UNIVERSIDADE FEDERAL DO PAMPA

CIÊNCIA DA COMPUTAÇÃO

BRUNO DE SOUZA MIRANDA

ALGORITMOS CLÁSSICOS DE APRENDIZADO DE MÁQUINA APLICADOS AO PROBLEMA DO RECONHECIMENTO DE IMAGENS

Trabalho de Conclusão de Curso

Alegrete

2011

BRUNO DE SOUZA MIRANDA

ALGORITMOS CLÁSSICOS DE APRENDIZADO DE MÁQUINA APLICADOS AO PROBLEMA DO RECONHECIMENTO DE IMAGENS

Trabalho de Conclusão de Curso apresentado como parte das atividades para obtenção do título de bacharel em Ciência da Computação na Universidade Federal do Pampa.

Orientador: Prof. Dr. Cleo Zanella Billa

Alegrete

2011

BRUNO DE SOUZA MIRANDA

ALGORITMOS CLÁSSICOS DE APRENDIZADO DE MÁQUINA APLICADOS AO PROBLEMA DO RECONHECIMENTO DE IMAGENS

Trabalho de Conclusão de Curso apresentado como parte das atividades para obtenção do título de bacharel em Ciência da Computação na Universidade Federal do Pampa.

Trabalho apresentado e aprovado em: 30 de Junho de 2011.
Banca Examinadora:

Prof. Dr. Cleo Zanella Billa
Orientador
Ciência da Computação - UNIPAMPA

Prof. Dr. José Carlos Bins Filho
Ciência da Computação - UNIPAMPA

Prof. Msc. Alessando de Oliveira Bof
Ciência da Computação - UNIPAMPA

RESUMO

O interesse pelo desenvolvimento de aplicações envolvendo visão computacional pode ser observado pelo número crescente de sistemas que buscam explorar os benefícios agregados por este ramo da ciência da computação. Exemplo disso é a utilização deste tipo de sistema em equipamentos de uso cotidiano, onde muitas vezes passam de forma despercebida pelas pessoas, como em alguns modelos de câmeras digitais e automóveis.

Sistemas de visão computacional envolvem várias etapas em seu processo de execução, desde a aquisição da imagem em um formato digital até a tomada de decisão, onde busca-se, através da integração de técnicas de processamento de imagens e inteligência artificial, fornecer aos sistemas computacionais meios para que sejam capazes de interagir através da percepção visual com o ambiente a sua volta.

O foco deste trabalho está na etapa de classificação e reconhecimento de um sistema de visão computacional, onde o objetivo é classificar os diferentes elementos que compõem uma determinada cena com base em informações extraídas de sua representação digital. Aqui serão abordadas diferentes formas de realizar esta etapa, utilizando-se algumas das técnicas de inteligência artificial, mais especificamente aprendizado de máquinas, e analisando o comportamento de cada uma quando aplicadas ao conjunto de imagens selecionado.

Palavras-chave: visão computacional, aprendizado de máquinas, classificação.

ABSTRACT

The interest for the development of applications involving computer vision can be observed by the increasing number of systems that seek to explore the benefits joined by this branch of computer science. Example of that is the use of this type of system in equipments of daily use, which often pass in an unnoticed way for the people, as in some models of digital cameras and automobiles.

Computer vision systems involve several stages in its process, from image acquisition in a digital format to the decision-making, which looks for, through the integration of techniques of images processing and artificial intelligence, to supply the computer systems means so that they are capable to interact through the visual perception with the environment around them.

The focus of this work is in the classification and recognition stage of a computer vision system, where the objective is to classify the different elements that compose a certain scene based on extracted information from its digital representation. Here different forms to accomplish this stage will be approached, using some of the techniques of artificial intelligence, specifically machine learning, and analyzing the behavior of each when applied to the group of selected images.

Keywords: Computer vision, machine learning, classification.

Lista de figuras

Lista de tabelas

SUMÁRIO

1 INTRODUÇÃO

A Visão Computacional, fortemente consolidada em vários países, tem crescido de maneira significativa nos últimos anos no Brasil (WVC, 2008), já estando presente em vários ramos da indústria, como na segurança e automação, além de estar presente em equipamentos de uso cotidiano, como nos sistemas de auxílio a pilotagem presentes em alguns modelos de automóveis, ou em câmeras digitais, nos detectores de sorriso. (CONCI et al, 2008) define a visão computacional como sendo o domínio da ciência da computação que estuda e aplica métodos que permitem aos computadores "compreenderem" o conteúdo de uma imagem. Isto pode ser feito quando as informações capazes de caracterizar os elementos que compõem uma cena podem ser padronizadas e extraídas a partir sua representação digital.

Dentre as vantagens proporcionadas por este ramo da computação podemos citar a possibilidade de tornar sistemas capazes de interagir e tomar decisões a partir de elementos observados em uma determinada imagem ou ambiente. Empresas de grande porte, como Intel[1] e Microsoft[2], vêm realizando investimentos significativos para o desenvolvimento de produtos e serviços capazes de explorar os benefícios proporcionados pela utilização da visão computacional.

Devido à vasta gama de aplicações e a impossibilidade de definirmos como um elemento deve ser representado em uma imagem, seja um carro, uma cadeira ou uma pessoa, dificilmente poderemos construir um algoritmo determinístico para reconhecer e classificar objetos

---

1 Intel compra empresa canadense de detecção facial. Disponível em: <http://g1.globo.com/tecnologia/noticia/2010/11/intel-compra-empresa-canadense-de-deteccao-facial.html> Acesso em: 03 dez. 2010

Site da Intel: <http://www.intel.com> Acesso em: 05 jul. 2011

2 Microsoft distribui convites para teste do Kinect. Disponível em: <http://g1.globo.com/tecnologia/noticia/2010/08/microsoft-distribui-convites-para-teste-do-kinect.html> Acesso em: 03 dez. 2010

Site da Microsoft: <http://www.microsoft.com > Acesso em: 05 jul. 2011

representados em imagens, fazendo-se necessária a aplicação de técnicas de inteligência artificial que proporcionem ao algoritmo a capacidade de aprendizado. Estás técnicas realizam o seu aprendizado a partir de dados característicos referentes a cada objeto representado em uma cena, reconhecendo e classificando-os, para a partir de então tomar alguma decisão ou ser capaz de representá-lo de maneira adequada.

Muito se tem pesquisado nesta área para descobrir como nós humanos interpretamos uma determinada cena e como podemos extrair informações de nossa interpretação de modo que possamos associá-las na resolução de certos problemas (ROCHA, 2009). Neste contexto, surge a aprendizagem de máquina, que é um subárea da inteligência artificial dedicado ao desenvolvimento de algoritmos e técnicas que permitam ao computador aprender, isto é, que permitam ao computador aperfeiçoar seu desempenho em alguma tarefa.

O aprendizado de máquina é um aliado valioso no desenvolvimento de sistemas de visão computacional voltados ao reconhecimento e classificação de imagens, visto que além de aprender, as técnicas de aprendizado de máquinas são capazes de generalizar o conhecimento adquirido durante o processo de treinamento. Estas características se enquadram perfeitamente ao comportamento esperado de um sistema de visão computacional, onde é necessário aprender a reconhecer elementos de uma cena e ser capaz de generalizar, reconhecendo elementos semelhantes em outras situações. Ao projetar um sistema deste tipo, a escolha da técnica de aprendizado de máquina mais adequada pode ser um fator decisivo quanto a viabilidade ou não do projeto. Isso ocorre devido a vários fatores, tais como a complexidade do problema e se o sistema deverá ou não ser executado em tempo real.

Dentro deste contexto surge a proposta deste trabalho, que é analisar o desempenho de algoritmos clássicos de aprendizado de máquinas quando aplicados a um problema de visão computacional, buscando a partir de um conjunto de imagens segmentadas identificar um grupo de objetos de interesse. Afim de atingir este objetivo faz-se necessário abordar a etapa de extração de características de um sistema de visão computacional, pois seja qual for o propósito do sistema, além de definir a técnica de reconhecimento mais adequada ao problema em questão, é necessário estabelecer parâmetros capazes de quantificar as características dos elementos presentes em uma cena para que o processo de reconhecimento possa ser realizado computacionalmente. Entretanto, a definição e extração destes parâmetros de um conjunto de

exemplos não é uma tarefa simples que possa ser realizada de forma única e sistemática (ROCHA, 2009).

Várias informações podem ser extraídas de uma cena afim de caracterizar os elementos que a compõem, porém, ao definir quais são as informações mais relevantes para caracterizalos é necessário considerar aspectos como: se o objeto é o único a estar presente na cena, se existem elementos sobrepostos, se o objeto está em movimento ou não. (CONCI et al, 2008) subdivide os tipos de características que podem ser extraídas de imagens em dois grupos, aspecto e forma, sendo que no primeiro são consideradas informações como rugosidade, cor e textura, e no segundo informações sobre contorno, dimensões, inércia e topologia.

Este trabalho busca realizar um estudo comparativo entre três algoritmos clássicos de aprendizado de máquina aplicados ao problema de classificação de imagens, considerando como medida de desempenho a eficácia na classificação e reconhecimento dos objetos de interesse presentes em um conjunto de imagens. Como o foco deste trabalho não é a parte de segmentação, que por si só poderia ser tema de outro trabalho de conclusão, o conjunto de imagens utilizado foi o disponibilizado pelo Visual Object Classes Challenge 2010 (VOC2010)[3], onde é possível encontrar várias imagens segmentadas.

Os algoritmos selecionados para o caso em estudo foram o K-Vizinhos mais próximos ponderado (K-Nearest Neighbors – KNN), as Redes Neurais Artificiais (RNAs) e a Máquina de Vetores de Suporte (Support Vector Machine - SVM), sendo que a escolha destes algoritmos deve-se a três fatos distintos. No caso do KNN o objetivo é comparar o desempenho de uma das técnicas de aprendizado de máquinas mais simples quando comparada a outras técnicas mais sofisticadas, as RNAs devido a sua relevância no contexto histórico e finalmente o SVM pelo fato de esta técnica, de acordo com (MARSLAND, 2008), normalmente apresentar resultados melhores do que outros classificadores e ter se tornado um dos algoritmos de aprendizado de máquina mais populares desde o seu lançamento.

Os fundamentos teóricos necessários para o desenvolvimento deste trabalho são apresentados nos capítulos dois e três, onde foram abordados os conceitos necessários sobre visão computacional e aprendizado de máquina. Os capítulos quatro e cinco apresentam a metodo-

3 Visual Object Classes Challenge 2010 (VOC2010). Disponível em: <http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2010/> Acesso em: 03 dez. 2010

logia adotada e os resultados obtidos, concluindo com o capítulo seis, onde são apresentadas as considerações finais.

2 VISÃO COMPUTACIONAL

Sistemas de visão computacional podem ser agrupados em diferentes áreas, de acordo com sua finalidade e com o uso que fazem de técnicas de inteligência artificial, dados e imagens. Aplicações que visam gerar imagens a partir de dados coletados são consideradas aplicações de síntese de imagens, já quando os dados são informações extraídas de uma imagem, a área em questão é a análise de imagens. Estando relacionada com a análise de imagens, a visão computacional pode ser vista como a área da computação que estuda e aplica métodos com o propósito de tornar sistemas computacionais capazes de perceber visualmente o ambiente que o cerca (CONCI et al, 2008).

No decorrer deste capítulo serão apresentados os conceitos envolvidos em cada uma das etapas que podem estar presentes em um sistema de visão computacional, com maior ênfase na etapa de classificação e reconhecimento, visto que esse é o foco deste trabalho, onde são apresentados trabalhos relacionados com a aplicação de cada técnica apresentada como alternativa para a execução desta etapa.

2.1 Principais etapas de um sistema de Visão Computacional

Os sistemas de visão computacional envolvem análise de imagens e comumente, mas nem sempre, técnicas de inteligência artificial para auxiliar na tomada de decisão, permitindo a identificação e classificação de objetos. A análise de imagens consiste em encontrar parâmetros descritivos que representem informações importantes da imagem, pois se a informação visual puder ser padronizada, o objetivo da análise da imagem pode ser o reconhecimento ou classificação. Essa forma particular é chamada de reconhecimento de padrões, classificando

dados visuais numéricos ou simbólicos baseando-se em informações contidas em bancos de dados de padrões (Conci et al, 2008).

2.1.1 Aquisição de imagens

A primeira etapa em um sistema de visão computacional consiste na aquisição de imagens, convertendo uma cena real de sua forma original, tridimensional, para uma representação digital bidimensional. Uma imagem pode ser vista como a distribuição de energia luminosa em uma posição espacial, sendo formada basicamente por dois fatores, a intensidade da luz no ambiente e a reflexão da luz pelos objetos presentes na cena (CONCI et al, 2008).

