

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta a fundamentação teórica, ou seja, o estudo das técnicas e algoritmos utilizados durante o trabalho. O capítulo divide-se em três seções.

A seção 2.1 apresenta técnicas e algoritmos que serão úteis para realizar a segmentação de imagens. A segmentação de imagens consiste em subdividir a imagem original em regiões ou objetos até que se tenha identificado a região ou objeto de interesse.

A seção 2.2 apresenta técnicas e algoritmos para realizar a descrição de imagens. Após a segmentação de uma imagem é necessário representar a imagem para que se possa processá-la. Existem duas abordagens para esse problema (GONZALEZ; WOODS, 2006): representação por meio de características externas da região, como bordas, ou representação por meio de características internas, como textura. A representação por características externas normalmente é utilizada quando se está interessado na forma da região. Já a representação por características internas é utilizada quando o foco está na cor ou textura da região. Nessa seção aborda-se principalmente a representação por meio de características externas de imagens.

Finalmente a seção 2.3 apresenta uma visão geral de reconhecimento de padrões utilizando redes neurais. O reconhecimento de padrões é feito tomando como base descritores, que podem ser obtidos a partir de técnicas como as vistas na seção 2.2.

2.1 SEGMENTAÇÃO

Esta seção apresenta técnicas e algoritmos para realizar a segmentação de imagens, tais como técnicas de subtração de fundo, limiarização, operações morfológicas, esqueletonização e análise de componentes principais.

2.1.1 Subtração de fundo

Técnicas de subtração de fundo são amplamente empregadas para a detecção de movimento em sequências de imagens de vídeo. A detecção do movimento se dá pela subtração da imagem

de referencia chamada "fundo", ou em inglês *background image*, da imagem a ser segmentada. A imagem de fundo não deve possuir objetos em movimento e deve estar sempre atualizada com relação à variações de iluminação (PICCARDI, 2004).

O método mais simples de subtração de fundo consiste em subtrair a imagem de fundo da imagem a ser segmentada e em seguida aplicar um limiar no resultado. Pode-se resumir esse método na equação 1. Onde, $D(t+1)$ é a imagem segmentada, $V(x,y,t+1)$ é a imagem de entrada, $V(x,y,t)$ é a imagem de referencia, ou fundo e Th é um limiar, que pode ser escolhido conforme os métodos apresentados na seção 2.1.2.

$$D(t+1) = |V(x,y,t+1) - V(x,y,t)| > Th \quad (1)$$

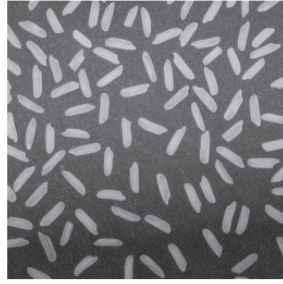
Existem outros métodos mais complexos, como *Running Gaussian average*, *Temporal median filter*, entre outros, que geram melhores resultados com fundos que não são completamente estáticos (PICCARDI, 2004), porém não são de interesse para este projeto.

2.1.2 Limiarização

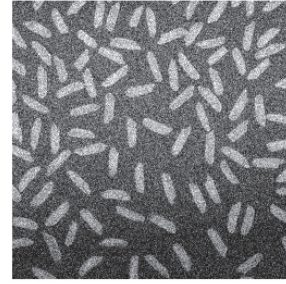
Pela simplicidade na implementação e pela baixa complexidade computacional os algoritmos de limiarização são bastante utilizados na segmentação de imagens. Os métodos de limiarização consistem na partição da imagem baseado na intensidade dos pixels. Caso a partição seja feita com base em um único limiar de intensidade a limiarização se diz global. Caso a partição seja feita levando em conta os valores de intensidade dos pixels em uma região de vizinhança de um certo pixel, então o método é chamado de limiarização local (GONZALEZ; WOODS, 2006). A equação 2 descreve esse processo, $g(x,y)$ é a imagem limiarizada, $f(x,y)$ é a imagem de entrada e T é um limiar, que pode variar para diferentes valores de x e y caso a limiarização seja local.

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{if } f(x,y) \leq T \end{cases} \quad (2)$$

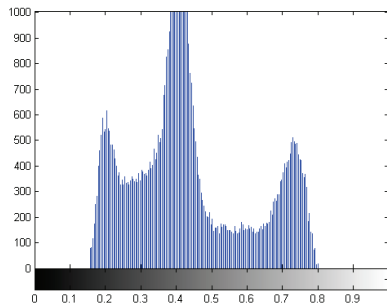
Na figura 1 mostra-se duas imagens e seus respectivos histogramas. A figura 1c mostra o histograma para a imagem sem ruído e a figura 1d mostra o histograma para a imagem com ruído gaussiano. Como pode-se ver na figura 1c, a escolha de um valor para o limiar T para a imagem sem ruído é uma tarefa fácil, porém a escolha do limiar para a imagem com ruído da figura 1d não é mais trivial. A seguir descreve-se um métodos para obtenção de limiar para limiarização global que pode ser aplicado nos dois casos.



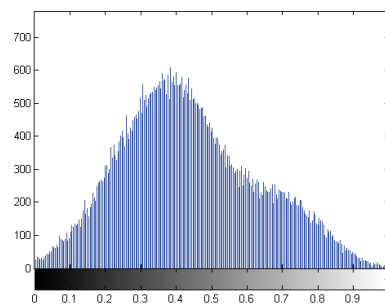
(a) Imagem sem ruído

Fonte: (MATHWORKS, 2013a)

(b) Imagem com ruído

Fonte: Autoria própria

(c) Histograma da imagem sem ruído

Fonte: Autoria própria

(d) Histograma da imagem com ruído

Fonte: Autoria própria**Figura 1:** Imagens com e sem ruído e seus respectivos histogramas

Limiarização global pelo método de *Otsu*

O método para determinar o valor do limiar T proposto por *Otsu* é um método ótimo que busca maximizar a variância entre as duas classes. Isso baseia-se no fato de que classes bem divididas possuem valores de intensidades distintos, e o valor ótimo para T é o valor tal que haja maior separação entre as intensidades (GONZALEZ; WOODS, 2006).

O cálculo do valor limiar pelo método de *Otsu* se dá pela equação 3.

$$\sigma_B^2(t) = \frac{[m_G P_1(t) - m(t)]^2}{P_1(t) [1 - P_1(t)]} \quad (3)$$

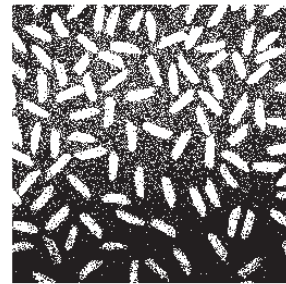
Onde:

- $\sigma_B^2(t)$ é a variância entre as classes para um valor limiar t ;
- m_G é a média global de intensidades da imagem;

- $P_1(t)$ é a probabilidade de um pixel qualquer da imagem estar na classe 1 dado o valor limiar t .
- $m(t)$ é a média acumulada das intensidades de $t = 0$ até a intensidade t ;

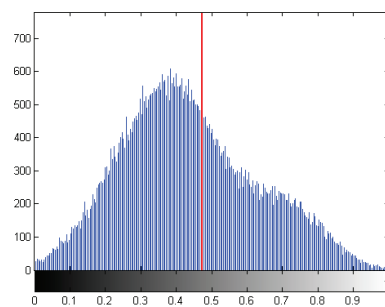
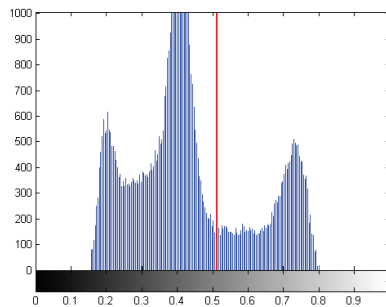
Portanto, para o cálculo do valor limiar pelo método de *Otsu* basta encontrar o valor limiar t tal que a variância entre as classes, $\sigma_B^2(t)$, é máximo. A dedução da equação 3 está fora do escopo deste projeto, mas pode ser encontrada em (GONZALEZ; WOODS, 2006).

