Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e Computação
Departamento de Comunicações

Extração de Características via Redes Neurais

Hugo Mauro Vasconcelos da Cunha Cavalcanti

Orientador: Prof. Dr. Lee Luan Ling

Banca Examinadora:

Prof. Dr. Lee Luan Ling
DECOM/FEEC/UNICAMP

Prof. Dr. Roseli Aparecida Francelin Romero
IMCM/USP

Prof. Dr. Márcio Andrade Neto
DCA/FEEC/UNICAMP

Dissertação de Mestrado apresentada à Faculdade de Engenharia Elétrica e de Computação (FEEC) da Universidade Estadual de Campinas (UNICAMP), como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica.

Dissertação de Mestrado
Campinas – SP – Brasil
fevereiro de 2000

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

C314e

Cavalcanti, Hugo Mauro Vasconcelos da Cunha
Extração de características via redes neurais / Hugo Mauro Vasconcelos da Cunha Cavalcanti.--Campinas, SP: [s.n.], 2000.

Orientador: Lee Luan Ling.
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Redes neurais (Computação). 2. Reconhecimento de padrões. I. Lee, Luan Ling. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

"Nenhum homem poderá revelar-vos nada senão o que já está meio adormecido na aurora de vosso entendimento. O mestre que caminha à sombra do templo, rodeado de discípulos, não dá de sua sabedoria, mas sim de sua fé e de sua ternura. Se ele for verdadeiramente sábio, não vos convidará a entrar na mansão de seu saber, mas antes vos conduzirá ao limiar de vossa própria mente. (...) Porque a visão de um homem não empresta suas asas a outro homem. E assim como cada um de vós se mantém só no conhecimento de Deus, assim cada um de vós deve ter sua própria compreensão de Deus e sua própria interpretação das coisas da terra".

Gibran Khalil

"A Deus pela inspiração,
à minha família pelo apoio,
aos amigos por acreditarem,
e ao meu amor pelo carinho e compreensão."

Agradecimentos

Valeu a pena chegar no final do nosso trabalho e ter o sentimento de dever cumprido. Mas, quem vê apenas o resultado final não sabe o quanto foi difícil chegar aqui. Desta forma, me sinto na obrigação de agradecer a algumas pessoas que muito me ajudaram neste trabalho.

Primeiro tenho que agradecer ao professor Lee, que mesmo nos piores momentos teve paciência e confiou no meu trabalho. Gostaria também de agradecer aos colegas de laboratório, que me acolheram muito bem e que me ajudaram na escolha de caminhos que hoje determinam a minha vida profissional.

Dentro dos colegas de laboratório, tenho um agradecimento especial a fazer ao "colega" Miguel que com o seu jeito de ser me ajudou a cumprir este tempo aqui em Campinas longe dos amigos e da família. Faço uma reverência especial a todos os amigos que longe ou perto sempre tiveram fé que eu conseguiria acabar este trabalho.

Como não poderia deixar de ser tenho que agradecer sobretudo aos meus pais, avós irmão e irmã que mesmo longe fisicamente sempre estiveram presentes nas minhas mais difíceis decisões.

E quero agradecer principalmente a minha Cecé, que mesmo a distância, me fez sentir a sua dedicação, carinho, companheirismo, paciência, amizade e amor dia após dia. Ela me faz ver a beleza da vida, e sonhar com um futuro muito feliz ao seu lado.

Finalmente, agradeço ao meu Deus, que me proporcionou o prazer de compartilhar da vida de tantas pessoas maravilhosas. E de ter tido pais, que sempre se preocuparam em me dar carinho, moral e educação.

Resumo

A implementação de um sistema de reconhecimento de padrões requer a solução de alguns problemas básicos: Aquisição de Dados, Extração de Características e Classificação dos padrões. Apesar de muitos trabalhos estarem sendo feitos na tentativa de resolver o problema de Reconhecimento de Padrões utilizando Redes Neurais, poucos são os trabalhos que abordam o Problema de Extração de Características.

Assim, nesta Tese propomos o Algoritmo de Extração de Características via Redes Neurais Lee/Cavalcanti. Este algoritmo encontra a quantidade mínima de características necessárias para resolver o problema de classificação de padrões utilizando uma rede neural do tipo Multilayer Parceptron (MLP). E baseia-se no fato de que todas as características informativas podem ser encontradas a partir da fronteira de decisão do problema.

Então, mostramos como construímos o algoritmo e apresentamos alguns experimentos que provam a eficiência do mesmo. Inicialmente, alguns experimentos foram feitos utilizando dados sintéticos, mostrando a relação entre a fronteira de decisão teórica e a fronteira de decisão prática encontrada a partir da rede treinada. Em seguida, implementamos uma rede neural para classificação de assinaturas estáticas. Neste experimento, utilizamos originalmente 32 características. E, em seguida, utilizando o algoritmo de extração de características Lee/Cavalcanti, conseguimos 98,84% de precisão de classificação, com apenas 16 características.

Desta forma, mostramos que o uso do algoritmo Lee/Cavalcanti pode encontrar a quantidade mínima de características de um problema de classificação de padrões. E, desta maneira, fazer com que a classificação do padrão seja realizada de forma mais rápida do que utilizando o conjunto original de amostras..

Abstract

The design and implementation of Pattern Recognition systems require the solution of the following problems: Data Acquisition, Feature Extraction and Pattern Classification. Although, much effort has been expended to solve a Pattern Recognition problem using the Neural Networks approach, not many works have being done to solve the Feature Extraction problem.

In this thesis, we propose the Lee/Cavalcanti Feature Extraction Algorithm via Neural Networks, which finds the minimum number of features necessary to solve the classification problem using Multilayer Perceptron (MLP) Neural Networks. This algorithm is based on informative features found from the Decision Boundary.

We present how the algorithm was built and some experiments to prove its efficiency. Some experiments using synthetic data are shown, indicating the relationship between the practical decision boundary, obtained from the trained neural network, and the theoretic one. Then, we design a neural classifier for a static signature recognition and we test it using 32 features. Finally, using only 16 features, we test the classifier obtaining a 98,84% accuracy in relation to the accuracy gained in the first test. The use of only 16 features was obtained using Lee/Cavalcanti Algorithm.

The use of the Lee/Cavalcanti Algorithm can reduce the number of features involved in a classification problem. Furthermore, it can make the system work faster with the same classification accuracy provided by the original set of features.

Índice Geral

Capítulo 1 - Objetivos

Capítulo 2 – Reconhecimento de Padrões e Características

Capítulo 3 – Teoria da Decisão de Bayes

Capítulo 4 – Reconhecimento de Padrões via Redes Neurais

Capítulo 5 – Extração de Características

Capítulo 6 – Multilayer Perceptron como uma aproximação da Função Discriminante de Bayes

Capítulo 7 – Algoritmo para Extração de Características via Redes Neurais

Capítulo 8 – Resultados Obtidos

Capítulo 9 – Conclusões Gerais

Capítulo 1

Objetivos

1.1 Introdução

A evolução da humanidade deve-se basicamente à capacidade do homem em discernir entre as várias opções que se lhe apresentam. Todos os cinco sentidos têm a sua importância e a cada um deles devemos algumas peculiaridades da vida moderna. Entretanto, o que realmente fez o homem diferente na sua evolução foi a percepção destes sentidos, ou seja, o fato de processar os sentidos e decidir o que representava cada um deles.

Evoluído, o homem desenvolveu o computador com o objetivo de o auxiliar em sua tarefas, o que este vem fazendo com bastante sucesso. Entretanto, fazer com que o computador aprenda a tomar decisões ainda é um desafio para muitos cientistas da nossa e das futuras gerações.

Neste contexto, surgiu Reconhecimento de Padrões, uma área de pesquisa voltada à inteligência artificial, com o objetivo básico de fazer com que o computador aprenda a tomar decisões e consequentemente auxiliar o homem também nesta tarefa.

Dentre as várias técnicas utilizadas em Reconhecimento de Padrões, poderíamos citar o Reconhecimento de Padrões Estatístico, Estrutural e Neural, sendo o Reconhecimento de Padrões Neural uma técnica relativamente nova, mas utilizada com muito sucesso nos últimos anos.

De um modo geral, poderíamos entender Reconhecimento de Padrões como sendo uma forma de fazermos algumas medições em um objeto qualquer com o objetivo de identificá-lo no meio de vários outros objetos.

As medidas ditas no parágrafo anterior são denominadas *características* e em poucas palavras estas características podem ser definidas como sendo medidas capazes de representar um objeto qualquer. E o ato de escolher e selecionar apenas as características que realmente incorporam uma discriminação entre os diversos objetos chama-se Extração de Características.

Apesar dos vários trabalhos realizados na área de redes neurais nos últimos anos, poucos são os trabalhos que enfocam Extração de Características. Assim, buscamos neste trabalho mostrar como podemos encontrar um conjunto ótimo de características e como fazê-lo utilizando uma rede neural feedforward como classificador.

1.2 Descrição dos Trabalhos

Em linhas gerais, o objetivo principal deste trabalho é o de propor um algoritmo que realiza Extração de Características utilizando para isto uma rede neural do tipo feedforward treinada pelo algoritmo backpropagation.

Assim, para expor este novo algoritmo dividimos o trabalho em 8 Capítulos como descritos sucintamente abaixo.

Inicialmente, no Capítulo 2, buscamos dar uma noção geral do que vem a ser Reconhecimento de Padrões, mostrando um pouco das técnicas utilizadas, das dificuldades e das possíveis soluções. Daremos uma definição formal a respeito de Características e como a partir dos objetos podemos encontrar características que representem o mesmo.

Em seguida, no Capítulo 3, mostraremos que num problema de classificação, a escolha ótima poderá ser encontrada sempre que utilizamos um classificador de Bayes. Também daremos uma exposição a respeito da função custo e das possíveis estratégias para procedermos com a classificação.

Então, no capítulo 4 mostraremos uma rede neural feedforward utilizando o algoritmo de backpropagation para treinamento e faremos uma abordagem sobre a utilização de redes neurais em Reconhecimento de Padrões. Estudaremos ainda como um processo de classificação via Redes Neurais deve ser realizado. Veremos então que uma rede do tipo feedforward particiona o espaço de entrada em várias células de classificação. Este resultado nos levará, nos capítulos posteriores, ao entendimento de que a fronteira de decisão é formada pelos hiperplanos gerados pelos neurônios da primeira camada escondida da rede neural feedforward..

No capítulo 5, mostraremos que todas as características necessárias a classificação de padrões podem ser encontradas a partir da fronteira de decisão. Isto justifica a importância da fronteira de decisão dentro de um problema de classificação de padrões.

No capítulo 6, mostraremos que uma rede neural do tipo Multilayer Perceptron (MLP) utilizando o algoritmo de backpropagation para treinamento tem a sua saída aproximada a função discriminante ótima de Bayes, segundo o erro quadrático médio.

Assim, no capítulo 7, proporemos um algoritmo que realiza a extração de características a partir de uma rede neural feedforward treinada utilizando o algoritmo de backpropagation.

Por fim, no capítulo 8, mostraremos algumas redes treinadas que mostram que uma rede MLP realmente aproxima a função discriminante ótima de Bayes, e, validaremos o nosso algoritmo de extração de características comprovando a sua eficiência. Citaremos ainda alguns trabalhos que podem ser feitos no futuro de forma a aprimorar algumas idéias e procedimentos.

Capítulo 2

Reconhecimento de Padrões e Características

2.1 Introdução

Neste Capítulo descreveremos inicialmente o que vem a ser Reconhecimento de Padrões e quais as possíveis técnicas utilizadas para realizá-lo.

Em seguida, definiremos características e mostraremos a sua importância dentro de um sistema de reconhecimento de padrões. Mostraremos então como podemos encontrá-las, e como devemos proceder a fim de que as mesmas sejam as mais significativas possível.

Por fim, apresentaremos um exemplo prático de Reconhecimento de Padrões, onde descreveremos um sistema de reconhecimento de assinaturas estáticas.

2.2 Reconhecimento de Padrões

Uma propriedade inerente ao ser humano é reconhecer padrões, e todo o seu conhecimento baseia-se na sua capacidade de reter, isolar, associar e comparar formas, sons e conceitos, identificando-os e os utilizando conforme as suas necessidades [2], [21] . Este processo é de tal forma complexo, que tem atraído cientistas na tentativa de exploração do seu mecanismo, e ao desenvolvimento de metodologias matemáticas como as redes neurais ou a inteligência artificial. De fato, a combinação da visão, do processo de memorização e

reconhecimento confere aos seres humanos habilidades que dificilmente serão ultrapassadas por outros seres.

A necessidade de comunicação entre homem e máquina através de linguagens naturais, e o interesse na idéia de projetar máquinas inteligentes que pudessem realizar certas tarefas com habilidades comparáveis às humanas geraram estudos que criaram e desenvolveram a chamada área de Reconhecimento de Padrões.

Assim, um sistema automático de Reconhecimento de Padrões tem por objetivo extrair informações dos objetos a serem reconhecidos, analisar as informações extraídas e agir de acordo com a análise das informações. Isto é, aceitar a entrada de uma pessoa, rejeite a assinatura de outra, encontre o alvo, etc...

2.3 Técnicas de Reconhecimento de Padrões

Reconhecimento de Padrões pode ser caracterizado como um processo de redução de informação, mapeamento de informação ou até mesmo de rotulação de informação. Assim, o nosso objetivo dentro de Reconhecimento de Padrões é obter informações de um objeto qualquer de forma que estas consigam representá-lo. Em seguida, as informações obtidas devem ser analisadas e o sistema de reconhecimento deve relacionar o objeto a um dos possíveis padrões do sistema.

De uma forma geral, a elaboração de um sistema de reconhecimento deverá seguir alguns passos descritos a seguir. Inicialmente, o projetista deve encontrar os padrões, ou seja, encontrar como as amostras se relacionam e se diferenciam uma das outras. Em seguida ele deve organizar as informações que podem ser extraídas dos padrões de acordo com o tipo de padrão a ser analisado. Ainda de acordo com os padrões, o projetista tem que escolher um método de classificação e treinar o classificador a partir de amostras conhecidas. Finalmente, o projetista deve associar cada um dos padrões do sistema a uma ação a ser tomada.

Existem duas abordagens básicas que podem ser tomadas no conjunto de dados onde se deseja fazer o reconhecimento; são elas Reconhecimento de Padrões Sintático e Reconhecimento de Padrões Numérico. Assim, descreveremos nas subseções que seguem estas duas abordagens.

2.3.1 Reconhecimento de Padrões Sintático

Uma das técnicas utilizadas em Reconhecimento de Padrões e explorada por Fu em [3] baseia-se na representação estrutural dos padrões. Esta representação deve ser feita utilizando uma gramática que descreve os padrões a partir de símbolos e as regras de como estes símbolos devem ser agrupados para formar cada padrão.

O mesmo Fu também em [3], afirma que não há soluções gerais e que a escolha da gramática é influenciada pela natureza dos dados disponíveis, da aplicação e da tecnologia. Assim, não existe uma gramática universal que sirva de base para todos os problemas de Reconhecimento de Padrões. Desta forma, o projetista do sistema deve criar uma gramática a cada problema de reconhecimento. A formação da gramática é por si só o aprendizado típico deste tipo de abordagem. E, a classificação dos padrões deve ser feita a partir da análise dos dados extraídos.

Como exemplo de reconhecimento sintático, podemos citar a árvore de decisão para reconhecimento de manuscritos, apresentada por Veloso em [23]. Nela, cada um dos caracteres é descrito segundo algumas características morfológicas, tais como: a quantidade de buracos que existem no caracter, a posição de cada um dos buracos, a quantidade de segmentos côncavos, o tipo de concavidade e a quantidade de interseções com um eixo determinado. Daí a árvore fará a análise destas características e fará o reconhecimento do caracter.

As técnicas de Reconhecimento Sintático podem ter um resultado bastante significativo, principalmente quando a gramática é bem elaborada. Entretanto, nem sempre é possível realizar a formulação gramatical do padrão, impossibilitando em alguns casos a utilização deste tipo de abordagem.

2.3.2 Reconhecimento de Padrões Numérico

Outra forma de se realizar reconhecimento de padrões é via o método de reconhecimento numérico de padrões que tem por objetivo classificar um padrão a partir de um vetor numérico, conhecido como vetor característico ou vetor do padrão. Este vetor deve então ser comparado a um vetor representativo de uma classe de padrões. O vetor de entrada será então classificado de acordo com a maior similaridade entre ele e o vetor representativo de uma das classes.

Dentro do Reconhecimento Numérico de Padrões, poderíamos incluir o Reconhecimento Estatístico e o Reconhecimento Neural.

2.3.2.1 Reconhecimento de Padrões Estatístico

O Reconhecimento de Padrão Estatístico se caracterizará pela geração de um modelo probabilístico para os padrões a serem reconhecidos [2], [21]. Assim, a partir dos dados que se dispõe para treinamento, o sistema deve gerar uma função densidade de probabilidade para as amostras.

A classificação das amostras será feita utilizando a função de densidade de probabilidade e aplicando a Teoria da Decisão Estatística. Vale salientar que o resultado deste tipo de sistema dependerá da quantidade de amostras disponíveis para o treinamento e do nível de generalização que o sistema pode oferecer. Entretanto, a falta de informações sobre a distribuição probabilística das amostras pode dificultar a utilização deste tipo de abordagem.

2.3.2.2 Reconhecimento de Padrões Neural

Como foi visto anteriormente, alguns padrões podem ser identificados como bem estruturados de forma a serem definidos por uma gramática, outros são de difícil modelagem ou difícil construção de uma gramática. Assim, não há soluções gerais e a escolha do Reconhecimento Estrutural de Padrões é influenciada pela natureza dos dados

disponíveis. O projetista deve realizar escolhas para a definição da gramática baseadas em suas experiências.

Esta natureza não estruturada do padrão a ser reconhecido torna o problema de reconhecimento difícil de ser tratado. Assim, utilizar Redes Neurais pode ser encarado como um método alternativo para resolução de problemas de reconhecimento, pois ao invés de criar procedimentos lógicos, a construção destas redes envolve o entendimento informal do comportamento desejado para atender ao problema [2], [21], [5].

Algumas das vantagens da utilização deste tipo de abordagem poderiam ser descritas como habilidade de se ajustar a novas informações, velocidade de classificação, capacidade de oferecer boas respostas mesmo com a falta de alguns dados ou com a presença de dados ruidosos.

A capacidade de generalização da rede neural também se apresenta como vantagem no problema de reconhecimento de padrões. Tomando como exemplo o reconhecimento de assinatura, vemos que a inclinação, tamanho, pressão da caneta sobre o papel e formato da assinatura são algumas das variáveis que podem afetar o reconhecimento da assinatura. Entretanto, uma rede neural após aprender a distinguir algumas assinaturas de tamanho ou formato diferentes, será capaz de distinguir assinaturas de diferentes tamanhos e formatos com sucesso. Desta forma, a capacidade de reconhecer padrões nunca antes vistos, porém semelhantes aos apresentados durante o treinamento, torna-se um diferencial perante muitas técnicas tradicionais, além de ajudar a superar ruídos indesejáveis nos dados de entrada do sistema de reconhecimento.

De uma forma geral, as redes neurais são altamente recomendáveis para se lidar com sistemas de reconhecimento pouco entendidos e que não podem ser adequadamente descritos por um conjunto de regras ou equações.

2.4 Características

Nas seções anteriores foi visto que para um sistema reconhecer padrões, este deve obter informações de um objeto antes de classificá-lo. Estas informações são chamadas de características, que pode ser uma informação concreta ou abstrata que representa um determinado padrão e o distingue de outros padrões [2], [23], [21].

A fim de que tenhamos uma melhor idéia do que vem a ser uma característica, consideremos o exemplo em que o cérebro humano é visto como um sistema de Reconhecimento de Padrões. A fim de que o cérebro reconheça um objeto, o cérebro verifica algumas informações tais como cor do objeto, pontos específicos, forma, contexto em que o objeto está, etc... Assim, estas informações que auxiliam o sistema (no caso o cérebro) constituem as características relevantes para o processo de reconhecimento.

Da mesma forma, um sistema automático de reconhecimento deve obter informações significativas do objeto de modo que o mesmo consiga distinguir dos outros objetos. Entretanto, sistemas automáticos utilizam informações as mais diversas tais como densidade de probabilidade, posicionamento de pixels, transformadas matemáticas como a de Fourier, dentre outras informações que podem diferenciar, em muitos casos, das características que um ser humano utilizaria para reconhecer o objeto.

Assim, não é incomum encontrarmos sistemas automáticos que possuam erros do tipo trocar um "S" por "M" ou por "E". Note que do ponto de vista do ser humano, um erro destes seria praticamente impossível, sendo muito mais fácil um ser humano trocar um "S" por um "5" por exemplo.

Assim, devido a possíveis erros de interpretação, sistemas automáticos muitas vezes exigem uma crítica sobre os resultados encontrados pelo classificador de padrões. Nesta crítica devemos analisar não só os objetos de reconhecimento, mas também o contexto em que foi encontrado cada objeto, e na medida do possível corrigi-los.

2.5 Sistema de Reconhecimento de Padrões

Genericamente, um sistema de Reconhecimento de Padrões desempenha a função conforme a descrição feita na figura 2.1, onde para cada entrada está relacionada uma única ação. Em termos de classificação de padrões, cada uma destas ações deve ser relacionada a uma classe, onde classes diferentes eqüivalem a ações diferentes [2], [21] e [5].

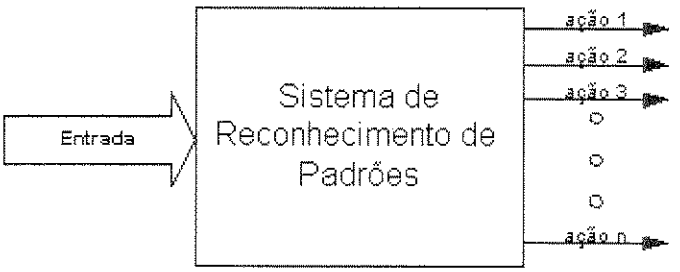

figura 2.1
Sistema de Reconhecimento de Padrões

De fato, um sistema de Reconhecimento de Padrões é composto de módulos distintos que desempenham individualmente funções diferentes, que incluem: adquirir os dados de entrada, processá-los, classificá-los e gerar uma ação de saída. Estes módulos são descritos nas subseções adiante. Na figura 2.2, mostramos um diagrama em blocos de um sistema de classificação de padrões. Note que ele é composto de um transdutor, representando o módulo de aquisição de dados, um extrator de características, representando o módulo de pré-processamento, e um estágio de saída, responsável por classificar as amostras de acordo com os dados extraídos no módulo de pré-processamento e agir de acordo com a classificação obtida [2].

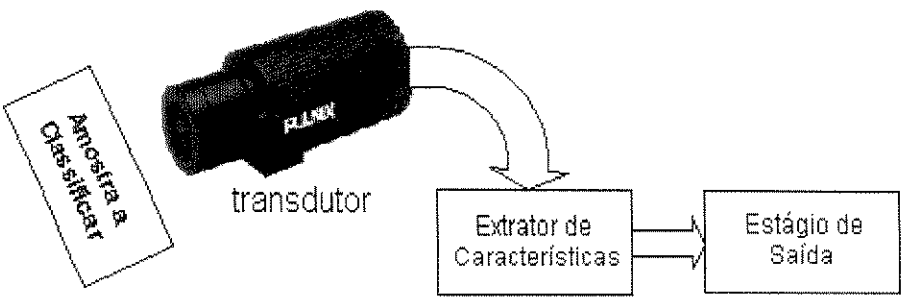

figura 2.2
Diagrama em blocos de um sistema de classificação de Padrões

2.5.1 Módulo de Aquisição de Dados

No módulo de aquisição de dados, temos como objetivo básico traduzir o objeto de reconhecimento em sinais possíveis de serem interpretados pelo módulo de pré-processamento. Este módulo será representado por um transdutor que dentre outras coisas pode ser uma câmera de vídeo, uma mesa digitalizadora, um microfone, um scanner, etc...

Suponha então um sistema de identificação de cheques bancários. Neste sistema teríamos como módulo de aquisição de dados, um Scanner com a saída digitalizada como entrada do módulo de pré-processamento [13]. Neste mesmo caso, também poderíamos imaginar um equipamento que pudesse extrair as informações bancárias pessoais impressas no cheque e de leitura magnética. Estes dados pessoais poderiam ser entregues ao módulo classificador para auxílio na identificação do cheque.

Os dados entregues ao módulo de pré-processamento devem conter todas as informações necessárias ao reconhecimento. Infelizmente, na maioria dos casos, é difícil avaliar se certos dados são relevantes ou redundantes ao sistema de reconhecimento. Daí, é importante utilizar um transdutor adequado a fim de que todas as informações relevantes possam estar contidas na saída do transdutor. Para exemplificar, na figura 2.3, mostramos a imagem digitalizada de um cheque e os dados complementares extraídos por uma máquina específica. Esta imagem e as informações complementares são enviadas ao módulo de pré-processamento para que este possa fornecer informações suficientes para o reconhecimento de algumas partes relevantes, tais como assinatura, valor numérico do cheque, data, etc...

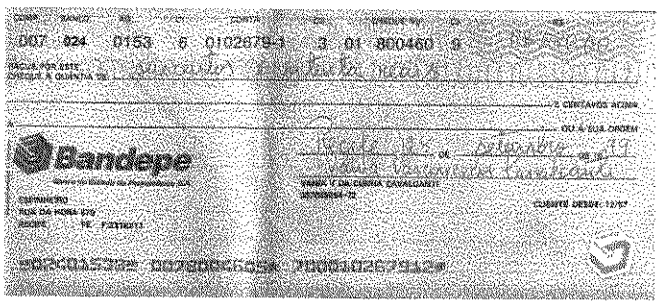

figura 2.3
Imagem digitalizada do cheque e informações extraídas

2.5.2 Módulo de Pré-Processamento

O módulo de pré-processamento é o responsável por extrair dos dados obtidos pelo módulo de aquisição de dados, as informações importantes ao reconhecimento, e se possível eliminar as informações desnecessárias e as redundâncias.