Atualmente os dispositivos de captura de imagem mais utilizados são as câmeras CCD(Charge Couple Device), que consistem em uma matriz de células semicondutoras fotossensíveis que atuam como capacitores, armazenando carga elétrica proporcional à energia luminosa incidente (CONCI et al, 2008).

Para a aquisição de imagens utilizando CCD, é necessário um conjunto de prismas e filtros de cor encarregados de decompor a imagem colorida em suas componentes R (red/vermelho), G (green/verde) e B (blue/azul), cada qual capturada por um CCD independente. Posteriormente, os sinais elétricos correspondentes a cada componente de cor podem ser combinados conforme o padrão de cor desejado (NTSC, National Television Standards Committee, ou PAL, Phase Alterning Line) (CONCI et al, 2008). De posse da representação digital de uma cena, torna-se possível desenvolver algoritmos de processamento e análise de imagens, visão computacional e qualquer outra área que utilize imagens digitais como dados entrada.

2.1.2 Realce

Esta etapa tem como objetivo principal melhorar a qualidade das imagens através da ampliação de seu contraste, eliminação de padrões periódicos ou aleatórios (ruídos ou imperfeições das imagens originados no processo de aquisição, transmissão ou compressão), melhoria no seu foco e acentuação de características. Por exemplo, quando a imagem encontra-se danificada e é possível identificar a função matemática que represente a deficiência ou construir

um modelo matemático adequado, é possível buscar a função inversa e aplicá-la na imagem deteriorada para restaurá-la (CONCI et al, 2008).

2.1.3 Segmentação

A segmentação é uma etapa muito importante da visão computacional, pois é nesta etapa onde ocorre a identificação e o agrupamento dos *pixels (picture elements)* de interesse na imagem, onde os elementos de cada região identificada possuem o mesmo conteúdo no contexto de uma aplicação; e são destacados dos demais para que posteriormente seja realizada a extração de atributos e o cálculo de parâmetros descritivos (CONCI et al, 2008).

Existem diversas formas de segmentar uma imagem, podendo ser realizada através da análise das descontinuidades, similaridades, limites ou áreas. Quando o processo de segmentação da imagem é baseado em um conjunto de características ou atributos é geralmente associada à identificação de um grupo de pixels ou área da imagem. Uma região em uma imagem é um grupo de pixels conectados com propriedades similares. Nesse caso, todos os pixels semelhantes são considerados como pertencentes a uma mesma região, sendo agrupados em um conjunto e demarcados de alguma forma (com cores únicas, por exemplo) para indicar que pertencem a determinada região (CONCI et al, 2008).

2.1.4 Extração de atributos e características

Em um sistema de visão computacional muitas vezes é necessário reconhecer elementos ou objetos de interesse em imagens, fazendo-se necessária a definição de parâmetros quantificáveis para que isso possa ser realizado computacionalmente. Esses parâmetros dependem de valores como cor, posição, orientação, dimensões, textura e outros que caracterizem os objetos (CONCI et al, 2008).

Para realizar qualquer tipo de reconhecimento, por classificação, é necessário ter inicialmente uma descrição o mais detalhada possível das classes que se desejam reconhecer. A partir desta descrição serão projetadas formas possíveis de se extrair as informações a serem obtidas analisando-se as imagens. Em geral é necessário selecionar informações que sejam des-

correlacionadas e tenham propriedades invariantes à translação, escala e rotação (CONCI et al, 2008).

Entre as informações que podem ser extraídas de uma imagem afim de classificar os elementos que à compõem estão as bordas, que além de delimitar o perímetro dos objetos, fornecem informações para a extração de outras características, como por exemplo a circularidade, que é uma medida para estimar a compactação de um objeto. A detecção de bordas é uma etapa importante no processamento de imagens e, segundo (GONZALEZ e WOODS, 2000), é de longe a abordagem mais utilizada para realizar a detecção de descontinuidades significantes em imagens, descontinuidades que destacam-se pela presença de pixels que estão na fronteira entre duas regiões com intensidades luminosas relativamente distintas.

(GRAVES e BATCHELOR, 2003) e (GONZALEZ e WOODS, 2000) citam vários métodos que podem ser utilizados para detectar-se as bordas de uma imagem, entre eles, os métodos de Canny, Sobel, Roberts, Prewitt e o método de vizinhança de 8, sendo este último aplicável apenas a imagens binárias, onde a definição de borda é dada pela presença de ao menos um pixel com intensidade igual a zero ao redor de um pixel com intensidade igual a 1.

Existem muitos descritores que podem ser utilizados para caracterizar objetos, entre os mais simples, baseados em heurísticas, (SONKA et al, 1998) cita área, alongamento, circularidade e direção, mas destaca que apesar de estes descritores apresentarem bons resultados para objetos simples, eles não são os mais indicados para formas mais complexas.

Uma outra abordagem baseada em descontinuidade que pode ser utilizada para extrair informações sobre os objetos que compõem uma cena é a detecção de pontos que representem cantos ou quinas. Entre as técnicas de detecção de cantos disponíveis na literatura, é possível citar o algoritmo de Harris, que faz uso do conceito de pontos de interesse introduzido por (MORAVEC, 1980).

O detector de cantos de Moravec localiza os pontos de interesse de uma imagem subdividindo-a em pequenas partes, para as quais é calculada a variação média na intensidade da imagem causada pelo deslocamento desta região em várias direções. (HARRIS e STEPHENS, 1988) contribuíram com o aperfeiçoamento desta técnica em seu trabalho “A Combined Cor-

ner and Edge Detector", onde demonstram e propõem melhorias às limitações apresentadas pelo algoritmo de Moravec.

De acordo com (SANTOS,2007), o algoritmo de Harris é largamente utilizado por normalmente apresentar vantagens e relação a outras técnicas quando todas as características de um detector de cantos ideal são consideradas. Exemplos destes requisitos são a capacidade de detectar todos os cantos que existem na imagem, não apresentar falsos positivos, os pontos marcados como cantos devem estar o mais próximo possível dos cantos verdadeiros, entre outros. Uma vez que as informações a serem utilizadas para caracterizar os objetos de interesse estejam definidas, é possível passar a próxima etapa do sistema.

2.1.5 Classificação e reconhecimento

Classificação e reconhecimento em visão computacional podem ser considerados como a atribuição de um rótulo (classe) a um objeto presente em uma cena, baseando-se nas características extraídas da imagem. Em um sistema de reconhecimento de padrões normalmente existem duas fases envolvidas, o treinamento ou aprendizado e a classificação, sendo que para a realização da primeira é necessário um conjunto de imagens de exemplos para que o programa seja capaz de "ajustar-se" antes de partir para a etapa de classificação. Um sistema de reconhecimento deve ser capaz de determinar a que classe um objeto pertence, mesmo que ele não tenha sido apresentado ao sistema durante a etapa de aprendizado (CONCI et al, 2008).

O reconhecimento é uma tarefa trivial para os seres humanos, pois é realizado de forma paralela com a visão e levando em consideração todo o conhecimento adquirido anteriormente sobre os objetos presentes na cena, ao contrário dos sistemas computacionais, onde é necessário subdividir uma imagem digitalizada em grupos de *pixels* para então iniciar a realização desta tarefa. Existem várias técnicas para realizar a classificação de objetos, das mais simples, baseadas em vizinhos mais próximos por exemplo, às mais complexas, como redes neurais e máquinas de vetores de suporte (SVM - Support Vector Machine) (CONCI et al, 2008).

2.1.5.1 Classificação Supervisionada

A classificação supervisionada busca estabelecer uma função discriminante que seja capaz de separar as diferentes classes presentes em um espaço de características para um dado problema (CONCI et al, 2008). Para que isto seja possível faz-se necessária a utilização de algum método para ensinar o programa a classificar os diferentes vetores de características em classes distintas, baseando-se em um conjunto de exemplos com entradas e saídas esperadas dentro do contexto da aplicação (HAYKIN, 2001).

As etapas deste processo consistem em escolher um conjunto de treinamento coerente com o conjunto de classes possíveis, definir as características que melhor descrevam os objetos a serem reconhecidos, obter e ajustar a função discriminante e realizar testes com objetos fora do conjunto de treinamento afim de verificar o progresso no aprendizado da aplicação (CONCI et al, 2008).

2.1.5.1.1 Classificação por Distribuição Livre

Classificadores de distribuição livre são importantes por não utilizarem informações sobre as probabilidades de ocorrência de uma classe dentro de um conjunto de classes possíveis. Entre os classificadores por distribuição livre existentes podemos citar a classificação por função discriminante (Redes Neurais Artificiais e SVM por exemplo) e a classificação por vizinhança mais próxima, onde todas as experiências passadas são armazenadas em uma memória de exemplo de entrada e saída classificadas corretamente, possibilitando a decisão de classificação de um conjunto de características x contando apenas com as informações obtidas a partir dos exemplos de treinamento (CONCI et al, 2008) e (HAYKIN, 2001).

2.1.5.2 Classificação não supervisionada

A classificação não supervisionada é utilizada quando não dispomos previamente de informações sobre as classes (Conci et al, 2008). Nesse tipo de classificação são dadas as condições necessárias ao classificador para realizar a medição da ocorrência das características observadas em um determinado ambiente, a partir das quais tenta-se criar agrupamentos no es-

paço de características através de regularidades estatísticas dos dados de entrada, criando automaticamente novas classes (HAYKIN, 2001).

Além disso, (HAYKIN, 2001) cita como exemplo de algoritmo para realizar a classificação não supervisionada uma rede neural de duas camadas, onde a primeira recebe os dados de entrada e a segunda contém um grupo de neurônios que competem entre si pela oportunidade de responder aos “estímulos” presentes nos dados de entrada.

3 APRENDIZADO DE MÁQUINA

A medida que a inteligência atificial (IA) evolui torna-se possível desenvolver aplicações capazes de solucionar problemas cada vez mais complexos e que antes eram possíveis somente ao cérebro humano (MASSAD et al, 2004). A capacidade dos sistemas inteligentes para solucionar esta nova gama de problemas deriva de sua habilidade para armazenar e recuperar grandes quantidades de informações de forma eficiente, que são utilizadas para modificar e adaptar o comportamento do sistema, de acordo com a área para qual ele é empregado (REZENDE, 2002).

Dentro do contexto de inteligência artificial surge o aprendizado de máquina, que é o subcampo da IA responsável pelo desenvolvimento de técnicas computacionais sobre o aprendizado, capazes de tornar um sistema apto a adquirir e representar o conhecimento de forma automática. Apesar de o aprendizado de máquina ser uma ferramenta poderosa no desenvolvimento de sistemas, é importante observar que não existe um único algoritmo que apresente o melhor desempenho para todos os problemas, em vista disso faz-se necessária a aplicação de alguma metodologia para estimar o desempenho futuro de uma técnica aplicada a um determinado problema (REZENDE, 2002).

Uma forma de avaliar o desempenho de uma técnica de aprendizado de máquina aplicada a um determinado problema consiste no cálculo estatístico da taxa de acertos para cada técnica em questão. De acordo com (MARSLAND, 2008), é comum utilizar o método de amostragem para realizar a comparação entre dois ou mais algoritmos. Este método consiste em subdividir o conjunto de exemplos utilizados para o sistema realizar o aprendizado em dois grupos, um para treinamento e outro para a avaliação, sendo que as estimativas para o desempenho futuro da técnica são realizadas com base no segundo, cujo objetivo é representar o mundo real (REZENDE, 2002).

Este capítulo está organizado de forma a apresentar os algoritmos de aprendizado de máquinas selecionados para o desenvolvimento deste trabalho, passando por Redes Neurais Artificiais (RNAs), Máquinas de Vetores de Suporte (SVM) e K-Vizinhos Ponderado (KNN).

3.1 Redes Neurais Artificiais (RNAs)

3.1.1 O que é uma rede neural artificial?

Redes Neurais Artificiais podem ser vistas como a representação computacional de um modelo matemático inspirado na estrutura neural de organismos inteligentes, capazes de adquirir e armazenar conhecimento através da experiência.

Segundo (HAYKIN, 2001), uma rede neural é uma máquina projetada para modelar a maneira como o cérebro realiza uma tarefa ou função de interesse, podendo ser vista como um processador extremamente paralelizado e distribuído, constituído de unidades de processamento simples (neurônios), normalmente implementada utilizando-se componentes eletrônicos ou simulada por software em computador.

Uma característica importante sobre redes neurais artificiais é que o conhecimento adquirido é representado e armazenado através de alterações nas forças das conexões que unem as unidades de processamento básico (neurônios artificiais). Esta característica tem enormes implicações para o processamento e o aprendizado, pois a representação do conhecimento é modelada para que o conhecimento armazenado através de experiências anteriores influencie no curso de novos processamentos, tornando cada experiência passada parte do processo a partir do momento em que foi adquirida (CASTRO, 2006).