Na figura 2 mostram-se as imagens da figura 1 limiarizadas pelo método de *Otsu* e seus respectivos valores de limiarização.



(a) Imagem sem ruído limiarizada

(b) Imagem com ruído limiarizada



(c) Histograma da imagem sem ruído com o valor do limiar T

(d) Histograma da imagem com ruído com o valor do limiar T

Figura 2: Imagens limiarizadas pelo método de *Otsu* e seus respectivos histogramas

Fonte: Autoria própria

2.1.3 Operações morfológicas básicas

Operações morfológicas como abertura e fechamento podem ser úteis para filtrar imagens. Principalmente para filtrar imagens binárias onde a operação de fechamento pode ser aplicada para diminuir ruído.

Para o estudo das operações morfológicas de abertura e fechamento precisa-se primeiramente esclarecer alguns pontos:

- Podemos representar imagens binárias como sendo um conjunto de pixels no espaço Z^2 , onde cada pixel é representado por uma tupla (x, y) e x e y são as coordenadas do pixel.
- A translação de um conjunto B para um ponto $z = (z_1, z_2)$, denotado por $(B)_z$ é o conjunto de pontos em B com as respectivas coordenadas (x, y) tendo sido substituídas pela coordenadas $(x + z_1, y + z_2)$.
- Elemento estruturante é um conjunto de pixels que é utilizado para fazer operações sobre uma imagem. Cada elemento estruturante possui um pixel de origem. A figura 3 mostra exemplos de elementos estruturantes e suas respectivas origens.

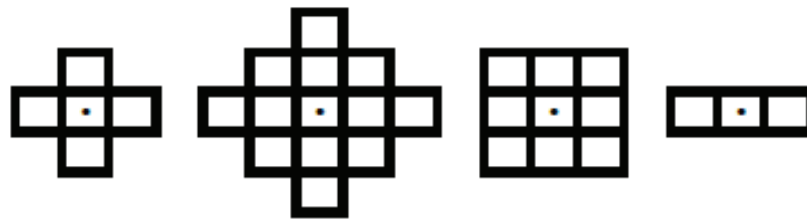


Figura 3: Exemplos de elementos estruturantes.

Fonte: Autoria própria

- O elemento estruturante pode ser utilizado para fazer operações sobre uma imagem binária. Considere por exemplo a operação no conjunto A , feita pelo elemento estruturante B (figuras 4a e 4b): Cria-se um novo conjunto C sobrepondo B em A da maneira que a origem de B passe por todos os pixels de A . Em cada ponto que a origem de B passa verifica-se se B está completamente contido em A . Se estiver marca-se esse ponto no conjunto C . A figura 4c mostra o resultado da operação, ou seja, o conjunto C .

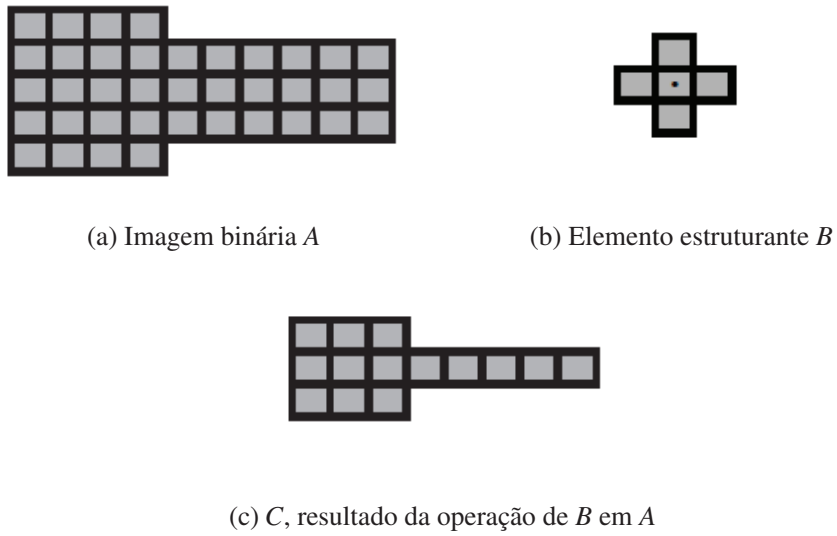


Figura 4: Imagem binária, elemento estruturante e operação de B em A
Fonte: Autoria própria

A seguir apresentam-se as operações de erosão e de dilatação, que são necessárias para as operações de abertura e fechamento que são apresentadas na sequência.

Erosão

Sendo A e B conjuntos no espaço Z^2 , a erosão do conjunto A pelo elemento estruturante B , denotada por $A \ominus B$, é definida na equação 4 (GONZALEZ; WOODS, 2006).

$$A \ominus B = \{z \mid (B)_z \subseteq A\} \quad (4)$$

Ou seja, a erosão de A por B é o conjunto de todos os pontos z tal que B , transladado por z , está completamente contido em A . Pode-se ver melhor esse processo na figura 5, onde o conjunto A é erodido pelo elemento estruturante B formando o conjunto $A \ominus B$.

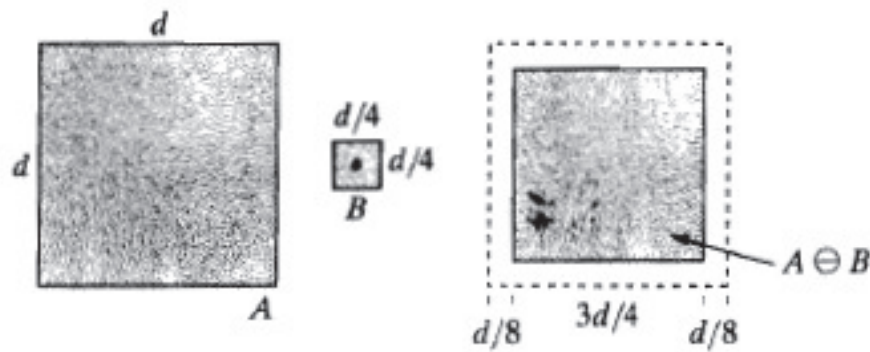


Figura 5: Exemplo de erosão do conjunto A por B.

Fonte: (GONZALEZ; WOODS, 2006)

Dilatação

Sendo A e B conjuntos no espaço Z^2 , a dilatação do conjunto A pelo elemento estruturante B , denotada por $A \oplus B$, é definida na equação 5 (GONZALEZ; WOODS, 2006).

$$A \oplus B = \{z \mid [(\hat{B})_z \cap A] \subseteq A\} \quad (5)$$

Onde \hat{B} é o conjunto B espelhado. Quando o elemento estruturante é simétrico então $\hat{B} = B$.

A equação 5 diz que a dilatação do conjunto A pelo elemento estruturante B é o conjunto de todos os pontos z tal que se \hat{B} e A se sobrepõem em pelo menos um ponto, então z faz parte de $A \oplus B$. Pode-se ver melhor esse processo na figura 6, onde o conjunto A é dilatado pelo elemento estruturante B formando o conjunto $A \oplus B$.

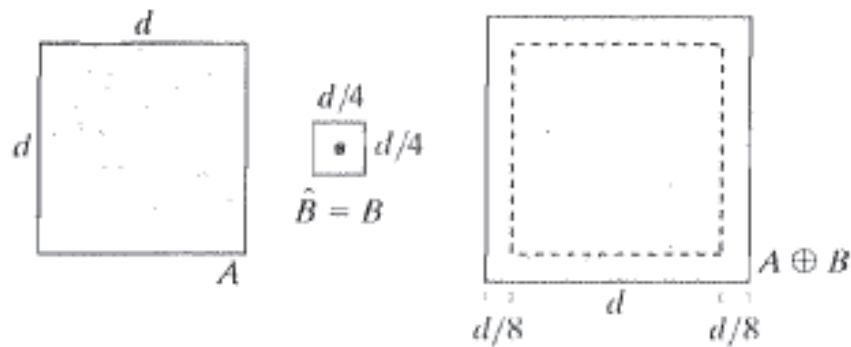


Figura 6: Exemplo de dilatação do conjunto A por B.