Por exemplo, suponhamos um sistema de identificação de cheques, podemos imaginar que o módulo de pré-processamento é dividido em blocos funcionais, responsáveis pela identificação, isolamento e classificação de cada parte de um cheque, incluindo a data, a assinatura, o valor numérico e o valor por extenso.

Uma maneira de fazer isto é subtraindo da imagem do cheque preenchido a imagem de um cheque sem preenchimento, onde reteríamos apenas as partes introduzidas pelo cliente, conforme ilustrado na figura 2.4.

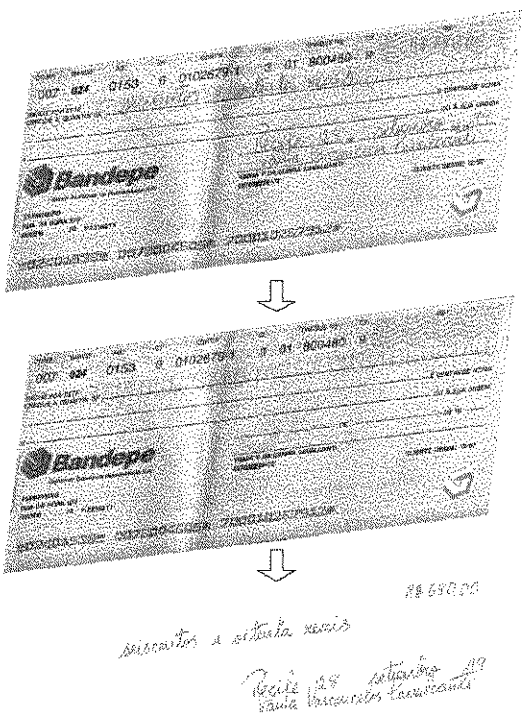

figura 2.4
Processamento da Imagem bancária

2.5.3 Módulo Classificador

As características extraídas no módulo de Pré-Processamento são então apresentadas ao classificador que deve definir a classe a que pertence a amostra e agir de acordo com esta classe. No caso de identificação pessoal, como no exemplo da assinatura, a ação poderia ser simplesmente aceitar ou rejeitar os dados relativos àquela pessoa. Entretanto, em outros sistemas de reconhecimento, como por exemplo em aplicações médicas, a ação poderia ser simplesmente identificar uma amostra a fim de contá-la ou medir a sua intensidade.

Ainda poderíamos incorporar, a este módulo, um bloco de extração de características, que teria como objetivo básico encontrar dentro das informações fornecidas pelo módulo de pré-processamento aquelas características que discriminem o objeto de classificação. Este bloco teria como finalidade reduzir a dimensão dos dados de entrada, fazendo com que o reconhecimento possa ser feito mais rapidamente.

Vale salientar que este bloco deve manter a precisão de classificação encontrada nos dados originais e é objetivo dos próximos capítulo mostrar como poderemos, via Redes Neurais, encontrar a quantidade mínima de características capaz de manter a mesma precisão de classificação.

Capítulo 3

Teoria da Decisão de Bayes

3.1 Introdução

A Teoria da Decisão de Bayes é uma abordagem estatística fundamental no que se refere ao problema de classificação de padrões. Esta abordagem baseia-se no fato de que um problema de decisão deve ser resolvido em termos de probabilidade, e que todos os valores probabilísticos podem ser computados. A importância da Teoria da Decisão de Bayes provém do fato de que ela garante, estatisticamente, uma taxa de erro de classificação mínima [2], [18], [5].

Desta forma, este capítulo tem por objetivo mostrar os fundamentos da Teoria da Decisão de Bayes e como, a partir dela, podemos encontrar um classificador ótimo. Assim, inicialmente definiremos custo e como este pode ajudar na classificação. Mostraremos também o problema da decisão no caso de apenas duas classes. Na seção subsequente, veremos que utilizando a Teoria da Decisão de Bayes teremos uma taxa de erro mínima na classificação, ou seja, encontraremos um classificador ótimo. Por fim, estenderemos nossa discussão sobre classificação envolvendo funções discriminantes e fronteiras de decisão.

3.2 Teoria da Decisão de Bayes

Suponha que para se fazer uma determinada classificação não tenhamos nenhuma informação referente às amostras. Então, deveremos classificá-la apenas de acordo com a sua natureza, ou seja, com as informações que podemos encontrar sem antes fazer qualquer análise. Por exemplo, suponha mais uma vez o sistema de reconhecimento de assinatura, descrito no capítulo anterior. Assumindo que não tenhamos qualquer informação a respeito das assinaturas, e que experiências anteriores mostram que uma assinatura qualquer tem uma probabilidade hipotética de 80% de ser verdadeira. Então, uma classificação destas assinaturas sem qualquer análise das mesmas nos levariam a classificá-las sempre como verdadeiras, visto que erraríamos em apenas 20% das oportunidades. A probabilidade que caracteriza a natureza ou deriva das experiências passadas é chamada de probabilidade *a priori* e reflete o conhecimento anterior que temos antes de qualquer análise das assinaturas, ou no contexto geral, antes da realização de qualquer experimento para obter informações sobre objetos [2], [21] e [18].

Felizmente, na maioria das vezes, um sistema de classificação tem a possibilidade de efetuar experimentos e conseguir amostras para análises antes de classificar um objeto. Assim, para ilustrar esta concepção, suponha mais uma vez como exemplo o sistema de reconhecimento de assinatura. Analisando assinaturas verdadeiras, adquiridas previamente, de uma determinada pessoa, podemos encontrar informações que nos levem a concluir que as mesmas possuem certas características comuns, que devem ser encontradas numa amostra de assinatura genuína, em teste, com um alto grau de probabilidade.

Denotamos x como sendo esta característica, devemos então encontrar a freqüência com que a mesma se apresenta nas assinaturas verdadeiras. Procedendo desta forma, e supondo que a mesma possua uma distribuição contínua, podemos encontrar uma função de densidade de probabilidade condicionada ao estado. Isto é, uma função que exprime a probabilidade desta característica aparecer numa assinatura verdadeira, e

que representamos por $p(x|\omega_1)$, onde ω_1 representa o estado de assinatura verdadeira, eω_2 representa o estado de assinatura falsa.

Na figura 3.1, mostramos duas funções de densidade de probabilidade amostral condicional $p(x/\omega_1)$ e $p(x/\omega_2)$, onde podemos defini-las como sendo a densidade de probabilidade de uma característica x, representar uma assinatura verdadeira (classe ω_1), ou pertencer a uma assinatura falsa (classe ω_2).

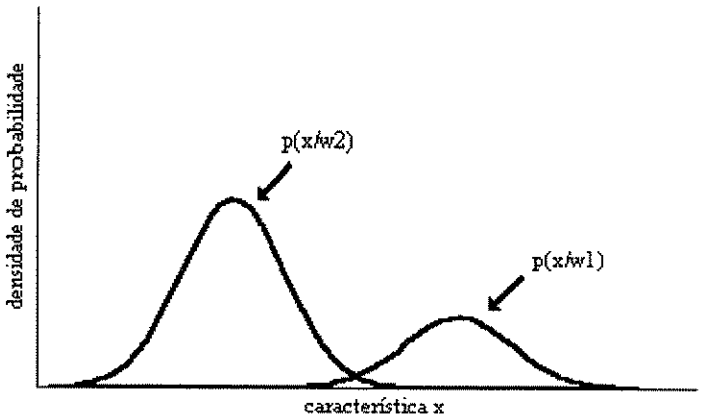

figura 3.1
Densidade de Probabilidade relativa à característica x quando pertencente a classe ω_1 ou ω_2

Da mesma forma, poderemos estimar, a partir das amostras de assinaturas falsas, a função densidade de probabilidade amostral $p(x/\omega_2)$.

Uma vez encontrado as funções de densidade de probabilidade amostral e a probabilidade *a priori* de cada classe, poderemos utilizar estas medidas para calcular a probabilidade a posteriori $P(\omega_i|x)$, a partir da *Regra de Bayes* descrita nas equações (3.1) e (3.2). Probabilidade *a posteriori* indica a probabilidade da amostra x pertencer a classe ω_i, uma vez que x seja observada [2], [21] e [18].

$$P(\omega_j \mid x) = \frac{p(x \mid \omega_j)P(\omega_j)}{p(x)}, \qquad (3.1)$$

a densidade amostral incondicional é dada por *p(x)*, conforme (3.2)

$$p(x) = \sum_{j=1}^{N} p(x \mid \omega_j)P(\omega_j), \qquad (3.2)$$

Uma vez encontradas as probabilidades *a posteriori*, a Regra de decisão de Bayes se torna relativamente simples. Devemos escolher a classe a qual a probabilidade *a posteriori* seja a máxima possível. Intuitivamente, vemos que esta é uma escolha razoável, na medida em que escolhemos a classe cuja probabilidade relacionada a amostra é a maior , mas vamos apresentar a prova a qual pode ser encontrada em [2].

Seja a probabilidade de erro na classificação de uma determinada amostra dada por:

$$P(Erro \mid x) = \begin{cases} P(\omega_1 \mid x), & \text{se decidirmos por } \omega_2 \\ P(\omega_2 \mid x), & \text{se decidirmos por } \omega_1 \end{cases}$$

então o erro total médio do sistema pode ser encontrado pela equação (3.3)

$$P(Erro) = \int_{-\infty}^{+\infty} P(Erro, x)dx$$

mas $P(Erro,x) = P(Erro|x)p(x)$

logo $P(Erro) = \int_{-\infty}^{+\infty} P(Erro \mid x)p(x)dx$

$$P(Erro) = \int_{P(\omega_2|x)>P(\omega_1|x)} P(\omega_1|x)p(x)dx + \int_{P(\omega_1|x)>P(\omega_2|x)} P(\omega_2|x)p(x)dx \qquad (3.3)$$

onde para qualquer valor de ***x*** o valor de *P(Erro|x)* calculado é o mínimo possível dado que a escolha foi feita com $P(\omega_i|x)$ máximo. Daí, minimizamos o valor de *P(Erro)*.

3.3 Função Custo

Mostraremos nesta seção a Função Custo. Para isto o nosso sistema de Reconhecimento de Padrões deve permitir as quatro situações descritas abaixo:

a. Usar mais de uma característica;

b. Existência de mais de dois estados (classes);

c. Permitir outras decisões que não somente o estado;

d. Introduzir uma função Perda, mais genérica do que a própria probabilidade de erro.

O uso de mais de uma característica é essencial no que se refere a confiabilidade e precisão da classificação, pois introduzimos mais informações úteis à classificação. Como conseqüência, é possível aumentar o grau de sensibilidade das classes, melhorando o desempenho do sistema em termos de probabilidade de classificação.

Devemos permitir mais do que dois estados, a fim de que tenhamos uma maior quantidade de ações possíveis no sistema.

A possibilidade de decidir entre outras decisões que não somente um dos estados, permite a rejeição de uma determinada amostra, ou seja, de não classificá-la quando não conseguirmos discernir com precisão o suficiente a qual classe pertence a amostra.

A função custo nos ajudará a generalizar a regra de Decisão, na medida em que poderemos determinar quanto custaria escolher uma ou outra ação. Por exemplo, o custo de aceitar uma amostra de uma assinatura em um cheque, pode ser muito maior para aceitação do que para rejeição.

Daremos então um tratamento formal.

Seja $\Omega=\{\omega_1, \omega_2, \ldots, \omega_N\}$ um espaço de N estados ou classes de um problema de Reconhecimento de Padrões. Seja $A=\{\alpha_1, \alpha_2, \ldots, \alpha_Q\}$ um espaço de ações que consiste de Q possíveis ações.

Então definiremos $\lambda(\alpha_i|\omega_j)$ como sendo o custo de realizar a ação α_i quando a amostra pertence ao estado ω_j.

Suponha então que tenhamos uma amostra particular $\boldsymbol{x}$, classificada por nós como pertencente à classe ω_i. A partir dela tomamos a ação α_i. Se a classe a que pertence $\boldsymbol{x}$ é, na verdade, ω_j, então teremos um erro e a ele estará relacionado a um custo designado por $\lambda(\alpha_i|\omega_j)$. Como a probabilidade de uma amostra qualquer pertencer à classe ω_j é dada por $P(\omega_j|\boldsymbol{x})$ o custo total associado a tomarmos a ação α_j é dado por (3.4):

$$R(\alpha_i \mid x) = \sum_{j=1}^{N} \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid x), \tag{3.4}$$

$R(\alpha_i|x)$ é chamado de Risco condicional e ele permite escolher, a partir de uma amostra qualquer, a ação que gera a menor perda. Isto eqüivale ao procedimento ótimo de Bayes, que iremos provar adiante.

Formalmente, o nosso problema é o de encontrar uma regra de decisão que nos permita minimizar o risco total. A Regra de Decisão é uma função $\alpha(\boldsymbol{x})$ que nos diz qual ação deve ser tomada para cada amostra $\boldsymbol{x}$.

O Risco total do sistema pode ser calculado então a partir da função (3.5)

$$R = \int R(\alpha(x) \mid x) p(x) dx, \tag{3.5}$$

Como $p(\boldsymbol{x})$ é sempre positivo R será minimizado se um $R(\alpha(\boldsymbol{x})|\boldsymbol{x})$ mínimo for escolhido para cada valor de $\boldsymbol{x}$. Assim, minimizar R eqüivale a minimizar a equação (3.4) para cada uma das amostras. Este valor mínimo de R é chamado de Risco de Bayes e é um resultado ótimo.

3.4 Classificação num caso de duas classes

Nesta seção veremos como podemos proceder para um sistema de classificação em duas classes. Simplificamos então as notações e definimos α_1 como sendo a ação correspondente à amostra pertencer a classe ω_1 e similarmente α_2 a ação correspondente à classe ω_2. Seja também $\lambda_{ij} = \lambda(\alpha_i|\omega_j)$ a perda relativa no caso de tomarmos a ação α_i quando a ação correta deveria ter sido α_j.

A partir da equação (3.4) poderíamos encontrar os riscos condicionais de cada uma das ações como segue.

$$R(\alpha_1 \mid x) = \lambda_{11}P(\omega_1 \mid x) + \lambda_{12}P(\omega_2 \mid x)$$
$$R(\alpha_2 \mid x) = \lambda_{21}P(\omega_1 \mid x) + \lambda_{22}P(\omega_2 \mid x)$$

Entretanto, sempre devemos tomar a decisão que nos gera a menor perda e consequentemente, devemos escolher de acordo com o seguinte critério:

$$\text{Decidir por} \begin{cases} \omega_1, & \text{se } R(\alpha_1 \mid x) < R(\alpha_2 \mid x) \\ \omega_2, & \text{se } R(\alpha_1 \mid x) > R(\alpha_2 \mid x) \\ \omega_1 \text{ ou } \omega_2, & \text{se } R(\alpha_1 \mid x) = R(\alpha_2 \mid x) \end{cases} \tag{3.6}$$

Em termos de probabilidade a posteriori, (3.6) pode ser descrita como:

$$\text{Decidir por} \begin{cases} \omega_1, & \text{se } (\lambda_{21} - \lambda_{11})P(\omega_1 \mid x) > (\lambda_{12} - \lambda_{22})P(\omega_2 \mid x) \\ \omega_2, & \text{se } (\lambda_{21} - \lambda_{11})P(\omega_1 \mid x) < (\lambda_{12} - \lambda_{22})P(\omega_2 \mid x) \\ \omega_1 \text{ ou } \omega_2, & \text{se } (\lambda_{21} - \lambda_{11})P(\omega_1 \mid x) = (\lambda_{12} - \lambda_{22})P(\omega_2 \mid x) \end{cases}$$

3.5 Taxa Mínima de Erro de Classificação

Num problema de classificação as ações α_i geralmente são interpretadas como a decisão correta se a amostra a ser classificada pertence a classe ω_i. Assim, se a ação α_i é

tomada e a amostra pertence a classe ω_j, então a decisão é correta se $i=j$ e é um erro se $i \neq j$.

Como os erros devem ser evitados, é natural que tenhamos uma regra que minimize a Probabilidade de erro, ou seja, da taxa de erro. Uma função erro particular que pode ser utilizada é a chamada função binária e que não pune o acerto e pune com um valor positivo fixo qualquer erro, ou seja:

$$\lambda(\alpha_i \mid \omega_j) = \begin{cases} 0, & \text{se } i = j \\ 1, & \text{se } i \neq j \end{cases}$$

Utilizando a equação (3.4), encontraremos o seguinte Risco condicional:

$$\begin{aligned} R(\alpha_i \mid x) &= \sum_{j=1}^{N} \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid x) \\ &= \sum_{j=1, j \neq i}^{N} P(\omega_j \mid x) \\ &= 1 - P(\omega_i \mid x) \end{aligned}$$

Assim, para minimizar a probabilidade de erro, devemos selecionar a classe que maximize a probabilidade a posteriori $P(\omega_i|\boldsymbol{x})$. Em outras palavras, para uma taxa de erro mínima, deveremos seguir a seguinte regra de Decisão:

Decidir por ω_i se $P(\omega_i \mid x) > P(\omega_j \mid x)$

Esta regra de decisão é chamada de Regra de Decisão de Bayes e basicamente nos diz que a classe a ser escolhida deve ser aquela tal que oferece a maior probabilidade *a posteriori*.

3.6 Classificadores e Função Discriminante

Existem muitas formas de representar uma Regra de Classificação. De modo genérico, num problema de N classes, poderemos definir um conjunto de funções discriminantes $g_i(\boldsymbol{x})$, $i=1,2,...,N$. E a regra de classificação consiste basicamente em

comparar estas funções para um dado vetor de características **x**. Por exemplo, seja a forma mais simples de comparação dada por:

$$\text{Decidir por } \boldsymbol{x} \in \omega_i \text{ se } g_i(\boldsymbol{x}) > g_j(\boldsymbol{x}) \text{ para todo } i \neq j$$

Neste caso, um classificador pode ser visto como uma máquina que calcula N funções discriminantes para a amostra apresentada e seleciona aquela classe cuja função discriminante obteve o maior resultado numérico. A figura 3.2 mostra conceitualmente como implementar este classificador a partir das funções discriminantes de cada uma das classes.

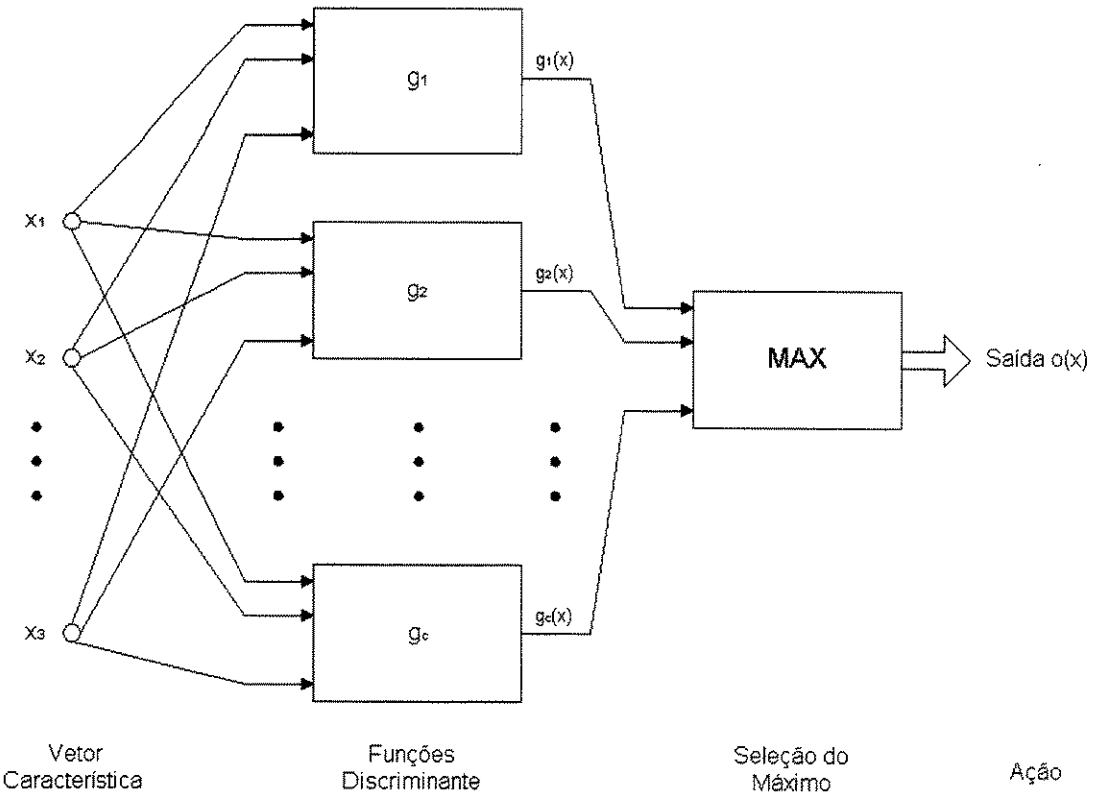

figura 3.2
Função discriminante e classificador

A escolha de funções discriminantes pode não ser uma tarefa trivial, uma vez que esta depende da aplicação. Entretanto, de uma maneira geral, poderíamos utilizar como função discriminante o inverso aditivo da função Risco, como mostra a equação (3.7). A

qual poderemos simplificar para a equação (3.8) no caso de um sistema de classificação que usa a taxa de erro mínimo como medida de desempenho.

$$g_i(x) = -R(\alpha_i \mid x), \tag{3.7}$$

$$g_i(x) = P(\omega_i \mid x), \tag{3.8}$$

É fácil verificar que, em geral, esta função discriminante não é única e que ao adicionarmos uma constante ou até mesmo multiplicarmos a função por uma constante positiva não alteraríamos o resultado das funções.

Vale salientar que podemos aplicar uma função monotonicamente crescente à função discriminante e obter uma outra função discriminante. Ou seja, $f(g(\mathbf{x}))$ também é uma função discriminante desde que $g(\mathbf{x})$ seja discriminante e que $f(.)$ seja uma função monotonicamente crescente [2].

3.7 Fronteira de Decisão

Apesar da função discriminante não ser única, sabemos que o resultado do sistema de classificação pode ser encontrado utilizando uma única regra de decisão, ou seja, escolher aquela classe cuja função discriminante tenha resultado no maior valor. Assim, se $g_i(\mathbf{x}) > g_j(\mathbf{x})$ para todo $i \neq j$ devemos escolher a classe ω_i.

Quando as funções discriminantes se igualam para uma dada observação $\mathbf{x}$, ou seja, quando $g_i(\mathbf{x}) = g_j(\mathbf{x})$, podemos escolher arbitrariamente as classes de decisão ω_i ou ω_j. O lugar geométrico dos pontos onde as funções são igualadas é definido como fronteira de decisão. Será mostrado nos próximos capítulos, como, a partir da fronteira de decisão poderemos encontrar a quantidade mínima de vetores de características capaz de realizar uma classificação ótima de padrões [2], [9] - [11].

Capítulo 4

Reconhecimento de Padrões via Redes Neurais

4.1 Introdução

Os estudos de Redes Neurais foram motivados pela percepção de que o cérebro humano processa as informações de forma inteiramente diferente da computação digital convencional. Estudos mostram que uma informação é processada por uma pastilha de silício aproximadamente 1 milhão de vezes mais rápida do que por um neurônio humano [5],[7]. Entretanto, a quantidade de neurônios, o paralelismo com que eles estão conectados e a sua não-linearidade os fazem bastante eficazes para processamento do tipo Reconhecimento de Padrões, percepção e controle motor.

Assim, Redes Neurais Artificiais podem ser vistas como máquinas desenvolvidas para modelar a forma com que o cérebro executa determinadas tarefas. Esta máquina pode ser um Hardware ou uma simulação em Software. Além disto, uma Rede Neural Artificial pode ainda ser vista como um processador de distribuição paralela que tem uma propensão natural para acumular conhecimento e fazê-lo disponível para uso. A sua utilização baseia-se em adquirir conhecimento, através de um processo de aprendizagem, e guardar as informações adquiridas a partir de interconexões, denominadas de sinápses, entre os neurônios.

Dentro de Reconhecimento de Padrões, um tipo de rede do nosso interesse especial são as redes do tipo feedforward. Mostraremos neste capítulo sucintamente, como funciona uma rede neural artificial do tipo feedforward utilizando o algoritmo de Backpropagation para treinamento. Definiremos também um tipo especial de rede

feedforward, chamada Multilayer Perceptron (**MLP**). Em seguida, mostraremos os passos necessários para se criar uma rede neural artificial para classificar ou associar padrões. Finalmente, mostraremos como a mesma pode dividir o espaço amostral de entrada, definindo regiões de decisão, assim como a fronteira de decisão para um problema de classificação de padrões [5], [7] e [14].

4.2 Multilayer Perceptron treinado pelo algoritmo Backpropagation

Tipicamente, uma Rede Neural artificial do tipo feedforward consiste de um conjunto de neurônios sensores na entrada, um conjunto de neurônios escondidos e um conjunto de neurônios de saída.