(CASTRO, 2006) e (HAYKIN, 2001) citam algumas das semelhanças entre as redes neurais artificiais e as redes neurais biológicas:

(i) O processamento básico ocorre nos neurônios.

(ii) Estes neurônios podem receber e enviar estímulos a partir de outros neurônios e a partir do ambiente.

(iii) Neurônios podem ser conectados uns aos outros, formando uma rede neural.

(iv) O conhecimento é adquirido pela rede a partir do ambiente através de um processo de aprendizado.

(v) O processo de aprendizagem ou treinamento é o responsável por adaptar a força das conexões de acordo com estímulos do ambiente.

(vi) Forças de conexão entre neurônios, conhecidas como pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido.

Devido a características como a capacidade de aprendizado, generalização, tolerância a ruídos e adaptabilidade, as redes neurais podem ser utilizadas para resolver problemas complexos, que são atualmente intratáveis. Na prática, de acordo com (HAYKIN, 2001), as redes neurais não podem fornecer uma solução trabalhando individualmente. Em vez disso, elas devem ser integradas à uma abordagem que vise decompor o problema em um número de tarefas relativamente simples, atribuindo-se as redes neurais um subconjunto de tarefas que coincidem com as suas capacidades.

Como exemplo de aplicação para redes neurais, (CARDOSO, 1994) cita o reconhecimento de áreas florestais desmatadas através da utilização de imagens de satélite. A interpretação da imagem feita pelo especialista, que identificou áreas de floresta, cerrado, desflorestamento, água, nuvem e sombra é “ensinada” à rede neural, que passa a reconhecer os padrões com o mesmo grau de acertos do especialista.

3.1.2 História

O primeiro modelo formal de rede neural artificial foi proposto em 1943 por W. McCulloch e W. Pitts em um artigo sobre a computação que poderia ser feita por um neurônio artificial de dois estados. Eles descreveram um modelo capaz de representar logicamente o comportamento de uma rede neural biológica. O neurônio de McCulloch e Pitts podia assumir somente os estados 0 e 1 em suas saídas, sendo 0 para não ativado e 1 para ativado (CASTRO, 2006).

Em 1949 Hebb's publica o livro *The Organization of Behavior*, onde é apresentada pela primeira vez as regras para a modificação dos pesos sinápticos das ligações entre os neurônios. Ele afirma que o aprendizado é alcançado pelo fortalecimento destas conexões, sempre que um determinado grupo de neurônios adjacentes está ativado, pois para Hebb's o fortalecimento ocorre incrementalmente pela ativação de um determinado grupo (CASTRO, 2006).

Aproximadamente 15 anos após a publicação do artigo de McCulloch e Pitts, uma nova abordagem foi introduzida por Rosenblatt (1958) em seu trabalho sobre o *perceptron*, um método inovador de aprendizagem supervisionada, capaz de aprender a classificar padrões linearmente separáveis, através da adição de pesos ajustáveis nas conexões entre os neurônios (HAYKIN,2001).

Pouco tempo após a contribuição de Rosenblatt, com *perceptron*, B. Widrow e seu estudante M. Hoff apresentam, no início dos anos 60, um neurônio articial similar ao neurônio apresentado por Rosenblat, chamado ADALINE (ADAptive LInear NEuron). Embora ambos sejam capazes de classificar apenas padrões linearmente separáveis, o ADALINE traz como novidade a regra de aprendizado Widrow-Hoff, ou regra do delta, devido a isso o seu trabalho ganhou relevância na comunidade científica (CASTRO, 2006).

No final dos anos 60, M. MinsKy e S. Papert (1969) publicaram um livro chamado *Perceptrons*, no qual demonstravam matematicamente as limitações de uma rede neural de uma única camada, ou seja, a incapacidade de classificar padrões não separáveis linearmente. Em uma breve sessão sobre o perceptron de múltiplas camadas, eles afirmam que não havia razão para supor que estas limitações poderiam ser superadas com a utilização de redes de múltiplas camadas (HAYKIN,2001). Isto causou um grande impacto no interesse por redes neurais durante a década de 70. Widrow e Rosenblatt, cientes das limitações dos modelos existentes, propuseram a adição de mais camadas na rede neural como forma de aumentar a capacidade de representação da rede. No entanto, eles não foram capazes de generalizar os seus algoritmos para treinar redes de múltiplas camadas (CASTRO, 2006).

Em 1982, J. Hopfield publicou um artigo onde utilizou a ideia de uma função de energia para formular um novo modo de entender a computação executada por redes recorrentes com conexões simétricas. Foi neste mesmo artigo em que pela primeira vez o princípio do armaze-

namento de informações em redes dinamicamente estáveis foi explicado. Um outro desenvolvimento importante ocorrido em 1982 foi a publicação do artigo de Kohonem, onde ele descreve uma rede neural artificial baseada em auto-organização. Esta rede é caracterizada pela auto-organização dos neurônios, que são posicionados de acordo com características estatísticas contidas nos padrões de entrada, formando um mapa dos padrões (HAYKIN,2001).

Cerca de dezessete anos após a publicação do livro *Perceptrons* (1969), Rumelhart e seus colaboradores publicam um livro intitulado *Parallel Distributed Processing* (1986), no qual demonstram que as afirmações de MinsKy e Papert a respeito das redes neurais de múltiplas camadas não estavam corretamente embasadas. Neste mesmo livro eles publicam o algoritmo *back-propagation*, o qual é capaz de treinar redes neurais de múltiplas camadas através da retropropagação do erro. Este livro também foi crucial para a retomada no interesse por pesquisas nesta área (CASTRO, 2006).

3.1.3 Neurônio de McCulloch e Pitts

A era moderna das redes neurais começou com o trabalho pioneiro de McCulloch e Pitts (HAYKIN,2001). De acordo com (CASTRO, 2006), embora este primeiro modelo de neurônio seja simples, ele já apresenta algumas características importantes em relação a modelos mais recentes, ou seja, a integração dos estímulos de entrada com a rede e a presença da função de ativação.

A descrição matemática a seguir foi retirada de (MARSLAND, 2008).

A figura 1 é uma representação do perceptron e será utilizada como base para realizar a descrição matemática deste modelo de neurônio artificial.

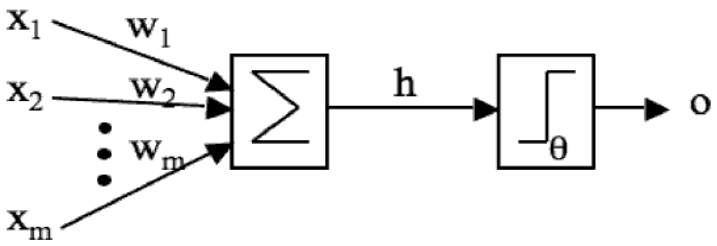


FIGURA 1 - Modelo matemático do neurônio de McCulloch e Pitt. As entradas x_i são multiplicadas pelos pesos w_i para então realizar a soma destes valores. Se a soma for maior do que 0 o neurônio é ativado, caso contrário, o neurônio permanece inativo (MARSLAND, 2008).

A esquerda da figura estão os neurônios de entrada, identificados por x_1 , x_2 ... x_m , com seus respectivos valores. Em um primeiro momento, nós devemos realizar a multiplicação das entradas x_i pelos pesos sinápticos w_i do neurônio, ao final deste processo nós teremos um valor *h* resultante. Isto pode ser escrito da seguinte forma:

$$h = \sum_{i=1}^{m} w_i x_i$$

EQUAÇÃO 1 - Somatório das entradas *X* multiplicadas pelos pesos sinápticos *W*. Onde *m* é o número de entradas e h o valor resultante.

A segunda metade do trabalho do neurônio é decidir se ele deve ou não ser ativado. Como o neurônio de McCulloch e Pitts é um neurônio de valores binários, produzindo somente zeros e uns em sua saída, nós podemos escrever a sua função de ativação da seguinte forma:

$$a = g(h) = \begin{cases} 1 \; se \; h > 0 \\ 0 \; se \; h \leq 0 \end{cases}$$

EQUAÇÃO 2 - Função de ativação do neurônio de McCulloch e Pitts. Também conhecida como função degrau.

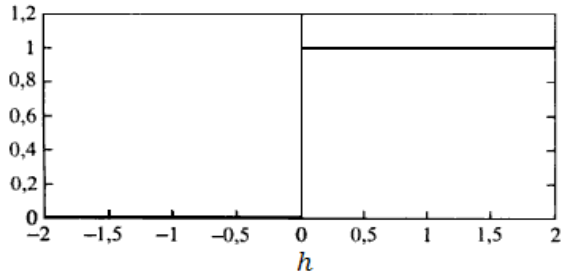
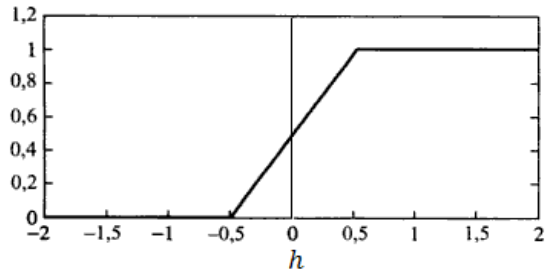
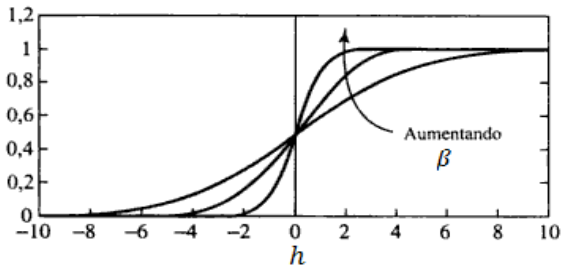
Uma questão que merece ser considerada é: O quão realista é este modelo de neurônio? A resposta, segundo (MARSLAND, 2008), é não muito, pois as entradas para um neurônio real não se resumem necessariamente a valores lineares e um neurônio artificial ainda deve ser capaz de produzir mais saídas entre os estados *ativado* e *não ativado*, o que pode ser obtido através de modificações na função de ativação.

3.1.4 Funções de Ativação

Uma função de ativação *g(h)*, segundo (CASTRO, 2006), determina a saída de um neurônio em relação a sua entrada. Elas podem assumir várias formas, a mais comum é a sigmoidal, graças ao seu balanço entre o comportamento das funções degrau e linear por partes, demonstradas na tabela 1.

TABELA 1

Comportamento das funções de ativação degrau, linear e sigmoidal.

Funções de ativação	
	$$a = g(h) = \begin{cases} 1 \; se \; h > 0 \\ 0 \; se \; h \leq 0 \end{cases}$$ EQUAÇÃO 2 - Função de ativação do neurônio de McCulloch e Pitts. Também conhecida como função degrau.
	$$a = g(h) = \begin{cases} 1 \;\; se \;\; h \geq \frac{1}{2} \\ h \;\; se \;\; \frac{1}{2} > h > -\frac{1}{2} \\ 0 \;\; se \;\; h \leq -\frac{1}{2} \end{cases}$$ EQUAÇÃO 3 - Função linear por partes.
	$$a = g(h) = \frac{1}{1 + e^{-\beta h}}$$ EQUAÇÃO 4 - Função de ativação sigmoidal, onde β é um inteiro positivo e h é a saída do processamento do neurônio.

Alterando-se o valor de β na equação quatro é possível tornar a curva da função mais ou menos suave, aproximando o seu comportamento ao das funções degrau e linear por partes (HAYKIN,2001).

3.1.5 Perceptron

De acordo com (CASTRO, 2006), o perceptron é a forma mais simples de uma rede neural, usada para classificar padrões linearmente separáveis (figura 2), como por exemplo os que se encontram em lados opostos de um hiperplano. Rosenblatt provou, em 1958, que se os padrões utilizados para treinar o perceptron são retirados de duas classes linearmente separáveis,

então o algoritmo converge e posiciona a superfície de decisão na forma de um hiperplano entre as duas classes. A prova da convergência do algoritmo é conhecida como o *teorema da convergência do perceptron* (HAYKIN,2001).

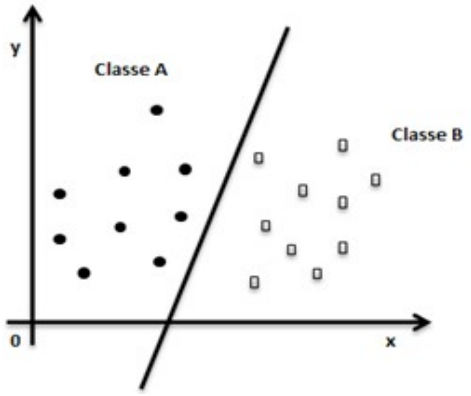


FIGURA 2 - Classes linearmente separáveis.

