática, identificada pelo índice p, estará relacionada a uma complexidade Θ_p, de modo que a complexidade total da operação de codificação, Θ, é definível como

$$\Theta = \sum_{p=1}^{P} \Theta_p. \tag{5.1}$$

Uma estratégia de mensuração da complexidade de um codificador é analisar individualmente a complexidade relacionada a cada operação, Θ_p. O valor de Θ_p será tão grande quanto a quantidade de vezes que tal operação for requisitada no codificador, diga-se η_p. Por outro lado, as operações podem ser mais ou menos custosas em comparação umas com as outras. Assim, para a utilização da Equação (5.1), há a necessidade de se ponderar as quantidades η_p por pesos, digam-se τ_p. Assim, a complexidade de cada operação matemática é avaliada como

$$\Theta_p = \tau_p \cdot \eta_p. \tag{5.2}$$

5.1 Complexidade da Codificação para Esquemas de Quantização

Nesta seção, o objetivo é modelar o cálculo da complexidade dos codificadores ECVQ e IVQ em termos de P operações matemáticas necessárias para sua realização e em termos do cálculo de suas quantidades, η_p, $p = 1, \ldots, P$. Para tanto, é necessário recorrer à modelagem de tais codificadores pelas respectivas redes neurais de implementação.

É possível iniciar a listagem de operações matemáticas pela análise da estrutura MLP, presente também no caso SOM. Sua operação foi definida pela Equação (3.29), reescrita aqui por conveniência:

$$\mathbf{u}_l = \mathbf{W}_l^{\mathsf{T}} \mathbf{o}_{l-1} + \mathbf{b}_l, \tag{5.3}$$

que corresponde ao mapeamento entre o vetor de saída da camada $l-1$, $\mathbf{o}_{l-1}$, no vetor interno da l-ésima camada de uma rede neural, $\mathbf{u}_l$. Definindo os operadores $\mathfrak{l}_{\mathrm{ROW}}(\cdot)$ e $\mathfrak{l}_{\mathrm{COL}}(\cdot)$, respectivamente, como o número de linhas e o número de colunas de uma matriz, a projeção do vetor $\mathbf{o}_{l-1}$ na matriz de sinapses $\mathbf{W}_l$ requer $\mathfrak{l}_{\mathrm{COL}}(\mathbf{W}_l)$ produtos internos.

Os produtos internos, por sua vez, podem ser pensados como a realização de $\iota_{\mathrm{ROW}}(\mathbf{W}_l)$ operações de somatório e $\iota_{\mathrm{ROW}}(\mathbf{W}_l) \cdot \iota_{\mathrm{COL}}(\mathbf{W}_l)$ de multiplicação.

No caso específico deste trabalho, é conveniente diferenciar a operação de multiplicação pelo sinal dos fatores de multiplicação $[W_l]_{i_1 i_2}$. Será visto na Seção 5.2 que a implementação de multiplicação é diferenciada, de acordo com o sinal de tais fatores. Como conseqüência, tratam-se de operações com diferentes pesos τ_p e que, portanto, devem ser tratadas separadamente.

As quantidades das operações de "multiplicação por fator positivo", η_1, e "multiplicação por fator negativo", η_2, são dadas por

$$\eta_1 = \sum_{l=1}^{L-1} \mathrm{card}\left([W_l]_{i_1 i_2} \;\middle|\; [W_l]_{i_1 i_2} > 0 \right), \tag{5.4a}$$

$$\eta_2 = \sum_{l=1}^{L-1} \mathrm{card}\left([W_l]_{i_1 i_2} \;\middle|\; [W_l]_{i_1 i_2} < 0 \right), \tag{5.4b}$$

$$i_1 = 1, \ldots, \iota_{\mathrm{ROW}}(\mathbf{W}_l), \quad i_2 = 1, \ldots, \iota_{\mathrm{COL}}(\mathbf{W}_l).$$

Nas Equações (5.4a) e (5.4b), os somatórios não incluem a última camada da rede. Embora essa equação seja empregada para as redes SOM e MLP, a razão para a não-inclusão da camada L é diferente para cada rede. No caso SOM, isso se deve à implementação da operação WTA pela última camada, cuja estrutura não é MLP. Já no caso MLP, embora a camada L assuma o formato da Equação (5.3), a matriz de sinapses $\mathbf{W}_L$ é composta por uma estrutura trivial que contém somente os valores 0 e 1. Essa estrutura pode ser implementada com custo desprezível diante do custo de implementação dos demais produtos internos.

Ainda na Equação (5.3), há a soma do vetor de vetor de polarização $\mathbf{b}_l$, operação comum para todas as redes consideradas. A geração de uma constante do vetor de polarização, $[b_l]_i$, a exemplo da operação de multiplicação, será diferenciada pelo sinal, seguindo o mesmo raciocínio de que a implementação dessa operação pode assumir peso τ_p diferente de acordo com o sinal. As quantidades de valores de constantes positivas e negativas que devem ser geradas para formar os vetores de polarização de todas as camadas da rede, η_3

e η_4 (respectivamente), são obtidas de maneira análoga às multiplicações:[1]

$$\eta_3 = \sum_{l=1}^{L-1} \text{card}\left(\left[b_l\right]_i \mid \left[b_l\right]_i > 0\right), \tag{5.5a}$$

$$\eta_4 = \sum_{l=1}^{L-1} \text{card}\left(\left[b_l\right]_i \mid \left[b_l\right]_i < 0\right), \tag{5.5b}$$

$$i = 1, \ldots, \mathfrak{l}(\mathbf{b}_l).$$

5.1.1 Operações Matemáticas Específicas para cada Tipo de Rede

A seguir, serão consideradas as operações matemáticas que, ou se diferem na quantidade de acordo com a rede neural, ou são exclusivas de uma determinada rede. Dentre as redes consideradas, a única operação que possui duas fórmulas para o cálculo de η é a soma. Esse fenômeno é ocasionado pela necessidade de se calcular a norma de um vetor do sinal de entrada, $\|\mathbf{x}\|^2$, para se gerar o vetor aumentado $\underline{\mathbf{x}}$ nas topologias SOM e RBF MLP, conforme as Equações (2.20) e (3.38), que descrevem respectivamente as primeiras camadas de cada rede e conforme suas ilustrações, apresentadas pelas Figuras 2.4 e 3.4.

Reconsiderando a Equação (5.3), a realização do produto $\mathbf{W}_l^\intercal \mathbf{o}_{l-1}$ requer $\mathfrak{l}_{\text{COL}}(\mathbf{W}_l)$ produtos internos e, por extensão, $\mathfrak{l}_{\text{COL}}(\mathbf{W}_l)$ operações de somatório. Todas as topologias de rede devem implementar tal operação, mas em especial as redes SOM e RBF MLP também devem levar em conta o adicional de uma operação de somatório para calcular a norma, obtida através de

$$\|\mathbf{x}\|^2 = \sum_{m=1}^{M} x_m^2. \tag{5.6}$$

Dessa maneira, a quantidade de operações de somatório, η_5, fica condicional ao tipo de rede e pode ser enunciada como

$$\eta_5 = \begin{cases} 1 + \sum_{l=1}^{L-1} \mathfrak{l}_{\text{COL}}(\mathbf{W}_l), \text{ para redes SOM e MLP RBF,} \\ \\ \sum_{l=1}^{L-1} \mathfrak{l}_{\text{COL}}(\mathbf{W}_l), \text{ para as demais.} \end{cases} \tag{5.7}$$

Nas próximas seções, serão analisadas as operações matemáticas necessárias e exclusivas para a operação de cada rede neural. Essas operações são incluídas somente no cálculo da complexidade das redes com as quais elas estão relacionadas.

[1] Para o caso IVQ, os vetores de polarização da camada L são limiares de decisão (confira pela Equação (3.30)), ou seja, $\mathbf{b}_L = \underline{\mathbf{t}}$. A geração de limiares de decisão será tratada conjuntamente com a operação de limiar de decisão, $\mathfrak{s}(\cdot)$, razão pela qual não se considera no somatório a camada L.