Fonte: (GONZALEZ; WOODS, 2006)

Abertura e Fechamento

A abertura do conjunto A pelo elemento estruturante B é definida pela equação 6 (GONZALEZ; WOODS, 2006).

$$A \circ B = (A \ominus B) \oplus B \quad (6)$$

O fechamento do conjunto A pelo elemento estruturante B é definido pela equação 7 (GONZALEZ; WOODS, 2006).

$$A \bullet B = (A \oplus B) \ominus B \quad (7)$$

Na figura 8 pode-se ver o resultado da execução da operação de abertura em cima do conjunto A da figura 7. Em seguida na figura 9 pode-se ver o resultado da execução da operação de fechamento em cima do mesmo conjunto da figura 7. Como pode-se ver, a operação de abertura "abre" a imagem em pontos estreitos, já a operação de fechamento "fecha" a imagem onde existem cavidades pequenas.

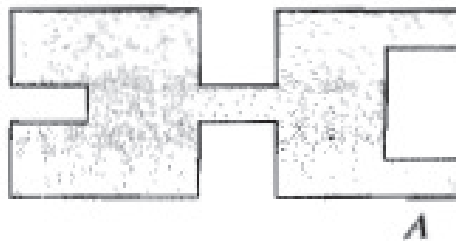


Figura 7: Conjunto A .

Fonte: (GONZALEZ; WOODS, 2006)

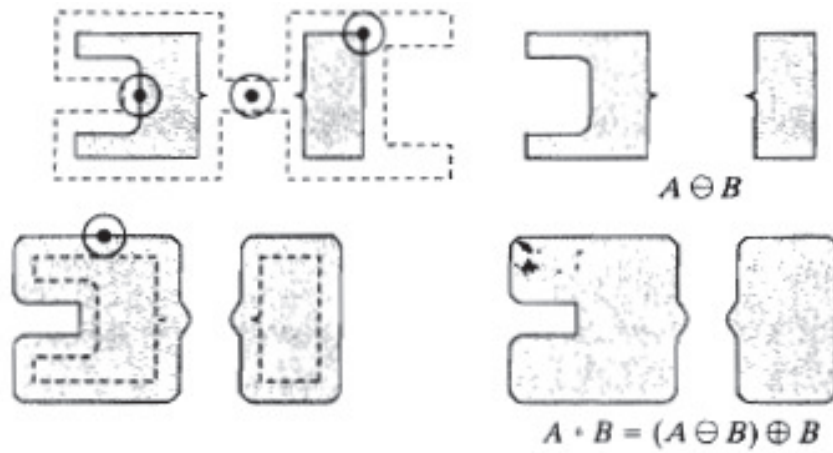


Figura 8: Conjunto $A \circ B$.

Fonte: (GONZALEZ; WOODS, 2006)

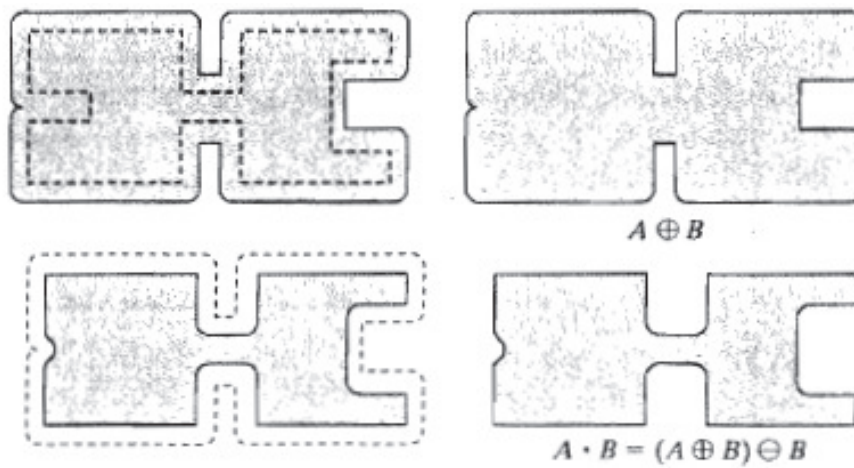


Figura 9: Conjunto $A \bullet B$.

Fonte: (GONZALEZ; WOODS, 2006)

2.1.4 Esqueletonização

Define-se esqueleto conforme segue (GONZALEZ; WOODS, 2006).

Se $S(A)$ é o esqueleto de A e $(D)_z$ o maior disco centrado em z e contido em A , pode se dizer que :

- Se z faz parte de $S(A)$ então não é possível encontrar um disco maior que $(D)_z$, não necessariamente centrado em z , que contenha $(D)_z$ e que esteja completamente incluso em A ;

- O disco $(D)_z$ toca a borda de A em pelo menos dois pontos distintos.

Isso pode ser visto melhor na figura 10.

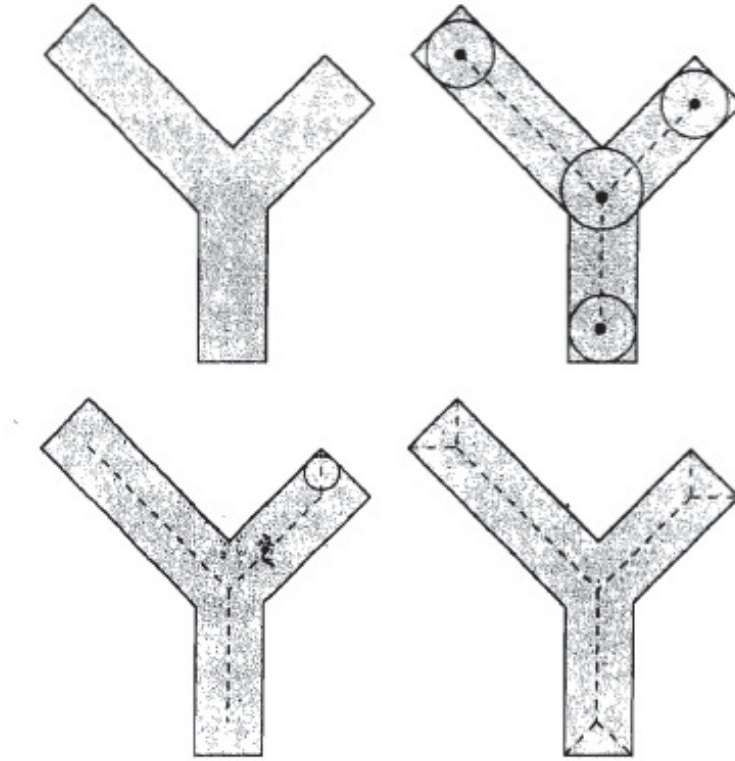


Figura 10: Conjunto A , disco máximo centrado em diversos pontos z e esqueleto completo.