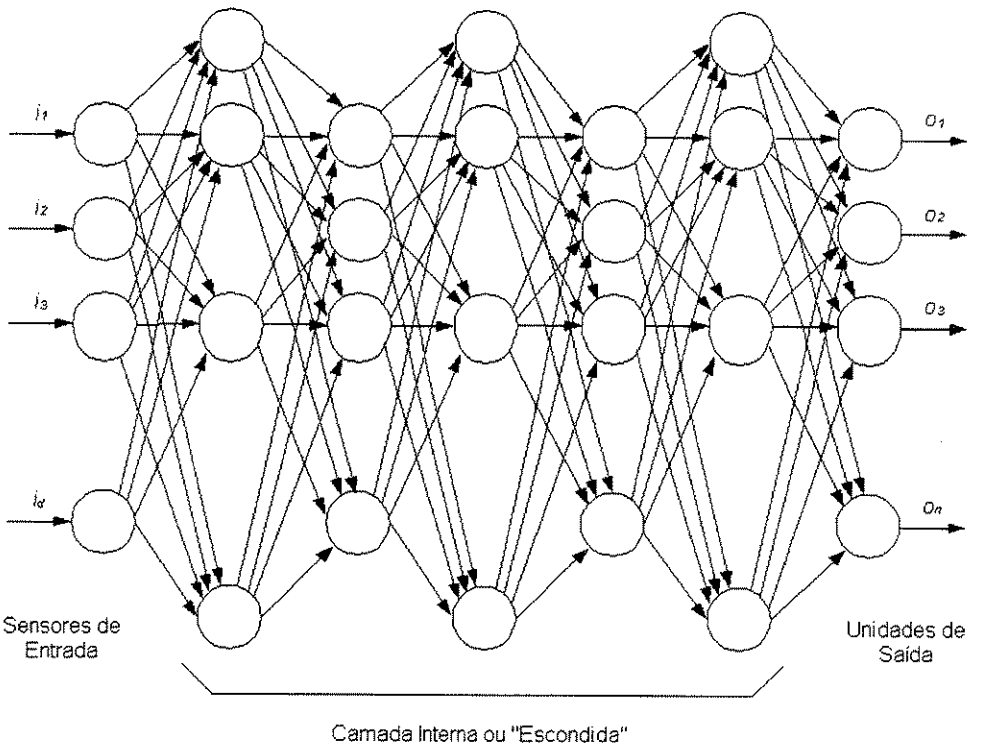

figura 4.1
Rede neural artificial do tipo *feedforward*

Podemos ver na figura 4.1 que o sinal de entrada d-dimensional propaga-se pela rede gerando então no final um sinal de saída que pode ser expresso como função de todos os neurônios da rede.

As Redes Multilayer Perceptron baseiam-se no treinamento supervisionado realizado através de um algoritmo bastante popular conhecido como backpropagation [5]. Este algoritmo baseia-se na regra de aprendizagem pela correção de erro, minimizando o erro quadrático médio encontrado no sinal de saída com relação ao sinal esperado, mostrado na equação (4.1). Note que *y(n)* representa a saída da rede relativa a n-ésima amostra de treinamento, *d(n)* a resposta esperada e *N* representa a quantidade total de amostras de treinamento.

$$\xi_{av} = \frac{1}{N}\sum_{n=1}^{N}(d(n) - y(n))^2 \tag{4.1}$$

O algoritmo backpropagation consiste de duas etapas simples, a propagação para frente(forward) e a propagação para trás(backward). Na propagação para frente um vetor de entrada deve ser aplicado ao conjunto de neurônios sensores (entrada da rede), e o sinal é percorrido camada por camada até a camada de saída. Nesta propagação os pesos das conexões entre os neurônios devem permanecer constantes.

Em seguida deverá haver uma comparação entre o sinal de saída produzido pela rede e a resposta desejada, gerando então um sinal de erro que por sua vez deverá ser propagado para trás. Daí, os pesos das conexões sinápticas são ajustados de forma a minimizar o erro da rede. A figura 4.2 ilustra a direção dos dois sinais básicos da Rede Neural, para frente e para trás de acordo com o sinal de erro ou da função sináptica.

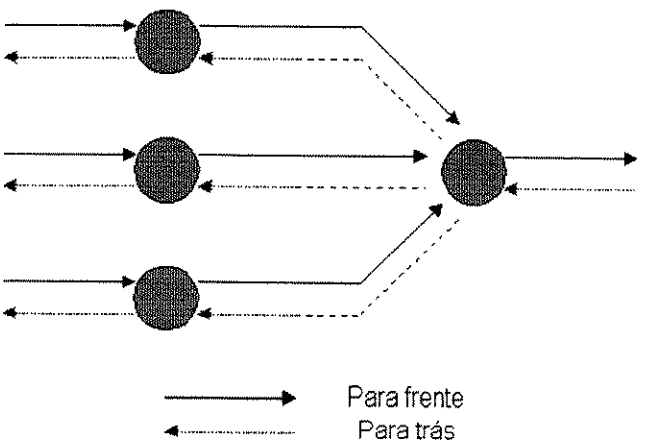

figura 4.2
Rede Neural *para frente*, como função dos neurônios, e p*ara trás* como propagação do erro.

4.3 Uma Rede MLP utilizando a função tangente hiperbólica

Um neurônio funciona como uma função $f(\mathbf{y})$, com $\mathbf{y}$ representando um vetor de entrada. Como foi dito nas seções anteriores, as conexões entre os neurônios são chamadas de sinápses e a elas estão relacionados pesos sinápticos que devem multiplicar cada sinal de entrada do neurônio. Uma vez multiplicadas por seus respectivos pesos as entradas serão somadas e passarão por uma função de ativação que adicionará uma não linearidade ao neurônio. A figura 4.3 mostra em forma de diagrama fluxo de sinais como funciona esta operação.

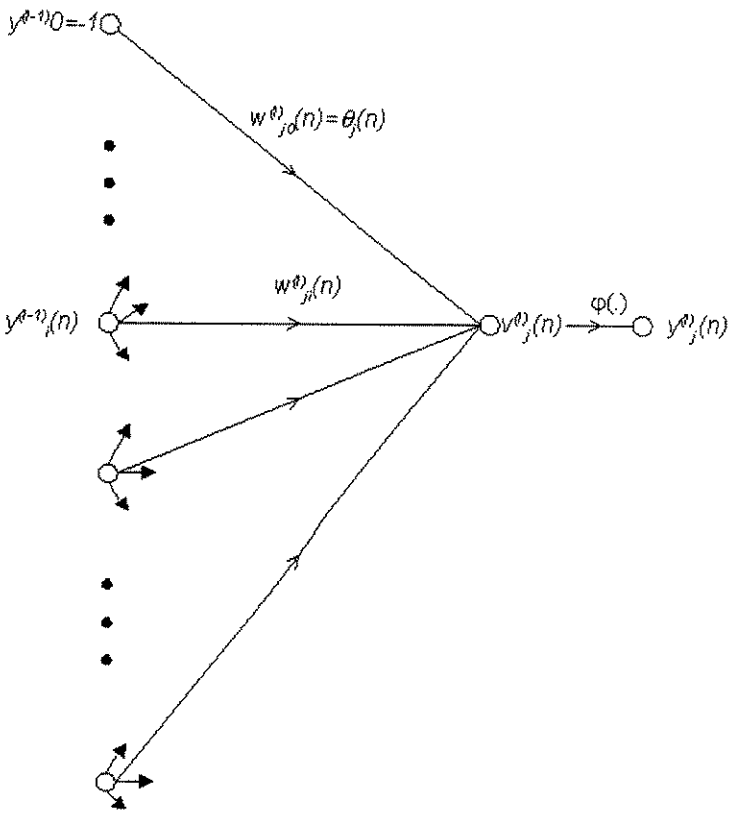

figura 4.3
Fluxo de entrada/saída de um neurônio j na Rede Neural

Traduzindo a figura para a linguagem matemática, haverá um peso sináptico relacionando cada componente de entrada y_i a um neurônio j da próxima camada l. Para cada neurônio j haverá um nível de ativação denotado por $v_j^{(l)}(n)$, e que será calculado pela equação (4.2).

$$v_j^{(l)}(n) = \sum_{i=1}^{p} w_{ji}^{(l)}(n) y_i^{(l-1)}(n) \tag{4.2}$$

Sem perda de generalidade, assumiremos a camada de entrada como sendo a camada 0. Assim, $y_j^{(0)}(n)$ representa a entrada da Rede. A saída da rede será representada por $o_k(n)$, onde $o_k(n) = y_k^{(L)}(n)$, com L sendo a quantidade de camadas da Rede Neural, ou seja, L é o número da camada de saída. Utilizamos também um limiar, também ajustável de acordo com o treinamento do perceptron, que pode ser visto como se existisse uma entrada adicional em todos os neurônios, constante, e de valor igual a -1.

A saída de cada neurônio será dada pela passagem do nível de ativação pela função de ativação $\varphi(.)$, que insere uma não linearidade ao neurônio. Se utilizarmos na nossa rede apenas um neurônio artificial, a função de ativação deveria ser a função sinal, $sgn(v)$ descrita em (4.3). A este tipo de Rede, com apenas um neurônio, chamaremos de perceptron, e podemos encontrar mais informações a seu respeito em [5], [7].

$$\text{sgn}(v) = \begin{cases} +1, & se \quad v > 0 \\ -1, & se \quad v < 0 \end{cases} \tag{4.3}$$

Entretanto, como pode ser visto em [5], para que o algoritmo que minimiza o erro quadrático médio possa convergir, a função de ativação deve ser contínua e diferenciável dentro de todo intervalo de saída. Uma função que segue esta regra e que é bastante utilizada é a função tangente hiperbólica mostrada pela equação (4.4) abaixo.

$$y_j^{(l)} = \varphi(v_j^{(l)}(n)) = \frac{e^{v_j^{(l)}(n)} - e^{-v_j^{(l)}(n)}}{e^{v_j^{(l)}(n)} + e^{-v_j^{(l)}(n)}} \tag{4.4}$$

Esta função é diferenciável e tem o seu gráfico mostrado na figura 4.4. Note que o gráfico desta função se assemelha bastante a função sinal, entretanto ela é diferenciável no intervalo de saída (-1,1).

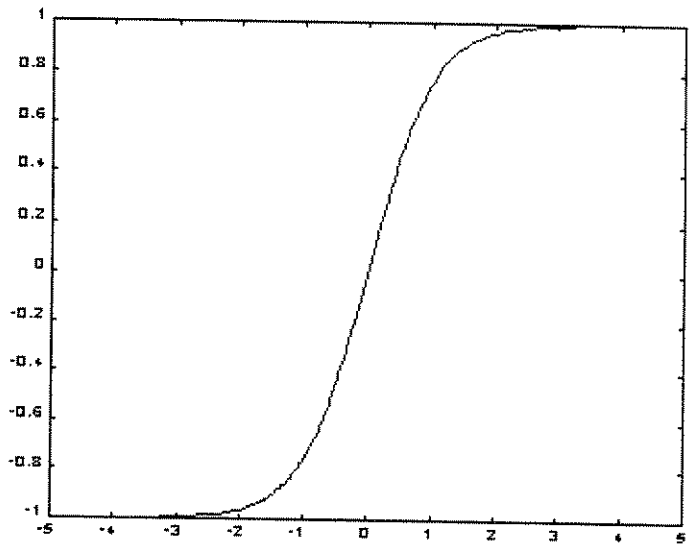

figura 4.4
Gráfico da Função Tangente Hiperbólica

4.4 Treinamento pelo algoritmo Backpropagation

Nesta seção, mostraremos como realizar o treinamento de uma Rede Neural MLP utilizando algoritmo backpropagation. Não é o escopo do trabalho mostrar a prova de convergência do algoritmo, que pode ser encontrada em livros específicos de Redes Neurais tais como [5] e [7].

O primeiro passo para o treinamento da Rede deve ser a sua inicialização. Uma boa escolha dos valores dos parâmetros pode ser decisivo na convergência da rede. Nos casos em que existem informações prévias a respeito da distribuição dos dados, estas informações podem ser utilizadas para determinar os pesos iniciais da rede. Por outro lado, quando estas informações não estão disponíveis, é prática inicializar todos os parâmetros com pequenos valores uniformemente distribuídos.

O treinamento da Rede Neural MLP consiste em dispor a entrada da rede um conjunto de dados de entrada com saída conhecida. Assim, teremos na saída da rede um valor que poderemos comparar com o valor verdadeiro ou desejado.

A diferença entre este valor desejado e o valor gerado na saída da rede é o erro da rede. Para cada neurônio *j* da camada de saída este erro pode ser encontrado pela equação (4.5) abaixo, onde d_j representa a resposta esperada no neurônio *j* e o_j representa a resposta gerada pela rede também no neurônio *j*.

$$e_j(n) = d_j(n) - o_j(n) \tag{4.5}$$

A partir do erro encontrado no neurônio *j* é necessário encontrar o gradiente local deste neurônio. A função deste gradiente, segundo o algoritmo de backpropagation, é indicar uma taxa para atualização dos pesos relativos ao neurônio *j* e ele pode ser encontrado a partir da equação (4.6).

$$\delta_j(n) = e_j(n)\varphi'(v_j(n)) \tag{4.6}$$

A derivada, $\varphi'(.)$, da função de ativação $\varphi(.)$ deve ser tomada com relação a ativação $v_j(n)$ e representa a variação da função de ativação com a própria ativação. No mesmo exemplo da função hiperbólica a derivada, $\varphi'(.)$ é mostrada na equação (4.7) e pode ser reduzida para a equação (4.8).

$$\varphi'_j(v_j(n)) = \frac{4}{\left[e^{v_j(n)} + e^{-v_j(n)}\right]^2} \tag{4.7}$$

$$\varphi'_j(v_j(n)) = [1 + y_j(n)][1 - y_j(n)] \tag{4.8}$$

A partir das equações (4.6) e (4.8) podemos encontrar o gradiente local da rede que será dado pela equação (4.9), quando o neurônio pertencer a camada de saída.

$$\begin{aligned} \delta_j(n) &= e_j(n)\varphi'_j(v_j(n)) \\ &= [d_j(n) - o_j(n)][1 + o_j(n)][1 - o_j(n)] \end{aligned} \tag{4.9}$$

Quando o neurônio pertencer a uma das camadas escondidas da rede, o gradiente local deve ser encontrado a partir da equação (4.10).

$$\begin{aligned}\delta_j(n) &= \varphi'_j(v_j(n))\sum_k \delta_k(n) w_{kj}(n) \\ &= [1+y_j(n)][1-y_j(n)]\sum_k \delta_k(n) w_{kj}(n)\end{aligned} \tag{4.10}$$

Uma vez encontrado o gradiente local de cada um dos neurônios com relação as suas respectivas saídas, deveremos encontrar a variação dos pesos sinápticos a partir da equação (4.11).

$$\Delta w_{ji}(n) = \alpha \Delta_{ji}(n-1) + \eta \delta_j(n) y_i(n) \tag{4.11}$$

Note na equação (4.11) os dois termos α e η que representam respectivamente o *momentum* do treinamento e a taxa de aprendizagem da Rede. Um estudo mais aprofundado destes parâmetros mostram que eles interferem diretamente na velocidade do treinamento da rede. Por exemplo, o *momentum* irá "ajudar" a variação dos pesos caso a variação anterior tenha sido na mesma direção. E a taxa de aprendizado tem a função de escalonar o gradiente do neurônio, permitindo variações mais ou menos rápidas.

4.5 Redes Neurais para Classificação

O desenvolvimento de redes neurais para classificação ou associação de padrões, deve seguir os seguintes passos[5]:

- Definir um mapeamento entrada/saída e estrutura da Rede;
- Escolher o método de treinamento e treinar a rede;
- Testar a Performance do Classificador;

4.5.1 Mapeamento Entrada/Saída:

Considere um problema de classificação de *m* classes distintas, onde a união destas *m* classes definem o espaço completo de classes. No treinamento, devemos informar à

entrada da rede neural cada um dos dados disponíveis. Assim, a rede processará as informações de entrada, e, no caso de um classificador neural, gerará uma saída, que deve determinar a que classe pertence cada uma das entradas da rede.

No caso de uma classificação de imagens, por exemplo, poderíamos ter como entrada todos os pixels da imagem a ser reconhecida, ou simplesmente algumas características que possam representar esta imagem. Por questões práticas, normalmente são utilizadas como vetor de entrada um conjunto de características representativas da imagem. O que reduz a quantidade de dados de entrada e consequentemente o custo computacional.

figura 4.5
Mapeamento Entrada/Saída de um Classificador Neural

No contexto geral da classificação de padrões, a saída da rede poderá ser representada por vários neurônios, cada um representando a saída relativa a uma determinada classe. A figura 4.5, mostra o conceito de mapeamento utilizado numa rede feedforward para classificação, onde X representa o vetor de características de entrada e os diversos y_i representam a saída relativa a classe i. Note que m representa a quantidade de classes possíveis do sistema.

Devemos nos atentar para o fato de que, idealmente, para cada entrada, uma única saída deve ser ativada, representando a classe a qual a amostra de entrada foi associada.

Em classificação de padrões, é conveniente que a rede neural tenha a seguinte saída formulada:

$$y_j = \begin{cases} 1, & se\ x \in \text{Classe}\, j \\ -1, & se\ x \notin \text{Classe}\, j \end{cases}, \text{com}\, j = \{1,2, ..., m\}$$

Entretanto, devido a função de ativação tangente hiperbólica, as saídas não são exatamente iguais a *1* ou *–1*, embora poderiam ser muito próximas a estes valores. Assim, um processo de binarização poderia ser aplicado para definir a classe de saída. Normalmente, o limiar de decisão deveria ser igual a zero. Mas, este tipo de resolução poderia tornar o processo de classificação pouco eficiente, na medida em que várias poderiam ser as saídas cuja ativação verificasse a hipótese $y_j>0$. Logo, não haveria uma classificação única. Para evitar problemas desta natureza, em problemas de classificação de Padrões, com redes do tipo MLP, deve-se utilizar um outro critério para classificação, que excluiria a possibilidade de uma classificação não única. Assim, segue-se o critério:

$$\boldsymbol{X} \in \omega_k, \text{ se } F_k(\boldsymbol{X}) > F_j(\boldsymbol{X}), \text{ para todo } k \neq j.$$

onde $F_i(\boldsymbol{X})$ eqüivale ao mapeamento entrada/saída da entrada $\boldsymbol{X}$, para cada uma das classes ω_k.

4.5.2 Planejamento do processo de treinamento:

O planejamento adequado do processo de treinamento pode maximizar o desempenho do classificador em termos da precisão e da confiabilidade. Num método de treinamento simples, as mesmas amostras utilizadas para o treinamento, poderiam ser utilizadas para testes, ou seja, se a rede já convergiu para um erro aceitável.

Entretanto, este método pode ser falho, na medida em que a rede pode estar realizando uma boa aproximação apenas para os dados do treinamento, mas uma má aproximação para outros dados que não os de treinamento. Um método alternativo, e que pode ser utilizado para atenuar este tipo de problema, é a técnica da validação cruzada, que divide o conjunto de treinamento em dois subconjuntos distintos, um de treinamento e um outro conjunto utilizado exclusivamente para testes. Normalmente, 10 ou 20% do conjunto total de dados deve ser utilizado para fazer o teste de validação cruzada [5].

4.5.3 Teste da Performance do Classificador:

O teste de performance mais comumente utilizado, é o do erro quadrático médio, que pode ser calculado através da expressão (4.1). Esta medida verifica o quão próximo os valores encontrados pela saída da rede estão se aproximando da saída desejada, e assim poderemos definir um critério que finalize o treinamento da rede tão logo este valor se aproxime de um valor aceitável.

4.6 Particionando o Espaço de Entrada

Considere neste momento uma rede neural MLP com apenas uma camada de neurônios escondida. Vamos analisar as funções desempenhadas por cada uma das duas camadas de neurônios (a camada escondida e a camada de saída). Neste temos, uma estrutura com duas camadas de neurônios independentes e com funções distintas. A primeira camada de neurônios é aquela cujos neurônios ligam a entrada aos neurônios de saída da rede e tem a função de particionar o espaço de entrada em células, através das combinações dos hiperplanos definidos pelos neurônios da primeira camada. A segunda camada, por sua vez, tem como objetivo agrupar estas células definindo as regiões de classificação [14].

Assumimos uma rede com uma entrada d-dimensional, p neurônios na camada escondida, e apenas uma saída, indicando um problema de duas classes.

Considere a entrada da rede dada por $\underline{x}=(x_1, x_2,..., x_d)$, e sejam os pesos sinápticos da rede neural denotados por w_{ij} com ij representando a conexão da entrada j da primeira camada, com o neurônio i da camada escondida. Seja também a saída de cada neurônio i da camada escondida representada por y_i. Note que θ_i representa um limiar para cada neurônio.

Idealmente, y_i é uma saída binária determinada pela função $g(z)$ descrita em (4.13). Note que $g(z)$ representa a função de ativação descrita nas seções anteriores. Daí, podemos então escrever:

$$y_i = g\left[\sum_{j=1}^{d} w_{ij} x_j - \theta_i\right], \qquad 1 \le i \le p, \tag{4.12}$$

onde

$$g(z) = \begin{cases} 1, & z \ge 0 \\ 0, & z < 0 \end{cases} \tag{4.13}$$

Note que as expressões (4.12) e (4.13) nos levam a uma função de um vetor d-dimensional, ou seja, $f(x_1, x_2, \ldots, x_d)$, para cada um dos p neurônios da camada escondida. Esta função tem uma saída binária determinada pela posição geométrica da entrada com relação a cada um dos hiperplanos determinados pela equação (4.14) abaixo.

$$\sum_{j=1}^{d} w_{ij} x_j - \theta_i = 0, \qquad 1 \le i \le p, \tag{4.14}$$

Note que $\underline{\mathbf{x}}=(x_1, x_2, \ldots, x_d)$ representa a entrada d-dimensional e que os termos w_{ij} são os pesos de cada sinapse que liga a entrada i com o neurônio j da camada escondida. Note também que os termos w_{ij} também representam os coeficientes da equação do hiperplano relativo ao neurônio j. Note que estes coeficientes também determinam o vetor normal deste hiperplano, e conjuntamente com os diversos θ_i especificam a posição do hiperplano no espaço de entrada.

Analisando então o neurônio da camada de saída, vemos que cada uma das saídas dos neurônios da camada escondida representa um elemento do vetor de entrada para o neurônio da saída. Desta forma, a saída da rede neural pode ser expressa por (4.15).

$$F(\underline{\mathbf{x}}, w) = g\left[\sum_{i=1}^{p} w_i y_i - w_0\right] \tag{4.15}$$

Os diversos w_i representam os pesos das sinápses que ligam os neurônios da camada escondida com os neurônios da última camada, e, determinam a direção do

hiperplano na segunda camada de neurônios. Note que w_0 determina a posição do hiperplano.

A figura 4.6 mostra um exemplo de uma rede com duas entradas, duas camadas de neurônio e apenas uma saída, como descrita acima.

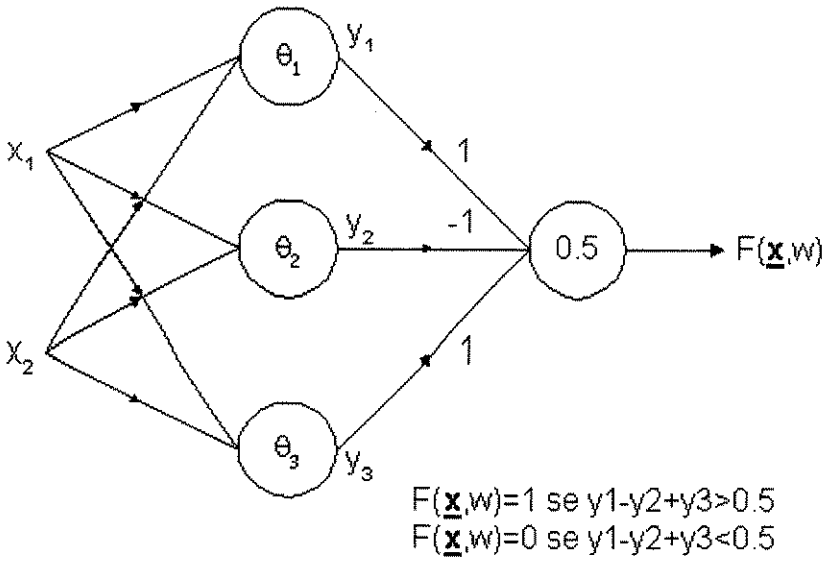

figura 4.6
Rede com duas entradas, três neurônios na camada escondida e uma saída

Como temos duas entradas e três neurônios na primeira camada escondida, teremos três retas denominadas y_1, y_2 e y_3 que podem dividir o espaço bidimensional de entrada em 7 células como mostra a figura 4.7. Note que cada uma das retas geradas, divide o espaço de entrada inteiro em duas regiões distintas, os quais rotulamos como 1 ou 0, dependendo da posição dos pontos com relação ao hiperplano relativo. Assim, as três retas geradas pelos neurônios da segunda camada subdividem o espaço em sete regiões, as quais rotulamos com um número binário $y_3y_2y_1$, onde y_3, y_2 e y_1 especificam o lado em que os pontos se encontram com relação a cada uma das retas.

Assim, quando as amostras de entrada passam por cada um dos neurônios da primeira camada, estes verificam onde a amostra se encontra com relação ao hiperplano e portanto indicarão como saída 1 ou 0. Assim, como a amostra passará em todos os três neurônios da rede, então se encontrará necessariamente em uma das sete regiões

encontradas anteriormente. Note que, neste caso, com hiperplanos na disposição mostrada na figura 4.7, não haveria a possibilidade de haver uma saída do tipo $y_3y_2y_1 = 101$.