Rede SOM

Como já observado, redes SOM exigem a implementação da operação de norma. Pela sua definição na Equação 5.6, segue que se exigem $\eta_6 = M$ operações quadráticas, além da já considerada operação de soma.

A última camada SOM implementa o bloco WTA (definido pela Equação (2.22)). Uma maneira menos complexa de implementá-lo consiste em simplificar o sinal de saída, passando do sinal $j(n)$, que é um número inteiro, para uma representação aumentada, $\underline{\mathbf{j}}(n)$. No caso WTA, o vetor $\underline{\mathbf{j}} \in \{0,1\}^K$ assume o formato

$$\left[\, \underline{j} \,\right]_k = \begin{cases} 1, \ k = j, \\ \\ 0, \ k \neq j, \end{cases} \qquad k = 1, \ldots, K, \tag{5.8}$$

o que permite implementar o bloco WTA com $\eta_7 = K$ sub-operações, como a definida pela Equação (5.8). A exemplo do caso IVQ, essa equação define uma nova implementação para o mapeamento $\pi : \{1, \ldots, K\} \to \{0,1\}^K$, agora para o caso ECVQ. Como justificado no Capítulo 3, o uso do sinal $\underline{\mathbf{j}}(n) = \pi(j(n))$ ao invés de $j(n)$ como saída da codificação não compromete a subseqüente operação VLC.

Redes MLP

As últimas camadas das redes MLP implementam o operador α para IVQ, ilustrado na Figura 3.2. A estrutura é modelável pela Equação (5.3), desde que se defina a operação degrau unitário, $\mathfrak{s}(\cdot)$, como sendo sua função de ativação. Uma vez que a estrutura da matriz de sinapses é trivial (representada por exemplo na Tabela 3.1), há motivação para se considerar o sinal $\mathbf{W}_L^{\mathsf{T}}\mathbf{o}_{L-1} \equiv \mathbf{u}'(n)$ e submetê-lo diretamente à operação $\mathfrak{s}(\mathbf{u}'(n) - \underline{\mathbf{t}})$, para se gerar o sinal de saída da rede. A operação $\mathfrak{s}(\mathbf{u}'(n) - \underline{\mathbf{t}})$ efetua T subcomparações com limiares de decisão, que são as entradas do vetor $\underline{\mathbf{t}}$. Novamente, a diferença de sinal para cada valor $\underline{t}_i$ será tratada, de modo que os números de comparações com limiares positivos e negativos são respectivamente iguais a:

$$\eta_8 = \mathrm{card}\left(\underline{t}_i \mid \underline{t}_i > 0\right), \tag{5.9a}$$

$$\eta_9 = \mathrm{card}\left(\underline{t}_i \mid \underline{t}_i < 0\right), \tag{5.9b}$$

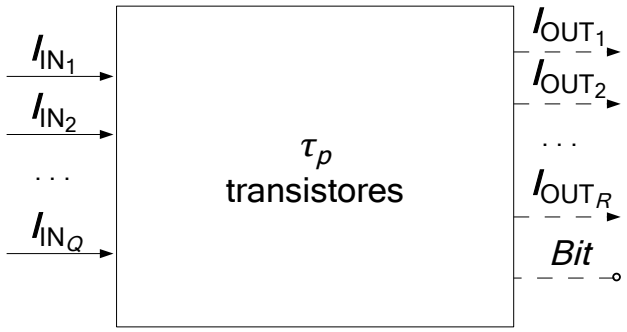


Figura 5.1: Bloco genérico para implementação da p-ésima operação matemática do codificador.

$$i = 1, \ldots, T.$$

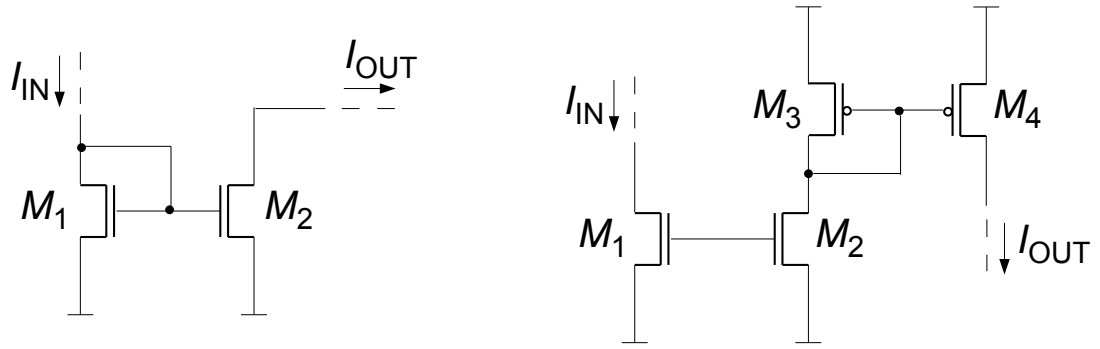
A primeira camada das redes KPCA MLP possuem diferentes funções de ativação, de acordo com o *kernel* utilizado. Em todos os casos, são calculadas $M + 1$ funções κ, exigindo as seguintes operações matemáticas para $f_1(\cdot)$:

- **HT PCA MLP:** Tangente hiperbólica, $\eta_{10} = M + 1$.
- **RBF PCA MLP:** Exponencial, $\eta_{11} = M + 1$. [2]
- **PCA Polinomial MLP:** Funções polinomiais de ordem r arbitrada. Nesse trabalho, serão consideras as ordens:
  - $r = 0,5$: (raiz quadrada) $\eta_{12} = M + 1$
  - $r = 3$: (cubo) $\eta_{13} = M + 1$

5.2 Parametrização da Complexidade em Transistores

Uma vez que se deseja atribuir uma única nota Θ de complexidade para cada codificador, deve-se somar a contribuição de cada operação matemática para a implementação do esquema de compressão. Isso não pode ser realizado sem que se conheça um peso relativo entre cada operação. Nesse trabalho, a finalidade prática é atribuída à compressão

[2] De modo análogo à rede SOM, deve-se também calcular $\|\mathbf{x}\|^2$, o que requer $\eta_6 = M$ operações quadráticas.



(a) (b)

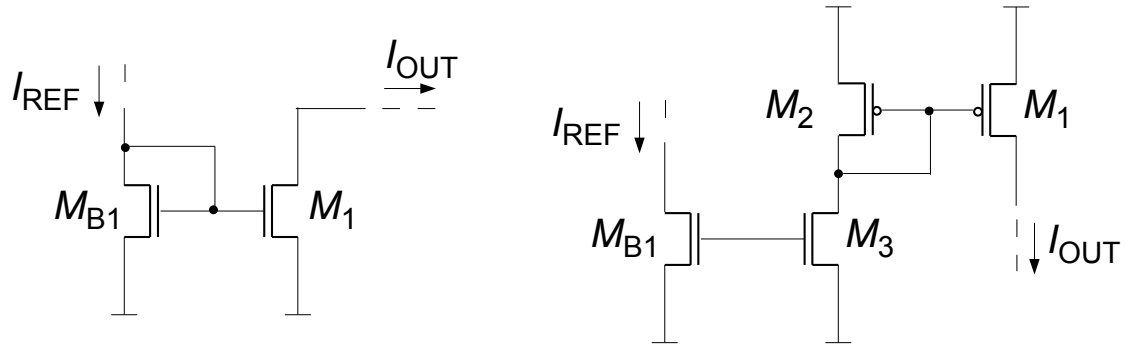
Figura 5.2: Implementação de multiplicação para (a) $[W_l]_{i_1 i_2} < 0$ e (b) $[W_l]_{i_1 i_2} > 0$.

de imagens em sensores fotográficos de câmeras portáteis, ainda no domínio analógico. Dessa forma, as operações matemáticas deverão ser implementadas por microcircuitos eletrônicos, de modo que todas as operações requeridas e descritas na Seção 5.1 para a implementação do codificador requerem como unidade básica transistores. O denominador comum para a ponderação, ou seja, os pesos τ_p, serão calculados através do número de transistores necessários para a realização de uma unidade da p-ésima operação. Uma vez que sensores fotográficos são dispositivos que podem ser modelados eletronicamente como fontes controladas de corrente, os sinais ao longo da topologia da rede serão correntes. Assim, um modelo genérico de bloco que implementa a p-ésima operação consiste em um circuito implementado por τ_p transistores (Figura 5.1). As Q correntes de entrada, $\{I_{\mathrm{IN}_1}, \ldots, I_{\mathrm{IN}_Q}\}$, serão em um número adequado para cada implementação de operação matemática, da mesma forma que o sinal de saída poderá ser composto por R correntes de saída, $\{I_{\mathrm{OUT}_1}, \ldots, I_{\mathrm{OUT}_R}\}$, ou opcionalmente no final da rede MLP, ser composto por um nível de tensão digital que represente um *bit*.