O número de entradas e o número de neurônios na camada de saída pode variar de acordo com o problema para o qual a rede está sendo modelada. A camada de saída, em um problema de classificação, pode conter um neurônio para cada padrão de entrada que deva ser reconhecido.

Basicamente, o perceptron consiste de uma única camada de neurônios de McCulloch e Pitts com pesos sinápticos ajustáveis e um *Bias* (CASTRO, 2006). Observando as equações 1 e 2, podemos verificar que se todos os valores de entrada forem iguais a zero o neurônio não será ativado sejam quais forem os pesos sinápticos, visto que o resultado da primeira equação será igual a zero. Para resolver este problema é necessário adicionar uma entrada extra com valor diferente de zero (normalmente são utilizados os valores -1 ou 1), esta entrada é chamada de *Bias* (MARSLAND, 2008).

3.1.6 Perceptron Multicamadas (Multi-Layer Perceptron - MLP)

O perceptron de múltiplas camadas (MLP) é uma generalização do perceptron de camada única, sendo constituído de um conjunto de unidades sensoriais (nós fonte) que formam a camada de entrada, uma ou mais camadas ocultas de nós computacionais e uma camada de saída

de nós computacionais. Onde o sinal se propaga para a frente, camada por camada (CASTRO, 2006).

Além disso, (CASTRO, 2006) cita que o MLP tem sido aplicado com sucesso em um grande número de problemas, sendo treinado com o algoritmo de aprendizado supervisionado conhecido como algoritmo de retropropagação do erro ou *back-propagation*.

Normalmente não é necessário utilizar mais do que três camadas, duas ocultas e uma de saída, ao modelar o MLP para realizar o reconhecimento de um determinado conjunto de padrões. Isso ocorre porque ao adicionarmos novas camadas ao MLP estamos combinando linearmente as funções de ativação dos neurônios em cada uma delas, pegando como exemplo a função sigmoidal (equação 3 da tabela 1), podemos verificar o resultado destas combinações nas figuras 3 e 4, retiradas de (MARSLAND, 2008).

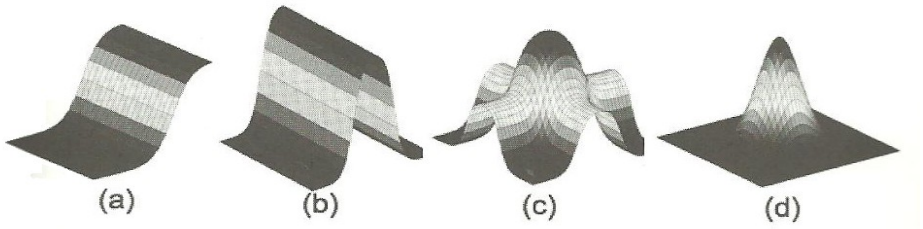


FIGURA 3 - O aprendizado do MLP pode ser visto em (a) como a saída de uma única função sigmoidal, que pode servir como entrada para outra camada, gerando a curva (b). Adicionando outra curva de 90º nós produzimos (c) uma crista, que pode ser aperfeiçoado (d) para a forma que queremos (MARSLAND, 2008).

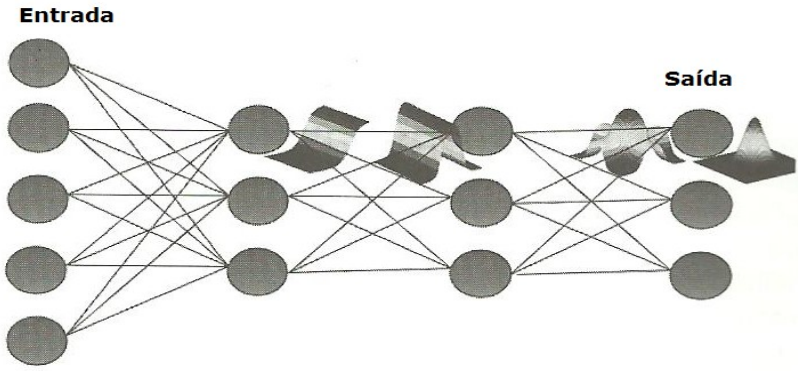


FIGURA 4 - Esquema da forma de aprendizado em cada estágio do MPL (MARSLAND, 2008).

A ideia básica por trás da adição de novas camadas ao MLP é que para cada nova camada inserida ocorre a combinação da função de ativação, neste caso a sigmoidal, resultando em funções de cristas, que se forem combinadas novamente podem gerar funções com um único ponto de máximo. Com isso são obtidas respostas localizadas e funções que podem ser muito arbitrárias, causando a perda da capacidade de generalização da rede (MARSLAND, 2008).

3.1.7 Treinamento

O primeiro passo para tornar uma rede neural apta a resolver algum problema é "ensina-la" a reconhecer as características pertencentes a cada padrão dentro um grupo pré-determinado de classes. Existem algumas técnicas para realizarmos esta tarefa, nesse trabalho será apresentado o aprendizado supervisionado, pois o objetivo é fazer com que a rede aprenda a reconhecer e classificar os padrões característicos dos elementos presentes em um grupo de exemplos pré-definidos.

O aprendizado supervisionado consiste em apresentarmos a rede neural repetidas vezes um grupo de exemplos de entrada com suas respectivas saídas esperadas, até que uma taxa mínima pré-determinada de acertos seja atingida. A medida que o processo de treinamento ocorre, os pesos sinápticos das conexões entre os neurônios são alterados gradativamente de acordo com os parâmetros utilizados, até que a rede apresente um comportamento satisfatório.

3.1.7.1 Parâmetros de aprendizado de uma RNA

A fim de obtermos um melhor ajuste da rede neural artificial em relação a um determinado problema, existem alguns parâmetro que podem ser modificados. A alteração destes parâmetros podem levar a rede a tornar-se demasiadamente especializada, perdendo a capacidade de generalização, ou torná-la instável, fazendo com que não consiga chegar a soluções satisfatórias. Não existem valores ótimos que possam ser aplicados a qualquer problema, fazendo-se necessária a aplicação de diferentes valores para cada parâmetro, até encontrarmos os que resultam em um melhor desempenho da rede para o conjunto de dados disponível para a realização do treinamento.

3.1.7.1.1 Taxa de aprendizado

A taxa de aprendizado controla o quanto os pesos sinápticos *w* podem ser alterados em cada repetição do processo de treinamento. Este valor deve variar entre 0 e 1, sendo que se utilizarmos valores muito altos, próximos a 1, a rede corre o risco de tornar-se instável, não convergindo para um ajuste adequado dos pesos. Ao utilizarmos valores muito pequenos, próximos a zero, o treinamento torna-se muito custoso, tipicamente são utilizados valores entre 0,1 e 0,4 (MARSLAND, 2008).

3.1.7.1.2 Momentum

O *momentum* é um valor, entre zero e um, que multiplica a variação dos pesos das conexões sinápticas nos tempos t e t - 1, tendo por objetivo melhorar o desempenho da RNA durante o processo de treinamento. A utilização desta técnica faz com que a RNA tenha a capacidade de superar mínimos locais e torna possível a utilização de valores pequenos para a taxa de aprendizado, permitindo um ajuste fino da rede (MARSLAND, 2008).

3.1.7.1.3 Quando devemos encerrar o treinamento?

O momento de encerrar o treinamento de rede neural artificial é uma decisão que afeta diretamente o seu desempenho. Esta decisão impacta no resultado final porque se o treinamento for encerrado muito tarde a rede neural pode tornar-se muito especializada, perdendo a capacidade de generalização, que de acordo com (CASTRO, 2006) é a capacidade que a rede possui para conhecer padrões que não estavam presentes durante o processo de treinamento. A perda de generalização devido ao excesso de treinamento também é conhecida como *overfitting*. Outra possibilidade é a de que o treinamento seja interrompido muito cedo, impedindo que a rede neural seja capaz de ajustar os pesos sinápticos da forma correta.

É possível observar a variação da média da soma quadrática dos erros durante o treinamento, tornando possível verificar o momento em que ocorre uma desaceleração desta variação, o que indica o local exato de um mínimo de erros para os exemplos apresentados. O objetivo não é interromper o treinamento antes de um mínimo ser encontrado, mas se o treina-

mento for muito longo a rede torna-se mais especializada do que o esperado (MARSLAND, 2008).

Para sermos capazes de detectar o momento em que a rede começa a perder a capacidade de generalização é necessário realizar o processo de validação periodicamente durante a execução do treinamento. Este processo consiste em apresentar a rede neural um grupo de entradas diferentes dos exemplos utilizados para o treinamento e cujas saídas também sejam conhecidas, tornando possível realizar o acompanhamento da variação da taxa de acertos para este segundo grupo de exemplos, e interrompendo o treinamento assim que esta taxa começa a diminuir. Além de sugerir a utilização deste método, também conhecido como *early stopping*, (MARSLAND, 2008) recomenda dividir os exemplos disponíveis para o treinamento em três grupos contendo 60%, 20% e 20% dos exemplos respectivamente, sendo um para o treinamento, um para a validação e o último para realizar o teste após o término do treinamento.

3.1.8 Treinando o Perceptron

Quando é preciso fazer com que o perceptron (figura 5) aprenda a reproduzir alguma saída em particular, isto é, que um padrão ative ou não os neurônios para uma dada entrada, devemos realizar um treinamento supervisionado, repetindo-o até que o erro da rede atinja valores aceitáveis.

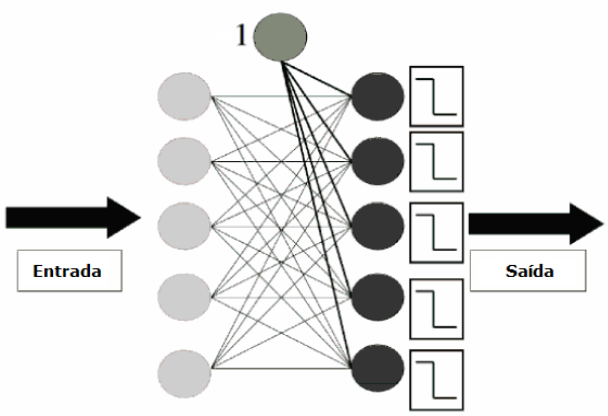


FIGURA 5 - A rede do Perceptron consiste em ligar as entradas (a esquerda) aos neurônios de McCulloch e Pitts (a direita) utilizando conexões com pesos sinápticos ajustáveis (MARSLAND, 2008).

A seguir é demonstrado o algoritmo de treinamento para uma rede neural formada por *n* perceptrons em uma única camada, cuja estrutura básica pode ser observada na figura 5, este algoritmo foi retirado de (MARSLAND, 2008).

- Inicie todos os pesos sinápticos w_{ij} com valores pseudoaleatórios pequenos, positivos e negativos, onde *i* corresponde ao índice do peso sináptico e *j* ao índice do neurônio.

- Para cada repetição *T* faça:

    - Para cada exemplo de treinamento faça:

        - Calcule o valor de saída *y* para cada neurônio da rede utilizando as equações 1 e 2, que podem ser escritas da seguinte forma:

$$y_j = g\left(\sum_{i=0}^{m} w_{ij}x_i\right) = \begin{cases} 1 \; se \; w_{ij}x_i > 0 \\ 0 \; se \; w_{ij}x_i \leq 0 \end{cases}$$

        - Calcule o erro na saída de cada neurônio

$$\mathrm{E} = (t_j - y_j)$$

Onde:

t_j é a saída esperada para a entrada x_i.

E é o erro na saída do neurônio j.

g(x) é a função de ativação.

        - Atualize cada peso sináptico individualmente fazendo:

$$w_{ij} = w_{ij} + \eta E x_i$$

Onde:

η é a taxa de aprendizado.

x_i é a entrada x na posição i.

3.1.9 Treinando o MLP com Backpropagation

O objetivo desta seção não é realizar a demonstração matemática das etapas do algoritmo *backpropagation* e sim apresentar o seu funcionamento, para que a partir do entendimento desta descrição seja possível compreender o processo de aprendizado de uma rede neural através da aplicação desta técnica. O algoritmo apresentado aqui foi retirado de (MARSLAND, 2008).

- Inicie todos os pesos sinápticos w_{ij} com valores pseudoaleatórios pequenos, positivos e negativos, onde *i* corresponde ao índice do peso sináptico e *j* ao índice do neurônio.

- Até que o nível de satisfação de atingido faça:

  - Calcule a saída para o exemplo de treinamento *x* utilizando a função de ativação *g(x)* escolhida.