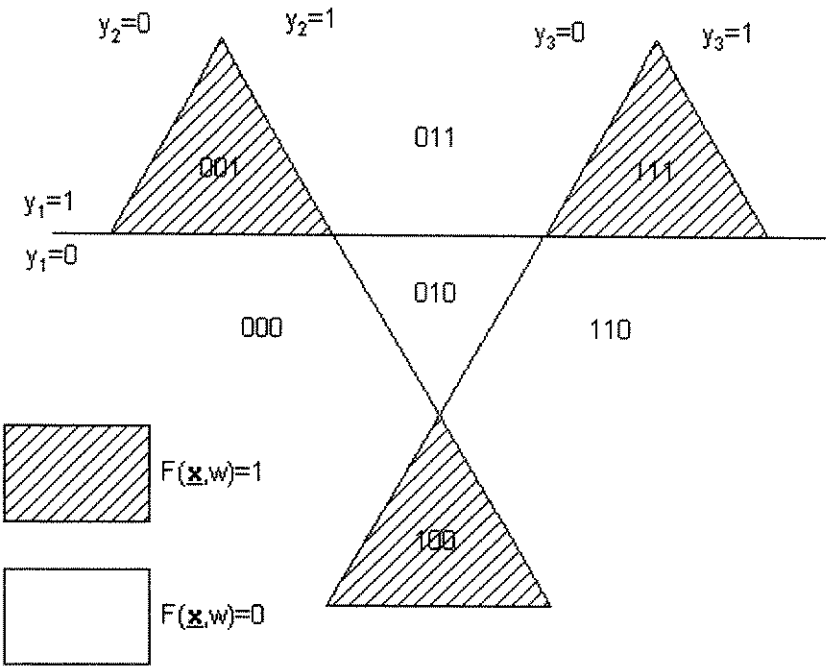

Figura 4.7
Divisão do espaço de entrada em 7 células

É função dos neurônios da segunda camada, classificar a amostra de acordo com a célula em que ela se encontra. Podemos mais uma vez entender o segundo conjunto de neurônios como subdividindo o espaço dos dados da saída da primeira camada de neurônios. No caso, existem três neurônios na camada escondida, logo teremos dados tridimensionais na entrada do neurônio da camada de saída. Além disto, como temos apenas um neurônio na camada de saída, este neurônio dividirá o espaço em apenas duas regiões de classificação. O neurônio da camada de saída representa um plano simples no espaço tridimensional diferente do espaço amostral de entrada.

No caso específico, queremos classificar as regiões 001, 100 e 111 como pertencente a classe $F(\underline{x},w)=1$ e as outras regiões como $F(\underline{x},w)=0$. Logo, podemos ver no gráfico da figura 4.8 que o plano $y_1 - y_2 + y_3 = 0.5$, realiza esta tarefa.

Note que, num caso genérico, o conjunto dos hiperplanos especificados pelos neurônios da primeira camada subdivide o espaço de entrada em regiões que classificam

a amostra. Assim, podemos dizer que numa rede com a estrutura descrita nesta seção, estes hiperplanos formam a fronteira de decisão das classes.

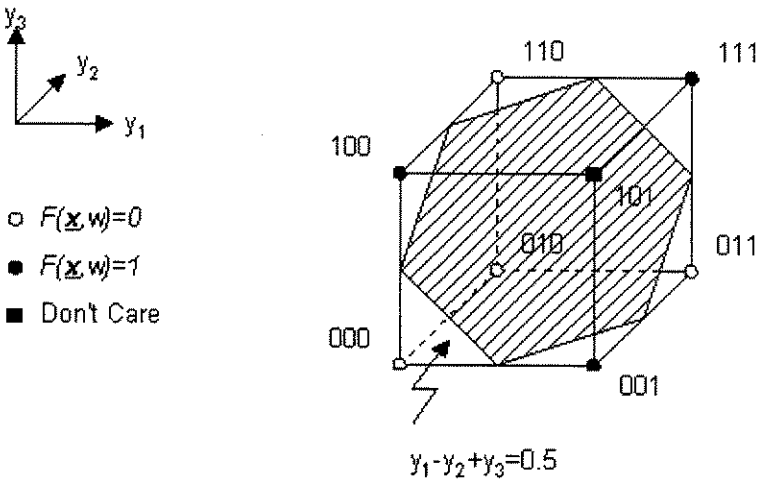

Figura 4.8
Classificação das células geradas pelos neurônios da primeira camada de neurônios

4.7 Consideração Importante

Na seção anterior, vimos que os hiperplanos formados a partir dos neurônios da primeira camada escondida subdividem o espaço amostral de entrada. Na seção 4.6, toda a análise foi feita levando em consideração a função de ativação dada por (4.13) e que representa o caso ideal.

Entretanto, num caso real, todo o treinamento da rede é feito a partir de funções de ativação diferenciáveis, e que consequentemente introduzem uma não-linearidade ao hiperplano. Desta forma, no capítulo 8, mostraremos fronteiras de decisão que dependendo da função de ativação, terá a fronteira de decisão mais ou menos aproximada dos hiperplanos gerados a partir dos neurônios da primeira camada escondida.

Capítulo 5

Extração de Características

5.1 Introdução

No capítulo 2, definimos características e sua função dentro de um sistema de Reconhecimento de Padrões. Vimos então, no capítulo 4, que podemos implementar um classificador de padrões baseado em redes neurais, e que numa rede do tipo MLP, o neurônio da rede com a maior ativação indicará a qual classe pertence a amostra de entrada.

Neste capítulo, temos como objetivo estudar extração de características, e mostrar que todas as características importantes a um problema de classificação podem ser encontradas a partir da fronteira de decisão definida no espaço amostral de entrada da rede.

5.2 Extração de Características

Extração de características, pode ser visto como um meio de encontrar um conjunto de vetores que representem uma observação, enquanto reduzem a dimensionalidade do conjunto original de características [9]-[11],[12].

É desejável extrair características que introduzam alta discriminação entre as possíveis classes dos dados de entrada, eliminando as características redundantes que não contribuem no processo de classificação.

Assim, após a extração de características, obtemos um novo conjunto de dados com dimensão menor do que o conjunto original, o que pode diminuir consideravelmente o custo computacional exigido no processo de reconhecimento. Entretanto, a diminuição da dimensão dos dados não deve comprometer o desempenho do sistema de classificação. É desejável, portanto, manter o mesmo nível de acerto de classificação conseguido com as características originais.

Em resumo, o objetivo principal da extração de características, é encontrar um número reduzido de vetores de características que possam representar uma observação, sem entretanto diminuir o poder de discriminação entre as classes.

5.3 Subespaço Amostral

Nesta seção daremos um tratamento formal a extração de características. Seja R^N um espaço Euclideano de dimensão N contendo as amostras para classificação. A extração de características mencionada aqui é equivalente a encontrar um Subespaço de R^N , R^M, com $M \leq N$, onde para cada vetor de característica no espaço original R^N deve-se ter um vetor de característica correspondente em R^M, que preserve a performance do classificador [9].

Seja X uma observação no Subespaço R^N. Então X pode ser representado por (5.1).

$$X = \sum_{i=1}^{N} a_i \alpha_i \tag{5.1}$$

onde $\{\alpha_1, \alpha_2, ..., \alpha_N\}$ é uma base de R^N.

Seja R^M um subespaço M-dimensional de R^N formado por M vetores linearmente independentes, $\beta_1, \beta_2, ..., \beta_M$.

$$W = Span\{\beta_1, \beta_2, ..., \beta_M\} \qquad \text{e} \qquad \dim(W) = M \leq N$$

Os novos vetores características em R^M são encontrados mapeando as observações originais de R^N em R^M. Assumindo que os diversos β_i's são ortonormais, então o novo conjunto de vetores características no subespaço R^M é dado pela expressão (5.2)

$$\{X^T\beta_1, X^T\beta_2, ..., X^T\beta_M\} = \{b_1, b_2, ..., b_M\} \tag{5.2}$$

onde o vetor $\boldsymbol{B}$, mostrado na expressão (5.3), representa as amostras no novo subespaço M-dimensional.

$$B = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{bmatrix} \tag{5.3}$$

Note que o vetor σ, dado pela expressão (5.4), é uma aproximação de X, em termos de uma combinação linear de $\{\beta_1, \beta_2, ..., \beta_M\}$ no espaço original N-dimensional. Idealmente, se a aproximação for tão boa que a precisão de classificação seja preservada, então teremos novas características, com dimensão M, menor que a dimensão N anterior. Ou seja, teremos diminuído a dimensão das amostras porém preservado a precisão de classificação.

$$\sigma = \sum_{i=1}^{M} b_i \beta_i \tag{5.4}$$

5.4 Regra de Decisão de Bayes

Nesta seção, consideraremos um problema de classificação de uma amostra X em duas classes, ω_1 e ω_2. Utilizando a Regra de Decisão de Bayes, teremos:

Se $P(\omega_1,X) > P(\omega_2,X)$,	Decidir por ω_1
Caso Contrário,	Decidir por ω_2

Onde:

- $P(\omega_i,X)$ é a probabilidade conjunta da amostra X pertencendo a classe ω_i; Expressando $P(\omega_i,X)$ após a aplicação da relação de Bayes, tem-se:

$$P(\omega_1,X) = P(X|\omega_1)\, P(\omega_1) = P(\omega_1|X)\, P(X)$$

- $P(X|\omega_i)$ a função de densidade condicional de X dado que ele pertence a classe ω_i;
- $P(\omega_i)$ a probabilidade *a priori* relacionada a classe ω_i;
- $P(\omega_i|X)$ a probabilidade *a posteriori*, ou seja, a probabilidade da classe ω_i dado uma amostra X qualquer;
- $P(X)$ é a função de densidade de probabilidade da amostra X.

Basicamente, esta regra nos diz que devemos escolher a classe cuja probabilidade conjunta da amostra com a classe for a maior dentre as duas possíveis classes em questão.

A figura 5.1 mostra a separação de 2 classes de dimensão 2 no espaço de características através de uma fronteira linear. Note que uma amostra será classificada como classe ω_1 sempre que $P(\omega_1,X) > P(\omega_2,X)$, e como classe ω_2 caso contrário. Vale salientar que, mesmo assumindo a maior probabilidade conjunta como critério de classificação, ainda teremos um erro de classificação remanescente, entretanto este, como foi mostrado no capítulo 2, será estatisticamente o menor possível.

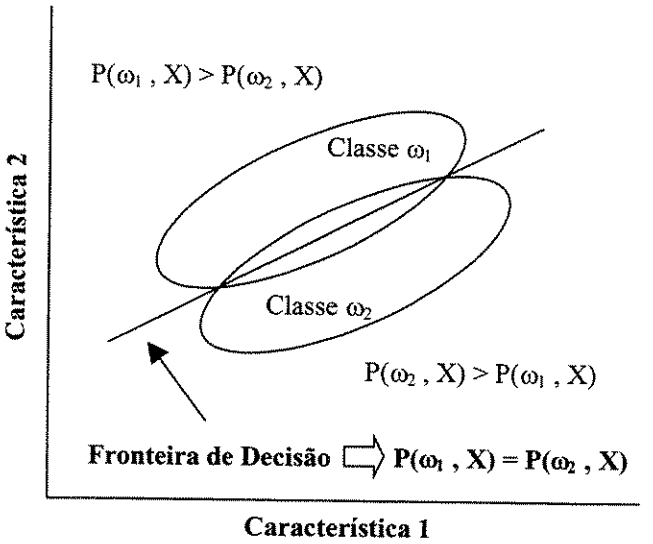

figura 5.1
Fronteira de Decisão

5.5 Fronteira de Decisão

Na Fronteira de Decisão temos $P(\omega_1, X) = P(\omega_2, X)$. Isto é, a probabilidade da amostra pertencer a cada uma das classes é igual. Desta forma, as amostras contidas na fronteira de decisão, podem ser classificadas indiferentemente como pertencentes a classe ω_1 ou ω_2.

DEFINIÇÃO 5.1:

A fronteira de decisão é definida como

$$\{X \mid P(X, \omega_1) = P(X, \omega_2)\}$$

ou seja, como o local geométrico onde estatisticamente não podemos precisar a classe a que pertence os seus pontos. A fronteira de decisão pode ser um ponto, uma linha, uma superfície curva ou uma hipersuperfície curva.

DEFINIÇÃO 5.2:

Apesar da fronteira de decisão poder se estender até o infinito na maioria das vezes grande parte da fronteira de decisão não é significante, ou seja, pouco contribui para a classificação de amostras. Definiremos então a fronteira de decisão efetiva, que representa a parte significante da fronteira de decisão.

A fronteira de decisão efetiva é definida como

$$\{X \mid P(X,\omega_1) = P(X,\omega_2),\ X \in R_1 \text{ ou } X \in R_2\}$$

ou seja, o local geométrico onde o erro estatístico de classificação das amostras, independe da escolha da classe. Onde R_1 representa a menor região que contem uma certa porção da classe ω_1 e R_2 a menor região que contem uma certa porção da classe ω_2.

A figura 5.2 nos mostra um exemplo de fronteira de decisão efetiva, onde a área em destaque contém a parte mais representativa da fronteira de decisão.

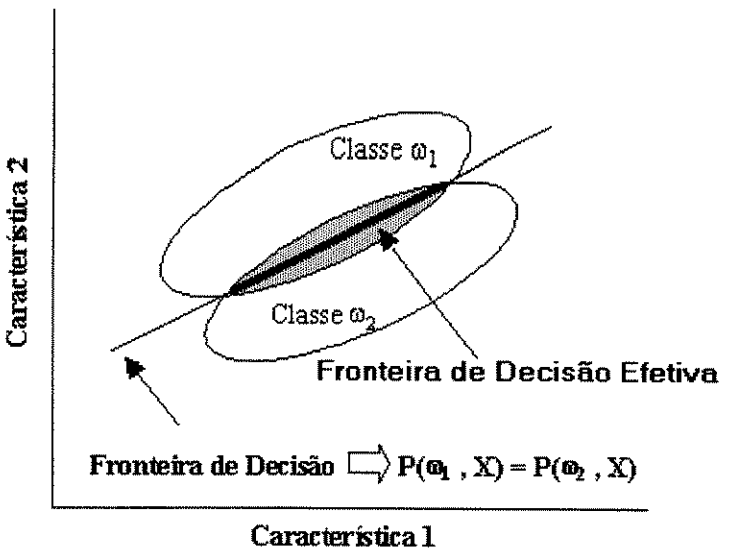

figura 5.2
Fronteira de Decisão Efetiva

5.6 Estudo das Características

Uma possível abordagem para Extração de Características seria o de achar um subespaço R^M com dimensão mínima M onde para qualquer observação X no espaço original R^N teremos uma observação correspondente σ, no subespaço R^M, tal que[9]:

$$[P(\omega_1,X) - P(\omega_2,X)]\ [P(\omega_1,\sigma) - P(\omega_2,\sigma)] \ > \ 0$$

isto é, para que a Extração de característica tenha êxito, as classificações de X no espaço original R^N e de σ no subespaço R^M devem ser necessariamente idênticas.

5.6.1 Característica Redundante

A extração de característica pode ser realizada pela exclusão de informações redundantes. São consideradas características redundantes aquelas que não contribuem na classificação.

Dizemos então que β_k é uma característica redundante se para **qualquer observação** X

$$[P(\omega_1,X) - P(\omega_2,X)]\ [P(\omega_1,\sigma) - P(\omega_2,\sigma)] \ > \ 0$$

onde $X = \sum_{i=1}^{N} b_i \beta_i$ e $\sigma = \sum_{i=1, i\neq k}^{N} b_i \beta_i$.

Em outras palavras, se $X \in \omega_i$ então $\sigma \in \omega_i$ para $i = 1,2$. Observando a figura 5.3 vemos que a característica somente será redundante quando a contribuição da característica não for suficiente para que uma amostra qualquer altere a sua classificação. Note que ela será redundante toda vez que o vetor característica for paralelo à Fronteira de Decisão [9]-[11].

Note pela figura 5.3 que mesmo que uma amostra X excursione pela direção de β_k esta não deixará de ser classificada como pertencente à classe ω_1, pois β_k é paralela à Fronteira de Decisão.

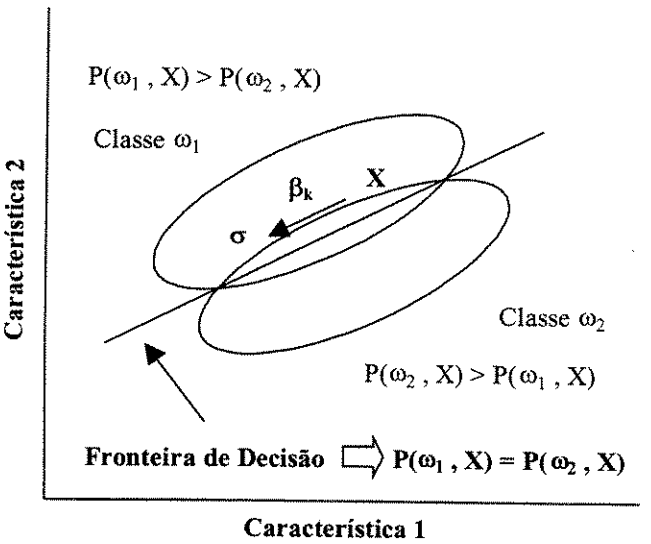

figura 5.3
Característica Redundante

5.6.2 Característica Informativa

Dizemos então que β_k é uma característica informativa se para **pelo menos uma observação** X

$$[P(\omega_1,X) - P(\omega_2,X)]\,[P(\omega_1,\sigma) - P(\omega_2,\sigma)] \;<\; 0$$

$$\text{onde } X = \sum_{i=1}^{N} b_i \beta_i \quad \text{e} \quad \sigma = \sum_{i=1, i\neq k}^{N} b_i \beta_i$$

Em outras palavras, se $X \in \omega_i$ então $\sigma \notin \omega_i$ para $i = 1,2$. Observando a figura 5.4 vemos um vetor característica informativo e verificamos que a característica será informativa toda vez que a contribuição deste vetor característica seja o suficiente para

que uma amostra altere a sua classificação. Note que isto ocorrerá de qualquer forma a não ser que o vetor característica seja paralelo à fronteira de decisão.

No caso, vemos na figura 5.4 que excursionando a amostra X pela direção de β_k esta deixará de ser classificada como pertencente à classe ω_1, pois a contribuição de β_k pode fazer com que a amostra X ultrapasse a Fronteira de Decisão e seja classificada como pertencente a classe ω_2.

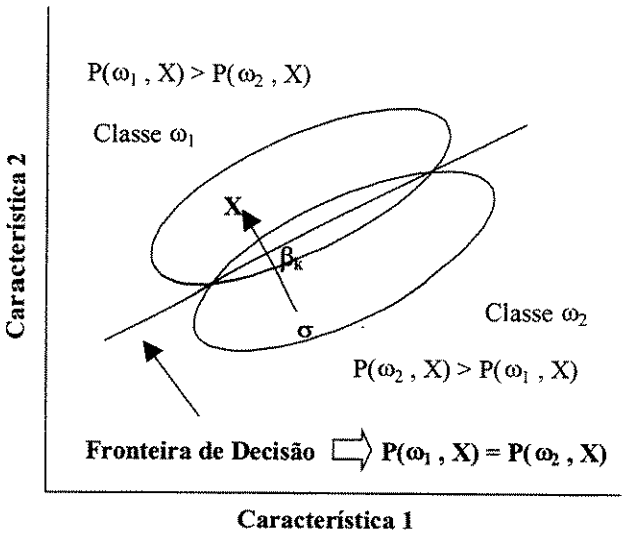

figura 5.4
Característica Informativa

TEOREMA 5.1:

Se num problema de classificação de padrões, um vetor é paralelo à Fronteira de Decisão em todos os pontos pertencentes a ela, então este vetor não contem nenhuma informação útil à classificação. Ele é um vetor REDUNDANTE[9].

Prova: Suponha $\{\beta_1,\beta_2,...,\beta_N\}$ uma base do Espaço Euclideano N-dimensional R^N, e β_N um vetor paralelo à Fronteira de Decisão em todos os pontos da mesma. Seja então R^{N-1} um subespaço formado por $N-1$ vetores, $\beta_1,\beta_2,...,\beta_{N-1}$.

Suponha então que β_N não é uma característica redundante, então deve haver pelo menos uma observação em que:

$$[P(\omega_1,X) - P(\omega_2,X)]\ [P(\omega_1,\sigma) - P(\omega_2,\sigma)] \ < \ 0$$

$$\text{onde } X = \sum_{i=1}^{N} b_i \beta_i \quad \text{e} \quad \sigma = \sum_{i=1}^{N-1} c_i \beta_i$$

Sem perda de generalidade podemos supor que o conjunto de vetores $\beta_1,\beta_2,...,\beta_N$ é um conjunto ortonormal. Então $b_i = c_i$ para $i=1, N-1$.

Logo

$$X' = X - \sigma \ = \ b_N \beta_N.$$

Entretanto, como foi suposto anteriormente, X e σ têm classificações diferentes, e isto implica que eles estão situados em lados opostos com relação à Fronteira de Decisão. Assim, como X' pode ser entendido como uma linha que une X a σ, e b_N apenas como uma constante, então β_N toca a Fronteira de Decisão em um ponto. Portanto β_N não poderia ser paralela a Fronteira de Decisão em todos os seus pontos. **CQD**

Deste teorema se derivam os Lemas 5.1 e 5.2 abaixo que são bastante úteis para encontrarmos características informativas.

LEMA 5.1:

Se um vetor **V** é ortogonal ao vetor normal da fronteira de decisão em todos os pontos da mesma, então este vetor **V** não contém qualquer informação útil a discriminação entre as classes, ou seja, ele é REDUNDANTE [9].

LEMA 5.2:

Se um vetor é normal a fronteira de decisão em pelo menos um ponto da mesma, então este vetor contém informações úteis a discriminação entre as classes, ou seja, o vetor é INFORMATIVO [9].

Capítulo 6

Multilayer Perceptron como uma aproximação da Função Discriminante de Bayes

6.1 Introdução

Neste Capítulo, mostraremos que uma Rede Neural MLP, quando treinada através do algoritmo de Backpropagation, pode aproximar a Função Discriminante de Bayes, em termos do erro quadrático médio. Isto é, a saída da Rede Neural aproxima a função de probabilidade *a posteriori* para cada uma das classes.

6.2 Definições Iniciais

Suponhamos inicialmente um problema de duas classes representadas por ω_1 e ω_2. Seja $\boldsymbol{x}$ um vetor de características a ser classificado e $F(\boldsymbol{w},\boldsymbol{x})$ a saída da Rede Neural em função da entrada, $\boldsymbol{x}$, e dos pesos sinápticos da Rede Neural, representados por $\boldsymbol{w}$. Seja ainda X_i o espaço amostral da classe ω_i e $X_1 \cup X_2 = X$. Também suponha a saída da rede dada por **1** quando o vetor pertencer a classe ω_1 e **–1** quando pertencer à classe ω_2.

6.3 Função Discriminante de Bayes

A Função Discriminante de Bayes, como foi vista no capítulo 2, minimiza a probabilidade de erro de classificação e pode ser escrita de várias formas [2],[21]. Para o nosso propósito, seja $g_0(x)$ uma Função Discriminante de Bayes dada por (6.1).

$$g_0(x) = P(\omega_1 \mid x) - P(\omega_2 \mid x) \tag{6.1}$$

onde $P(\omega_1|x)$ e $P(\omega_2|x)$ são as probabilidades *a posteriori* das classes ω_1 e ω_2, respectivamente. Note que $g_0(x)$ é positivo quando a probabilidade *a posteriori* da classe ω_1 é maior e vice-versa.

A distribuição do vetor característica amostral é dada por (6.2).

$$p(x) = p(x \mid \omega_1)P(\omega_1) + p(x \mid \omega_2)P(\omega_2) \tag{6.2}$$

onde $P(\omega_i)$ é a probabilidade *a priori* da classe ω_i e $p(\mathbf{x}|\omega_i)$ é a função de densidade de probabilidade condicional de $\mathbf{x}$ dado que $\mathbf{x}$ pertence à classe ω_i. Vale salientar que na nossa notação $p(\cdot)$ minúsculo representa uma função de densidade de probabilidade enquanto que o $P(\cdot)$ maiúsculo representa a função de probabilidade.

6.4 Backpropagation aproximando a saída da rede à função discriminante de Bayes

Dada as definições das seções anteriores, vamos mostrar que utilizando o algoritmo de backpropagation, a saída da Rede, $F(\mathbf{x},\mathbf{w})$, aproxima a função discriminante de Bayes, g_0, segundo o critério de erro quadrático médio [20].

Seja $X_1=\{x_1, x_2, \ldots, x_{n1}\} \subset X$ e $X_2=\{x_{n1+1}, x_{n1+2}, \ldots, x_{n1+n2}\} \subset X$. Definamos então a Função Erro Amostral $E_S(w)$ como mostrado na equação (6.3).

$$E_S(w) = \sum_{x \in X_1} (F(x,w) - 1)^2 + \sum_{x \in X_2} (F(x,w) + 1)^2 \tag{6.3}$$

O erro quadrático médio entre a saída da rede, dada por $F(\boldsymbol{x},\boldsymbol{w})$, e a resposta desejada da rede é definido por (6.4).

$$E_a(w) = \lim_{n \to \infty} \frac{1}{n} E_S(w) \tag{6.4}$$

onde n é o número total de amostras.