Ao longo dessa seção, serão apresentados esquemas de microcircuitos eletrônicos que efetuam os processamentos matemáticos necessários para a operação de codificação.

Multiplicação

As operações de multiplicação são implementadas por espelhos de corrente [12], de acordo com o sinal desejado para os fatores $[W_l]_{i_1 i_2}$, provenientes das matrizes de sinapses. Para o caso negativo, o amplificador é a configuração básica ilustrado na Figura 5.2a, de valor $\tau_2 = 2$ transistores. Já a multiplicação positiva requer a combinação de dois amplificadores



(a) (b)

Figura 5.3: Implementação das constantes de polarização para (a) $[b_l]_i < 0$ e (b) $[b_l]_i > 0$.

em série, para a inversão do sinal de saída, totalizando $\tau_1 = 4$ transistores (Figura 5.2b). Em ambos os circuitos, o fator $[W_l]_{i_1 i_2}$ desejado é obtido através do projeto da razão entre as *razões de aspecto* dos transistores. [3]

Constantes de Polarização

A geração de uma constante de polarização $[b_l]_i$ pode ser encarada como um caso particular da multiplicação, porém aplicada para uma corrente constante e unitária ao longo da operação. Assim, para se produzir constantes negativas, utiliza-se um amplificador de ganho negativo. Uma vez que uma constante pode ser implementada pela combinação de qualquer valor de corrente de referência I_{REF} e o ganho de amplificação, é possível utilizar a corrente I_{REF} em outras partes do circuito. Dessa forma, o transistor de polarização M_{B1} não é considerado para a contagem, de modo que o número de transistores é $\tau_4 = 1$. No caso de constantes positivas, o circuito é um amplificador de sinal positivo e novamente exclui-se da contagem o transistor M_{B1}, restando $\tau_3 = 3$ transistores para serem considerados. Os circuitos de implementação para os casos negativo e positivo são ilustrados, respectivamente, pelas Figuras 5.3a e 5.3b.

Somatório

O somatório de Q correntes, $I_{\mathrm{OUT}} = I_{\mathrm{IN}_1} + \ldots + I_{\mathrm{IN}_Q}$ pode ser interpretado como uma extensão da Lei de Kirchhoff para nós. O número Q não modifica a arquitetura do circuito

[3] A razão de aspecto de um transistor MOSFET é o quociente obtido pela divisão das dimensões de largura por comprimento de seu canal [22].

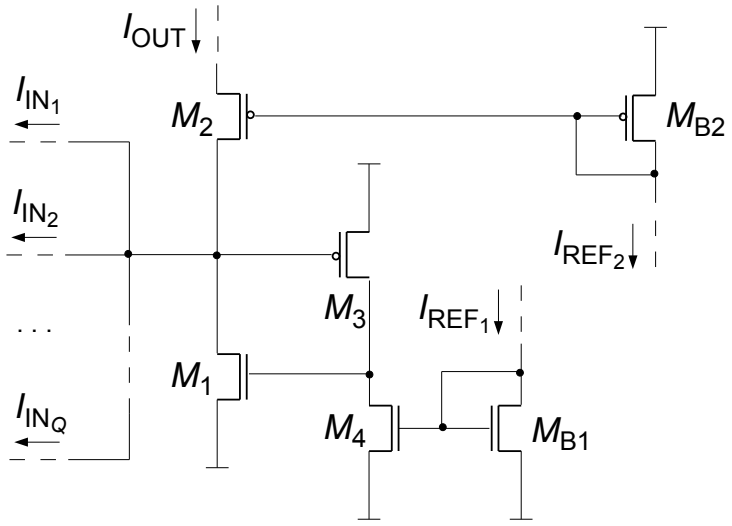


Figura 5.4: Circuito para implementação da soma $I_{\mathrm{OUT}} = I_{\mathrm{IN}_1} + \ldots + I_{\mathrm{IN}_Q}$.

(ver Figura 5.4), conhecida como *current conveyor* [5]. A função de $\{M_1, \ldots, M_4\}$ é assegurar que a tensão no nó de entrada (na porta de M_3) seja constante. O número de transistores necessários vale $\tau_5 = 4$.

Função Quadrática

O circuito proposto para o processamento da função quadrática, $I_{\mathrm{OUT}} = I_{\mathrm{IN}}^2$, é composto por $\tau_6 = 3$ transistores [3]. Pela análise da Figura 5.5, observa-se que o transistor de polarização M_{B1} tem como finalidade gerar uma tensão de referência em sua porta, razão pela qual é desconsiderado na contagem.

Bloco WTA

O bloco WTA é implementado por uma arquitetura ilustrada na Figura 5.6 [1]. Trata-se de um circuito composto por K células, lembrando que K é o tamanho do dicionário ECVQ. A k-ésima célula é composta por $\tau_7 = 2$ transistores, $\{M_{2k-1}, M_{2k}\}$. Das K correntes de entrada, somente a maior corrente, nomeada I_{IN_j}, se sobressairá, de tal modo que a corrente de saída I_{OUT_j} existirá e as demais I_{IN_k}, $k = 1, \ldots, K$, $k \neq j$ serão zero. Dessa forma, a saída desse circuito representa um vetor binário de tamanho K em que somente a j-ésima saída é não-nula, conforme a Equação (5.8).

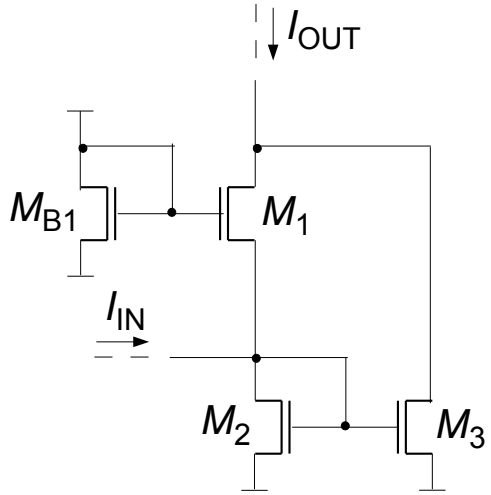


Figura 5.5: Circuito para cálculo do quadradro, $I_{\mathrm{OUT}} = I_{\mathrm{IN}}^2$.

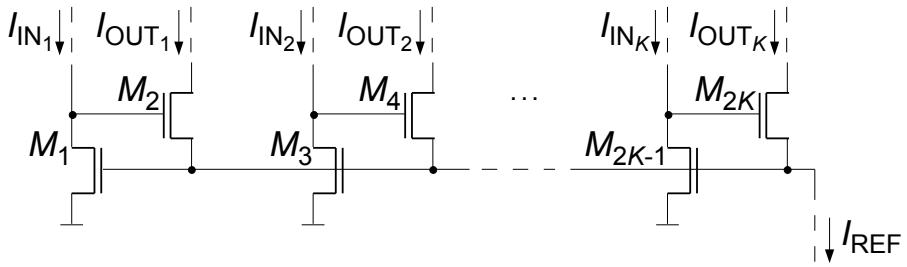


Figura 5.6: Circuito para implementação de um bloco WTA de tamanho K.