    - Calcule o erro para cada neurônio da camada de saída.

$$E = (y_k - t_k)y_k(1 - y_k)$$

    - Calcule o erro para os neurônios das camadas intermediárias

$$E = y_k(1 - y_k)\sum_k w_{jk}E_j$$

    - Atualize todos os pesos sinápticos da rede.

$$w_{ij} = w_{ij} + \eta E x_i + \alpha \Delta w_{ij}^{t-1}$$

Onde:

η é a taxa de aprendizado.

x_i é a entrada x na posição i.

E é o erro na saída do neurônio j.

α é o momentum.

Δw_{ij}^{t-1} é a variação de W_{ij} nos tempos *t* e *t-1*.

3.2 Máquinas de Vetores de Supote (Support Vector Machine – SVM)

3.2.1 O que é Support Vector Machine?

Support Vector Machine (SVM) é outro método que pode ser utilizado para analisar e reconhecer padrões, um dos algoritmos de aprendizado de máquinas mais populares desde que foi publicado. Isto deve-se principalmente ao fato de ele normalmente, mas não sempre, apresentar resultados melhores do que outras técnicas (MARSLAND, 2008).

Esta técnica foi desenvolvida em 1992 por um grupo de pesquisa da AT&T Bell Laboratories e foi conhecida inicialmente como "algoritmo de treinamento para classificadores de margens ótimas" (Boser et al. 1992). No ano seguinte este mesmo grupo de pesquisa publicou o trabalho "Automatic Capacity Tuning of Very Large VC-dimension Classifiers" (Boser et al. 1993), no qual demonstram que uma grande quantidade de classificadores, incluindo o SVM e o perceptron, podem ser treinados com uma pequena quantidade de dados, os vetores de suporte, sem prejudicar o desempenho final da técnica.

O algoritmo de aprendizagem por vetores de suporte utilizado pelo SVM apresenta um comportamento mais genérico que o *backpropagation*, visto que através dele é possível treinar máquinas de aprendizagem polinomial, redes de função de base radial e perceptrons de duas camadas (com uma única camada oculta), entre outras, enquanto o *backpropagation* é capaz de treinar apenas o perceptron de múltiplas camadas (HAYKIN, 2001).

3.2.2 Características

(CARVALHO, 2005) cita em seu trabalho sobre "O estado da arte em métodos para reconhecimento de padrões: Support Vector Machine", como características das SVMs a detecção automática dos exemplos mais relevantes nas bases de dados utilizadas, chamados vetores de suporte; a robustez aos exemplos das bases de dados que são notadamente errôneos, conhecidos como outliers; e o mapeamento implícito dos exemplos em um espaço de dimensões elevadas, através das funções de kernel.

3.2.3 Funcionamento

A superfície de decisão posicionada pelo SVM busca maximizar a margem de separação entre as classes, tornando-a capaz de fornecer um bom desempenho de generalização e evitando o problema de overfitting (super aprendizado) (HAYKIN, 2001). Considerando-se um conjunto de pontos distribuídos em um plano, os pontos que estão mais próximos da classe a qual não pertencem, chamados de vetores de suporte, são separados em um subconjunto e utilizados para posicionar a superfície de decisão que irá separar as duas classes (figura 6), uma positiva e outra negativa (MARSLAND, 2008).

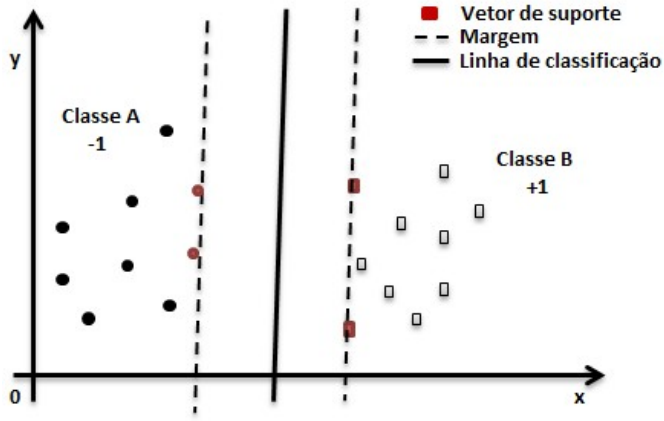


FIGURA 6 - As classes *A* e *B* são separada por uma linha que maximiza a distância margem das duas classes.

Partindo-se do princípio de que a melhor linha de separação é aquela que divide o espaço entre as classes em duas partes iguais, é possível afirmar que a margem entre as classes e a linha de classificação deve ser a maior possível e que os melhores pontos para serem utilizados ao posicionar a linha que separa as classes são os que foram classificados como vetores de suporte, pois eles são os que estão mais próximos de causar erro na classificação.

Quando o problema de classificação apresenta padrões que não podem ser separados linearmente o SVM realiza um mapeamento dos vetores de características apresentados para um espaço de dimensões superior ao espaço original, onde o conjunto de padrões tem uma maior probabilidade de ser separado por um hiperplano. Para realizar este mapeamento o SVM utiliza geralmente funções não-lineares, chamadas de funções de kernel, como funções poliomiais, de base radial e sigmoidal (REZENDE, 2002) e (AMORIM, 2007).

De acordo com (LORENA e CARVALHO, 2007), uma função de kernel "*k*" pode ser vista como uma função que recebe dois pontos (vetores de suporte), $\vec{w}$ e $\vec{x}$, do espaço de características original e retorna o seu produto escalar no espaço de características de dimensões mais elevadas. Esta função pode ser expressa matematicamente por $k(\vec{w},\vec{x})=\Phi(\vec{w}).\Phi(\vec{x})$, onde $\Phi(\vec{x})$ representa a função de mapeamento utilizada. Considerando-se uma situação em que a função de mapeamento é dada por $\Phi(\vec{x})=\Phi(x_1,x_2)=(x_1^2,\sqrt{2\,x_1 x_2},x_2^2)$, e duas amostras, w e x , definidas em R^2 respectivamente por $(\vec{w_1},\vec{w_2})$ e $(\vec{x_1},\vec{x_2})$, o kernel é dado por:

$$k(\vec{w},\vec{x})=(w_1^2,\sqrt{2\mathrm{w}_1 w_2},w_2^2).(x_1^2,\sqrt{2\mathrm{x}_1 x_2},x_2^2)=(\vec{w}.\vec{x})^2$$

O resultado da operação de mapeamento $\Phi(\vec{x})$ realizada no kernel é um novo ponto em um espaço de dimensional maior que o original, mas que ainda representa as características que geraram o ponto inicial. A tabela 2 apresenta alguns exemplos de funções que podem ser utilizadas para realizar o mapeamento de pontos no kernel do SVM, nestes exemplos as variáveis *gama* e *b* representam, nesta mesma ordem, o coeficiente angular e o deslocamento linear das equações.

TABELA 2

Exemplos de funções de mapeamento utilizadas pelo kernel do SVM.

Funções de kernel	
Kernel	**Função de mapeamento Φ**
Linear	$\vec{w}.\vec{x}$
Polinomial	$(gama*\vec{w}.\vec{x}+b)^{grau}$
Base Radial	$e^{-gama\lvert\vec{w}-\vec{x}\rvert^2}$
Sigmoidal	$\tanh(gama*\vec{w}.\vec{x}+b)$

(LORENA e CARVALHO, 2007) ainda ressaltam a necessidade da utilização de variáveis de folga para permitir presença de alguns pontos entre as margens do hiperplano de separação das classes e a ocorrência de erros durante o treinamento, uma vez que dificilmente

duas classes serão linearmente separáveis, a não ser em casos especiais ou em condições de estudo, seja pela presença de ruídos na base de dados ou pela própria natureza do problema.

Apesar dos benefícios proporcionados por está abordagem na classificação e reconhecimento de padrões, este método pode ter um custo computacional muito elevado quando aplicado a um conjunto de informações (datasets) "extremamente grande", visto que o SVM utiliza inversões de matrizes e envolve problemas de otimizações (MARSLAND, 2008).

3.2.3.1 Algoritmo do SVM

(KECMAN, 2001) apresenta o algoritmo de treinamento do SVM para uma base de dados linearmente separável, definida por:

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n), x \in \Re^n, y \in \{-1, +1\}$$

Como sendo:

1 – Processe as entradas presentes no conjunto de treinamento atribuindo-lhes às classes -1 ou +1.

2 – Entre todos os hiperplanos *d('w,x,b)* capazes de minimizar o erro durante o treinamento, encontre o que maximiza as margens entre o hiperplano e cada uma das classes.

$$d(x, w, b) = w \,.\, x + b = \sum_{i=1}^{n} w_i x_i + b$$

Onde w e $x \in \Re^n$ e $b \in \Re$.

3 - Uma vez que o treinamento seja concluído, basta aplicar uma função de sinal em *d(w,x,b)* para determinar a qual classe uma nova amostra pertence.

$$sgn\big(d(w, x, b)\big) = \begin{cases} +1 \; se \; w.x + b > 0 \\ -1 \; se \; w.x + b < 0 \end{cases}$$

3.3 K-Vizinhos mais próximos (K-Nearest Neighbors - KNN)

3.3.1 O que é K-Vizinhos?

O método dos k-vizinhos mais próximos, formalmente estudado por Cover & Harts em 1967, está entre as técnicas de aprendizado de máquinas mais simples para realizar o reconhecimento de padrões e difere das anteriores pela ausência do processo de treinamento, bastando apenas armazenar na memória as informações dos exemplos conhecidos (HAYKIN, 2001). De acordo com (MARSLAND, 2008), a ideia por trás do algoritmo K-Vizinhos é que se a amostra apresentada não é conhecida, a melhor coisa a fazer é analisar os exemplos similares a nova amostra para determinar a qual classe ela pertence.

Uma diferença chave entre esta técnica e os métodos anteriores é que nesta abordagem uma função de aproximação, para determinar a que classe o vetor de características pertence, é gerada para cada nova amostra apresentada ao algoritmo. Este comportamento proporciona uma vantagem significativa a esta técnica quando a função discriminante é muito complexa, mas pode ser descrita por um conjunto de funções de aproximação de complexidade reduzida (MITCHELL,1997).

No entanto, o aprendizado baseado em instâncias pode ter um custo computacional elevado para classificar novas amostras. Isto ocorre devido ao fato de que todos os cálculos ocorrem no momento da classificação e não quando os exemplos de treinamento são armazenados. Outra desvantagem deste tipo de abordagem é o fato de que todos os exemplos armazenados na memória são utilizados nos cálculos de aproximação para cada nova amostra, quando somente os mais próximos seriam o bastante para realizar a classificação (MITCHELL,1997).

3.3.2 Funcionamento do método KNN

Este algoritmo assume que cada instância corresponde a um ponto em um espaço de dimensão N, onde N é definido pelo número de características de cada instância, atribuindo para a nova amostra a mesma classe do seu vizinho mais próximo, portanto, o mais semelhante. Uma forma de definir o vizinho mais próximo a cada ponto no espaço é através do cálculo da distância Euclidiana (MITCHELL,1997):

$$d(x_i, x_j) \equiv \sqrt{\sum_{r=1}^{n} \left(a_r(x_i) - a_r(x_j)\right)^2}$$

Onde:

$d(\vec{x}_i, \vec{x}_j)$ é distância entre os pontos $\vec{x}_i$ e $\vec{x}_j$.

$a_r(\vec{x}_i)$ é a característica r da instância $\vec{x}_i$.

n é o número total de coordenadas de cada ponto.

Uma variação do método do vizinho mais próximo é o método dos *k-vizinhos mais próximos*, onde k é o número de vizinhos a serem considerados no momento da classificação. Assim, cada novo exemplo apresentado ao algoritmo será classificado de acordo com a classe mais frequente dentre seus vizinhos mais próximos (HAYKIN, 2001). Ainda é possível tornar está variante mais eficiente atribuindo pesos a cada elemento considerado no processo de classificação, de acordo com sua distância em relação a amostra que está sendo classificada, diminuindo a importância de cada ponto conforme sua distância aumenta (MITCHELL,1997).

A importância de cada elemento no processo de classificação e a função que define a qual classe uma determinada amostra de entrada pertence podem ser definidas matematicamente, segundo (MITCHELL,1997), como demonstrado abaixo.

$$f'(x_q) \leftarrow argmax_{v \in V} \sum_{i=1}^{k} w_i \delta(v, f(x_i))$$

Onde:

w_i é a importância do exemplo x_i no processo de definição da classe da amostra x_q .

$f(x_q)$ retorna a classe que será atribuída a amostra x_q .

v é uma classe no conjunto de V classes possíveis.

K é o número de vizinhos mais próximos utilizados para realizar o processo de definição da classe de uma amostra.

argmax retorna a classe *v* com o maior somatório de pesos nos *K* vizinhos considerados.