Fonte: (GONZALEZ; WOODS, 2006)

Pode-se definir também o esqueleto $S(A)$ em função de erosões e aberturas sucessivas como mostrado nas equações 8, 9, 10 e 11 (GONZALEZ; WOODS, 2006).

$$S(A) = \bigcup_{k=0}^K S_k(A) \quad (8)$$

Com:

$$S_k(A) = (A \ominus kB) - (A \ominus kB) \circ B \quad (9)$$

Onde B é um elemento estruturante e $(A \ominus kB)$ indica k sucessivas erosões de A :

$$(A \ominus kB) = (((\dots((A \ominus B) \ominus B) \ominus \dots) \ominus B) \quad (10)$$

Sendo K o ultimo passo antes de A erodir para um conjunto vazio. Ou seja:

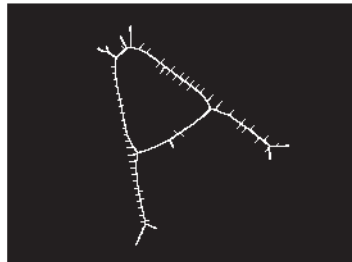
$$K = \max \{k \mid (A \ominus kB) \neq \emptyset\} \quad (11)$$

Poda (*Pruning*)

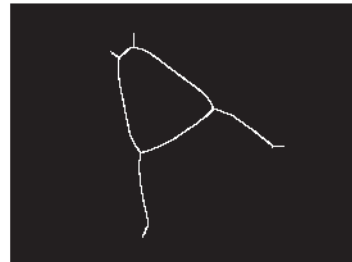
Aplicando-se o método de esqueletonização mencionado anteriormente em figuras reais, como a figura 11a, pode-se obter resultados como a figura 11b. Devido a esse tipo de resultados aplica-se após a esqueletonização um método de poda. A poda do esqueleto consiste em avaliar o tamanho dos ramos gerados, caso sejam menores que um valor limiar são removidos. A figura 11c mostra o método de poda aplicado com um limiar de 20 pixels.



(a) Imagem de entrada



(b) Imagem esqueletonizada



(c) Imagem esqueletonizada e podada

Figura 11: Exemplo de esqueletonização e poda.

Fonte: Autoria própria

2.1.5 Normalização

A análise de componentes principais pode ser empregada para normalizar imagens em função de rotação, translação e tamanho (GONZALEZ; WOODS, 2006). Explica-se a seguir alguns conceitos necessários para efetuar a normalização utilizando a análise de componentes principais e em seguida o processo de normalização utilizando componentes principais.

Matriz de covariâncias

Considere as imagens da figura 12. Os pontos da imagem, podem ser considerados como sendo vetores $x_k = (x_1, x_2)^T$, onde x_1 e x_2 são as coordenadas do ponto x_k , e T indica a matriz transposta. Os pontos x portanto formam uma população de vetores no espaço Z^2 . Sendo assim pode-se calcular a matriz de covariâncias C_x e o vetor da média m_x para a população de vetores.

Para K amostras de uma população aleatória pode-se aproximar a média m_x pela equação 12 (GONZALEZ; WOODS, 2006).

$$m_x = \frac{1}{K} \sum_{k=1}^K x_k \quad (12)$$

Agora utilizando a média m_x , pode-se aproximar a matriz de covariâncias C_x pela equação 13 (GONZALEZ; WOODS, 2006).

$$C_x = \frac{1}{K} \sum_{k=1}^K (x_k x_k^T) - m_x m_x^T \quad (13)$$

Na matriz de covariâncias os elementos c_{ii} de C_x representam a variância entre os valores na coordenada x_i . Os elementos c_{ij} representam a covariância entre os valores na coordenada x_i com os valores da coordenada x_j . Portanto, a matriz de covariância possui informações que dizem respeito à distribuição dos pixels da imagem.

Pode-se ver isso melhor nas matrizes de covariância apresentadas nas figuras 12d, 12e e 12f que foram calculadas para as imagens das figuras 12a, 12b e 12c respectivamente. Percebe-se que na figura 12d a maior variância ocorre entre os elementos da coordenada x_1 , visto que c_{11} possui o maior valor na matriz de covariâncias. Semelhantemente percebe-se que na figura 12e a maior variância ocorre entre os elementos da coordenada x_2 . A terceira imagem, figura 12f, mostra uma distribuição mais continua em x_1 e x_2 , mas valores maiores para as covariâncias entre os elementos das coordenadas x_1 e x_2 . A terceira imagem mostra portanto o relacionamento entre as duas coordenadas da imagem: quando x_1 aumenta, x_2 tende a aumentar também,

e vice-versa.

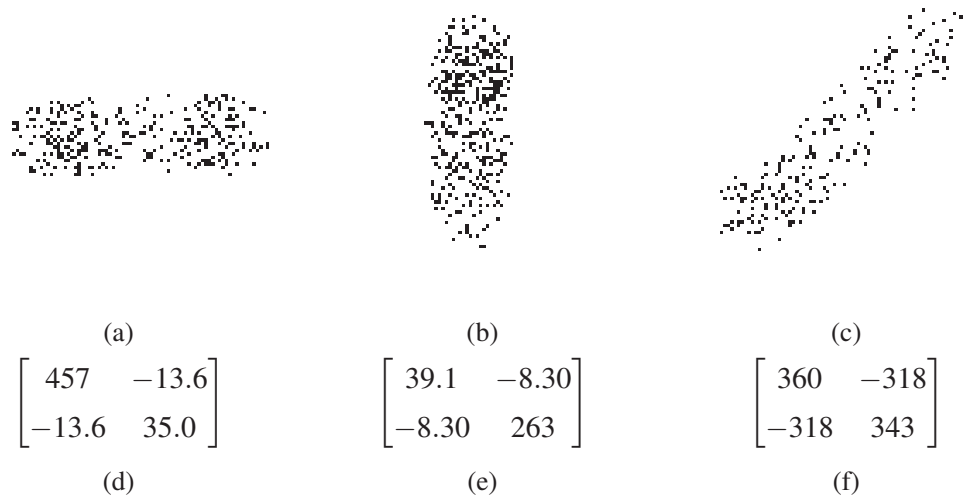


Figura 12: Imagens de exemplo e suas respectivas matrizes de covariância.

Fonte: Autoria própria

Autovetores e autovalores

Um autovetor de uma matriz A é um vetor não nulo v , que quando a matriz é multiplicada por v tem-se um vetor que é múltiplo de v por uma constante λ denominada autovalor. Ou seja, a equação 14 é verdadeira (LEON, 1998).

$$Av = \lambda v \quad (14)$$

Algumas propriedades de autovetores:

- Os autovetores podem ser calculados apenas para matrizes quadradas (LEON, 1998);
- Para uma matriz $n \times n$ existem n autovetores (LEON, 1998);
- Todos os autovetores são ortogonais entre si (GONZALEZ; WOODS, 2006).

Análise de componentes principais

Pode-se demonstrar que o autovetor associado ao maior autovalor de uma matriz de covariâncias aponta na direção de maior variação dos dados (GHAOUI, 2013). Pode-se demonstrar também que o segundo autovetor, associado ao segundo maior autovalor, aponta na segunda

direção de maior variação dos dados. A esses autovetores se dá o nome de componentes principais da imagem.

Considere por exemplo as imagens da figura 13. Calculamos os autovetores para cada uma das imagens tomando como base as matrizes de covariância apresentadas nas figuras 12d, 12e e 12f. Os autovetores podem ser vistos nas figuras 13d, 13e e 13f. O autovetor associado ao maior autovalor está na primeira linha de cada matriz e o segundo autovetor está na segunda linha. Os autovetores foram representados sobre as respectivas imagens nas figuras 14a, 14b e 14c.

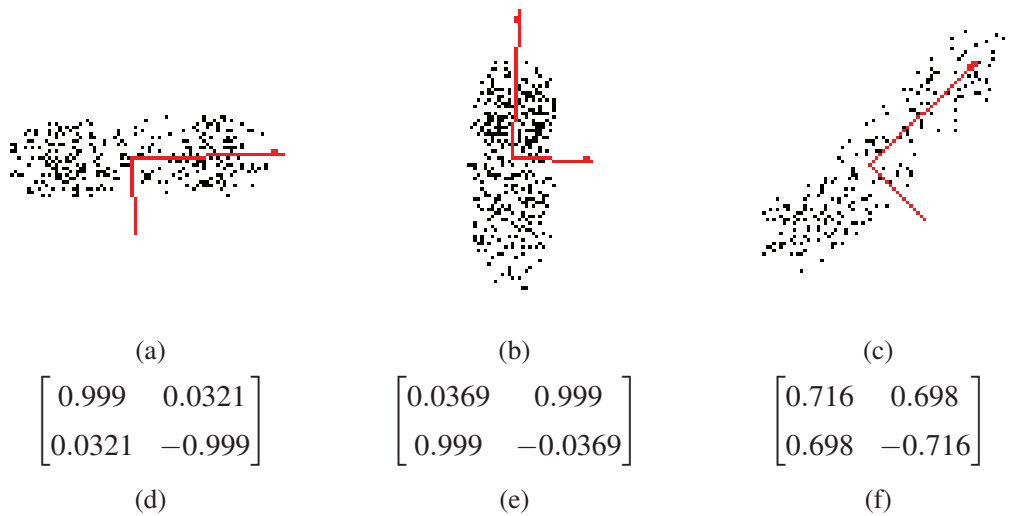


Figura 13: Imagens de exemplo e seus respectivos autovetores.