Comparação de Limiares

As funções de limiares de decisão produzem em sua saída níveis de tensão digitais, para produzir o sinal digital, com a finalidade de implementar a operação $\underline{\mathbf{j}}(n) = \mathfrak{s}\left(\mathbf{u}'(n) - \underline{\mathbf{t}}\right)$. Essa operação pode ser implementada por T blocos. Considere o caso do i-ésimo bloco, associado ao limiar de decisão $\underline{t}_i > 0$. Para esse caso o circuito de implementação, ilustrado pela Figura 5.7a [20], requer $\tau_8 = 3$ transistores para implementação. O inversor lógico digital formado pelos transistores $\{M_2, M_3\}$ fica em função da tensão no dreno de M_1 que, por sua vez, é função da corrente de entrada, I_{IN}, e do ganho gerado pelo par $\{M_{B1}, M_1\}$. No caso em que $\underline{t}_i < 0$, o circuito de implementação dual é apresentado na Figura 5.7b, também composta por $\tau_9 = 3$ transistores.

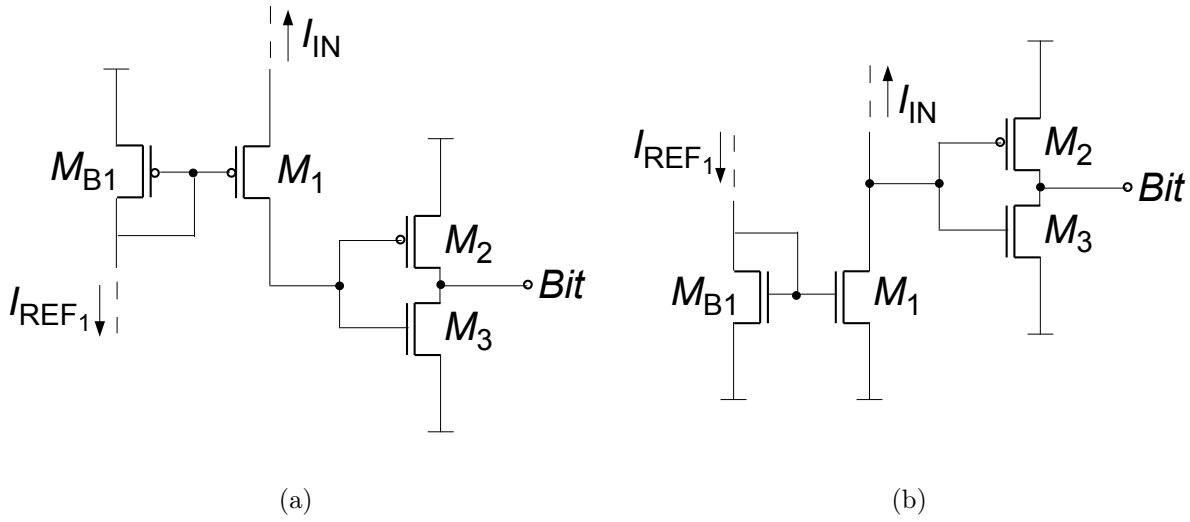


Figura 5.7: Implementação dos comparadores, para limiar (a) positivo e (b) negativo.

Tangente Hiperbólica

Na comparação com as operações anteriores, a operação de tangente hiperbólica requer uma arquitetura mais laboriosa (Figura 5.8). Pode-se inicializar a análise do circuito pelo amplificador diferencial implementado pelos transistores $\{M_3, \ldots, M_7\}$. Sua operação não-linear é modelável como uma sigmóide que se aproxima da curva da função tangente hiperbólica [22]. No entanto, o sinal de entrada do par diferencial deve ser tensão, de modo que um bloco para conversão do sinal de corrente para tensão se faz necessário. A implementação de tal conversor é realizada por M_1 e M_2.

A arquitetura resultante do amplificador diferencial e do conversor, composta pelos transistores $\{M_1, \ldots, M_7\}$ só opera para correntes I_{IN} positivas. Quando $I_{\text{IN}} < 0$, a saída do amplificador diferencial correlato vale zero. Assim, faz-se necessário construir o circuito dual, válido para correntes de entrada negativas ($I_{\text{IN}} < 0$). Esse circuito é implementado pelos transistores $\{M_8, \ldots, M_{14}\}$. Para unificar os sinais de saída dos dois amplificadores diferenciais, é necessário realizar a operação de soma, implementada pela arquitetura *current conveyor* com os transistores $\{M_{15}, \ldots, M_{18}\}$. Logo, chega-se à contagem de transistores $\tau_{10} = 18$.

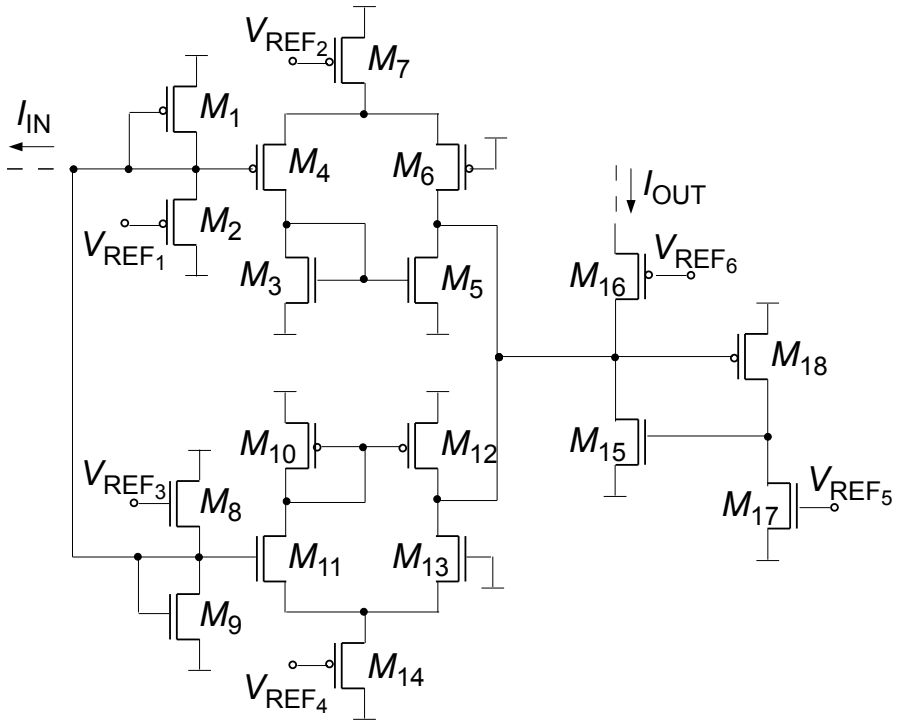


Figura 5.8: Circuito para o cálculo de tangente hiperbólica, $I_{\mathrm{OUT}} = \tanh(I_{\mathrm{IN}})$.

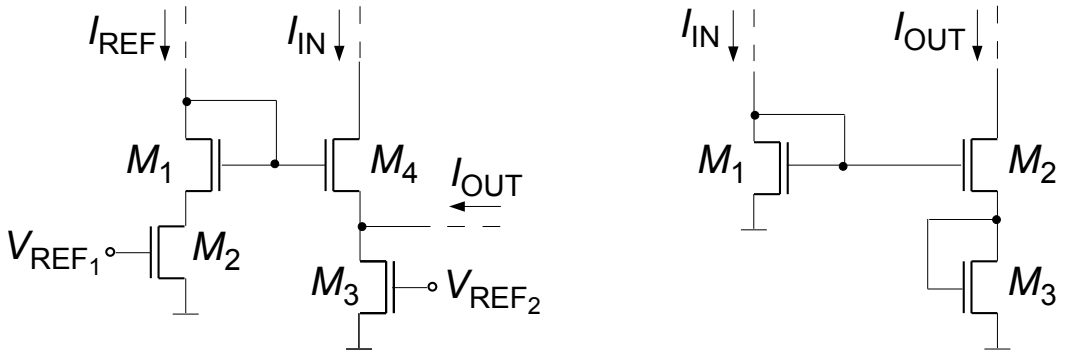
Funções Exponencial e Raiz Quadrada

Para a realização da função exponencial, é proposto um circuito em [18]. Sua estratégia é utilizar as características não-lineares do transistor, que pode ser modelado através de curvas exponenciais. Há a necessidade de $\tau_{11} = 2$ transistores. O circuito é apresentado na Figura 5.9a.

Um circuito que implementa raiz quadrada é apresentado em [20]. Sua apresentação é ilustrada pela Figura 5.9b e requer $\tau_{12} = 3$ transistores.