Rescrevendo a expressão (6.4) de acordo com o número de amostras de cada classe, encontramos então a equação (6.5).

$$E_a(w) = \lim_{n \to \infty} \left[\frac{n_1}{n} \frac{1}{n_1} \sum_{x \in X_1} (F(x,w) - 1)^2 + \frac{n_2}{n} \frac{1}{n_2} \sum_{x \in X_2} (F(x,w) + 1)^2 \right] \tag{6.5}$$

Utilizando o número de amostras utilizadas no treinamento de uma determinada classe proporcional à probabilidade *a priori* desta classe, então quando o número $\boldsymbol{n}$ cresce, os números $\boldsymbol{n_i}$ para todas as classes crescem proporcionalmente, contanto que a classe em questão tenha probabilidade *a priori* diferente de zero. Desta forma, pela Lei dos grandes números, encontraremos a equação (6.6)[18].

$$\begin{aligned} E_a(w) &= P(\omega_1) \int_X (F(x,w) - 1)^2 \, p(x \mid \omega_1) dx + P(\omega_2) \int_X (F(x,w) + 1)^2 \, p(x \mid \omega_2) dx \\ E_a(w) &= \int_X \{ \left[F^2(x,w) + 1 \right] * \left[p(x \mid \omega_1) P(\omega_1) + p(x \mid \omega_2) P(\omega_2) \right] \\ &\qquad - 2F(x,w) \left[p(x \mid \omega_1) P(\omega_1) - p(x \mid \omega_2) P(\omega_2) \right] \} dx \end{aligned} \tag{6.6}$$

Multiplicando então a equação (6.1) por $p(x)$ encontramos a equação (6.7).

$$g_0(x) p(x) = p(x \mid \omega_1) P(\omega_1) - p(x \mid \omega_2) P(\omega_2) \tag{6.7}$$

Manipulando as expressões (6.2), (6.6) e (6.7) encontraremos a equação (6.8) mostrada abaixo

$$\begin{aligned} E_a(w) &= \int_X \left[F^2(x,w)p(x) - 2F(x,w)g_0(x)p(x)\right]dx + 1 \\ &= \int_X \left[F(x,w) - g_0(x)\right]^2 p(x)dx + \left\{1 - \int_X g_0^2(x)p(x)dx\right\} \\ &= \varepsilon^2(w) + \left\{1 - \int_X g_0^2(x)p(x)dx\right\} \end{aligned} \tag{6.8}$$

onde $\varepsilon^2(w) = \int_X \left[F(x,w) - g_0(x)\right]^2 p(x)dx$

Entretanto, o algoritmo de backpropagation minimiza E_S com relação a $\boldsymbol{w}$, portanto é certo afirmar que ele também minimizará E_a com relação a $\boldsymbol{w}$. Como o termo entre as chaves não depende de $\boldsymbol{w}$, minimizar E_a com relação a $\boldsymbol{w}$ é o mesmo que minimizar ε^2 com relação a $\boldsymbol{w}$, ou seja, eqüivale a dizer que o algoritmo de backpropagation minimiza em termos do erro quadrático médio, a diferença entre a saída da rede e a função discriminante ótima de Bayes.

Note entretanto que este resultado depende da arquitetura da rede treinada. Se a rede não for bem dimensionada não haverá uma boa aproximação. Por exemplo, se houver poucos neurônios nas camadas escondidas ou se as amostras escolhidas para treinamento não forem representativas o suficiente, então a aproximação não dará um bom resultado[20].

6.5 Aproximação no caso de várias classes

Nesta seção mostraremos que o resultado obtido na seção anterior, ou seja, que o algoritmo backpropagation minimiza em termos do erro quadrático médio a diferença entre a saída da rede e a função discriminante ótima de Bayes, vale não só para o caso de duas classes como também para um problema com várias classes [20]. Começaremos com algumas definições a serem utilizadas no processo de generalização.

Considere agora um problema de classificação de k padrões (classes) distintos. Seja $F_i(x,w)$ a i-ésima saída de uma rede MLP, com $i=1, 2, ..., k$. As saídas desejadas da rede são dadas por:

$$d_i(x) = \begin{cases} 1 & \text{se } x \in X_i \\ 0 & \text{se } x \notin X_i \end{cases}$$

Define-se a função Discriminante de Bayes para a classe i como:

$$g_i(x) = P(\omega_i \mid x) \qquad \text{para todo } i = 1,2,...,k$$

Neste caso, a regra de decisão deve ser escolher a classe ω_i se $g_i(x) > g_j(x)$ para todo $j \neq i$. Esta regra de decisão, como foi vista na seção anterior, é ótima no sentido de minimizar a probabilidade de erro de classificação ou o custo total da decisão.

O erro pode então ser definido pela expressão (6.9).

$$E_S(w) = \sum_{x \in X_1} \left[\sum_{j \neq 1} F_j^2(x,w) + (F_1(x,w)-1)^2 \right] + \ldots + \sum_{x \in X_k} \left[\sum_{j \neq k} F_j^2(x,w) + (F_k(x,w)-1)^2 \right] \quad (6.9)$$

Pela equação (6.5) encontraremos o erro médio através da equação (6.10):

$$\begin{aligned} E_a &= \lim_{n \to \infty} \frac{1}{n} E_S(w) \\ &= \lim_{n \to \infty} \left\{ \sum_{k=1}^{K} \frac{n_k}{n} \frac{1}{n_k} \sum_{x \in X_k} \left[\sum_{j \neq k} F_j^2(x,w) + (F_k(x,w)-1)^2 \right] \right\} \end{aligned} \quad (6.10)$$

Aplicando novamente a lei dos Grandes Números e simplificando a equação teremos como resultado a equação abaixo descrita.

$$
\begin{aligned}
E_a(w) &= P(\omega_1)\left[\int_X (F_1(x,w)-1)^2\, p(x\mid\omega_1)dx + \int_X F_2^2(x,w)p(x\mid\omega_1)dx + \ldots + \int_X F_k^2(x,w)p(x\mid\omega_1)dx\right] \\
&\quad + P(\omega_2)\left[\int_X F_1^2(x,w)p(x\mid\omega_2)dx + \int_X (F_2(x,w)-1)^2\, p(x\mid\omega_2)dx + \ldots + \int_X F_k^2(x,w)p(x\mid\omega_2)dx\right] \\
&\quad + \ldots + \mathrm{P}(\omega_k)[\ldots] \\
&= \sum_{i=1}^{k}\left[\int_X (F_i(x,w)-1)^2\, p(x\mid\omega_i)P(\omega_i)dx + \sum_{j\neq i}\int_X F_i^2(x,w)p(x\mid\omega_j)P(\omega_j)dx\right] \\
&= \sum_{i=1}^{k}\int_X \left\{F_i^2(x,w)\left[\sum_{j=1}^{k} p(x\mid\omega_j)P(\omega_j)\right] - [2F_i(x,w)-1]p(x\mid\omega_i)P(\omega_i)\right\}dx
\end{aligned}
\tag{6.11}
$$

Pela relação de Bayes, encontramos as seguintes identidades:

$$
\sum_{j=1}^{k} p(x\mid\omega_j)P(\omega_j) = p(x)
$$

$$
\begin{aligned}
p(x\mid\omega_i)P(\omega_i) &= p(x,\omega_i) \\
&= P(\omega_i\mid x)p(x) \\
&= g_i(x)p(x)
\end{aligned}
$$

Assim, a equação (6.11) se transformará em (6.12) que pode ser vista abaixo.

$$
\begin{aligned}
E_a(w) &= \sum_{i=1}^{k}\left[\int_X \left\{F_i^2(x,w)p(x) - [2F_i(x,w)-1]g_i(x)p(x)\right\}dx\right] \\
&= \sum_{i=1}^{k}\left[\int_X [F_i(x,w)-g_i(x)]^2\, p(x)dx + \left\{\int_X g_i(x)(1-g_i(x))p(x)dx\right\}\right]
\end{aligned}
\tag{6.12}
$$

Note que, assim como aconteceu na seção anterior, o termo entre chaves não depende de w, logo o algoritmo Backpropagation ao minimizar o erro médio também minimiza o seguinte termo:

$$
\varepsilon^2(w) = \sum_{i=1}^{k}\int_X [F_i(x,w)-g_i(x)]^2\, p(x)dx \tag{6.13}
$$

Em outras palavras, utilizando o algoritmo de Backpropagation para treinamento, a função da rede aproxima em termos do erro quadrático médio, a função discriminante de Bayes.

6.6 Saída aproxima a Função Discriminante de Bayes

A partir de (6.13) podemos verificar que ε^2 é minimizado segundo o algoritmo backpropagation. Analisando ε^2 verificamos que no caso ideal, ou seja, quando igual a zero, nos gera a expressão (6.14):

$$F_i(x,w) = g_i(x) \tag{6.14}$$

daí,

$$F_i(x,w) \approx P(\omega_i \mid x) \tag{6.15}$$

Ou seja, a saída de uma rede MLP aproxima em termos do erro quadrático médio a probabilidade *a posteriori* de uma amostra pertencer a determinada classe. Daí, poderíamos concluir que idealmente, uma rede treinada até a exaustão nos geraria um classificador ótimo de Bayes.

Note, entretanto que isto é um caso ideal e que depende sensivelmente da arquitetura da rede se por algum acaso o número de células escondidas na rede for insuficiente, a aproximação com relação à função de densidade de probabilidade *a posteriori* será bastante pobre.

Portanto, ao minimizarmos o erro quadrático médio numa rede neural a partir do algoritmo de backpropagation, também aproximamos a saída da nossa rede à função discriminante de Bayes, tanto no problema de duas classes como no problema de multiclasses. Assim, podemos dizer que cada saída da rede treinada aproxima a probabilidade *a posteriori* da sua classe correspondente.

6.7 Backpropagation Bayesiano

Vimos na seção anterior que o algoritmo de backpropagation aproxima a função ótima discriminante de Bayes em termos do erro quadrático médio.

Entretanto, em [17] Nedeljkovic mostrou que apesar do algoritmo backpropagation aproximar a função discriminante de Bayes, em todo o espaço amostral, em pontos específicos podem existir funções que apresentem um erro menor do que a encontrada pela rede treinada.

Como exemplo, suponha a figura 6.1. Nela, vemos que apesar da função 1 aproximar a função discriminante de Bayes em todo o espaço amostral, a função 2 bem diferente no que se refere ao espaço, possui uma aproximação melhor quando analisamos apenas o ponto da fronteira de decisão entre as duas classes.

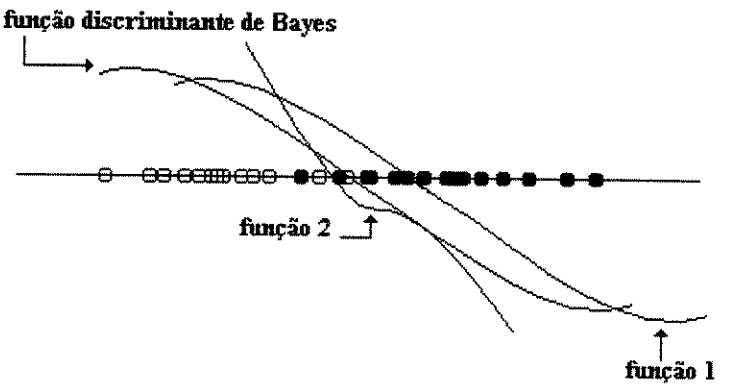

figura 6.1
Aproximação da função discriminante

Como em problemas de classificação de padrões o maior interesse é em diminuir o erro de classsificação, então utilizar um método que minimize a função discriminante de Bayes apenas na fronteira de decisão pode melhorar o desempenho do classificador.

Assim, Nedeljkovic sugeriu em [17] um algoritmo que minimiza a probabilidade de erro de classificação, ou seja, aproxima ainda mais a fronteira de decisão teórica da fronteira de decisão encontrada pela rede neural. A este algoritmo ele deu o nome de backpropagation Bayesiano.

Entretanto, em [12] Ling fez alguns experimentos e verificou que, na prática, a melhoria do método proposto em [17] se mostrou muito pequena em alguns casos, e em alguns outros até piorou o desempenho do sistema de classificação. Desta forma, não chegamos a testar este algoritmo e trabalhamos simplesmente com o algoritmo backpropagation convencional.

Capítulo 7

Algoritmo para Extração de Características via Redes Neurais

7.1 Introdução

Neste capítulo proporemos um novo método para extração de características discriminantes utilizando redes neurais. Este método baseia-se na função discriminante de Bayes, discutida no capítulo 3, e também no fato de que as características discriminantes para fim de classificação podem ser obtidas a partir da fronteira de decisão, o que foi mostrado no capítulo 5.

Assim, mostraremos inicialmente como, a partir dos resultados obtidos nos capítulos anteriores, encontrar a fronteira de decisão do classificador. Uma vez encontrada a fronteira de decisão, vamos definir a Matriz de Característica da Fronteira de Decisão, e mostrar quais as propriedades da mesma que a fazem capaz de encontrar um novo espaço de características de dimensão reduzida, onde as características originais devem ser projetadas.

Por fim, proporemos um algoritmo capaz de realizar a extração de características discriminantes a partir de uma rede neural treinada pelo algoritmo backpropagation.

7.2 Revisão Bibliográfica

A maioria dos algoritmos de extração de características podem ser vistos como transformações lineares. Um dos algoritmos mais utilizados atualmente é a transformada Karhunem-Loève, também conhecida como Análise dos Componentes Principais, e que decompõe os sinais segundo a sua distribuição no espaço de características. Esta transformada é ótima para representação de sinais na medida em que reduz a dimensão dos dados originais encontrando um erro quadrático médio mínimo. Entretanto, nem sempre a transformada gera características ótimas no que se refere a separação de classes [5], [11].

Em análise do discriminante, um conjunto de vetores características é selecionado para maximizar um determinado critério de seleção. Entretanto, análise do discriminante utiliza a diferença entre as médias dos vetores características, daí se esta diferença for pequena o uso da análise do discriminante não é funcional. Além disto, análise do discriminante também apresenta uma limitação por não funcionar em sistemas onde existem dados multi-modais [11].

Patrick apresentou em [25] um processo para extração de características onde uma função distância não quadrática, definida entre as classes, é definida para encontrar o melhor sobespaço linear. Mas, este método seria difícil de ser implementado em redes neurais.

Em [22] Short e Fukunaga propuseram um algoritmo para extração de características particionando o espaço de características em regiões de interesse. Assim, o problema de extração é simplificado e, para cada subregião, é realizado uma estimação linear. Entretanto, este método também seria difícil de ser aplicado utilizando redes neurais, uma vez que deveríamos treinar uma rede para cada subregião.

Redes neurais têm sido bastante utilizadas para resolver problemas de extração de características [111]. E, muitos métodos para extração de características e para seleção de características têm sido propostos. Mao e Jain fizeram uma análise extensiva sobre esta área, e pode ser vista em [15]. Em [4] e [24] é mostrado um estudo sobre a relação entre

análise do discriminante e redes neurais, e em [16] Mao e Jain adaptaram a análise do discriminante para redes neurais.

Em [10], Lee et al. propõe um algoritmo que baseia-se diretamente da fronteira de decisão, e tem por propriedades encontrar a quantidade mínima de característica necessária para obter a mesma precisão de classificação do que no espaço original, e não deteriorar a sua performance mesmo quando existem diferenças pequenas de médias e covariâncias. Este método tem por desvantagem exigir um custo computacional muito alto. Desta forma, neste capítulo mostramos este método e propomos um algoritmo que encontra as características diretamente a partir de uma rede neural treinada sem exigir um custo computacional elevado.

7.3 Revisão dos Capítulos Anteriores

Vimos no capítulo 2 que, dentro de Reconhecimento de Padrões, um dos principais objetivos é encontrar um conjunto de dados que possa representar um objeto de forma eficiente. Assim, a partir de um conjunto de medidas, devemos encontrar aquele conjunto de características que possa distinguir este objeto de qualquer outro.

Entretanto, como foi visto no capítulo 5, o conjunto de dados que se extrai de um sinal normalmente possui uma dimensão muito grande, contendo em geral uma quantidade significativa de informações redundantes. Desta forma, utilizamos extração de características para encontrar um conjunto de vetores de dimensão reduzida que possa representar eficientemente um objeto, ou de forma equivalente, reter somente as informações relevantes.

Vimos também que a utilização de redes neurais pode trazer vantagens significativas com relação a outros métodos de classificação e de extração de características, uma vez que, mesmo sem informações probabilísticas ou estatísticas podemos implementar, idealmente, um classificador de Bayes baseado somente nos dados

experimentais. Além de que, uma vez treinadas, as redes neurais podem ser utilizadas para determinar a fronteira de decisão de um problema de classificação com a vantagem de ter um custo computacional reduzido em relação a outros métodos de classificação.

Mostramos ainda no capítulo 5 que a partir dos neurônios treinados de uma rede MLP, podemos encontrar uma aproximação da fronteira de decisão no espaço amostral de entrada através dos hiperplanos formados pelos neurônios da primeira camada escondida da rede. Ainda neste capítulo, vimos que através desta fronteira de decisão podemos reter somente as informações úteis a classificação.

No Capítulo 6, mostramos que uma rede MLP, adequadamente treinada, aproxima a função discriminante de Bayes. Daí, concluímos que encontrando a fronteira de decisão a partir da rede treinada, aproximamos a fronteira ótima para classificação em termos do erro quadrático médio.

7.4 Matriz de Características da Fronteira de Decisão

No capítulo 5, vimos que um vetor normal em pelo menos um ponto a fronteira de decisão é um vetor característica discriminante informativo. Vimos também no capítulo 5 que os hiperplanos que formam a fronteira de decisão podem ser encontrados a partir dos neurônios da primeira camada escondida da rede neural treinada.

Assim, temos como objetivo nesta seção, definir a Matriz de Características da Fronteira de Decisão[9], Σ_{MCFD}, e em seguida, mostrar algumas propriedades de Σ_{MCFD}, que nos ajudam a encontrar um conjunto de vetores independentes entre si e que formem os vetores normais à fronteira de decisão. Procedendo desta forma, teremos encontrado um novo subespaço onde, quando projetarmos as amostras no espaço original, preservaremos o mesmo nível de classificação das mesmas.

DEFINIÇÃO 7.1: A Matriz de Características da Fronteira de decisão:

Seja $N(X)$ o vetor coluna normal unitário a Fronteira de Decisão no ponto X. Então a Matriz de Características da Fronteira de Decisão Σ_{MCFD} é definida como:

$$\sum_{MCFD} = \frac{1}{K}\int_S N(X)N^T(X)p(X)dX \tag{7.1}$$

onde X representa um ponto pertencente ao espaço amostral de entrada, $p(X)$ representa a função densidade de probabilidade destes pontos, e S o lugar geométrico onde a integral é realizada, ou seja, a própria fronteira de decisão. K é apenas uma constante de normalização dada por $K = \int_S p(X)dX$, e equivale a probabilidade dos pontos pertencerem a fronteira de decisão.

Note que a Matriz Σ_{MCFD} acima é obtida por um processo de integração que fornece a esperança matemática de $N(X)N(X)^T$.

Desta forma, a matriz obtida é uma média sobre a superfície de decisão. Isto quer dizer que a equação acima determinará um hiperplano médio que aproximará a superfície de decisão. Portanto, só fará sentido utilizá-la se a superfície, ou trecho dela, puder ser aproximada razoavelmente por hiperplanos.

7.4.1 Propriedades da Matriz de Características da Fronteira de Decisão

Nesta seção listaremos algumas propriedades da Matriz de Característica da Fronteira de Decisão. As provas destas propriedades podem ser encontradas em [9]-[11].

- Propriedade 1: A Matriz de Característica da Fronteira de Decisão é real e simétrica.
- Propriedade 2: Os autovetores da Matriz de Característica da Fronteira de Decisão são ortogonais.

- Propriedade 3: A Matriz de Característica da Fronteira de Decisão é positiva e semidefinida.

- Propriedade 4: A Matriz de Característica da Fronteira de Decisão pode ser expressa como uma soma de matrizes calculadas por segmentos que compõem a fronteira de decisão.

Estas propriedades nos levam a dois teoremas descritos a seguir:

TEOREMA 7.1: O Rank da Matriz de Característica da Fronteira de Decisão Σ_{MCFD} de um problema de Classificação de Padrões representará a quantidade mínima de características capazes de classificar corretamente um objeto. Em outras palavras, será a dimensão mínima que o vetor de características pode ter, preservando a precisão de classificação da dimensão original [9].

Prova: Seja X uma observação no espaço Euclideano N-dimensional R^N sobre a hipótese H_I: $X \in \omega_i$ $\{i=1, ..., J\}$ onde J é o número das classes. Seja Σ_{MCFD} a Matriz de Características da Fronteira de Decisão como definida anteriormente. Suponha então que

$$rank(\Sigma_{MCFD}) = M \leq N$$

Sejam $\{\phi_1, \phi_2, \ldots, \phi_M\}$ os autovetores de Σ_{MCFD} correspondentes aos autovalores não nulos. Então um vetor normal a fronteira de decisão em qualquer ponto da fronteira de decisão pode ser representado como uma combinação linear dos autovetores de Σ_{MCFD}, isto é,

$$V_{NORMAL} = \sum_{i=1}^{M} a_i \phi_i$$

Qualquer conjunto de vetores linearmente independentes pode ser estendido para uma base do espaço vetorial. Assim, nós podemos expandir $\{\phi_1, \phi_2, \ldots, \phi_M\}$ para formar uma

base para o espaço Euclideano N-dimensional. Seja então a base estendida dada por $\{\phi_1, \phi_2, \ldots, \phi_M, \phi_{M+1}, \ldots, \phi_N\}$.

Sem perda de generalidade, podemos assumir que $\{\phi_1, \phi_2, \ldots, \phi_M, \phi_{M+1}, \ldots, \phi_N\}$ formam uma base ortonormal, pois sempre é possível encontrar uma base ortonormal para um espaço vetorial usando o procedimento de Gram-Schmidt. Como a base é assumidamente ortonormal, todos os vetores $\{\phi_{M+1}, \ldots, \phi_N\}$ são ortogonais a qualquer vetor V_{NORMAL} normal a fronteira de decisão. Isto porque para $i=M+1, \ldots, N$.

$$\phi_i^T V_{NORMAL} = \phi_i^T \sum_{k=1}^{M} a_k \phi_k$$

$$= \sum_{k=1}^{M} a_k \phi_i^T \phi_k = 0 \qquad \text{dado que } \phi_i^T \phi_k = 0 \text{ se } i \neq k$$

Desta forma, de acordo com o que foi visto no LEMA 5.1 do Capítulo 5, os vetores $\{\phi_{M+1}, \ldots, \phi_N\}$ não contribuem para discriminar as classes portanto podem ser considerados redundantes. Portanto, o número de características redundantes é N-M e a dimensão mínima do vetor de características é o rank da matriz de características da Fronteira de Decisão. **CQD**

Note que não assumimos nada a respeito da quantidade de classes do problema. Desta forma, este resultado serve para qualquer quantidade de classes de um problema de classificação de padrões.

TEOREMA 7.2: Os autovetores da Matriz de Características da Fronteira de Decisão num problema de Classificação de Padrões correspondentes aos autovalores não nulos são os vetores características necessários para preservar a precisão de classificação encontrada no espaço original do problema. [9]

Prova: Na prova do Teorema 7.1, foi mostrado que os autovetores Σ_{MCFD} correspondentes aos autovalores não nulos são os únicos vetores discriminante informativo. Assim,

podemos desprezar os autovetores de Σ_{MCFD} cujos autovalores são nulos, uma vez que os mesmos possuem informações redundantes.

7.4.2 Procedimento para encontrar a Matriz de Característica da Fronteira de Decisão

No capítulo 4 vimos que, dada uma rede neural MLP treinada para classificação, podemos encontrar os vetores normais $N(X)$ aos hiperplanos que formam a fronteira de decisão Σ_{MCFD} a partir dos pesos sinápticos dos neurônios que formam a primeira camada escondida.

Lembramos que, como foi dito na seção 4.7, para a nossa dedução, a rede MLP ideal é aquela apresentada na seção 4.6 cujos neurônios da camada escondida têm função de ativação abrupta e não diferenciável. Daí, os vetores normais à fronteira de decisão equivalem exatamente aos pesos sinápticos dos neurônios da primeira camada escondida. Entretanto, nos nossos experimentos mostraremos como as funções de ativação diferenciáveis influenciam na fronteira de decisão proposta.

Já na seção anterior vimos que a Matriz de Característica da Fronteira de Decisão é encontrada pelos vetores normais à fronteira de decisão. Esta Matriz tem as características e propriedades descritas na seção anterior, e os autovetores correspondentes aos autovalores não nulos desta Matriz devem gerar um subespaço onde as características originais poderão ser projetadas sem haver perda na precisão da classificação.

Teoricamente, a Matriz pode ser encontrada a partir de (7.1). Entretanto, podemos esperar que a fronteira de decisão entre todas as classes do problema de classificação seja uma combinação de todos os hiperplanos, no espaço amostral de entrada, encontrados a partir dos pesos sinápticos dos neurônios da primeira camada escondida da Rede Neural.

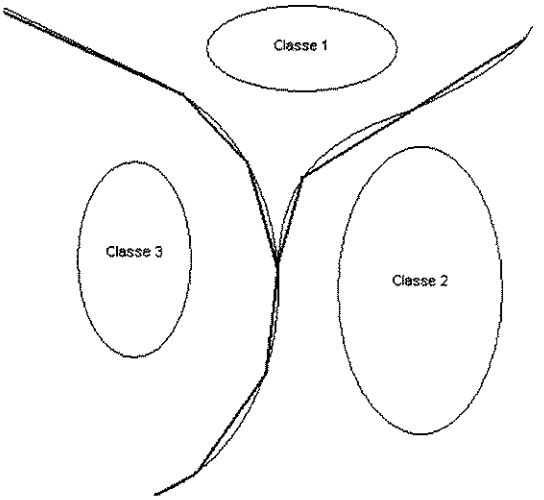

figura 7.1
Fronteira de Decisão Aproximada

Como foi visto no Capítulo 5, todos os hiperplanos encontrados serão combinados pelos outros neurônios das camadas seguintes da Rede Neural gerando assim a fronteira de decisão, que será uma aproximação em termos de hiperplanos da fronteira real entre as classes. Na figura 7.1 temos um exemplo de um possível sistema de classificação de padrões com três classes. Nela mostramos a distribuição de três classes, a fronteira de decisão real e uma fronteira de decisão que aproxima a fronteira real por hiperplanos.