Fonte: Autoria própria

Vê-se portanto que é possível definir um eixo absoluto para cada imagem com base na distribuição dos pixels sobre a imagem.

Normalização

A normalização das imagens pode ser feita por meio de uma matriz de transformação que mapeia valores do sistema de coordenadas original da imagem para o sistema de coordenadas formado pelos autovetores da imagem (GONZALEZ; WOODS, 2006). A transformação em questão pode ser expressa pela equação 15.

$$y = A(x - m_x) \quad (15)$$

Onde:

- y é o valor no novo sistema de coordenadas formado pelos autovetores.
- x o valor da coordenada no sistema original.
- A a matriz de autovetores.
- m_x vetor da média.

As imagens da figura 12 podem ser vistas normalizadas na figura 14.

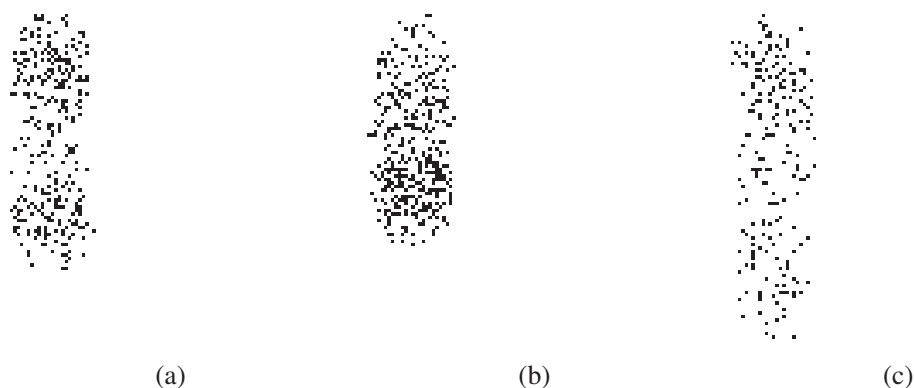


Figura 14: Imagens normalizadas.

Fonte: Autoria própria

Caso se deseje normalizar a imagem com relação ao tamanho pode-se dividir o valor das coordenadas y pelos autovalores correspondentes (GONZALEZ; WOODS, 2006).

2.2 DESCRIÇÃO

Esta seção apresenta técnicas e algoritmos para realizar a descrição de imagens, tais como extração de propriedades básicas de regiões e descritores de *fourier*.

2.2.1 Propriedades de regiões

Propriedades básicas de regiões, como área e excentricidade, podem ser usadas para descrever imagens. A seguir descreve-se algumas propriedades básicas de regiões, principalmente aquelas que possuem como saída um valor escalar, visto que são facilmente adicionados à um vetor descritor.

Área

A área de uma região é o número de pixels da região (MATHWORKS, 2013b).

Área convexa

A área convexa é dada pela área do menor polígono capaz de conter a região (MATHWORKS, 2013b).

Área Preenchida

Área preenchida corresponde a área da região cujas cavidades foram todas preenchidas (MATHWORKS, 2013b).

Solidez

A solidez de uma região é dada pela razão Área / Área convexa (MATHWORKS, 2013b).

Extensão

Semelhantemente à área convexa, a extensão é dada pela razão entre a área da região (definida anteriormente) e a área do menor retângulo capaz de conter a região (MATHWORKS, 2013b).

Excentricidade

Para a obtenção da excentricidade de uma região deve-se primeiramente obter o momento de inércia de área da região (MATHWORKS, 2013b). O momento de inércia de área para uma região qualquer no sistema de coordenadas polares é definido pela equação 16:

$$J_{BB} = \int_A \rho^2 dA \quad (16)$$

Onde A é a área da região sobre a qual se deseja calcular o momento de inércia de área e ρ é a distância do centro do sistema de coordenadas até dA . Uma representação da área pode ser vista na figura 15.

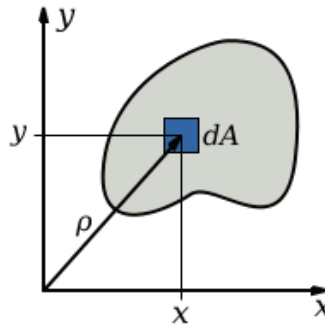


Figura 15: Representação da área para cálculo do momento de inércia.

Fonte: (WIKIPEDIA, 2013)

A excentricidade da região é então a excentricidade da elipse que possui o mesmo momento de inércia que a área da região. A excentricidade para uma elipse com semi-eixo maior a e semi-eixo menor b é dada pela equação 17:

$$e = \sqrt{1 - \frac{b^2}{a^2}} \quad (17)$$

Tamanho do eixo principal e eixo secundário

O tamanho do eixo principal de uma região diz respeito ao tamanho do maior eixo da elipse que possui o mesmo momento de inércia de área que a área da região, conforme visto anteriormente na seção Excentricidade (MATHWORKS, 2013b). Da mesma forma o tamanho do eixo secundário diz respeito ao tamanho do menor eixo da elipse que possui o mesmo momento de inércia de área que a área da região.

Número de Euler

O número de Euler é dado pelo total de objetos desconexos da região subtraindo o total de cavidades da região (MATHWORKS, 2013b).

2.2.2 Descritores de *fourier*

Considere a figura 16a. Os pixels com valor 1 da imagem binária podem ser representados como sendo pares de coordenadas $(x_0, y_0), (x_1, y_1), (x_2, y_2), \dots, (x_{K-1}, y_{K-1})$, onde K é o número de pixels na imagem. Além disso os pixels podem ser representados como uma sequência

$x(k) = x_k$ e $y(k) = y_k$ ou como $s(k) = [x(k), y(k)]$ para $k = 0, 1, 2, \dots, K-1$. Além disso pode-se tratar as coordenadas dos pixels como números complexos: $s(k) = x(k) + jy(k)$.

Tem-se que a transformada discreta de *fourier* de um sinal $s(k)$ é dado pela equação 18 (GONZALEZ; WOODS, 2006):

$$a(u) = \sum_{k=0}^{K-1} s(k) e^{-j2\pi uk/K}, u = 0, 1, 2, \dots, K-1 \quad (18)$$

Os coeficientes complexos $a(u)$ da transformada discreta de *fourier* são chamados de descritores de *fourier* (GONZALEZ; WOODS, 2006).

Da mesma forma que se aplicou a transformada discreta de *fourier* para obter os descritores de *fourier* pode-se aplicar a transformada discreta inversa de *fourier* nos coeficientes complexos para obter novamente os números complexos representando as coordenadas dos pixels (equação 19).

$$s(k) = \frac{1}{K} \sum_{u=0}^{K-1} a(u) e^{j2\pi uk/K}, k = 0, 1, 2, \dots, K-1 \quad (19)$$

Suponha agora que ao invés de utilizar todos os descritores de *fourier* utiliza-se apenas os P primeiros descritores para reconstituir $s(k)$. Tem-se portanto uma aproximação para $s(k)$ dada por $\hat{s}(k)$ na equação 20.

$$\hat{s}(k) = \frac{1}{P} \sum_{u=0}^{P-1} a(u) e^{j2\pi uk/P}, k = 0, 1, 2, \dots, K-1 \quad (20)$$

Quando os P coeficientes de menor frequência são escolhidos para reconstituir a imagem tem-se uma imagem aproximada da imagem original porém que mantém as principais características. Pode-se ver esse comportamento nas imagens da figura 16. A imagem da figura 16b foi formada utilizando todos os 19410 coeficientes da imagem original, a imagem 16c foi formada com 9705 coeficientes, e assim por diante, até apenas 2 coeficientes na figura 16o.