Função Cúbica

Por último, o circuito que eleva um sinal ao cubo é ilustrado na Figura 5.10. Optou-se por utilizar o circuito para multiplicação de correntes $I_{\mathrm{OUT}} = I_{\mathrm{IN}_1} \cdot I_{\mathrm{IN}_2}$ proposto por [3], conhecido como circuito de quatro quadrantes. Sua representação é ilustrada pela Figura 5.10c. Uma vez que se deseja implementar a operação $I_{\mathrm{OUT}} = I_{\mathrm{IN}}^3$, é necessário implementar uma fonte como $I_{\mathrm{IN}_1} = I_{\mathrm{IN}}$ e outra como $I_{\mathrm{IN}_2} = I_{\mathrm{IN}}^2$. No caso específico desse trabalho, a corrente de entrada I_{IN} é advinda da saída de um bloco de soma, que drena sua corrente de saída (Confira pelo sentido da corrente I_{OUT} do circuito da Figura 5.4). Logo,

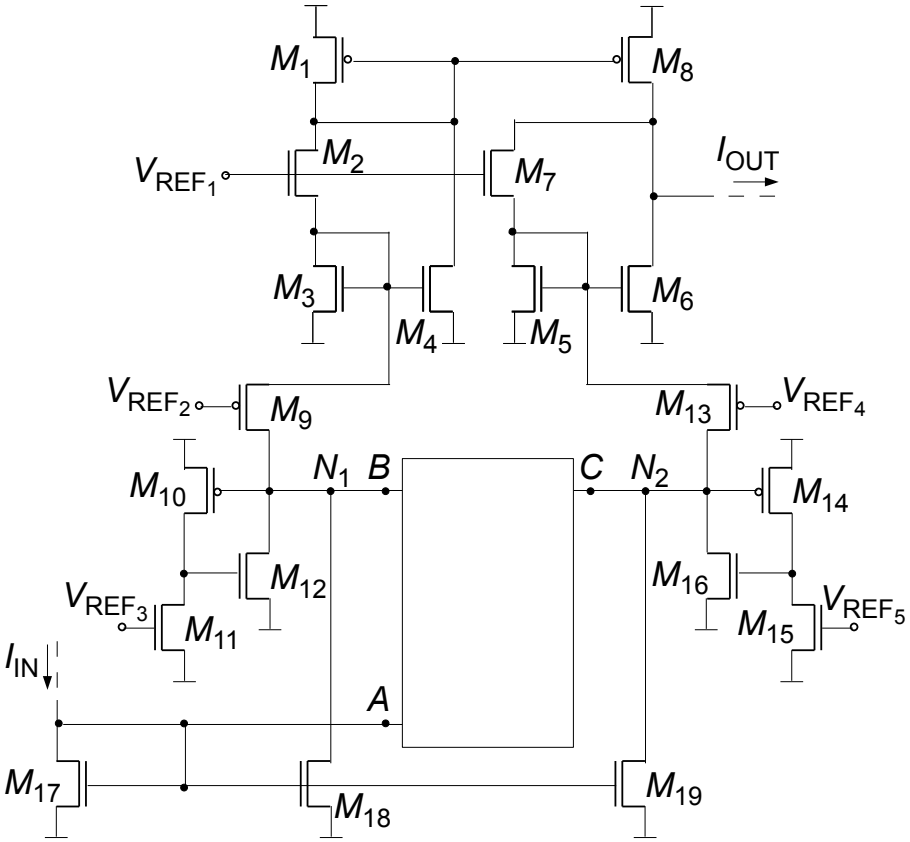


(a) (b)

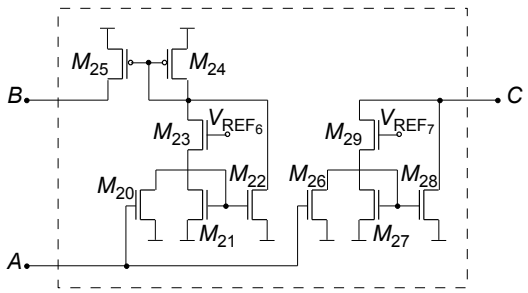
Figura 5.9: Circuitos para as implementações das operações (a) exponencial, $I_{\mathrm{OUT}} = \exp(I_{\mathrm{IN}})$, e (b) raiz quadrada, $I_{\mathrm{OUT}} = \sqrt{I_{\mathrm{IN}}}$.

deve-se submeter I_{IN} a um ganho de -1 para se obter a corrente I_{IN_1}. A implementação desse ganho é realizada pelos pares de transistores $\{M_{17}, M_{18}\}$ e $\{M_{17}, M_{19}\}$.

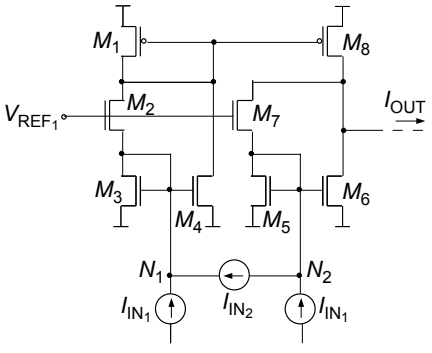
A geração da fonte de corrente $I_{\mathrm{IN}_2} = I_{\mathrm{IN}}^2$ é implementada pelo circuito de função quadrática, já analisado e ilustrado pela Figura 5.5. Uma vez que a fonte de corrente I_{IN_2} possuiu seus dois terminais ligados aos nós N_1 e N_2, para a utilização do circuito de função quadrática, é necessário gerar duas correntes de valor I_{IN}^2, a primeira para a injeção no nó N_1 e a segunda para a drenagem do nó N_2. A fonte de corrente I_{IN_2} resultante é implementada pelo circuito de terminais ABC, ilustrado na Figura 5.10b. Os conjuntos de transistores $\{M_{20}, \ldots, M_{23}\}$ e $\{M_{26}, \ldots, M_{29}\}$ implementam a operação quadrática. Ao analisar o circuito quadrático (Figura 5.5), nota-se que o sentido da corrente de saída é preferencial para a implementação da corrente que é drenada pelo nó N_2, ou seja, $-I_{\mathrm{IN}}^2$. Assim, para a geração da corrente que é injetada no nó N_1, é necessário a inversão do sinal, realizada pelos transistores $\{M_{24}, M_{25}\}$. Finalmente, as somas das correntes nos nós N_1 e N_2 utilizam *current conveyors*, implementados por $\{M_9, \ldots, M_{12}\}$ e $\{M_{13}, \ldots, M_{16}\}$. O número resultante de transistores para implementar a função cúbica é $\eta_{13} = 29$.



(a) Circuito geral



(b) Subcircuito com terminais ABC



(c) O circuito de quatro quadrantes, $I_{\mathrm{OUT}} = I_{\mathrm{IN}_1} \cdot I_{\mathrm{IN}_2}$.

Figura 5.10: Circuito para o cálculo do cubo, $I_{\mathrm{OUT}} = I_{\mathrm{IN}}^3$.