$f(x_i)$ retorna a classe do exemplo x_i .

δ retorna um se a classe *v* for igual a $f(x_i)$ e zero caso contrário.

4 METODOLOGIA

Nos capítulos anteriores foram apresentados os fundamentos teóricos necessários ao desenvolvimento deste trabalho. Nesta seção é definida a abordagem utilizada durante a avaliação das técnicas de aprendizado de máquinas abordadas neste trabalho, passando pela definição dos objetos de interesse, das informações utilizadas para caracterizá-los, pela forma como estas informações foram extraídas e pelas configurações adotadas por cada uma das técnicas durante o processo de reconhecimento e classificação.

Para o caso em estudo, foram selecionados quatro tipos de objetos (carro, bicicleta, garrafa e TV/monitor) dentre os disponibilizados na base de dados da competição Visual Object Classes Challenge[4], a partir da qual também foram obtidas as imagens segmentadas por objetos. As figuras 7 e 8 apresentam respectivamente um exemplo de imagem encontrada nesta base de dados, onde os objetos de interesse são duas TV/Monitor, e a representação desta mesma cena após a imagem ter passado pelo processo de segmentação.

Juntamente com as imagens contidas nesta base de dados estão disponíveis arquivos em formato XML cujo conteúdo descreve os elementos presentes em cada cena. Esta descrição inclui informações como as de quais são os objetos de interesse presentes em uma determinada imagem, sua posição, se o objeto está truncado ou se é de difícil reconhecimento. Um objeto é dito truncado se ele não esta representado em sua totalidade dentro da região abrangida pela imagem. Se o objeto for claramente visível na cena mas não puder ser identificado sem a avaliação do contexto em que aparece, ele é descrito como de difícil reconhecimento. Para a avaliar o desempenho dos métodos de aprendizado de máquinas abordados neste trabalho foram desconsiderados os objetos descritos como difíceis.

---

4 Visual Object Classes Challenge 2010 (VOC2010). Disponível em: <http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2010/> Acesso em: 21 abr. 2011

FIGURA 7 - Imagem antes de passar pelo processo de segmentação.

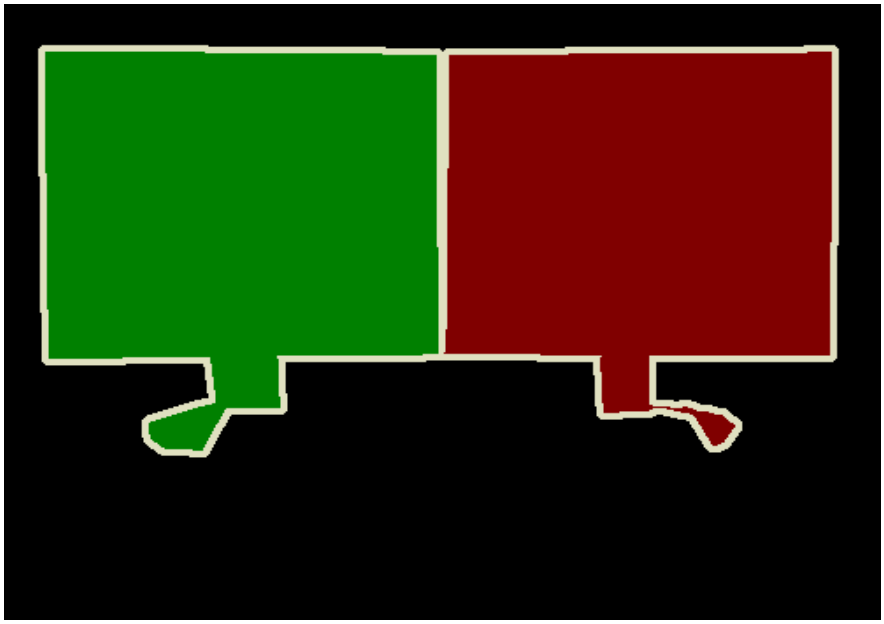


FIGURA 8 - Imagem segmentada por objetos.

Para o caso em estudo foram selecionados os objetos TV/monitor, garrafa, carro e bicicleta, os quais foram divido em dois grupos: TV/monitor e garrafa; carro e bicicleta. Cada um

dos dois grupos foi avaliado em duas etapas, primeiramente os objetos descritos como truncados ou difíceis foram desconsiderados, na segunda etapa apenas os objetos descritos como difíceis foram desconsiderados. Após definir quais são os objetos de interesse e como eles serão apresentados aos classificadores, é necessário definir quais serão as informações utilizadas para caracterizá-los e permitir que os classificadores realizem a distinção entre um ou outro objeto.

Com a finalidade de facilitar o aprendizado necessário para o desenvolvimento deste trabalho foi desenvolvida uma interface gráfica, que pode ser observada na figura 9, para visualizar as imagens e o conteúdo de cada arquivo XML. A interface tem o propósito de permitir a visualização dos resultados da manipulação das imagens sem que para isso seja necessário criar um novo arquivo contendo a imagem modificada, mas ela não está ligada de forma alguma aos resultados obtidos pelos classificadores ou ao processo de extração de características.

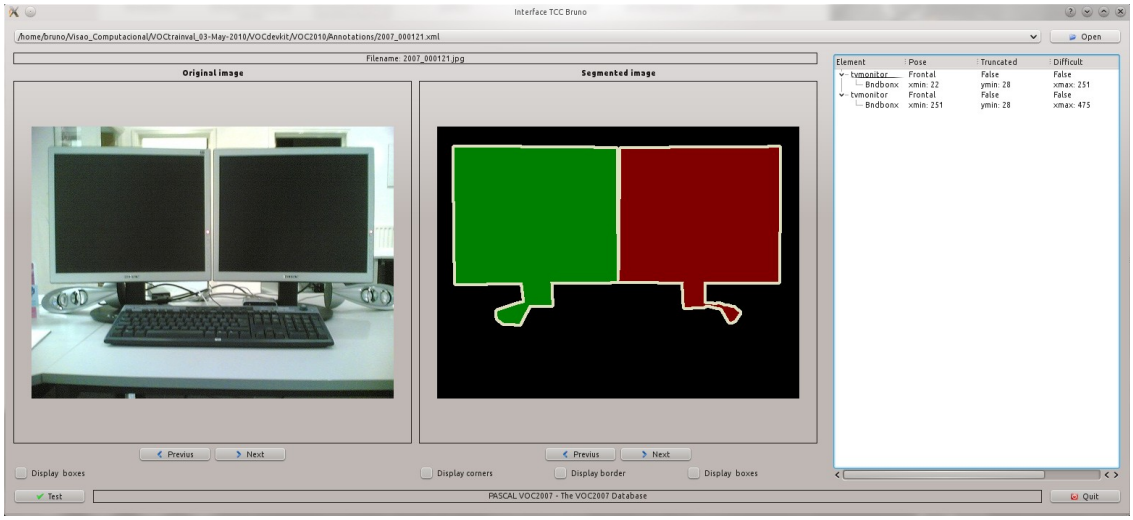


FIGURA 9 - Interface gráfica desenvolvida para auxiliar no desenvolvimento deste trabalho.

4.1 Extração de características

Dentre os atributos utilizados para caracterizar os objetos de interesse em imagens citados na literatura foram selecionados e avaliados quatro para cada par de objetos, buscando descrevê-los com base em sua forma. Três dos atributos utilizados são mencionados por (SONKA et al, 1998) como sendo capazes de fornecer bons resultados para regiões simples, a

compactação/circularidade, a porção da área do menor retângulo capaz de envolver o objeto que é ocupada pelo objeto e a razão entre a largura e a altura do menor retângulo capaz de envolver o objeto (alongamento). A última informação selecionada foi o número de cantos de dos objetos de interesse, utilizado em várias aplicações de visão computacional, como é o caso de (SAMPAIO, 2010).

Com o propósito de tornar possível a visualização da distribuição espacial dos objetos e a posição dos erros cometidos pelos classificadores, as informações extraídas foram avaliadas três a três para cada um dos grupos, sendo selecionadas de acordo com a distribuição de cada grupo de objetos no espaço de caraterísticas quando representado por cada conjunto de características.

A extração destas informações a partir das imagens segmentadas foi realizada por meio de um programa desenvolvido C++, ocorrendo a medida que as descrições das imagens são lidas e filtradas. Quando uma imagem possui mais algum objeto além dos desejados para o caso em estudo, os indesejados são eliminados, atribuindo-lhes a mesma cor do fundo da imagem (preto por padrão). Se a imagem não possuir nenhum objeto de interesse, ela é descartada antes de as informações a respeito de cada objeto serem extraídas.

Para casos como o demonstrado na figura 8, onde mesmo após o processo de filtragem existem dois ou mais objetos de interesse na mesma imagem, são feitas "n" cópias em memória desta imagem para que cada um dos objetos pode ser processado de forma isolada, sendo "n" igual ao número de objetos de interesse presentes na cena. A separação dos objetos de uma imagem ocorre fazendo com que apenas a cor de um objeto, além do preto, permaneça em cada cópia da imagem. A partir deste ponto, cada uma das imagens em memória é tratada como uma imagem binária, onde os pixels com valores diferentes de zero (preto) são considerados como parte integrante do objeto ali representado.

Uma vez que os objetos de interesse estejam separados uns dos outros é iniciada a extração das informações que serão utilizadas para reconhecê-los posteriormente. Este processo parte pela detecção de cantos, que foi realizada com o uso do algoritmo de Harris, disponível na biblioteca OpenCV[5] através da função *cvGoodFeaturesToTrack()*. Os parâmetros passados

5 OpenCV (Open Source Computer Vision). Disponível em: <http://opencvlibrary.sourceforge.net/> Acesso em: 30 mai. 2011.

a esta função foram: blocos de tamanho 5x5, k igual a 0.1, distância euclidiana mínima entre cada canto definida como 7, nível de qualidade em 0.005 e número máximo de cantos para cada objeto em 100. Estes valores foram selecionados pela observação dos cantos retornados pelo algoritmo com cada configuração, e uma vez definidos não variaram mais.

Dando continuidade ao processo de extração de características é realizada a detecção de bordas, através do algoritmo de vizinhança de 8, seguida pelo cálculo da circularidade, descrito em (SONKA et al, 1998) como sendo:

$$Circularidade = \frac{(região\ da\ borda\ do\ objeto)^2}{área\ do\ objeto}$$

Concluindo esta etapa do sistema, é realizado o cálculo da diferença entre a área do menor retângulo capaz de envolver o objeto e a área do objeto, e o cálculo do alongamento, que é a razão entre a largura e a altura do menor retângulo capaz de envolver o objeto. O resultado deste processamento é um arquivo de texto contendo os valores, que ainda devem ser normalizados antes de serem apresentados aos algoritmos de aprendizado de máquina, das quatro características extraídas de cada objeto e o rótulo/nome de cada um.

Outra questão que deve ser destacada aqui é que apesar de existirem falhas resultantes do processo de segmentação, elas não foram tratadas, ficando a cargo dos classificadores tratar as variações nas informações obtidas de cada objeto causadas por tais falhas. As figuras 10, 11 e 12 apresentam de forma mais clara uma falha decorrente do processo de segmentação, apontada por uma seta azul na figura 11, e o resultado desta falha ao detectar-se o menor retângulo capaz de envolver um objeto.

Com a etapa de extração de características concluída, a eficácia de cada conjunto de atributos ao representar os objetos de interesse foi observada através da análise da distribuição dos objetos no espeço de características, quando representados por um ou outro conjunto de informações, conforme o demonstrado a seguir.

FIGURA 10 - Imagem antes de passar pelo processo de segmentação, com um objeto de interesse a esquerda da imagem, uma TV/Monitor.

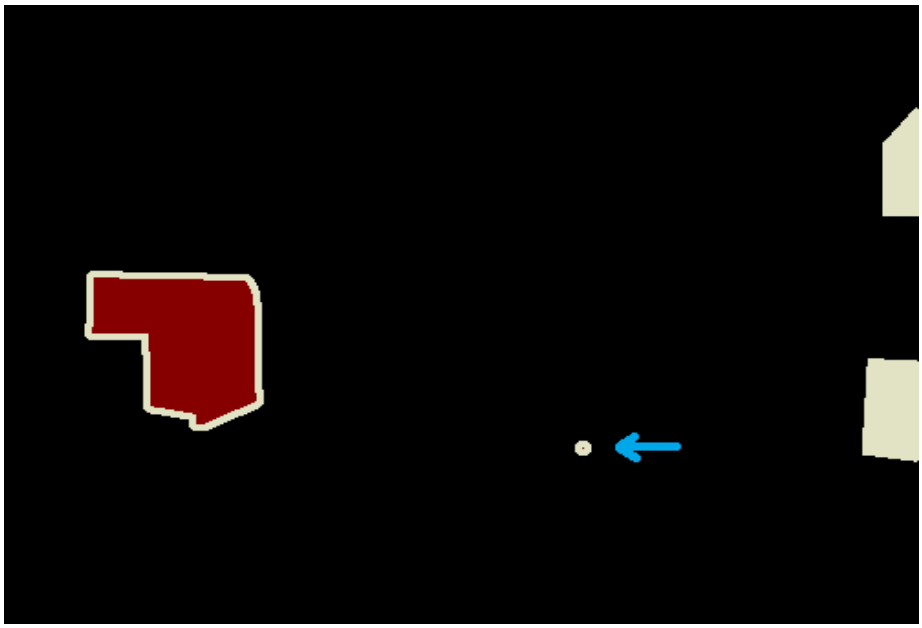


FIGURA 11 - Imagem após passar pelo processo de segmentação, com um pixel erroneamente marcado como região de interesse.