Vale salientar que a estrutura da rede poderá ser determinante para se obter uma fronteira de decisão próxima da real. Idealmente, quando o número de neurônios na camada escondida tender a infinito teremos então a fronteira de decisão neural igual a fronteira de decisão real entre as classes.

Devemos então encontrar Σ_{MCFD} a partir da equação (7.1), entretanto, como a fronteira de decisão é constituída por pedaços de hiperplanos, o cálculo da Matriz de Características da Fronteira de Decisão pode ser simplificado resultando na equação (7.2) mostrada abaixo:

$$\begin{aligned}\sum_{MCFD} &= \sum_{i=1}^{L} N_i N_i^T \frac{1}{K} \int_S p(X)dX \\ &= \sum_{i=1}^{L} N_i N_i^T\end{aligned} \tag{7.2}$$

onde N_i equivale ao vetor gerado pelos pesos sinápticos do i-ésimo neurônio da camada escondida e L representa a quantidade total de neurônios da camada escondida.

7.4.3 Subespaço gerado

Pelos teoremas 7.1 e 7.2, sabemos que os autovetores correspondentes a autovalores não nulos de Σ_{MCFD}, formam um subespaço onde as projeções das amostras originais neste novo subespaço preserva a precisão de classificação das amostras.

Procedendo desta forma, estaremos criando um novo espaço com a mesma capacidade de classificação do espaço anterior, entretanto com uma dimensão menor do que a original. Daí, mapeando as amostras originais no novo espaço gerado, encontraremos uma nova representação para as amostras, que são separadas por uma outra fronteira de decisão equivalente no novo espaço.

7.5 Algoritmo de Extração de Características Lee/Landgrebe

Nesta seção mostraremos um outro método de extração de características proposto em [9]-[11] por Lee e Landgrebe.

Seja uma rede MLP treinada pelo algoritmo backpropagation, e assuma um problema de duas classes. Daí, podemos pensar numa rede com apenas duas saídas, sendo uma para cada classe.

Assim, utilizando as definições feitas anteriormente, define-se fronteira de decisão de uma Rede Neural, como a região geométrica onde ambos os neurônios de saída têm o mesmo valor de ativação.

Denotando-se a saída de cada neurônio i como $OUT_i(X)$ onde $\boldsymbol{X}$ é um vetor de entrada e seja $h(X) = OUT_1(X) - OUT_2(X)$, então a região de decisão pode ser definida como:

$$\{ \boldsymbol{X} \mid \boldsymbol{h(X) = 0} \}. \tag{7.3}$$

Como não podemos resolver (7.3) analiticamente, devemos encontrar a fronteira de decisão numericamente. Daí, utilizamos o seguinte procedimento numérico:

- Assuma Y e Z classificados corretamente em classes distintas. A linha que une Y a Z, deve então passar pela fronteira de decisão.

- Movendo então um ponto sobre esta linha, podemos encontrar um ponto X na vizinhança da fronteira de decisão dentro de um intervalo ε, como pode ser verificado na figura 7.2.

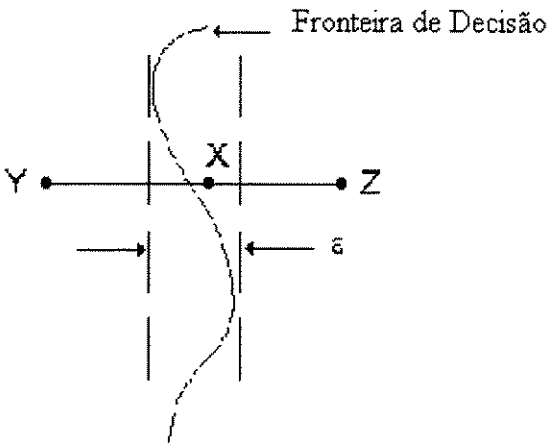

figura 7.2
Fronteira de Decisão achada numericamente

Uma vez achado o ponto X, o vetor normal a fronteira de decisão no ponto X é dado por 7.4:

$$\nabla h(X) = \frac{\partial h}{\partial x_1} x_1 + \frac{\partial h}{\partial x_2} x_2 + ,\ldots, \frac{\partial h}{\partial x_n} x_n \tag{7.4}$$

onde

$$h(X) = OUT_1(X) - OUT_2(X)$$

e

$$OUT_1(X) = F(W_h^1 F(W_i X))$$

$$OUT_2(X) = F(W_h^2 F(W_i X))$$

sendo:

W_i - *Matriz de pesos de entrada;*

W_h^i - *Matriz dos pesos escondidos;*

F - *Função de ativação;*

Entretanto, como o número de camadas escondidas pode ser variável, então o cálculo analítico de (7.4) pode se tornar bastante complexo, daí utilizou-se o cálculo numérico do gradiente de h, pela expressão 7.5:

$$\nabla h(X) \approx \frac{\Delta h}{\Delta x_1} x_1 + \frac{\Delta h}{\Delta x_2} x_2 + ,\ldots, \frac{\Delta h}{\Delta x_n} x_n \tag{7.5}$$

Pode-se então achar o vetor normal unitário N_i através de 7.6:

$$N_i = \frac{\nabla h(X)}{|\nabla h(X)|} \tag{7.6}$$

E a nova matriz de características da fronteira de decisão poderá ser estimada por $\Sigma_{DBFM} = \frac{1}{L}\sum_i N_i N_i^T$, onde L é o número de amostras corretamente classificadas.

Daí, devemos encontrar os autovetores e autovalores correspondentes a partir da matriz de características da fronteira de decisão, e extrair as características como descrito no capítulo 5.

7.6 Algoritmo Lee/Cavalcanti

Na seção anterior vimos o algoritmo de extração de características apresentado por Chulle Lee et al. em [9]. De uma forma geral, o algoritmo é válido, e pode ser utilizado em qualquer sistema de reconhecimento de padrões que utilizar uma rede MLP para treinamento. Entretanto, o algoritmo possui uma grande limitação, na medida em que o mesmo requer um custo computacional muito elevado para encontrar a fronteira de decisão do problema.

Desta forma, nesta seção propomos um algoritmo capaz de minimizar o custo computacional para encontrar a fronteira de decisão, onde encontramos a fronteira de decisão diretamente da rede neural treinada. Chamamos então o algoritmo de extração de características de algoritmo de extração de características Lee/Cavalcanti, e como será mostrado nesta seção e no capítulo destinado a apresentação dos resultados, o algoritmo possui um desempenho semelhante ao algoritmo proposto por Chulle Lee et al. em [9].

Vimos no capítulo 4, que podemos encontrar os vetores normais aos hiperplanos que formam a fronteira de decisão a partir dos neurônios da primeira camada escondida de uma rede neural MLP.

Os vetores normais a cada um dos hiperplanos que formam a fronteira de decisão são encontrados pelos pesos sinápticos dos neurônios que ligam a entrada de dados aos neurônios da primeira camada.

Assim, sendo i o número do neurônio da camada escondida e d a dimensão dos dados de entrada, encontraremos em (7.7) os vetores normais, φ_i, aos hiperplanos que formam a fronteira.

$$\varphi_i = \begin{bmatrix} w_{i1} \\ w_{i2} \\ \vdots \\ w_{id} \end{bmatrix} \qquad 1 \leq i \leq L \tag{7.7}$$

onde L representa a quantidade de neurônios da primeira camada escondida da Rede Neural MLP de classificação.

Neste momento, dispomos de todos os vetores normais $\{\varphi_1, \varphi_2, \ldots, \varphi_L\}$ aos hiperplanos que formam a fronteira de decisão. Poderemos então encontrar a Matriz Característica da Fronteira de Decisão a partir da equação (7.8).

$$\sum_{DBFM} = \sum_{i=1}^{L} N_i N_i^T \tag{7.8}$$

onde

$$N_i = \frac{\varphi_i}{|\varphi_i|}$$

Isto é, N_i representa o vetor unitário normal ao hiperplano i. E a Matriz Σ_{MCFD} será a responsável pela elaboração de um subespaço onde as amostras projetadas neste subespaço deverão manter a mesma precisão de classificação.

Deveremos então encontrar os autovetores e autovalores correspondentes da matriz encontrada em (7.8). Sendo D a dimensão dos dados originais, então deveremos encontrar D autovetores e autovalores correspondentes relacionados à Matriz da Fronteira de Decisão.

Entretanto, os autovetores correspondentes a autovalores nulos deverão ser desprezados, sobrando apenas M vetores que também formam um subespaço R^M, com dimensão M, com $M<D$. Desta forma,

$$\{\phi_1, \phi_2, ..., \phi_M\} \text{ forma uma base ortonormal para } R^M \tag{7.9}$$

Deveremos então projetar as amostras originais, representadas por X, no novo subespaço R^M. Assim, a nova amostra será representada por Y e será obtida em (7.10):

$$Y=\{X^T\phi_1, X^T\phi_2, ..., X^T\phi_M\} \tag{7.10}$$

Fazendo então esta mudança de base para cada uma das amostras teremos um novo conjunto de dados que mantém a mesma precisão de classificação do conjunto de dados original, entretanto possuem uma dimensão menor que a original. Vale salientar que este novo conjunto de amostras deve ser treinado numa nova rede neural. Entretanto, apesar de existir um novo treinamento, este será mais rápido que o primeiro por se tratar de um conjunto de dados de menor dimensão, e como normalmente uma aplicação de classificação tem um uso bastante repetitivo, então o algoritmo poderá reduzir substancialmente o tempo de processamento de classificação.

7.6.1 Resumo do Algoritmo Lee/Cavalcanti

PASSO 1) Treinar a rede MLP com todas as amostras na dimensão original;

PASSO 2) Encontrar todos os vetores normais aos hiperplanos que formam a fronteira de decisão diretamente da Rede Neural.

$$\varphi_i = \begin{bmatrix} w_{i1} \\ w_{i2} \\ \vdots \\ w_{id} \end{bmatrix} \qquad 1 \leq i \leq L$$

PASSO 3) Normalizar os vetores encontrados no passo 2:

$$N_i = \frac{\varphi_i}{|\varphi_i|}$$

PASSO 4) Encontrar a Matriz de Característica da Fronteira de Decisão utilizando os vetores normais encontrados no passo 3:

$$\sum_{DBFM} = \sum_i N_i N_i^T$$

PASSO 5) Selecionar os autovetores representativos de acordo com a magnitude dos seus autovalores. Escolheremos aqueles cujos autovalores são não-nulos;

PASSO 6) Encontrar um novo conjunto de amostras usando os autovetores encontrados no passo 5;

$$Y=\{X^T\phi_1, X^T\phi_2, \ldots, X^T\phi_M\}$$

PASSO 7) Treinar uma nova Rede a partir das amostras encontradas no passo 6;

Capítulo 8

Resultados Obtidos

8.1 Introdução

No capítulo 4 mostramos que a partir dos pesos sinápticos dos neurônios da primeira camada escondida de uma rede neural MLP, como a mostrada na seção 4.6, treinada para classificação, encontramos os vetores normais aos hiperplanos que formam a fronteira de decisão. Já no capítulo 7 vimos o algoritmo Lee/Cavalcanti que tem como principal objetivo extrair características baseada na fronteira de decisão a partir de uma Rede Neural MLP treinada para classificação. Assim, neste capítulo, descreveremos inicialmente as simulações realizadas e mostraremos como a fronteira de decisão pode ser realmente encontrada a partir dos hiperplanos encontrados pelos neurônios da camada escondida. Em seguida, comprovaremos a eficiência do algoritmo de extração de características Lee/Cavalcanti exposto no Capítulo 7.

Desta forma, na seção 8.2 descreveremos algumas características do classificador MLP utilizado para os testes. A seção 8.3 é dividida em várias subseções que mostram separadamente a fronteira de decisão obtida em vários exemplos. Na seção 8.4, mostraremos o efeito sobre os hiperplanos quando incluímos funções de ativação suaves aos invés de abruptas, como citado no capítulo 4. Na seção 8.5, mostraremos alguns exemplos em que realizamos a extração de características. Na seção 8.6 faremos uma comparação entre a fronteira de decisão encontrada pelo método Lee/Cavalcanti e a fronteira encontrada pelo método Lee/Landgrebe, que encontra a fronteira de decisão a

cada ponto. É importante ressaltar que nesta seção utilizaremos dados sintéticos com distribuição gaussiana gerados a partir do programa SAS.

Na seção 8.7 mostraremos os resultados encontrados a partir da extração de características realizados em dados reais. Os dados reais utilizados foram os mesmos utilizados por Lizarraga/Lee para reconhecimento de assinaturas estáticas em [13]. Finalmente, na seção 8.8, tiraremos nossas conclusões a respeito do trabalho realizado e daremos algumas sugestões para trabalhos futuros.

8.2 A Implementação do classificador neural

Para cada conjunto de dados da seção 8.3 tivemos de implementar um classificador baseado numa rede MLP treinada pelo algoritmo de backpropagation.

Realizamos então o treinamento da Rede Neural utilizando metade dos dados gerados, e utilizando a outra metade dos dados para o teste de classificação. A Rede foi implementada utilizando a linguagem "C" e também o programa MATLAB.

O critério de parada é dado pela convergência da rede em termos do erro quadrático médio. Quando a rede não consegue convergir para um erro pré-determinado ε, então utilizamos como critério de parada a quantidade de épocas de treinamento. Neste trabalho, adotamos 50.000 épocas. Este valor foi escolhido experimentalmente já que havia pouca mudança no erro quadrático médio da rede após 50.000 ciclos de treinamento.

A mesma configuração de rede é adotada para todos os exemplos mostrados, a qual consiste de uma rede com apenas uma camada de neurônios escondidos. Porém, com quantidade de neurônios escondidos variável a fim de verificarmos como a fronteira de decisão se comporta de acordo com a quantidade de neurônios.

O passo do treinamento, η, é fixo e igual a 0,9 com um momentum, α, nulo. Assim, a taxa com que os pesos sinápticos dos neurônios da rede são atualizados é fixa e a variação anterior do peso não será considerada.

Implementamos a rede neural utilizando a função tangente hiperbólica como a função de ativação da rede. Entretanto em alguns casos modificamos o coeficiente *coef* da função tangente hiperbólica *tanh(coef*x)*. A alteração deste coeficiente faz com que a função de ativação modifique a suavidade de sua transição, o que, como veremos nos exemplos da próxima seção e como foi descrito no capítulo 4, faz com que a fronteira de decisão se torne mais ou menos abrupta.

8.3 Estudo experimental da fronteira de decisão utilizando dados sintéticos

Com o objetivo de encontrar a fronteira de decisão, utilizamos alguns exemplos de tipos de fronteira de decisão sugeridos em [2] por Duda & Hart. Neste caso, como os dados utilizados são gaussianos, é possível encontrar a função discriminante de Bayes na forma analítica para cada uma das classes, a partir da expressão (8.1). A fronteira de decisão teórica pode ser obtida, e mostrada explicitamente fazendo $g_1(x) = g_2(x)$. Assim, para cada caso abaixo mostraremos também a fronteira de decisão teórica. Os exemplos a seguir mostram variadas formas de fronteira como uma parábola, hiperbole, circular, reta e elíptica.

$$g_i(x) = x^t W_i x + w_i^t x + w_{i0}$$

onde

$$W_i = -\frac{1}{2}\Sigma_i^{-1}$$
$$w_i = \Sigma_i^{-1}\mu_i \qquad (8.1)$$

e

$$w_{i0} = -\frac{1}{2}\mu_i^t\Sigma_i^{-1}\mu_i - \frac{1}{2}\log|\Sigma_i| + \log P(\omega_i)$$

8.3.1 Exemplo 8.1

No primeiro exemplo, as classes ω_1 e ω_2 têm distribuição normal com os seguintes parâmetros:

$$\omega_1: \qquad \mu_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \Sigma_1 = \begin{bmatrix} 9 & 0 \\ 0 & 9 \end{bmatrix}$$

$$\omega_2: \qquad \mu_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \Sigma_2 = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

Geramos um conjunto de 10.000 amostras de cada uma das classes. Sendo que deste total, 5.000 amostras de cada classe foram utilizadas para testar o classificador. Podemos ver na figura 8.1 o conjunto de dados gerados. As amostras representadas em vermelho pertencem a classe ω_1 enquanto as amostras representadas em verde pertencem a classe ω_2. Note que pare que pudéssemos visualizar as amostras de ambas as classes numa só figura, plotamos inicialmente as amostras da classe ω_1, e em seguida sobrepomos os pontos relativos a classe ω_2. Desta forma, existem pontos pertencentes a classe ω_1 que não podem ser visualizados nesta figura.

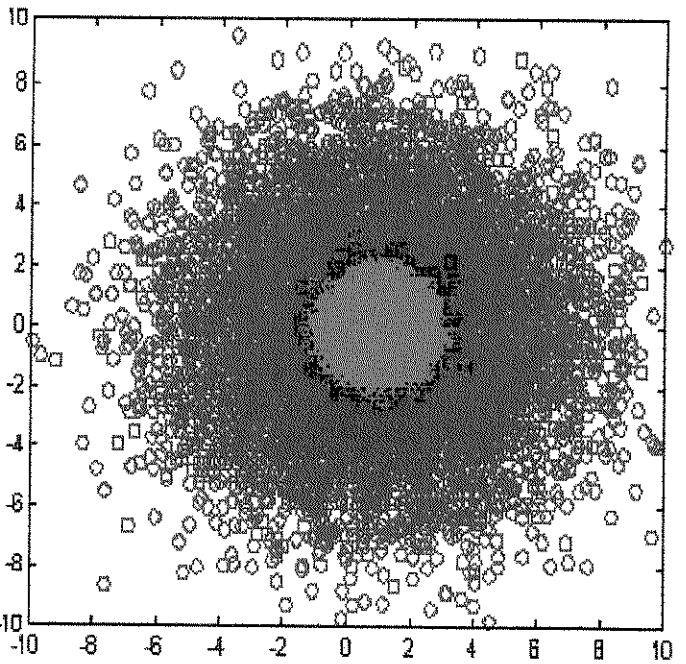

figura 8.1
Conjunto de dados do exemplo 8.1

A rede foi implementada inicialmente com 10 neurônios na camada intermediária. A figura 8.2 nos mostra as amostras classificadas corretamente em cada uma das classes As amostras representadas em vermelho são aquelas que pertencem a classe ω_2 e que foram classificadas corretamente. Já as amostras representadas em verde pertencem a classe ω_1 e também foram classificadas corretamente. O limiar entre as amostras representadas em verde e as amostras representadas em vermelhas representa a fronteira de decisão prática para este exemplo. Note também que a figura também mostra a fronteira de decisão teórica circular, dada pela expressão (8.2) derivada de (8.1) e que pode ser visualizada na figura 8.2 como a linha circular em azul.

$$y^2 + (x\text{-}1)^2 = 3.06 \tag{8.2}$$

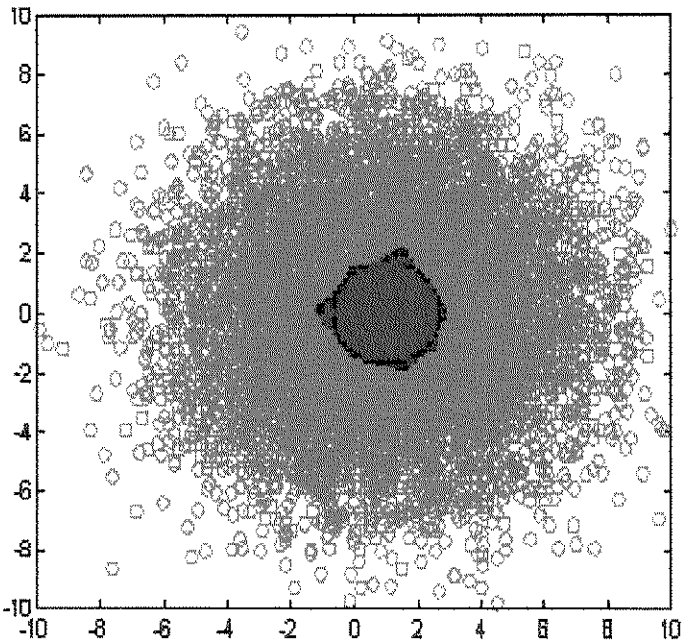

figura 8.2
Classificação e fronteira teórica do exemplo 8.1

Mostramos então na figura 8.3 os diversos hiperplanos obtidos experimentalmente e que contribuem para a formação da fronteira de decisão. Estes hiperplanos foram extraídos dos pesos sinápticos dos neurônios da primeira camada escondida da rede neural. Note que os hiperplanos determinam células de classificação e que como foi mostrado no capítulo 4,

é função da segunda camada de neurônios separar estas classes e consequentemente determinar a fronteira de decisão, que será então formada por parte dos hiperplanos encontrados a partir dos neurônios da camada escondida de neurônios.

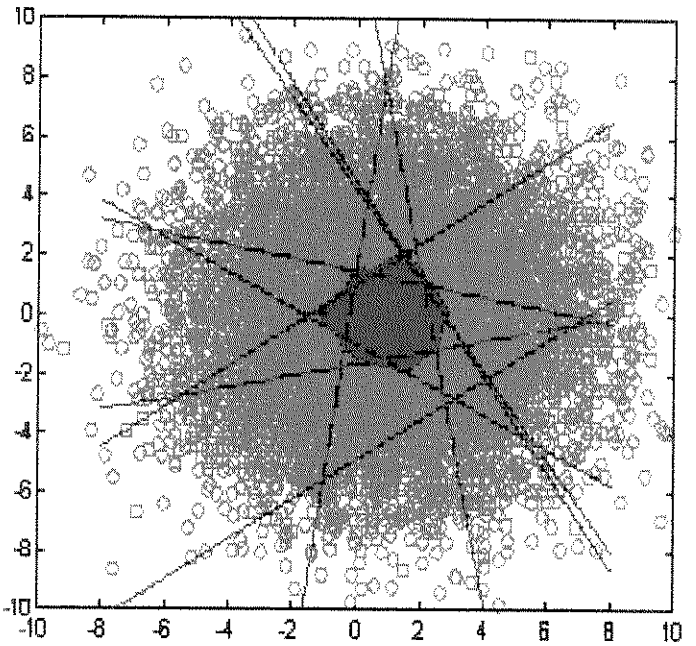

figura 8.3
Hiperplanos que formam a fronteira de decisão do exemplo 8.1

Na figura 8.4 temos uma visão em ZOOM da figura 8.3. Note que apesar dos hiperplanos não determinarem exatamente a fronteira de decisão, ele indicam uma boa aproximação para a mesma, e este fato provém das não-linearidades introduzidas nos neurônios da camada escondida. Entretanto, a influência da função de ativação será melhor discutida na seção 8.4.

Vemos também nas figuras 8.3 e 8.4 que a fronteira de decisão prática não é exatamente circular como esperávamos, mas isto provém do fato de que esta era uma rede experimental e com apenas 10 neurônios. No caso ideal, numa rede com infinitos neurônios, e infinitas amostras significativas para o treinamento teríamos a fronteira de decisão prática encontrada pela rede igual à fronteira de decisão ideal.

Existem alguns hiperplanos, como podemos ver nas figuras 8.3 e 8.4, que não contribuem para a formação da fronteira de decisão. Por exemplo, na figura 8.3 vemos

apenas 9 hiperplanos quando esperávamos 10, uma vez que tínhamos uma rede com 10 neurônios na camada escondida. Isto se deve ao fato de que no treinamento da rede alguns neurônios convergiram para uma região diferente daquela que se esperava. Desta forma, sugerimos desde já que seja feito um trabalho de prunning a fim de que no final do treinamento da rede tenhamos uma rede composta apenas pelos neurônios que contribuem para a classificação das amostras.

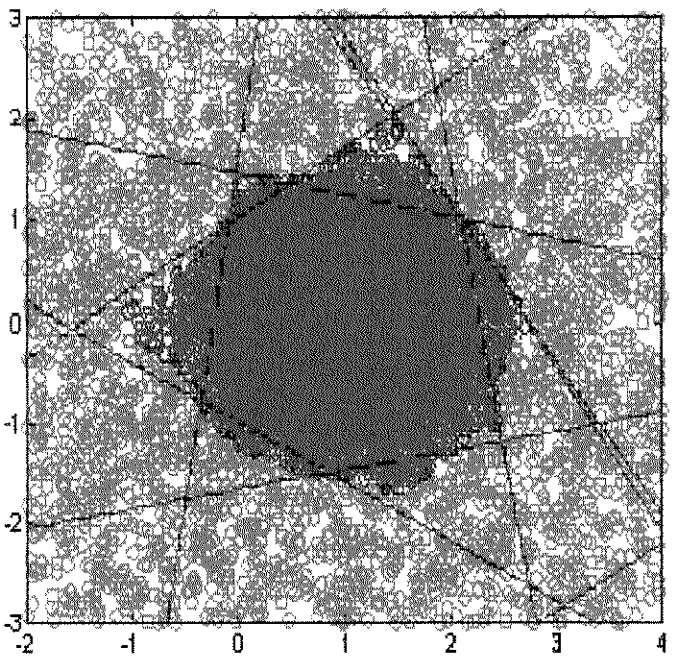

figura 8.4
Zoom da figura 8.3

8.3.2 Exemplo 8.2

Neste segundo exemplo, as classes ω_1 e ω_2 têm distribuição normal com os seguintes parâmetros:

$$\omega_1: \qquad \mu_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \Sigma_1 = \begin{bmatrix} 9 & 0 \\ 0 & 9 \end{bmatrix}$$

$$\omega_2: \qquad \mu_2 = \begin{bmatrix} -4 \\ 0 \end{bmatrix}; \Sigma_2 = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

Geramos mais uma vez um conjunto de 10.000 amostras de cada uma das classes. Sendo que deste total, 5.000 amostras de cada classe foram utilizadas para testar o classificador. Podemos ver na figura 8.5 o conjunto de dados gerados.