Tabela 5.1: Sumário da Métrica de Complexidade

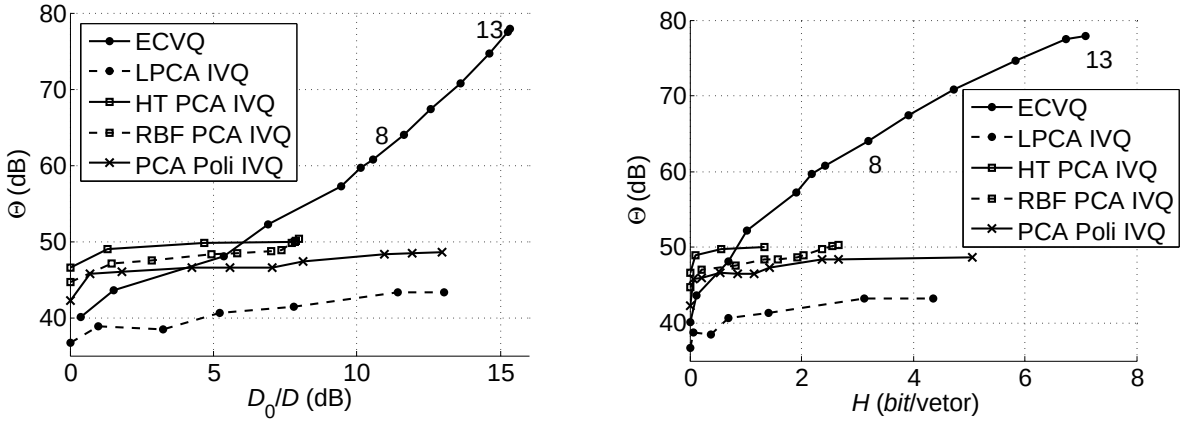
Operação	p	η_p	τ_p
Multiplicação (+)	1	Equação (5.4a)	4
Multiplicação (−)	2	Equação (5.4b)	2
Polarização (+)	3	Equação (5.5a)	3
Polarização (−)	4	Equação (5.5b)	1
Somatório	5	Equação (5.7)	4
Quadrado	6	M (redes SOM e RBF MLP)	3
WTA	7	K (rede SOM)	2
Comparação com limiar (+)	8	Equação (5.9a) (redes MLP)	3
Comparação com limiar (−)	9	Equação (5.9b) (redes MLP)	3
Tangente hiperbólica	10	$M+1$ (rede HT MLP)	18
Exponencial	11	$M+1$ (rede RBF MLP)	4
Raiz quadrada	12	$M+1$ (rede MLP Polinomial)	3
Cubo	13	$M+1$ (rede MLP Polinomial)	29

5.3 Avaliação da Complexidade em Sistemas de Compressão de Imagens

A métrica exposta nas Seções 5.1 e 5.2 é resumida pela Tabela 5.1. De posse dessa nova ferramenta, é possível iniciar a análise dos esquemas de compressão de imagens projetados no Capítulo 4, em termos de complexidade. Com a finalidade de avaliar os sistemas de quantização vetorial comparativamente, foram submetidos à métrica de complexidade os sistemas ECVQ e os sistemas IVQ. Uma vez que o desempenho dos sistemas IVQ-2 são superiores aos IVQ-1 mantendo a mesma estrutura, serão considerados aqui somente os sistemas IVQ-2. Dessa forma, serão considerados os sistemas ECVQ, LPCA MLP IVQ-2, HT MLP IVQ-2, RBF MLP IVQ-2 e MLP Polinomial IVQ-2.

A Figura 5.11 apresenta o comportamento da complexidade em função do desempenho do sistema em termos de distorção (Figura 5.11a) e entropia e (Figura 5.11b).

Para se investigar a vantagem de se utilizar IVQ ao invés de ECVQ, são propostas duas análises: para uma mesma entropia, quanto se perde em distorção com a redução de complexidade utilizando esquemas IVQ e, para uma mesma distorção, quanto se perde



(a) (b)

Figura 5.11: Avaliação da complexidade em função do desempenho em termos de (a) distorção e (b) entropia. Note que os quantizadores ECVQ rotulados pelos índices 8 e 13 na curva ECVQ da Figura (a) são os mesmos quantizadores identificados com os respectivos índices na Figura (b). De modo análogo, esse raciocínio é estendido para os demais quantizadores e para as demais curvas.

em entropia ao se utilizar esquemas menos complexos. A Tabela 5.2 avalia a perda de qualidade em termos de aumento de distorção em detrimento da redução de complexidade. Foram utilizados como dados sistemas IVQ do tipo LPCA. A tabela mostra que a perda de qualidade é desprezível $(0,22$ dB) para um ganho de 20 dB de complexidade. Essa constatação revela que esquemas IVQ de fato atingem desempenhos comparáveis ao ECVQ, mas com significativa redução de complexidade.

Pela análise da Figura 5.11, constata-se que o comportamento da complexidade é de aumentar à medida que sistemas com melhor desempenho em termos de distorção e en-

Tabela 5.2: Comparação entre ECVQ e IVQ para entropia equivalente

H_{ECVQ}	H_{IVQ}	Θ_{ECVQ}	$\Theta_{\mathrm{ECVQ}}/\Theta_{\mathrm{IVQ}}$	$(D_{\mathrm{IVQ}}/D_{\mathrm{ECVQ}})$
bit/vetor	bit/vetor	dB	dB	dB
0,683	0,696	48,0967	7,509	0,153
3,198	3,132	63,973	20,686	0,220

Tabela 5.3: Comparação entre ECVQ e IVQ para distorção equivalente

$(D_0/D)_{\mathrm{ECVQ}}$ dB	$(D_0/D)_{\mathrm{IVQ}}$ dB	Θ_{ECVQ} dB	$\Theta_{\mathrm{ECVQ}}/\Theta_{\mathrm{IVQ}}$ dB	$H_{\mathrm{ECVQ}}/H_{\mathrm{IVQ}}$
5,378	5,225	48,097	7,509	0,982
11,630	11,409	63,973	20,686	1,021
13,624	13,053	70,799	27,512	1,085

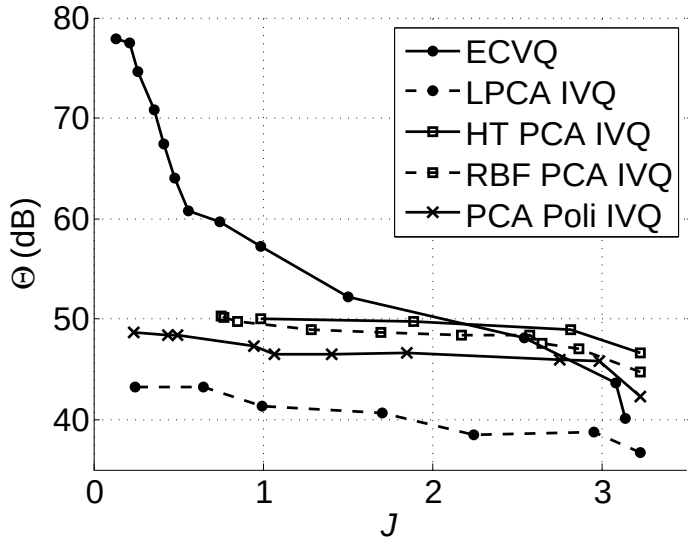


Figura 5.12: Avaliação da complexidade em função do custo J.

tropia são considerados. Com relação à comparação entre a complexidade de codificadores ECVQ e IVQ, o gráfico justifica o uso de técnicas de projeto mais sofisticadas, caso da IVQ, para se projetar sistemas de satisfatórios desempenhos com significativas reduções de complexidade.

No cenário de iguais distorções, a comparação entre a entropia de diferentes sistemas (perda menor que $1,1$), como apresentado pela Tabela 5.3, prova que a redução maior de 27 dB em complexidade foi conseguida sem relevante sacrifício de aumento de entropia.

Por fim, a Figura 5.12 apresenta o comportamento da complexidade dos sistemas Q em função dos respectivos custos $J_Q = D_Q + \lambda_Q H_Q$. A relevância dessa análise está no fato de que o projeto de quantização vetorial é guiado através da minimização de J. O comportamento das curvas confirma a tese de que sistemas de menor custo são projetados

através de codificadores com maiores complexidades [11].

Capítulo 6

Conclusão

Este trabalho objetivou analisar esquemas de compressão de imagens para implementação no plano focal. A nível de implementação em dispositivos portáteis, o grau de complexidade se torna uma consideração crítica, de modo que este estudo se guiou na proposição de esquemas que alcançassem desempenho em termos de taxa e distorção a nível de estado-da-arte, mas que mantivessem a restrição de complexidade.

No **Capítulo 2**, foi estudada ECVQ, técnica de quantização que leva em conta no seu projeto o compromisso entropia-distorção. O uso de IVQ foi analisado no **Capítulo 3**, para a implementação de esquemas de baixa complexidade. A adoção de Redes Neurais MLP como sua arquiteura foi modelada. Além disso, a extensão do método de projeto, com a proposição do algoritmo de otimização MLP, foi desenvolvida com a finalidade de aproximar o desempenho de IVQ a ECVQ.