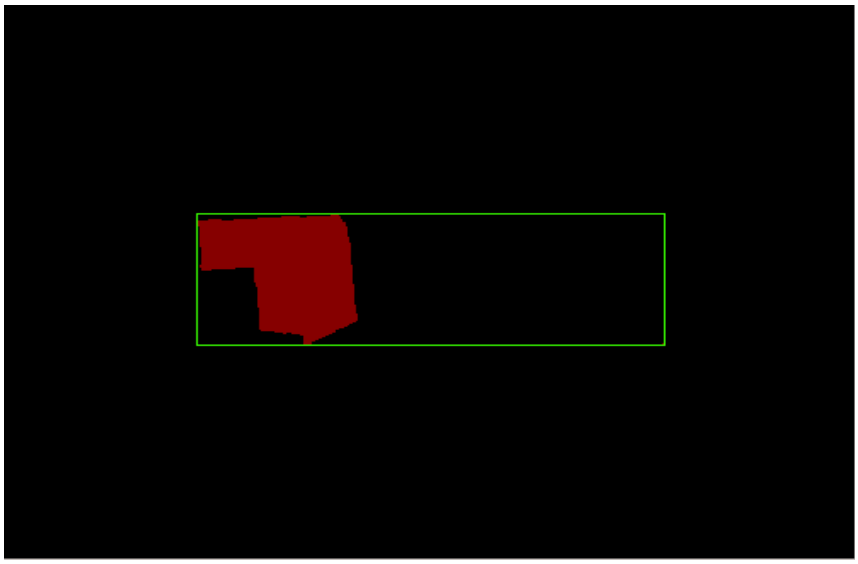


FIGURA 12 - Falha no processo de segmentação causando um resultado indesejado ao extrair uma informação da região de interesse.

Ao observar-se os gráficos apresentados nas figuras 13 e 14 é possível notar a diferença na distribuição espacial dos objetos carro e bicicleta quando representados por diferentes grupos de informações, todas obtidas a partir da representação destes objetos nas imagem segmentada disponíveis na base de dados utilizada. Em ambos os gráficos cada objeto é representado por um ponto, sendo que os carros estão representados cada um por um triângulo azul-escuro e as bicicletas por um ”x” azul-claro cada. Na distribuição espacial observada na figura 13, as informações utilizadas para representar cada um dos objetos foram o seu alongamento, o número de cantos e a porção de área do menor retângulo capaz de envolver o objeto que é ocupada pelo objeto em questão. Vale ressaltar que os valores apresentados nos eixos X, Y e Z variam entre zero e um em todos os gráficos de distribuição espacial dos quatro objetos de interesse, fato ocasionado pelo processo de normalização das informações referentes a cada característica extraída antes de os gráficos serem gerados.

Comparando os gráficos das figuras 13 e 14 fica claramente evidenciada a melhor caracterização dos objetos quando representados pelas informações utilizadas no gráfico da figura 14 (onde o alongamento foi substituído pela circularidade), o que vai de encontro ao objetivo desta etapa do sistema, chegar ao mais próximo possível de agrupar cada objeto de interesse em uma região distinta do espaço de características.

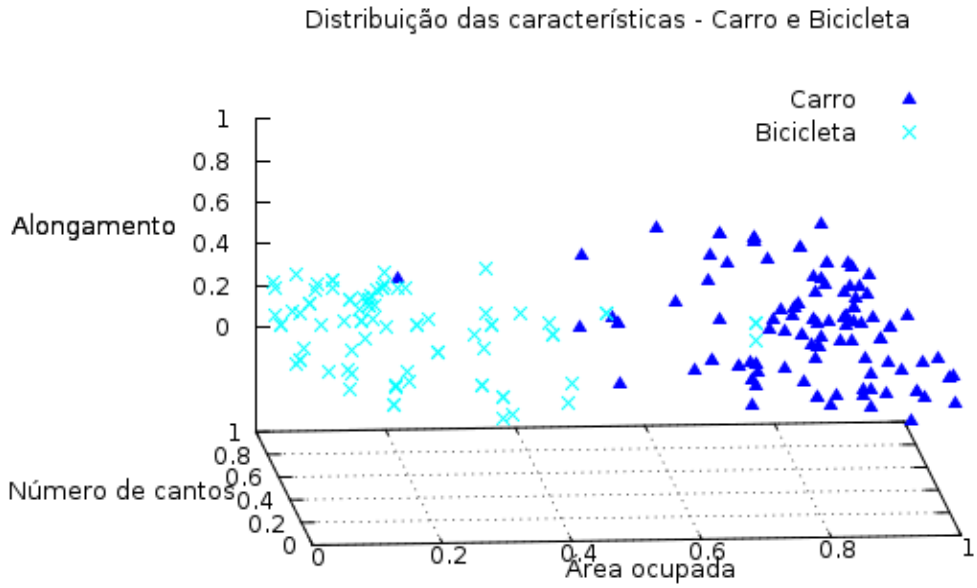


FIGURA 13 - Distribuição dos objetos carro e bicicleta no espaço de características quando representados por seu alongamento, número de cantos e área ocupada.

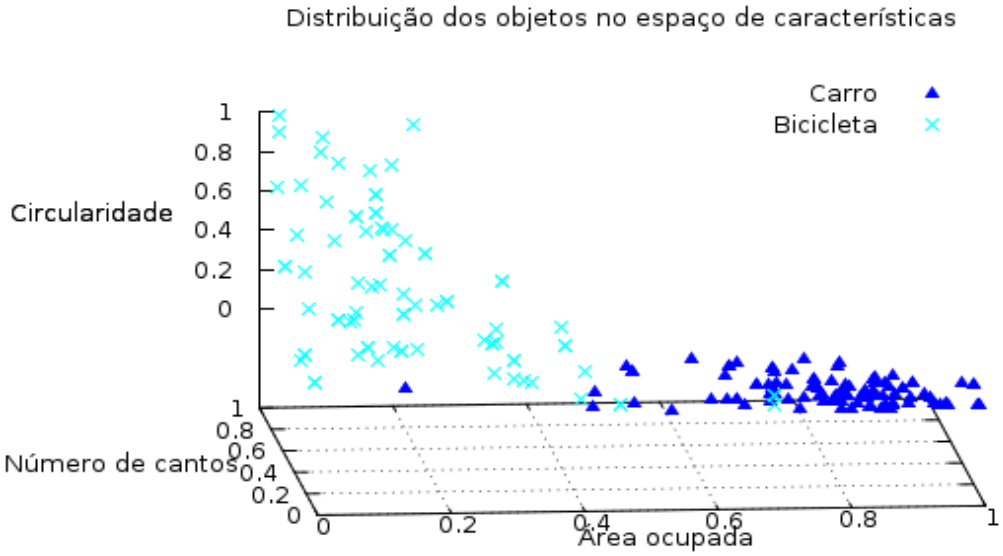


FIGURA 14 - Distribuição dos objetos carro e bicicleta no espaço de características quando representados por sua circularidade, número de cantos e área ocupada.

As figuras 15 e 16 apresentam dois gráficos de distribuição dos objetos TV/Monitor e garrafa no espaço de características. No primeiro este objetos são representados pelo seu alongamento, circularidade e porção de área do menor retângulo capaz de envolver o objeto ocupada pela TV/Monitor ou garrafa. No segundo gráfico, apresentado na figura 16, o alonga-

mento foi substituído pelo número de cantos, resultando em um agrupamento maior dos pontos na mesma região. Devido a melhor distribuição dos objetos, as informações selecionadas para representar os objetos TV/Monitor e garrafa durante a etapa de classificação e reconhecimento foram as apresentadas no gráfico da figura 15.

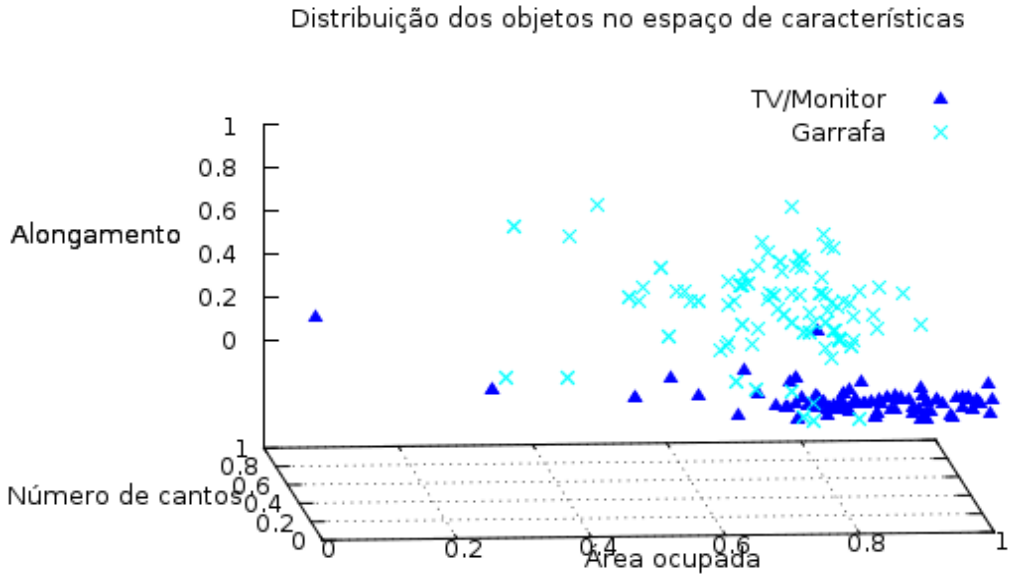


FIGURA 15 - Distribuição dos objetos TV/Monitor e garrafa no espaço de características quando representados por seu alongamento, circularidade e área ocupada.

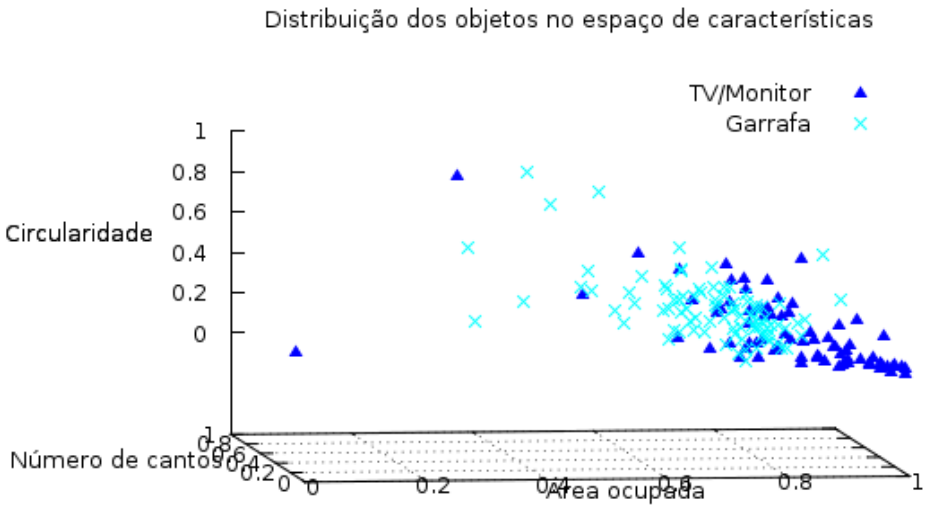


FIGURA 16 - Distribuição dos objetos TV/Monitor e garrafa no espaço de características quando representados por sua circularidade, número de cantos e área ocupada.