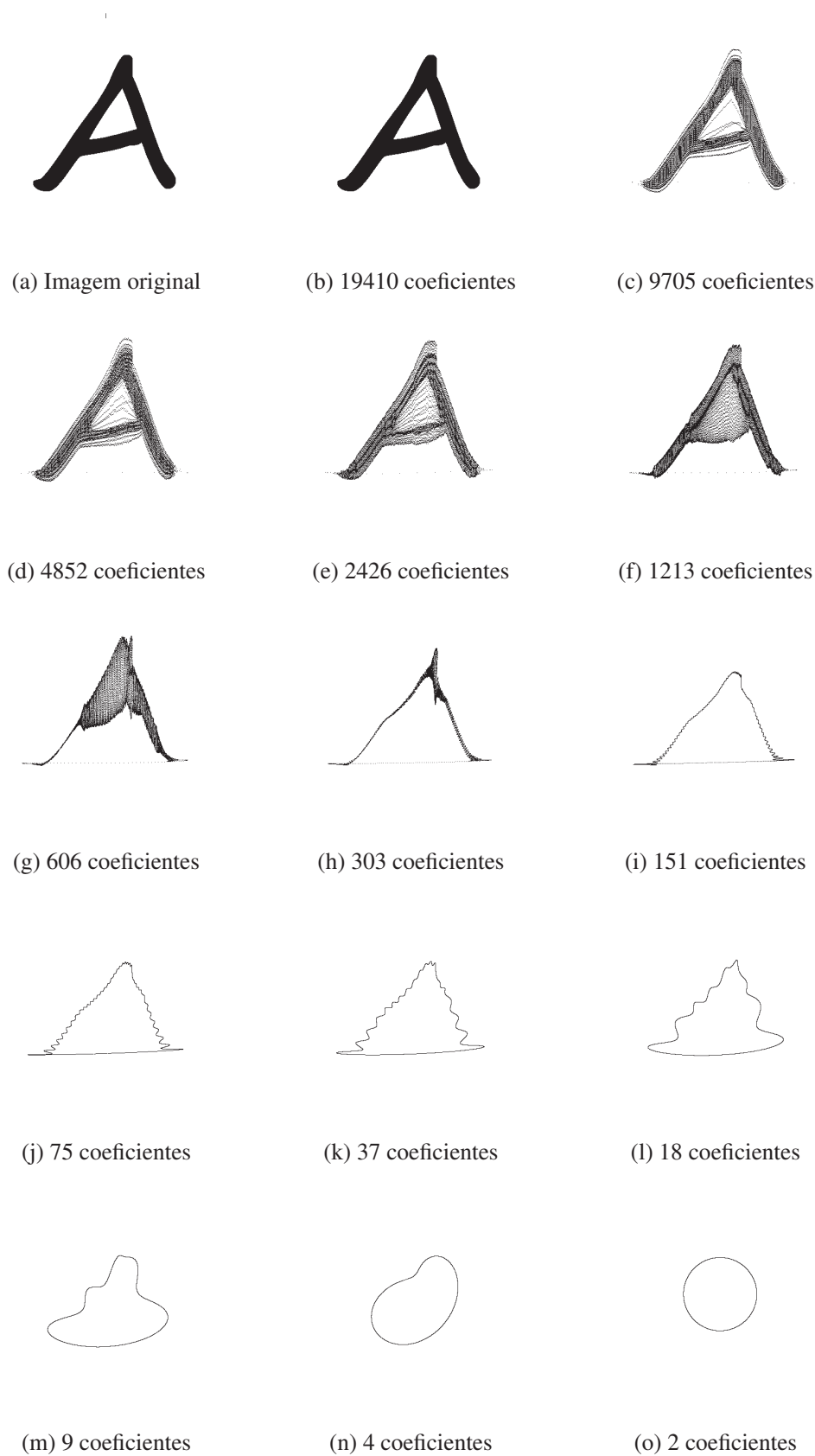


Figura 16: Imagem original e imagens reconstituídas utilizando n descritores de *fourier*.

Fonte: Autoria própria

2.3 CLASSIFICAÇÃO

Esta seção apresenta uma visão geral de reconhecimento de padrões utilizando redes neurais.

2.3.1 Redes neurais

Redes neurais são elementos não lineares (chamados neurônios) interconectados da mesma maneira que se acredita que os neurônios estão conectados no cérebro. Usam-se essas redes para tomadas de decisão após seus coeficientes terem sido ajustados por meio de apresentações sucessivas de conjuntos de dados de treinamento (GONZALEZ; WOODS, 2006).

Na forma mais simples no entanto a rede é um elemento linear e é composta por um único neurônio, também chamado *perceptron*. Esse neurônio então aprende uma função linear para separar dois conjuntos de dados de treinamento linearmente separáveis. A figura 17 apresenta o modelo básico para um *perceptron*. A resposta desse *perceptron* é baseada na soma ponderada das entradas, dada pela equação 21.

$$d(x) = \sum_{i=1}^n (\omega_i x_i) + \omega_{n+1} \quad (21)$$

Os coeficientes $w_i, i = 1, 2, \dots, n, n+1$, são chamados de pesos e modificam a soma antes que os valores de entrada passem pelo elemento de ativação, também chamado de função de ativação. Os pesos podem ser comparados às sinapses que ocorrem no cérebro humano. No caso particular da figura 17 a saída do *perceptron* será +1 caso a entrada pertença a classe ω_1 e -1 caso a entrada pertença a classe ω_2 .

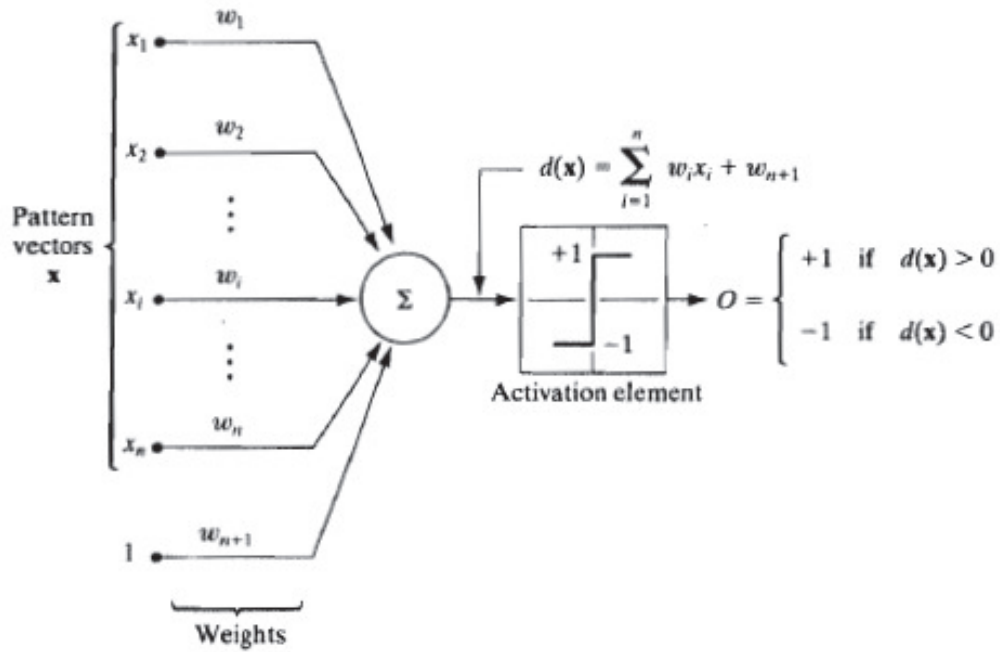


Figura 17: Modelo de *perceptron*.