A comparação de desempenho, em termos de taxa e distorção, entre compressores de imagens implementados por IVQ e ECVQ foi realizada no **Capítulo 4**, demonstrando que a evolução algorítmica do MLP OA de fato melhora significativamente o desempenho IVQ, tornando-o equivalente ao ECVQ. Finalmente, o **Capítulo 5** apresentou uma métrica para a avaliação da complexidade de compressores de imagens. Quando aplicada para a comparação de complexidade entre a codificação IVQ e ECVQ, confirmou-se que os compressores IVQ possuem desempenho equivalente aos ECVQ, mas com um nível de complexidade inferior.

Logo, as principais contribuições desse trabalho consistiram em

- Desenvolver métodos de projeto IVQ com desempenho equivalente a ECVQ;
- Fornecer subsídios que permitam ao projetista adequar as especificidades de sua

aplicação mediante uma análise RDC.

6.1 Diretivas Futuras

Os resultados alcançados pelo presente trabalho confluem aos seguintes tópicos para pesquisas futuras:

- Investigação de métodos de otimização KPCA mais robustos a parâmetros de inicialização arbitráveis;
- Simulação de compressores para taxas superiores, para verificação de limites de desempenho;
- Análise de sensibilidade da codificação, levando em conta as imprecisões inerentes aos projetos de microcircuitos eletrônicos;
- Proposição de novos esquemas de pré-processamento, para a compressão de imagens.

Apêndice A

Lista de Símbolos e Siglas

As siglas e os principais símbolos matemáticos utilizados nesta dissertação são especificados neste apêndice.

A.1 Lista de Símbolos

$\mathbb{R}$	Conjunto dos números reais
$\mathbf{A}^\mathsf{T}$	A matriz transposta de $\mathbf{A}$
$\|\mathbf{x}\|^2$	Norma do vetor $\mathbf{x}$
$a \triangleq b$	A entidade a é definida como b
$\mathbf{0}$	Matriz ou vetor de entradas nulas
$\mathbf{0}_T$	Matriz de tamanho $T \times T$ de entradas nulas
$\mathbf{1}$	Matriz ou vetor de entradas unitárias
$\mathbf{1}_T$	Matriz de tamanho $T \times T$ de entradas unitárias
$\mathbf{I}$	Matriz identidade
$\mathbf{I}_T$	Matriz identidade de tamanho $T \times T$

$\mathrm{vec}(\mathbf{A})$	Vetor-coluna cujas entradas são os coeficientes da matriz $\mathbf{A}$
$\mathrm{diag}(a, b, \ldots, z)$	Matriz diagonal cuja diagonal principal é composta pelos escalares $a, b, \ldots, z$
$\mathfrak{C}(k)$	Comprimento do código VLC que representa o índice de valor k
$\mathrm{card}(\mathcal{X})$	Cardinalidade ou quantidade de elementos do conjunto $\mathcal{X}$
$\mathfrak{l}(\mathbf{x})$	Tamanho do vetor $\mathbf{x}$
$\mathfrak{l}_{\mathrm{ROW}}(\mathbf{X})$	Número de linhas da matriz $\mathbf{X}$
$\mathfrak{l}_{\mathrm{COL}}(\mathbf{X})$	Número de colunas da matriz $\mathbf{X}$
$\mathfrak{s}(\cdot)$	Função degrau unitário (definição na Equação (3.6))
$\exp(\cdot)$	Função exponencial, $\exp(x) = e^x$
$\tanh(\cdot)$	Função tangente hiperbólica, $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
$\mathrm{std}(\cdot)$	Desvio padrão de um conjunto de vetores $\mathcal{X}$, definido por $\sqrt{D_0}$, com D_0 dado pela Equação (4.7).
$\kappa(\mathbf{x}, \mathbf{l})$	Função de *kernel*, $\kappa : (\mathbb{R}^M, \mathbb{R}^M) \rightarrow \mathbb{R}^M$
$\mathfrak{h}(\mathcal{Q})$	Critério LCH aplicado ao conjunto de quantizadores $\mathcal{Q}$

A.2 Lista de Siglas

CMOS	*Complementary Metal-Oxide Silicon*
DCT	*Discrete Cossine Transform*
DPCM	*Differential Pulse-Codem Modulation*
ECVQ	*Entropy-Constrained Vector Quantization*
GLA	*Generalized Lloyd Algorithm*
IVQ	*Interpolative Vector Quantization*
IVQA	*Interpolative Vector Quantization Algorithm*
HT	*Hiperbolic Tangent*
KPCA	*Kernel Principal Component Analysis*
LCH	*Lower Convex Hull*
LPCA	*Linear Principal Component Analysis*

MLP	*Multilayer Perceptron*
MLP OA	*Multilayer Perceptron Optimization Algorithm*
MSE	*Mean Squared Error*
PCA	*Principal Component Analysis*
PSNR	*Peak Signal-to-Noise Ratio*
RBF	*Radial Basis Function*
RDC	*Rate, Distortion and Complexity*
SOM	*Self-Organized Maps*
SQ	*Scalar Quantization*
SQNR	*Signal-to-Quantization Noise Ratio*
VLC	*Variable-Length Coder*
WTA	*Winner Takes All*

Apêndice B

O Critério da Casca Convexa Inferior

A solução para o problema que define o projeto de quantização vetorial,

$$Q' = \arg\min_{J} Q, \tag{B.1}$$

deve ser resolvida através de algoritmos de otimização, porque as Condições da Partição e do Centróide (Equações (2.16) e (2.17)), que definem as operações de codificação e decodificação para a quantização vetorial ótima, são funções não-analíticas. Lembrando que a expressão do custo é dada pela função lagrangeana $J = D + \lambda H$, deseja-se investigar como o multiplicador de Lagrange λ influencia na solução numérica da Equação (B.1), que é obtida através de otimização. Considere, por hipótese, que a distorção $D(Q)$ e a entropia $H(Q)$ de um determinado quantizador vetorial Q sejam funções analíticas. Por extensão, o custo $J(Q)$ será analítico e o quantizador Q', solução para a Equação (B.1), é dada por

$$\left(\frac{\partial J}{\partial Q}\right)_{Q=Q'} = 0. \tag{B.2}$$

Exprimindo J em função de D e H:

$$\begin{aligned}
&\frac{\partial J}{\partial Q} = \frac{\partial}{\partial Q} \cdot \big(D(Q) + \lambda H(Q)\big) = \frac{\partial D}{\partial Q} + \lambda \frac{\partial H}{\partial Q} \\
&\Rightarrow \left(\frac{\partial D}{\partial Q} + \lambda \frac{\partial H}{\partial Q}\right)_{Q=Q'} = 0 \\
&\Rightarrow \left(\frac{\partial D}{\partial Q}\right)_{Q=Q'} = \left(-\lambda \frac{\partial H}{\partial Q}\right)_{Q=Q'} \\
&\therefore \left(\frac{\partial D}{\partial H}\right)_{Q=Q'} = -\lambda.
\end{aligned} \tag{B.3}$$

Esse resultado não é prático para fins de cálculo de λ, pois leva em conta a hipótese de que $J(Q)$ é analítica. Todavia, a expressão $\lambda = -\partial D/\partial H$ permite ser aproximada através