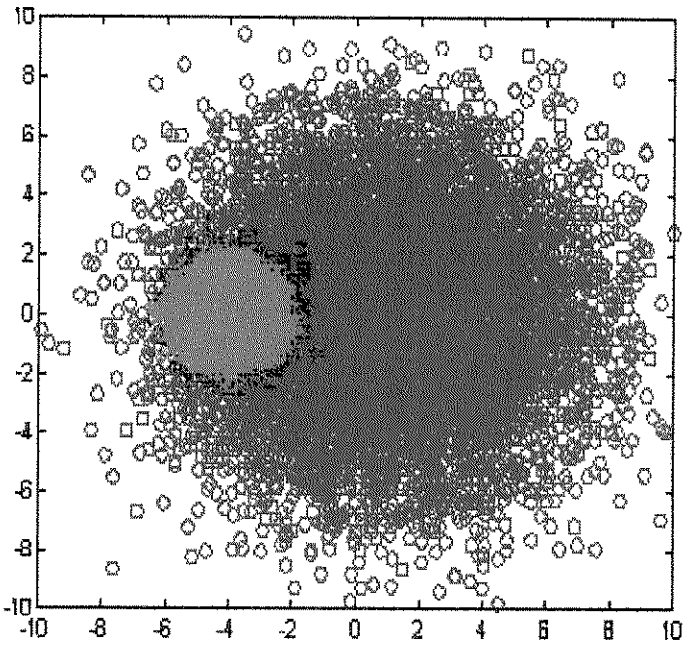

figura 8.5
Conjunto de dados do exemplo 8.2

A rede foi implementada com 15 neurônios na camada intermediária. Da mesma forma que no exemplo anterior, a figura 8.6 nos mostra as amostras classificadas corretamente em cada uma das classes. A figura também mostra a fronteira de decisão teórica circular, dada pela expressão (8.3).

Note porém, que mais uma vez a fronteira de decisão prática é apenas uma aproximação da fronteira teórica, e inclusive todas amostras que ficam do lado esquerdo da classe ω_2 não são classificadas corretamente. Ao verificarmos a figura 8.7, vemos que isto se deve ao fato de que a rede não conseguiu encontrar hiperplanos que formassem células nesta posição do espaço.

$$(y\text{-}0)^2 + (x + 4.29)^2 = 4.6174 \qquad (8.3)$$

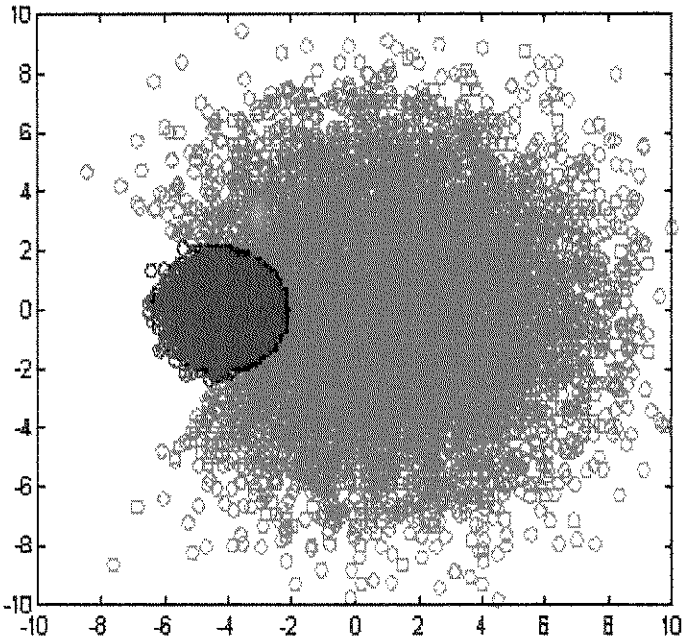

figura 8.6
Classificação e fronteira teórica do exemplo 8.2

Mostramos então na figura 8.7 os diversos hiperplanos que contribuem na formação da fronteira de decisão. Estes hiperplanos foram extraídos dos pesos sinápticos dos neurônios da primeira camada escondida da rede neural.

Na figura 8.8 mostramos a mesma figura 8.7, mas numa escala diferente, na qual podemos comprovar que a fronteira de decisão é formada através da combinação dos segmentos dos hiperplanos formados pela primeira camada de neurônios da rede neural.

Da mesma forma que no exemplo 8.1, em alguns pontos a fronteira de decisão ainda não é exatamente determinada pelos hiperplanos encontrados, mas ainda assim, eles representam uma boa aproximação da fronteira de decisão prática.

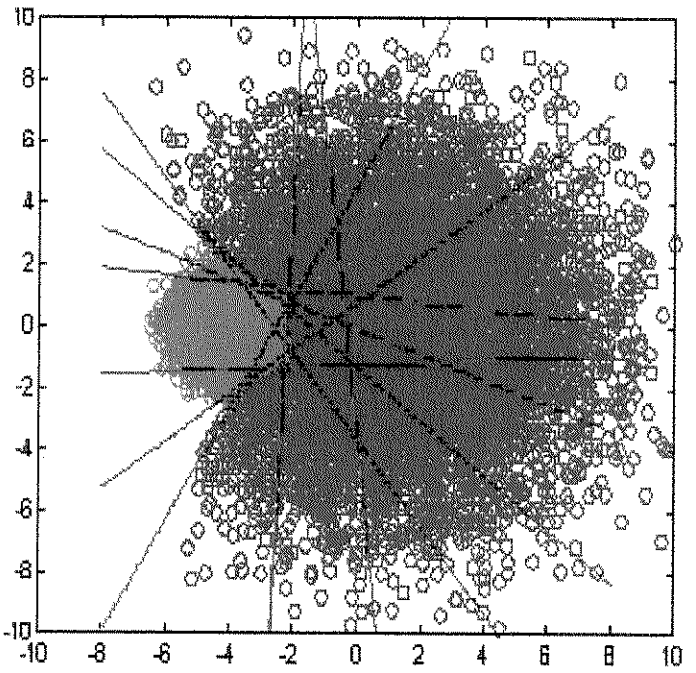

figura 8.7
Hiperplanos que formam a fronteira de decisão do exemplo 8.2

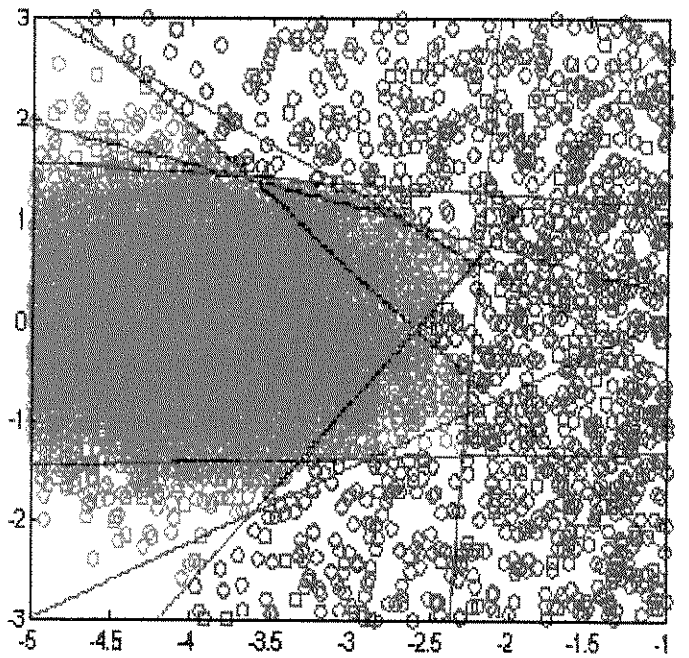

figura 8.8
Zoom da figura 8.7

8.3.3 Exemplo 8.3

A partir deste exemplo, mostraremos apenas a fronteira de decisão teórica e a classificação encontrada pela rede neural. onde mais uma vez a diferença de cores entre as amostras classificadas corretamente, indica a fronteira de decisão da rede.

Neste terceiro exemplo, as classes ω_1 e ω_2 têm distribuição normal com os seguintes parâmetros:

$$\omega_1: \qquad \mu_1 = \begin{bmatrix} -4 \\ 0 \end{bmatrix}; \Sigma_1 = \begin{bmatrix} 9 & 0 \\ 0 & 36 \end{bmatrix}$$

$$\omega_2: \qquad \mu_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \Sigma_2 = \begin{bmatrix} 0.5 & 0 \\ 0 & 2 \end{bmatrix}$$

Geramos mais uma vez um conjunto de 10.000 amostras de cada uma das classes. Sendo que deste total, 5.000 amostras de cada classe foram utilizadas para testar o classificador.

Na figura 8.9 vemos a classificação resultante da rede, e a fronteira de decisão teórica do problema, dada pela expressão (8.4) abaixo. Neste caso a fronteira de decisão descreve uma elipse.

$$\frac{y^2}{4} + \left(x - \frac{22}{17} \right)^2 = 4.6174 \qquad (8.4)$$

Mais uma vez a rede foi implementada com 15 neurônios na camada intermediária.

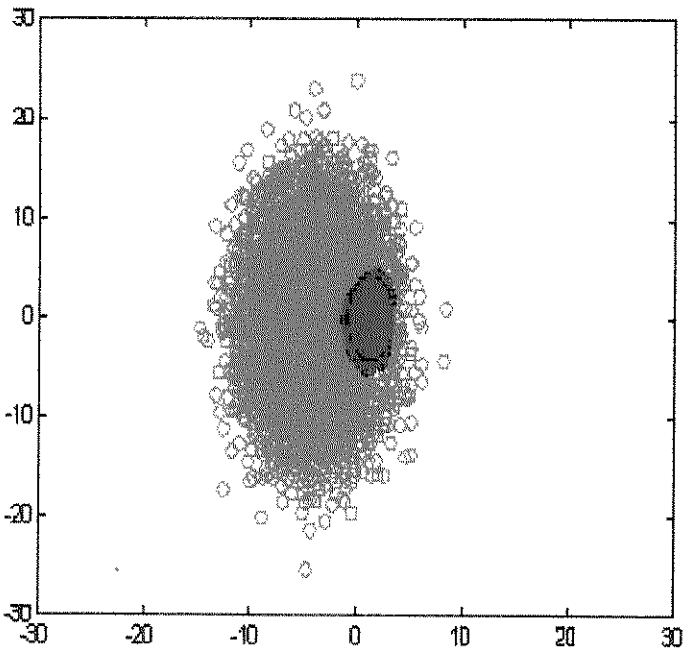

figura 8.9
classificação e fronteira teórica do exemplo 8.3

8.3.4 Exemplo 8.4

No quarto exemplo, as classes ω_1 e ω_2 têm distribuição normal com os seguintes parâmetros:

$$\omega_1: \qquad \mu_1 = \begin{bmatrix} -4 \\ 0 \end{bmatrix}; \Sigma_1 = \begin{bmatrix} 4 & 0 \\ 0 & 36 \end{bmatrix}$$

$$\omega_2: \qquad \mu_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \Sigma_2 = \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix}$$

Geramos mais uma vez um conjunto de 10.000 amostras de cada uma das classes. Sendo que deste total, 5.000 amostras de cada classe foram utilizadas para testar o classificador.

Na figura 8.10 vemos a classificação resultante da rede, e a fronteira de decisão teórica do problema, dada pela expressão (8.5) abaixo. Note que teoricamente a fronteira de decisão é uma parábola.

$$\frac{68}{36}y^2 = 10x + (4\ln 18 + 15) \tag{8.5}$$

Mais uma vez a rede foi implementada com 15 neurônios na camada intermediária.

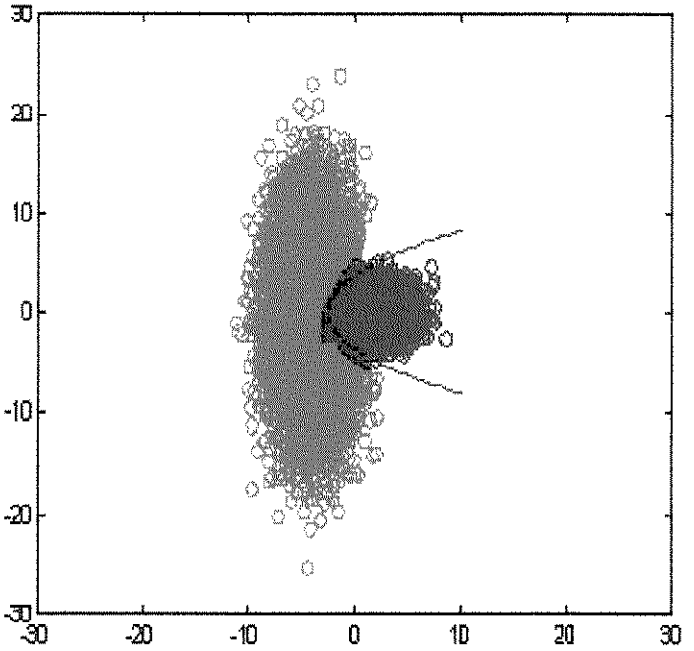

figura 8.10
classificação e fronteira teórica do exemplo 8.4

8.3.5 Exemplo 8.5

No quinto exemplo, as classes ω_1 e ω_2 têm distribuição normal com os seguintes parâmetros:

$$\omega_1: \qquad \mu_1 = \begin{bmatrix} -4 \\ 0 \end{bmatrix}; \Sigma_1 = \begin{bmatrix} 4 & 0 \\ 0 & 36 \end{bmatrix}$$

$$\omega_2: \qquad \mu_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \Sigma_2 = \begin{bmatrix} 36 & 0 \\ 0 & 4 \end{bmatrix}$$

Geramos mais uma vez um conjunto de 10.000 amostras de cada uma das classes. Sendo que deste total, 5.000 amostras de cada classe foram utilizadas para testar o classificador.

Na figura 8.11 vemos a classificação resultante da rede, e a fronteira de decisão teórica do problema, dada pela expressão (8.6) abaixo. Neste caso, temos uma fronteira de decisão hiperbólica.

$$y^2 = \left(x + \frac{35}{8}\right)^2 - \frac{9}{8} \tag{8.6}$$

Mais uma vez a rede foi implementada com 15 neurônios na camada intermediária.

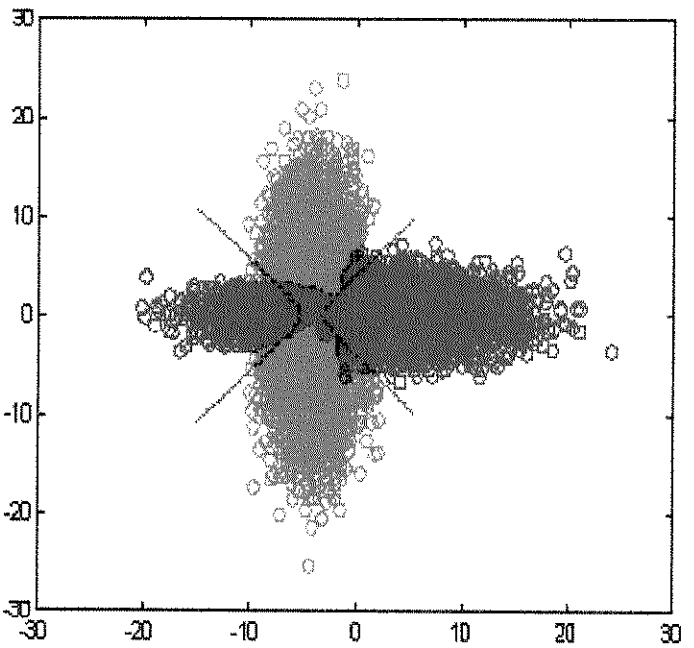

figura 8.11
classificação e fronteira teórica do exemplo 8.5

8.3.6 Exemplo 8.6

No sexto exemplo, as classes ω_1 e ω_2 têm distribuição normal com os seguintes parâmetros:

$$\omega_1: \quad \mu_1 = \begin{bmatrix} 0 \\ 4 \end{bmatrix}; \Sigma_1 = \begin{bmatrix} 4 & 0 \\ 0 & 36 \end{bmatrix}$$

$$\omega_2: \quad \mu_2 = \begin{bmatrix} 4 \\ 0 \end{bmatrix}; \Sigma_2 = \begin{bmatrix} 36 & 0 \\ 0 & 4 \end{bmatrix}$$

Geramos mais uma vez um conjunto de 10.000 amostras de cada uma das classes. Sendo que deste total, 5.000 amostras de cada classe foram utilizadas para testar o classificador.

Na figura 8.12 vemos a classificação resultante da rede, e a fronteira de decisão teórica do problema, dada pela expressão (8.7) abaixo. A fronteira de decisão é descrita como um par de retas.

$$\begin{aligned} y &= -1 - x \\ y &= x \end{aligned} \tag{8.7}$$

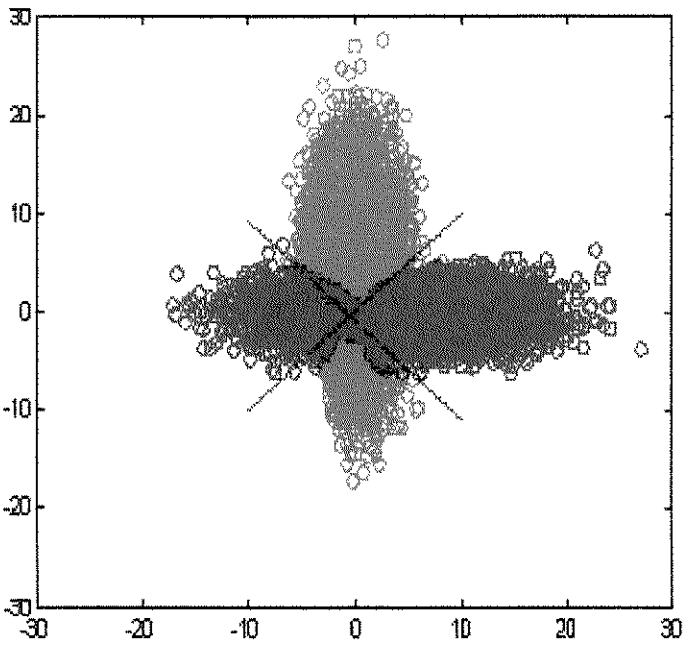

figura 8.12
classificação e fronteira teórica do exemplo 8.6

8.4 Influência da função de ativação na fronteira de decisão

Nesta seção investigamos a influência que a função de ativação terá para encontrarmos a fronteira de decisão. Como foi mostrado no capítulo 4, a fronteira de decisão será formada pelos pesos sinápticos dos neurônios da primeira camada escondida.

No estudo teórico, utilizamos uma função de ativação ideal e binária, como foi mostrado na expressão (4.3). Entretanto, o algoritmo de backpropagation garante convergência apenas para funções de ativação diferenciáveis. Daí, neste trabalho, utilizamos a função tangente hiperbólica como nossa função de ativação [5].

Nossa investigação sobre o efeito da função de ativação com relação à fronteira de decisão foi feita sobre a família de função tangente hiperbólica variando apenas a forma de transição da mesma. Isto é, variamos sempre o argumento da função, ou seja, o coeficiente *coef* da função *tanh(coef*x)*. A figura 8.13 mostra as várias funções de ativação utilizadas nesta seção.

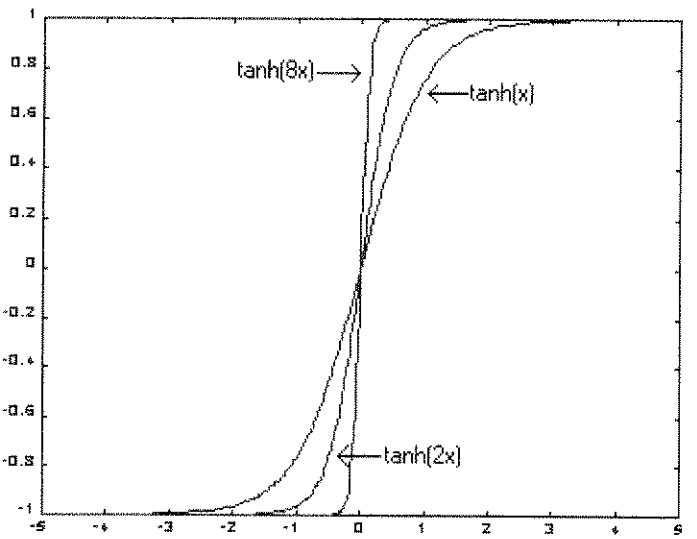

figura 8.13
Vários coeficientes para a função tangente hiperbólica

As figuras 8.14, 8.15 e 8.16 mostram os resultados de classificação incluindo os hiperplanos para a função de ativação tangente hiperbólica assumindo coeficientes igual a 1, 2 e 8 respectivamente.

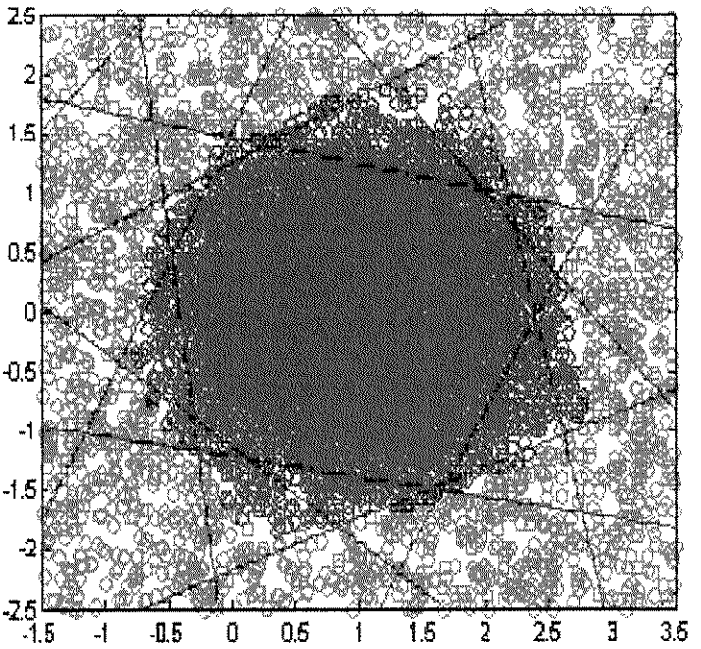

figura 8.14
Fronteira de Decisão para Função de Ativação *tanh(x)*

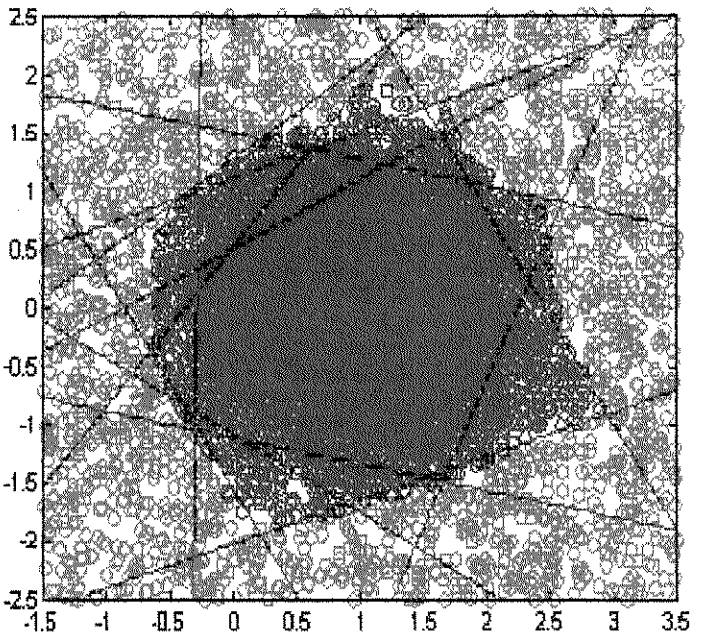

figura 8.15
Fronteira de Decisão para Função de Ativação *tanh(2x)*

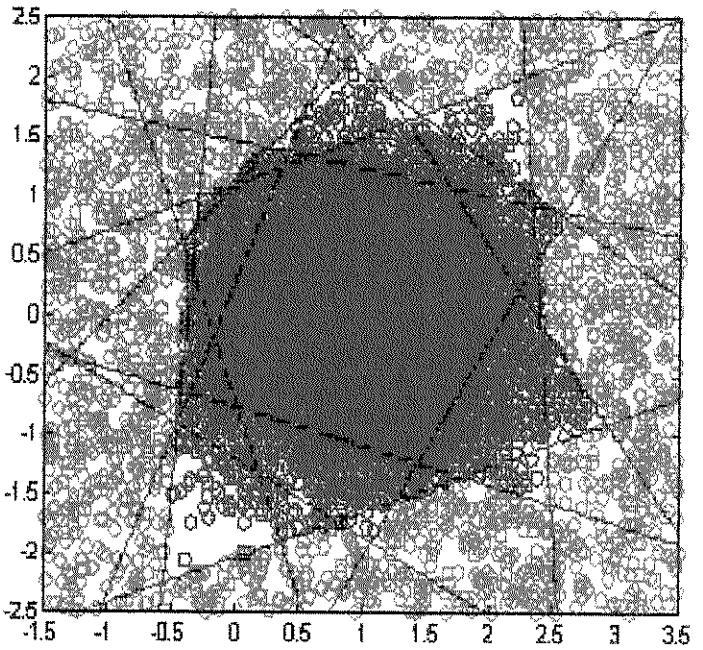

figura 8.16
Fronteira de Decisão para Função de Ativação *tanh(8x)*

Comparando estas figuras, notamos que quanto maior for o coeficiente, maior a aproximação entre a fronteira de decisão experimental encontrada pela rede neural se assemelha ao conjunto de hiperplanos gerados a partir dos neurônios da camada escondida.

8.5 Estudo experimental da extração de características utilizando dados sintéticos

Nesta seção, temos como objetivo comprovar a eficiência do algoritmo Lee/Cavalcanti de extração de características. Para cada um dos exemplos descritos abaixo implementamos uma rede neural com 10 neurônios na camada escondida.