Fonte: (GONZALEZ; WOODS, 2006)

Esse tipo de abordagem formada por um único *perceptron* não é capaz de separar mais do que duas classes e também não é capaz de separar classes que não são linearmente separáveis. Isso se dá devido ao fato de que a equação 21 quando expandida define um hiperplano em um espaço n -dimensional.

Utilizam-se portanto para problemas mais complexos redes de neurônios, também chamada de *multi-layer perceptron* ou *multi-layer feedforward network*. Cada neurônio é formado por uma estrutura do mesmo tipo da estrutura do *perceptron*, com a diferença de a função de ativação ser geralmente uma função sigmoide ao invés de um simples limiar. A função sigmoide pode ser vista na equação 22 e na figura 18. As saídas dos neurônios estão interconectadas com as entradas de outros neurônios. Essa estrutura pode ser vista na figura 19.

$$h_j(I_j) = \frac{1}{1 + e^{-(I_j + \theta_j)/\theta_0}} \quad (22)$$

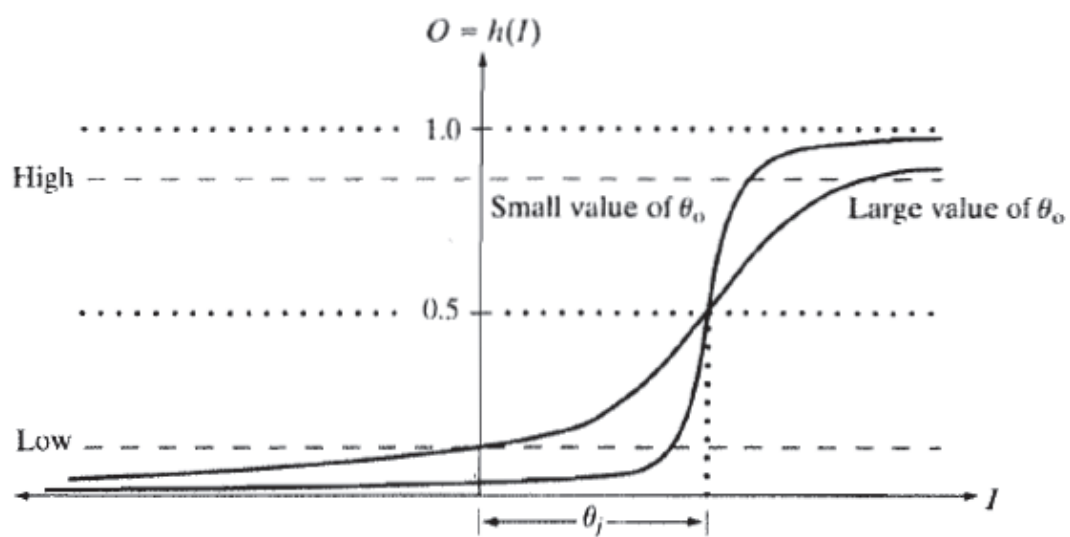


Figura 18: Gráfico da função de ativação sigmoide.

Fonte: (GONZALEZ; WOODS, 2006)

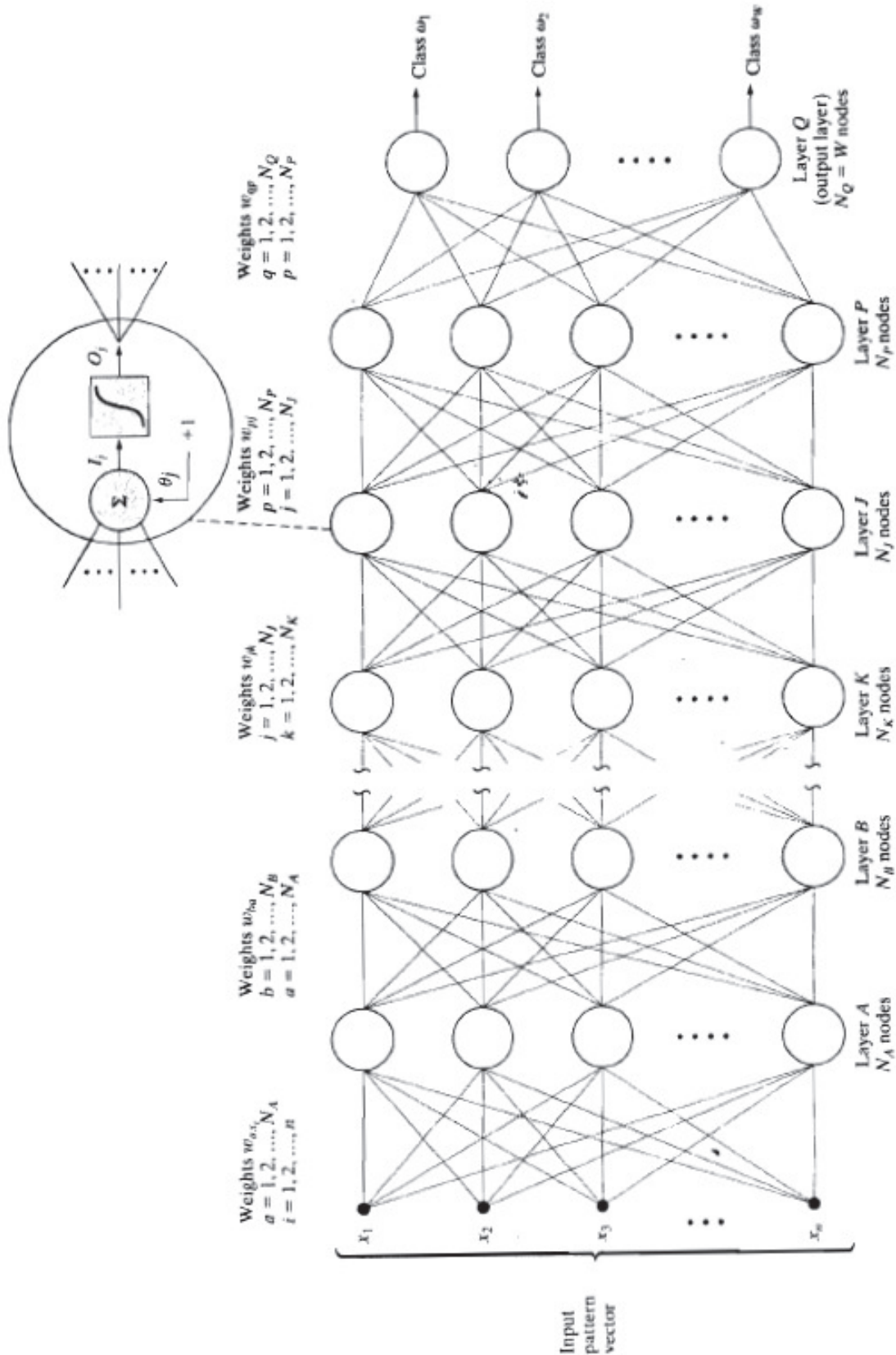


Figura 19: Modelo de *multi-layer feedforward network*.

Fonte: (GONZALEZ; WOODS, 2006)

Nesse tipo de rede a primeira camada, A , possui o mesmo número de neurônios que a dimensão do vetor descritor da entrada, $N_A = n$. Semelhantemente o número de neurônios da camada de saída, Q , é igual ao número de classes que a rede deve reconhecer, $N_Q = W$. A rede reconhece uma entrada como sendo parte de uma classe quando a saída da classe é um e as demais saídas são zero.

Pode-se demonstrar que redes desse tipo são capazes de separar regiões convexas no espaço n -dimensional de entradas utilizando duas camadas de neurônios. São também capazes de separar classes em regiões arbitrárias utilizando três camadas de neurônios (GONZALEZ; WOODS, 2006). A complexidade das regiões separadas por uma rede de três camadas se dá pelo número de neurônios em cada camada. Isso está resumido na figura 20.