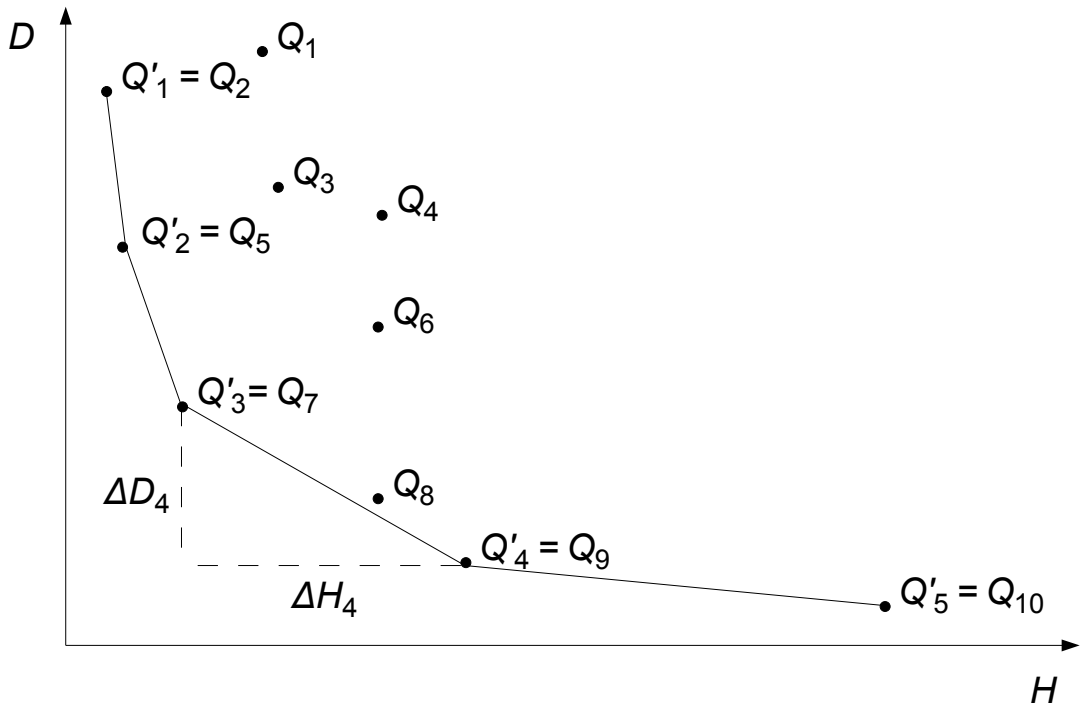


Figura B.1: Exemplo de seleção de quantizadores pelo critério da casca convexa inferior. O conjunto de $I = 10$ quantizadores $\mathcal{Q} = \{Q_1, \ldots, Q_{10}\}$ define uma curva na qual somente os sistemas $\mathcal{Q}' = \mathfrak{h}(\mathcal{Q})$ são mantidos. Particularmente, o valor de λ_4, associado ao quantizador Q'_4, é calculado pela relação $-\Delta D_4/\Delta H_4$, onde $\Delta D_4 = D_4 - D_3$ e $\Delta H_4 = H_4 - H_3$.

de uma solução geométrica conhecida como "casca convexa inferior" (*lower convex hull* – LCH).

Considere um conjunto de I quantizadores vetoriais $\mathcal{Q} = \{Q_1, \ldots, Q_I\}$. Cada quantizador $Q_i \in \mathcal{Q}$ é associado a um par $(D, H)_i$, que especifica o seu desempenho em termos de distorção e entropia. Geometricamente, é possível selecionar os melhores sistemas através da LCH, como exemplificado pela Figura B.1. Os I' pontos da curva que definem a casca são aqueles que possuem o melhor compromisso (D, H) dentre os sistemas do conjunto $\mathcal{Q}$, de modo que se define um critério de seleção dos melhores quantizadores $\mathcal{Q}' \subset \mathcal{Q}$:

$$\mathcal{Q}' \subset \mathcal{Q} \mid \mathcal{Q}' = \mathfrak{h}(\mathcal{Q}), \tag{B.4}$$

onde o operador $\mathfrak{h}(\cdot)$ recebe o nome de "critério LCH".

Dessa forma, a curva da casca pode ser utilizada para associar a cada quantizador $\mathcal{Q}'_i$

um λ_i *local*, através do cálculo da inclinação de cada reta que forma a casca:

$$\begin{aligned} \lambda_i &= -\frac{D_i - D_{i-1}}{H_i - H_{i-1}},\ i = 2, \ldots, I', \\ \lambda_1 &= \lambda_2. \end{aligned} \tag{B.5}$$

A Equação (B.5) torna-se uma útil ferramenta para o projeto de quantizadores vetoriais, pois permite inicializar os algoritmos de otimização com um quantizador vetorial Q'_i e tomar como valor de λ, para o cálculo do custo J, o respectivo valor λ_i.

Referências Bibliográficas

[1] A. G. Andreou, K. A. Boahen e P. O. Pouliquen. Current-mode subthreshold MOS for analog VLSI neural systems. *IEEE Transactions of Neural Networks*, 2(2):205–213, março de 1991.

[2] R. E. Blahut. Computation of channel capacity and rate-distortion functions. *IEEE Transactions on Information Theory*, IT-8(4):460–473, julho de 1972.

[3] K. Bult e H. Wallinga. A class of analog CMOS circuits based on the square-law characteristic of an MOS transistor in saturation. *IEEE Journal of Solid-State Circuits*, SC-22(3):357-365, junho de 1987.

[4] P. A. Chou, T. Lookabaugh e R. M. Gray. Entropy-Constrained Vector Quantization. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(1):31-42, janeiro de 1989.

[5] G. Ferri e N. C. Guerrini. *Low-Voltage Low-Power CMOS Current Conveyors*. Kluwer, Londres, Reino Unido, 2003.

[6] B. Foo, Y. Andreopoulos e M. van der Schaar. Analytical rate-distortion-complexity modeling of wavelet-based video coders. *IEEE Transactions on Signal Processing*, 56(2):797-815, fevereiro de 2008.

[7] A. Gersho. Optimal nonlinear Interpolative Vector Quantization. *IEEE Transactions on Communications*, 38(9):1285-1287, setembro de 1990.

[8] A. Gersho e R. M. Gray. *Vector Quantization and Signal Compression*. Kluwer, Boston, Estados Unidos, 1992.

[9] J. G. R. C. Gomes e S. K. Mitra. Kernel PCA for quantization of analog vectors on a pyramid. *IEEE International Workshop on Neural Networks for Signal Processing*, páginas 597 - 606, Toulouse, França, setembro de 2003.

[10] J. G. R. C. Gomes. *Mixed-Signal Multilayer Perceptron Implementation of Low-Complexity Vector Quantizers for Image Compression*. Tese de Doutorado, Universidade da Califórnia, Santa Bárbara, Estados Unidos, 2004.

[11] J. G. R. C. Gomes e S. K. Mitra. A comparative study of the complexities of Neural Networks based focal-plane image compression schemes. *The IECE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, J88-A(11):1185 - 1196, novembro de 2005.

[12] H. L. Haas, J. G. R. C. Gomes e A. Petraglia. Analog hardware implementation of a vector quantizer for focal-plane image compression. *Proceedings of the 21st Symposium on Integrated Circuits and Systems Design*, Gramado. Artigo aceito para publicação em maio de 2008.

[13] S. Haykin. *Neural Networks - A Comprehensive Foundation*. Prentice Hall, Nova Jersey, Estados Unidos, 1999.

[14] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall, Nova Jersey, Estados Unidos, 1992.

[15] T. Kohonen. *Self-Organizing Maps*. Springer-Verlag, Berlim, Alemanha, 2001.

[16] J. C. Lagarias, J. A. Reeds, M. H. Wright e P. E. Wright. Convergence properties of the Nelder Mead simplex method in low dimensions. *SIMA Journal of Optimization*, 9(1):112-147, dezembro de 1998.

[17] Y. Linde, A. Buzo e R. M. Gray. An algorithm for Vector Quantizer design. *IEEE Transactions on Communications*, COM-28(1):84-95, janeiro de 1980.

[18] J. Madrenas, M. Verleysen, P. Thissen e J. L. Voz. A CMOS analog circuit for Gausian functions. *IEEE Transactions on Circuits and Systems II*, 43(1):70-74, janeiro de 1996.

[19] H. S. Malvar, A. Hallapuro, M. Karczewicz e L. Kerofsky. Low-complexity transform and quantization in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):598 - 603, julho de 2003.

[20] C. Mead. *Analog VLSI and Neural Systems*. Addison-Wesley, Ontário, Canadá, 1989.

[21] B. Schölkopf, A. Smola e K. R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299 - 1319, 1998.

[22] A. S. Sedra e K. C. Smith. *Microeletrônica*. Makron Books, São Paulo, 2004.

[23] Y. Tsividis. Mixed-domain systems and signal processing based on input decomposition. *IEEE Transactions on Circuits Systems I*, 53(4):2145-2156, outubro de 2006.