Para cada exemplo, 10 conjuntos de dados foram gerados com uma distribuição gaussiana, sendo que para cada uma das classes foram geradas 1000 amostras, totalizando-se 2000 amostras, onde a metade foi utilizada para treinamento e a outra metade para testes.

8.5.1 Exemplo 8.7

As classes ω_1 e ω_2 têm distribuição normal com os seguintes parâmetros:

$$\omega_1: \quad \mu_1 = \begin{bmatrix} -2 \\ 0 \end{bmatrix}; \Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\omega_2: \quad \mu_2 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}; \Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Para efeito de ilustração, a figura 8.1 mostra um dos 10 conjunto de dados gerados.

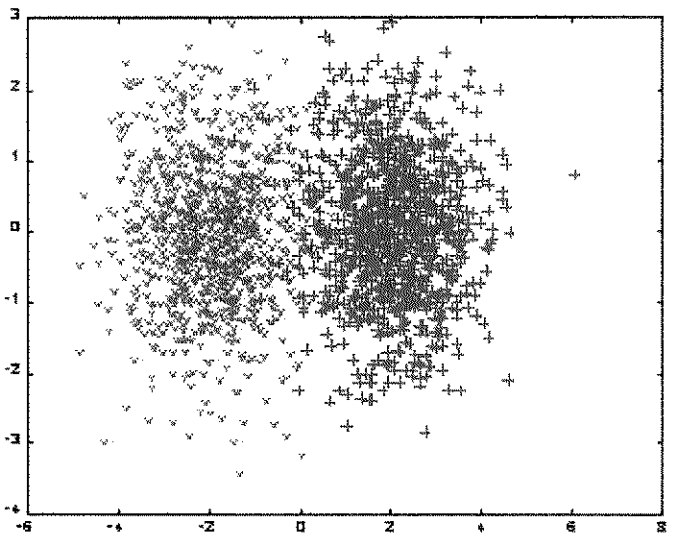

figura 8.17
Primeiro conjunto de dados gerados

Para cada um dos conjuntos foi treinada uma rede neural. Em seguida, para cada rede, foi feito o teste de classificação da rede, com amostras diferentes daquelas utilizadas para o treinamento. Assim, fizemos uma média aritmética da precisão de classificação para cada uma das redes, e para este exemplo, conseguimos uma precisão de classificação de 97.62%.

A partir dos neurônios da camada escondida construímos a matriz de característica da fronteira de decisão, Σ_{DBFM}, para cada um dos 10 conjuntos de dados. Assim, encontramos em média aritmética, autovetores e autovalores com os seguintes valores:

Autovalores:

λ_1=356.7534 λ_2=0.1992

Autovetores:

$$\phi_1 = \begin{bmatrix} -0.9996 \\ -0.0005 \end{bmatrix} \phi_2 = \begin{bmatrix} 0.0005 \\ -0.9996 \end{bmatrix}$$

Analisando os valores numéricos dos autovalores, observamos que λ_1 é da ordem de1500 vezes maior do que λ_2, logo podemos considerar λ_2 desprezível com relação a λ_1. Desta forma, podemos encontrar um novo conjunto de amostras ao projetar as amostras originais no subespaço formado pelo autovetor ϕ_1. Note que o novo conjunto de amostras tem dimensão 1, assim, este novo subespaço é uma reta na direção do vetor ϕ_1.

O conjunto original de amostras teve em média um total de 97.62% de classificação correta. Já o conjunto de amostras encontrado após a extração de características apresentou também em média 97.56% de classificação correta, o que representa 99.94% da precisão conseguida pelo conjunto original das amostras.

A tabela 8.1 apresenta os resultados obtidos realizando a classificação das amostras obtidas antes e após a extração de características. O campo dos **Autovalores**, mostra a média dos autovalores obtidos em todas as redes. Em parênteses, apresentamos a variância encontradas nestes dados. Note que o campo **Proporção de Autovalores** nos mostra a porcentagem de cada autovalor com relação a soma de todos os autovalores. E consequentemente o campo **Acúmulo de Autovalores** indica o acúmulo dos autovalores. A **Precisão de classificação** também é mostrada num campo exclusivo, e em parênteses aparece a variância encontrada com relação a todos os experimentos. Finalmente, o campo

Precisão de Classificação Normalizada, mostra a percentagem obtida com relação ao conjunto original de amostras.

Número de Características	Autovalores	Proporção de Autovalores (%)	Acumulo de Autovalores (%)	Precisão de classificação (%)	Precisão de Classificação Normalizada
1	356.7534(102)	99.94	99.94	97.56 (4.5)	99.94
2	0.1992(0.1)	0.06	100	97.62 (5.2)	100

tabela 8.1
Resultado do Exemplo 8.7

8.5.2 Exemplo 8.8

Neste exemplo, a classe ω_1 é normal com a seguinte distribuição estatística:

$$\mu_{11} = \begin{bmatrix} -8 \\ -8 \end{bmatrix} \quad \sum_{11} = \begin{bmatrix} 5 & 2 \\ 2 & 5 \end{bmatrix} \qquad \mu_{12} = \begin{bmatrix} 6 \\ 2 \end{bmatrix} \quad \sum_{12} = \begin{bmatrix} 7 & 2 \\ 2 & 7 \end{bmatrix}$$

e a classe ω_2 possui a seguinte distribuição:

$$\mu_{21} = \begin{bmatrix} -7 \\ 5 \end{bmatrix} \quad \sum_{21} = \begin{bmatrix} 4 & 2 \\ 2 & 4 \end{bmatrix} \qquad \mu_{22} = \begin{bmatrix} 3 \\ -12 \end{bmatrix} \quad \sum_{22} = \begin{bmatrix} 7 & 4 \\ 4 & 7 \end{bmatrix}$$

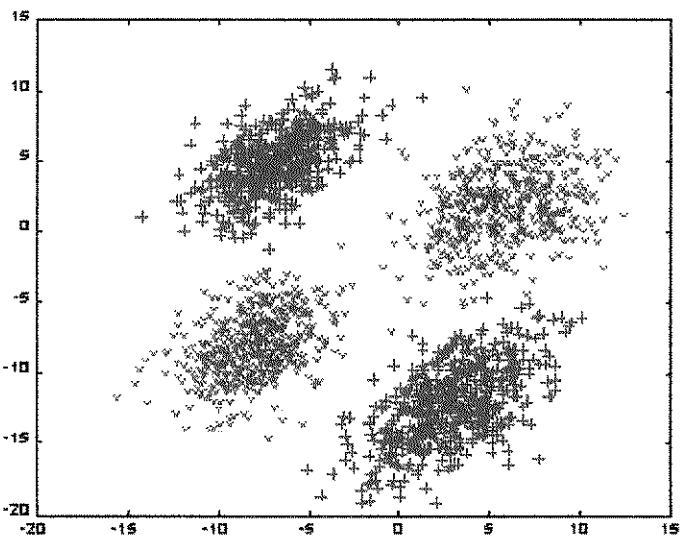

figura 8.18
Segundo conjunto de dados gerados

A figura 8.18 mostra o segundo conjunto de dados gerados. Note que neste exemplo utilizamos classes multi-modais, daí esperamos que as células de classificação geradas pela rede neural de treinamento seja não conexa. O objetivo então deste exemplo é exatamente o de mostrar que a extração de características pode ser realizada mesmo em problemas que utilizam regiões de classificação desconexas.

Segue os resultados obtidos, onde na tabela 8.2 encontramos um resumo dos resultados. Em média, estes conjuntos de autovetores e autovalores tiveram os seguintes valores:

Autovalores:

λ_1=119.0461 λ_2=279.9206

Autovetores:

$$\phi_1 = \begin{bmatrix} 0.7392 \\ 0.6659 \end{bmatrix} \quad \phi_2 = \begin{bmatrix} -0.6659 \\ 0.7392 \end{bmatrix}$$

Note que, neste caso, ambos os autovalores são da mesma ordem de grandeza, e isto mostra que teoricamente ambos os vetores de características são relevantes para classificação. Entretanto, a fim de verificarmos o comportamento do algoritmo num caso onde possamos reduzir a precisão de classificação, realizamos a extração de características apenas com o autovetor correspondente ao maior autovalor. No caso, utilizamos apenas ϕ_2 para a extração de características. Assim, podemos encontrar um novo conjunto de amostra projetado no subespaço formado pelo autovetor ϕ_2. Assim, o novo conjunto de amostras tem apenas uma dimensão.

O conjunto original de amostras classifica corretamente 99.53% das amostras. Após a extração de características a rede apresentou 78.85% de classificação correta, o que representa 79.22% da precisão conseguida pelo conjunto original das amostras. Note pela

tabela que com apenas uma característica o resultado é bem pior do que o obtido com o conjunto original de características.

A tabela 8.2 resume o resultado obtido amostras com a mesma dimensão original e com dimensão reduzida após a extração de características.

Número de Características	Autovalores	Proporção de Autovalores (%)	Acumulo de Autovalores (%)	Precisão de classificação (%)	Precisão de Classificação Normalizada
1	279.9206(17)	70.16	70.16	78.85(0.56)	79.22
2	119.0461(30)	29.84	100	99.53(0.29)	100

tabela 8.2
Resultado do Exemplo 8.8

8.5.3 Exemplo 8.9

Neste exemplo, temos dados tridimensionais. A classe ω_1 tem distribuição normal com os seguintes parâmetros:

$$\mu_1 = \begin{bmatrix} -6 \\ -4 \\ -8 \end{bmatrix} \quad \sum_1 = \begin{bmatrix} 9 & 4 & 5 \\ 4 & 8 & 4 \\ 5 & 4 & 9 \end{bmatrix}$$

a classe ω_2 também possui distribuição normal com parâmetros:

$$\mu_2 = \begin{bmatrix} 5 \\ 5 \\ 5 \end{bmatrix} \quad \sum_2 = \begin{bmatrix} 8 & 2 & 6 \\ 2 & 9 & 2 \\ 6 & 2 & 10 \end{bmatrix}$$

Mais uma vez, 10 conjunto de dados foram gerados com cada uma das classes contendo 1000 amostras. Assim, temos ao todo 2000 amostras, das quais 1000 foram

utilizadas para treinamento e 1000 para teste. E procedemos da mesma forma que nos exemplos 8.7 e 8.8. Daí, obtivemos os seguintes resultados.

Autovalor:

λ_1=255.8037 λ_2=1.1008 λ_3=0.5121

Autovetor:

$$\phi_1 = \begin{bmatrix} 0.4077 \\ 0.5819 \\ 0.7037 \end{bmatrix} \phi_2 = \begin{bmatrix} -0.8651 \\ 0.4927 \\ 0.0938 \end{bmatrix} \phi_3 = \begin{bmatrix} 0.2920 \\ 0.6470 \\ -0.7043 \end{bmatrix}$$

Neste caso, teremos três autovetores com autovalores correspondentes. Daí, podemos realizar a extração de característica para diminuirmos a dimensão das amostras para 1 ou 2. No caso, tomamos sempre os autovetores correspondentes às maiores magnitudes de autovalores. No quadro 8.3 abaixo veremos o resultado obtido utilizando uma e duas características para classificação.

Número de Características	**Autovalores**	**Proporção de Autovalores (%)**	**Acumulo de Autovalores (%)**	**Precisão de classificação (%)**	**Precisão de Classificação Normalizada**
1	255.8037	99.37	99.37	99.15(0.32)	99.96
2	1.1008	0.43	99.80	99.19(0.31)	99.99
3	0.5121	0.20	100	99.20 (0.19)	100

tabela 8.2
Resultado do Exemplo 8.9

8.6 Lee/Landgrebe X Lee/Cavalcanti

Nesta seção faremos uma breve comparação entre os dois métodos de extração de características mostrados neste trabalho.

O método Lee/Landgrebe possui uma vantagem com relação ao método Lee/Cavalcanti, na medida em que ele serve para qualquer tipo de rede neural MLP utilizando o algoritmo de backpropagation para treinamento. Afinal concebemos o algoritmo Lee/Cavalcanti para redes como as descritas na seção 4.6.

Também implementamos o método Lee/Landgrebe nas plataformas C e MATLAB, e encontramos as fronteiras de decisão para ambos os algoritmos. Descobrimos então que o custo computacional para a utilização do algoritmo Lee/Lendgrebe é algumas milhares de vezes maior do que o necessário para o algoritmo Lee/Cavalcanti.

Citaremos então algumas desvantagens do algoritmo Lee/Landgrebe.

- A rede neural deve ser treinada para cada uma das classes separadamente. Então serão realizadas vários treinamentos de rede para um caso de multi-classes.

- Para cada uma das redes deve ser encontrada uma Matriz de Característica da Fronteira de Decisão.

- Além de que para cada uma das amostras deveremos encontrar o ponto correspondente a esta amostra na fronteira de decisão, e o seu vetor normal.

Entretanto, na figura 8.19 utilizando uma rede utilizada para o algoritmo Lee/Cavalcanti e aplicamos o algoritmo Lee/Landgrebe, e encontramos que a fronteira de decisão do algoritmo Lee/Landgrebe é formada pelos hiperplanos encontrados pelo algoritmo Lee/Cavalcanti, o que esperávamos. Desta forma, as duas fronteiras de decisão são equivalentes e portanto devem gerar o mesmo resultado na extração de características, comprovando assim a eficiência do nosso algoritmo.

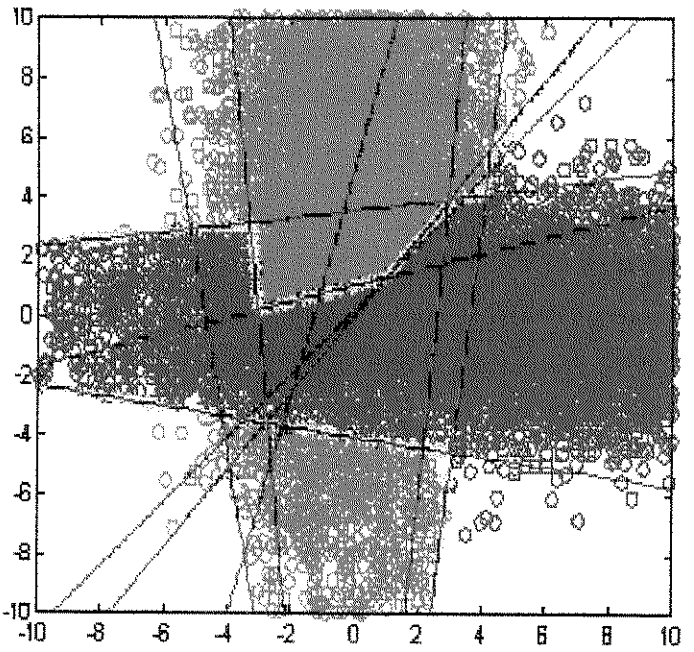

figura 8.19
Comparação entre os métodos de extração de Características Lee/Landgrebe X Lee/Cavalcanti

8.7 Resultado obtido com dados reais

Nesta seção, mostraremos os resultados obtidos para extração de características numa rede neural que classifica assinaturas estáticas. Os experimentos foram realizados utilizando uma rede neural igual a utilizada nos exemplos das seções anteriores.

Foi tomado um banco de dados utilizado por Lee/Lizarraga [13] com 550 assinaturas, sendo 110 de cada uma das pessoas a ter a sua assinatura reconhecida, sendo que para cada assinatura foram extraídas, 32 características discriminantes [13].

Implementamos então uma Rede Neural de classificação, onde para cada pessoa foi atribuída uma saída da rede, ou seja, associamos cada pessoa a uma classe. Separamos também metade das amostras para treinamento da rede e metade dos dados para testes.

Uma vez treinada, a rede conseguiu classificar 93.48% das amostras corretamente. Daí, encontramos a Matriz de Características da Fronteira de Decisão Σ_{DBFM}, onde encontramos 32 autovetores e autovalores correspondentes. Daí, tomando sempre os autovetores correspondentes aos maiores autovalores, realizamos a extração de características cujos resultados mostramos na tabela abaixo.

Número de Características	Autovalores	Proporção de Autovalores (%)	Acumulo de Autovalores (%)	Precisão de classificação (%)	Precisão de Classificação Normalizada
1	328.7452	34.9327	34.9327	27.27	29.12
2	259.9265	27.6199	62.5526	59.09	63.11
3	147.2716	15.6492	78.2018	76.36	81.55
4	96.8837	10.2949	88.4967	86.54	92.42
5	76.3852	8.1167	96.6134	90.00	96.12
6	24.5991	2.6139	99.2273	90.36	96.50
7	3.4758	0.3693	99.5966	90.36	96.50
8	1.9930	0.2118	99.8084	90.90	97.08
9	0.5434	0.0577	99.8662	90.72	96.89
10	0.3159	0.0336	99.8997	91.45	97.67
11	0.1723	0.0183	99.9181	91.81	98.06
12	0.1171	0.0124	99.9305	92.18	98.45
13	0.0844	0.0090	99.9395	92.18	98.45
14	0.0735	0.0078	99.9473	92.36	98.64
15	0.0694	0.0074	99.9547	92.72	99.03
16	0.0606	0.0064	99.9611	92.54	98.84
17	0.0531	0.0056	99.9667	92.72	99.03
18	0.0485	0.0052	99.9719	92.72	99.03
19	0.0391	0.0042	99.9761	92.90	99.23
20	0.0334	0.0036	99.9796	92.90	99.23
21	0.0317	0.0034	99.9830	93.09	99.42
22	0.0279	0.0030	99.9859	93.09	99.42
23	0.0250	0.0027	99.9886	93.27	99.62
24	0.0220	0.0023	99.9909	93.27	99.62
25	0.0177	0.0019	99.9928	92.90	99.23
26	0.0164	0.0017	99.9946	93.27	99.62
27	0.0135	0.0014	99.9960	93.45	99.81
28	0.0115	0.0012	99.9972	93.27	99.62
29	0.0100	0.0011	99.9983	93.45	99.81
30	0.0066	0.0007	99.9990	93.63	100
31	0.0060	0.0006	99.9996	93.27	99.62
32	0.0036	0.0004	100	93.63	100

tabela 8.3
Resultados da Extração de Características num caso real

Capítulo 9

Conclusões Gerais

9.1 Conclusões Gerais

Neste trabalho, mostramos como poderíamos extrair características a partir de uma Rede Neural treinada para classificação. Assim, iniciamos os nossos trabalhos mostrando que a extração de características baseia-se no fato de que podemos encontrar todas as características importantes a classificação a partir da fronteira de decisão.

Mostramos então que a partir de uma Rede Neural treinada, nós podemos estimar os vetores normais a fronteira de decisão. Com os vetores normais à fronteira de decisão, pudemos descrever a matriz característica da fronteira de decisão e com ela, um novo conjunto de dados poderia ser encontrado aplicando o algoritmo Lee/Cavalcanti de Extração de Características.

Também discutimos a respeito dos autovetores e autovalores correspondentes da Matriz de Característica da Fronteira de Decisão. Observamos então que os autovetores correspondentes a autovalores nulos, não contribuem no aumento da precisão de classificação do sistema.

O algoritmo proposto preserva as propriedades de uma rede neural para classificação e o seu emprego faz com que tenhamos uma diminuição do número de características sem com isto diminuir a precisão de classificação. Vale salientar que com a

diminuição do número de características a rede neural de classificação se torna mais simples e rápida.

Assim, podemos concluir que este algoritmo deve ser utilizado em aplicações que utilizem grande número de dados ou em aplicações onde haja uma grande repetição no uso da rede. Além disto, como a rede neural pode ser utilizada sem assumirmos qualquer conhecimento a respeito da probabilidade dos dados, a utilização do algoritmo pode servir também para mostrar algumas propriedades do problema de classificação de padrões.

Em resumo, concluímos que:

- Uma Rede Neural MLP pode ser utilizada como uma aproximação da Função Discriminante de Bayes.
- Podemos encontrar, a partir da Rede MLP treinada, a fronteira de decisão de um problema de classificação de padrões.
- algoritmo proposto encontra a quantidade mínima de características necessária para um problema de classificação de padrões.
- Com a diminuição do número de características podemos encontrar uma nova rede neural de classificação mais simples e rápida.

9.2 Sugestões para trabalhos futuros

A partir do trabalho que realizamos construímos algumas indagações que sugerimos para um trabalho futuro. Por exemplo, como deveríamos proceder a fim de encontrar a quantidade mínima de neurônios numa rede neural para classificação. Talvez dentro da mesma pergunta, como poderíamos proceder para encontrar aqueles neurônios que são relevantes para a classificação.

Quanto à questão da função de ativação, sugerimos uma pesquisa mais profunda no que diz respeito a qual função de ativação poderíamos utilizar e verificar a eficiência do algoritmo backpropagation Bayesiano proposto por Nedeljkovic em [17].

Em [6] Hérbert et al critica a atuação da rede neural MLP para classificação de padrões na medida em que normalmente uma rede deste tipo sempre vai encontrar uma saída correspondente a cada entrada, nunca rejeitando padrões. O que faz com que mesmo uma amostra sem qualquer sentido prático seja classificada segundo um dos padrões. Daí, segue a sugestão para que seja feita uma análise crítica em cima de todas as saídas da rede a fim de que possamos determinar uma probabilidade de uma determinada saída ser representativa ou não.

Referências

[1] **Bernardini, A. e De Fini, S.** *Optimal Decision Boundaries for M-QAM Signal Formats Using Neural Classifiers.* **IEEE Transactions on Neural Networks, Vol 9 No. 2, March 1998**

[2] **Duda, R. O. e Hart, P. E.** *Pattern Classification and Scene Analysis.* **New York: Wiley, 1973.**

[3] **FU, K. S.** *Syntactic pattern recognition and applications.* **Englewood Cliffs: Prentice Hall, 1982.**

[4] **Gallinari, P., Thiria, S., Badran, F. e Fogelman-Soulie, F.** *On the relations between discriminant analysis and multilayer perceptrons.* **Neural Networks, Vol 4 pp 349-360, 1991**

[5] **Haykin, S.** *Neural Networks – A Comprehensive Foundation.* **New Jersey: Prentice Hall, 1994**

[6] **Hérbet, J. F. e Parizeau, M.** *Cursive Character Detection using Incremental Learning.* **International Conference on Document Analysis and Recognition - Bangalore, India, 1999.**

[7] **Hertz, J., Krogh, A. e Palmer, R. G.** *Introduction to the Theory of Neural Computation.* **Redwood City: Wesley 1991**

[8] **Jain, A. K., e Mao, J.** *Artificial neural network for nonlinear projection of multivariate data.* **Proc. IEEE Int. Joint Conference Neural Networks, Baltimore, MD, pp 335-340, 1992**

[9] **Lee, C.** *Feature Extraction and Classification Algorithms for High-Dimensional data.* **Ph.D. dissertation, Purdue University, W. Lafayette, IN, 1992**

[10] **Lee, C. e Landgrebe, D. A.** *Feature Extraction Based on Decision Boundaries.* **IEEE Transaction on Pattern Analysis & Machine Intelligence Vol 15 No. 4, April 1993**

[11] **Lee, C. e Landgrebe, D. A.** *Decision Boundary Feature Extraction for Neural Networks.* **IEEE Transactions on Neural Networks, Vol 8 No. 1, January 1997**

[12] **Ling, L. L. e Leskow, L. A.** *Extração de Características Discriminantes Baseadas em Fronteiras de Decisão Ótima Obtidas Através do Aprendizado de uma Rede Neural.* **XV SBT, Recife, Brasil, 1997**

[13] **Lizárraga, M. G.** *Um Sistema Automático de Consulta e verificação de Assinaturas Estáticas.* **Tese de Mestrado, UNICAMP, Campinas, SP, 1996**

[14] **Makhoul, J., El-Jaroudi, A. e Schwarts, R.** *Formation of Disconnected Decision Regions with a Single Hidden Layer.* **BBN Laboratories, Cambridge**

[15] **Mao, J. e Jain, A. K.** *Artificial neural networks for feature extractions and multivariate data projection.* **IEEE Transactions on Neural Networks, Vol 6 pp 296-317, 1995**

[16] **Mao, J. e Jain, A. K.** *Discriminant analysis neural networks. Proc. IEEE Int. Joint Conference Neural Networks.* **San Fracisco, CA, MD, pp 300-305, 1993**

[17] **Nedeljkovic, V.** *A Novel Multilayer Neural Networks Training Algorithm that Minimizes the Probability of Classification Error.* **IEEE Transactions on Neural Networks, Vol 4 No. 4, July 1993**

[18] **Papoulis, A.** *Probability, Random Variables, and Stochastic Processes.* **New York: McGraw-Hill, 1991**

[19] Patrick, E.A. and F. P. F. II *Nonparametric feature Selection.* **IEEE Transaction on Information Theory, Vol IT-15, pp 577-584, 1969**

[20] Ruck, D.W., Rogers, S. K., Kabrisky, M., Oxley, M. E. e Suter, B. W. *The Multilayer Perceptron as an Approximation to a Bayes Optimal Discriminant Function.* **IEEE Transactions on Neural Networks, Vol 1 No. 4, December 1990**

[21] Schalkoff, R. *Pattern Recognition - Statistical, Structural and Neural Approaches.* **New York: Wiley, 1991.**

[22] Short, R. D. e Fukunaga, K. *Feature Extraction using problem localization* **IEEE Transaction on Pattern Analysis & Machine Intelligence Vol PAMI-4 pp 323-326, May 1982**

[23] Veloso, L. R. e Carvalho, J. M. *Neural versus Syntactic Recognition of Handwritten Numerals.* **International Conference on Document Analysis and Recognition - Bangalore, India, 1999.**

[24] Webb, A. R., Lowe, D. *The optimized internal representation of multilayer classifier networks performs nonlinear discriminant analysis.* **Neural Networks Vol 3, pp 367-375, 1990**