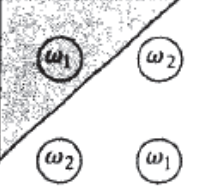
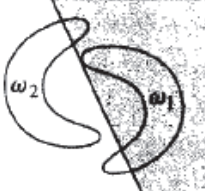
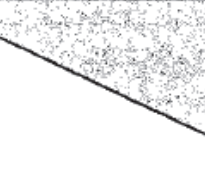
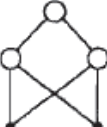
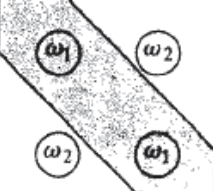
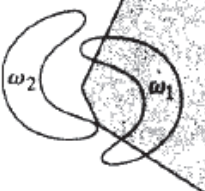





Network structure	Type of decision region	Solution to exclusive-OR problem	Classes with meshed regions	Most general decision surface shapes
Single layer 	Single hyperplane			
Two layers 	Open or closed convex regions			
Three layers 	Arbitrary (complexity limited by the number of nodes)			

Figura 20: Estruturas de redes neurais e sua respectiva capacidade de decisão.

Fonte: (GONZALEZ; WOODS, 2006)

Treinamento de *perceptrons*

O treinamento de *perceptrons* consiste em determinar os valores adequados de ω_i para que a soma ponderada das entradas forme a saída adequada para a função de ativação. Existem diversos métodos de treinamento para determinar os valores de ω_i para um único perceptron.

Apresenta-se a seguir o método denominado *delta rule* ou *least-mean-square*.

A principal característica do método *delta rule* é que ele minimiza o erro entre a saída desejada e a saída real do *perceptron* em cada passo de treinamento. Considere a função da equação 23.

$$J(w) = \frac{1}{2}(r - w^T y)^2 \quad (23)$$

$J(w)$ representa o erro quadrático para um dado conjunto de pesos w . O erro é calculado como sendo a diferença entre a saída desejada do *perceptron* r e a saída real dada pelo produto do vetor de pesos transposto w^T com o valor de entrada do conjunto de treinamento y .

O objetivo do treinamento pelo método *delta rule* é minimizar $J(w)$ ajustando os valores dos pesos w .

Se $w(k)$ representa o vetor de pesos no passo de treinamento k , então pode-se calcular $w(k+1)$ de forma a reduzir o valor de $J(w)$ utilizando a equação 24 (GONZALEZ; WOODS, 2006).

$$w(k+1) = w(k) - \alpha \left[\frac{\partial J(w)}{\partial w} \right]_{w=w(k)} \quad (24)$$

Onde $\alpha > 0$ é uma constante escolhida de acordo com o grau de correção que se deseja a cada passo. Da equação 23 tem-se que:

$$\frac{\partial J(w)}{\partial w} = -(r - w^T y)y \quad (25)$$

Substituindo a equação 25 na equação 24 resulta na equação 26 que pode ser utilizada para o cálculo dos coeficientes $w(k+1)$ dados os coeficientes $w(k)$, a saída desejada r e o valor de entrada do conjunto de treinamento y .

$$w(k+1) = w(k) + \alpha [r(k) - w^T(k)y(k)]y(k) \quad (26)$$

Pode-se demonstrar que utilizando esse método para calcular os coeficientes, tem-se a redução do erro do *perceptron* em um fator de $\alpha y(k)^T y(k)$ a cada ciclo de treinamento (GONZALEZ; WOODS, 2006).

Treinamento de *multi-layer perceptrons*

O treinamento de redes multicamadas pode ser efetuado de maneira semelhante ao treinamento de um único *perceptron*. Considere a equação 27. O erro quadrático E_Q obtido na camada de saída é dado pela soma dos erros quadráticos em todos os *perceptrons* da camada de saída (*perceptrons* de 1 a N_Q). O erro em cada *perceptron* é dado pela diferença entre a saída desejada r_q , no *perceptron* q , e a saída obtida O_q .

$$E_Q = \frac{1}{2} \sum_{q=1}^{N_Q} (r_q - O_q)^2 \quad (27)$$

Novamente deseja-se minimizar o erro E_Q em cada passo de treinamento. Minimizando a equação 27 chega-se a dois casos. O primeiro é quando o *perceptron* está na última camada da rede. Nesse casos sabe-se o valor esperando na saída do *perceptron*, que é dado por O_q . Por outro lado quando o *perceptron* está em uma camada intermediária é necessário que primeiro se defina o erro em função de parâmetros conhecidos na rede. Esse processo foi desenvolvido por (GONZALEZ; WOODS, 2006) e a seguir apresenta-se o resultado.

Para *perceptrons* na camada de saída Q os pesos $w(k+1)$ podem ser calculado pelas equações 28 e 29. K é a camada que precede a camada de saída Q .

$$w_q(k+1) = w_q(k) + \alpha \delta_q O_K \quad (28)$$

$$\delta_q = (r_q - O_q) h'_q(I_q) \quad (29)$$

Onde:

- $w_q(k+1)$ são os pesos atualizados do *perceptron* q ;
- $w_q(k)$ são os pesos antigos do *perceptron* q ;
- $\alpha > 0$ é uma constante escolhida de acordo com o grau de correção que se deseja a cada passo;
- O_K são as saídas dos *perceptrons* da camada K ;
- r_q é a saída esperada do *perceptron*, dada pelo conjunto de treinamento;
- O_q é a saída atual no *perceptron* q ;

- $h'_q(I_q)$ é a derivada da função de ativação do *perceptron* q para a entrada I_q .

Para *perceptrons* em uma camada intermediária J , onde a camada seguinte é a camada P e a camada antecedente é a camada K , os pesos $w(k+1)$ podem ser calculados pelas equações 30 e 31.

$$w_j(k+1) = w_j(k) + \alpha \delta_j O_K \quad (30)$$

$$\delta_j = h'_j(I_j) \sum_{p=1}^{N_P} \delta_p \omega_{jp} \quad (31)$$

Onde:

- $w_j(k+1)$ são os pesos atualizados do *perceptron* j ;
- $w_j(k)$ são os pesos antigos do *perceptron* j ;
- $\alpha > 0$ é uma constante escolhida de acordo com o grau de correção que se deseja a cada passo;
- O_K são as saídas dos *perceptrons* da camada K ;
- $h'_j(I_j)$ é a derivada da função de ativação para a entrada I_j ;
- p até N_P são todos os *perceptrons* da camada P ;
- δ_p é o valor de δ que havia sido calculado para o *perceptron* p da camada P quando os pesos da camada P foram atualizados.
- ω_{jp} é o valor do peso ω que havia sido calculado para o *perceptron* j em cada *perceptron* da camada P .

Caso a função de ativação escolhida seja a função sigmoide da equação 22, com $\theta_O = 1$, pode-se demonstrar que $h'_j(I_j)$ assume o valor da equação 32 e semelhantemente $h'_q(I_q)$ assume o valor da equação 33 (GONZALEZ; WOODS, 2006).

$$h'_j(I_j) = O_j(1 - O_j); \quad (32)$$

Onde:

- O_j é a saída atual no *perceptron* j ;

$$h'_q(I_q) = O_q(1 - O_q); \quad (33)$$

Onde:

- O_q é a saída atual no *perceptron* q ;

Como se pode ver é necessário iniciar o processo de atualização dos pesos da rede pela camada de saída. Na camada de saída da rede é o único lugar onde se sabe qual é a saída que cada *perceptron* deveria ter. Para as camadas antecedentes a atualização deve ser feita em função dessas saídas. Portanto cada camada depende de parâmetros da camada seguinte na rede. Esse processo de atualização dos pesos iniciando pela ultima camada e voltando até a primeira recebe o nome de *back propagation*. O método mencionado anteriormente recebe o nome de *delta rule*, no entanto existem diversos métodos diferentes mas que utilizam a mesma estratégia de propagação das correções na rede. Todos esses métodos são classificados como métodos de *back propagation*.