4.2 Classificação e reconhecimento

Para avaliar o desempenho das técnicas de aprendizado de máquina abordadas neste trabalho, os algoritmos KNN e MLP foram implementados em C++ para classificar um grupo de objetos (composto por dois objetos) por vez, funcionando de forma similar ao SVM, onde foi utilizada a LIBSVM[6]. Os conjuntos de exemplos, já normalizados, apresentados aos classificadores durante o processo de treinamento e testes foram os mesmo para todas as três técnicas e repetições de execução, sendo divididos e apresentados de forma aleatória na proporção de 60% dos exemplos para treinamento e 40% para testes, seguindo as recomendações de (MARSLAND, 2008). As tabelas 3 e 4 apresentam de forma mais detalhada a distribuição das amostras dos quatro objetos utilizados para realizar este trabalho em cada uma das etapas da avaliação.

TABELA 3

Distribuição dos exemplos de carros e bicicletas durante a avaliação de desempenho do classificadores.

Distribuição das amostras – Carro e Bicicleta			
Objeto	**Treinamento**	**Teste**	**Total**
Etapa 1	**Número de amostras – Sem objetos truncados**		
Carro	53	37	90
Bicicleta	38	26	64
Etapa 2	**Número de amostras – Com objetos truncados**		
Carro	127	86	213
Bicicleta	68	47	135

6A Library for Support Vector Machines (LIBSVM). Disponível em: <http://www.csie.ntu.edu.tw/~cjlin/libsvm/> Acesso em: 25 abr. 2011

TABELA 4
Distribuição dos exemplos de TVs/Monitores e garrafas durante a avaliação de desempenho do classificadores.

Distribuição das amostras – TV/Monitor e Garrafa			
Objeto	**Treinamento**	**Teste**	**Total**
Etapa 1	**Número de amostras – Sem objetos truncados**		
TV/Monitor	46	32	78
Garrafa	53	36	89
Etapa 2	**Número de amostras – Com objetos truncados**		
TV/Monitor	76	51	127
Garrafa	97	65	162

4.2.1 RNA (Rede Neural Artificial)

Para realizar a avaliação da RNA foi implementado um MLP (Multi-Layer Perceptron) de três camadas, sendo a primeira composta por três neurônios, a segunda com o número de neurônios variando de dois a dez e a terceira com dois neurônios, como pode ser observado na figura 17. O treinamento foi realizado utilizando o algoritmo backpropagation, a função de ativação sigmoidal e com os pesos sinápticos inicializados de forma aleatória, variando entre -0.5 e 0.5. Ainda no que se refere as configurações da RNA, o parâmetro β da função de ativação sigmoidal, demonstrada na equação 4 da tabela 1, foi mantido fixo em 1, da mesma forma que o número de camadas intermediárias.

Durante o treinamento da RNA, o processo de validação foi realizado ao término de cada época, visto que cada época corresponde a uma apresentação completa do conjunto de treinamento, onde a taxa de erro da validação no tempo "*t*" é comparada com a taxa de erro no tempo "*t-1*", para que a cada melhora os pesos sinápticos da RNA sejam salvos. O objetivo desta abordagem é possibilitar que ao término do treinamento, limitado a três mil épocas ou até que a taxa de erro de validação caia a zero, seja possível restaurar os pesos da RNA ao momento em que ela obteve o melhor resultado de validação. Esta abordagem foi inspirada no método *early stopping*, recomendado por (MARSLAND, 2008), diferenciando-se pelo fato de permitir pioras na taxa de erro da validação ao pressupor que ela pode estar em um mínimo local.

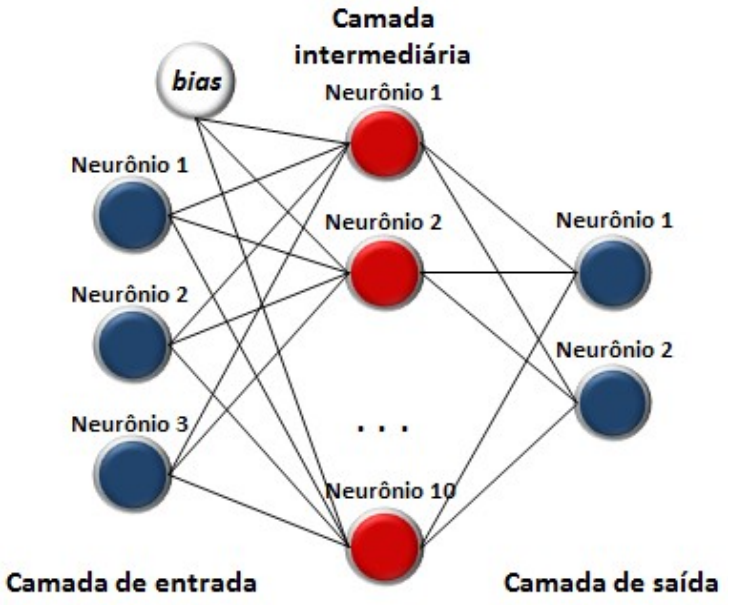


FIGURA 17 - Estrutura da RNA utilizada neste trabalho, com as camadas de entrada e saída em azul e a camada intermediária em vermelho.

O processo de avaliação da RNA ocorreu em duas fases para cada etapa de avaliação dos grupos de objetos, totalizando 4590 execuções do algoritmo em cada grupo. Na primeira fase de avaliação deste método foram utilizados valores abaixo do recomendado na literatura para a taxa de aprendizado, deixando para a segunda a atribuição dos valores recomendados, entre 0,1 e 0,4. A relação completa de configurações utilizadas para avaliar o desempenho da RNA em cada fase pode ser observada na tabela 5, onde o número de neurônios indicado é referente a camada intermediária, uma vez que o número de entradas e saídas da rede permanecem constantes.

TABELA 5

Configurações adotadas durante a avaliação de desempenho da RNA.

Configurações avaliadas da RNA				
Taxa de aprendizado	**Momentum**	**Nº Neurônios**	**Repetições**	**Total de execuções**
0,001 até 0,01 com intervalo de 0,001	0.2 até 0,4 com intervalo de 0,1	2 a 10	5	1350
0,1 até 0,4 com intervalo de 0,05	0,2 até 0,4 com intervalo de 0,1	2 a 10	5	945

Com o propósito de facilitar o entendimento ao comparar-se o desempenho entre os métodos aqui abordados, os exemplos utilizados durante a validação da RNA foram reapresentados durante a etapa de testes, permitindo assim que 40% dos exemplos fossem utilizados para montar a matriz de confusão e avaliar a taxa de acertos, da mesma forma que ocorre com os métodos KNN e SVM. A matriz de confusão é uma forma de avaliar o desempenho de um classificador, onde o número de amostras classificadas corretamente é apresentado na diagonal principal, nas demais posições estão o número de erros de classificação para cada classe de objetos (REZENDE, 2002).

4.2.2 SVM (Support Vector Machine)

Durante a avaliação do classificador SVM foram utilizadas as quatro funções de kernel disponibilizadas na LIBSVM, cujos valores adotados para cada um dos parâmetros podem ser observados na tabela 6. Sendo que para cada configuração adotada foi realizada uma execução, totalizando vinte e uma execuções do classificador para cada uma das etapas de avaliação dos grupos. Lembrando que na primeira etapa são desconsiderados os objetos descritos como difíceis ou trucados e na segunda apenas os objetos descritos como difíceis são desconsiderados.

TABELA 6

Configurações adotadas durante a avaliação de desempenho do SVM.

Configurações avaliadas do SVM			
kernel	**Função de mapeamento**	**Grau**	**Gama**
Linear	$\vec{w}.\vec{x}$	-	-
Polinomial	$(gama*\vec{w}.\vec{x}+b)^{grau}$	2 a 4	0,2 a 0,5 com intervalos de 0,1
Base Radial	$e^{-gama\lvert\vec{w}-\vec{x}\rvert^2}$	-	0,2 a 0,5 com intervalos de 0,1
Sigmoidal	$\tanh(gama*\vec{w}.\vec{x}+b)$	-	0,2 a 0,5 com intervalos de 0,1

4.2.3 KNN (K-Nearest Neighbors)

No algoritmo KNN utilizado nesta avaliação a importância de cada vizinho durante o processo de atribuição de um rótulo a um novo exemplo é dada de acordo com a distância que cada um dos "n" vizinhos está da nova entrada. Esta abordagem segue as recomendações de (MARSLAND, 2008) e busca reduzir o impacto dos exemplos mais distantes, valorizando os mais próximos da nova amostra durante o processo de votação. O número de vizinhos considerados para avaliar os dois grupos de objetos selecionados para este trabalho variam de um até cinquenta, totalizando cinquenta execuções do KNN para cada uma das etapas de avaliação dos grupos de objetos.

Inicialmente o peso do voto para cada um dos vizinhos mais próximos foi calculado através da divisão de 1 pelo quadrado da distância da nova amostra em relação a cada um de seus vizinhos. Porém, realizando uma análise preliminar dos resultados obtidos pelo KNN, foi possível notar que a variação da importância de cada vizinho demonstrou-se muito sensível a pequenas variações na distância. No gráfico apresentado na figura 18 está representada em tons de azul a base de conhecimento do KNN para os objetos carro e bicicleta (considerando os truncados), obtida durante o treinamento, e destacada por um circulo vermelho a região de um carro erroneamente classificado como bicicleta. Ao observar-se a região interna ao circulo, é possível notar que apesar de aquela ser uma região onde predominam os exemplos de carros, o classificador atribuiu o rótulo errado a nova amostra devido ao fato de os dois exemplos mais próximos serem de bicicletas.

Com o propósito de reduzir o impacto destes ruídos na base de conhecimento o método utilizado para calcular a importância de cada voto foi modificado, passando para 1 dividido pela distância do novo exemplo até cada um de seus vizinhos, reduzindo assim a diferença de a importância entre os vizinhos mais próximos e os mais distantes, mas ao mesmo tempo mantendo esta relação entre eles. Devido a melhora apresentada nos resultados obtidos pelo classificador KNN quando este método foi utilizado no o cálculo do peso para cada voto, durante a avaliação dos seus resultados foi esta a forma utilizada para calcular a importância dos votos.

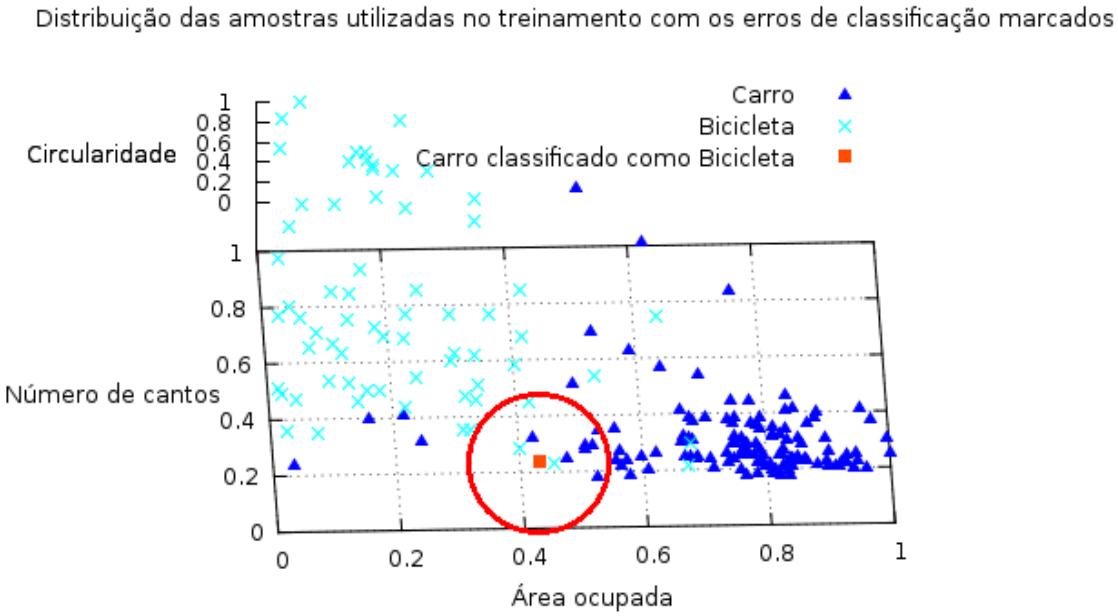


FIGURA 18 - Distribuição da base de conhecimento do classificador KNN com os erros de classificação em destaque.

5 RESULTADOS

O objetivo dos testes aqui executados é avaliar a taxa de acertos de cada um dos classificadores abordados neste trabalho ao atribuir rótulos aos objetos de interesse a eles apresentados. Porém, os resultados obtidos podem variar de acordo com as informações utilizadas para caracterizar cada objeto e com qualidade das imagens utilizadas no processo de reconhecimento como um todo. Afim de demonstrar esta variação, os resultados são apresentados em duas partes. Inicialmente são apresentados os resultados obtidos com cada classificador quando os objetos truncados, que não estão representados em sua totalidade dentro da cena, são excluídos do processo de classificação, e posteriormente, quando eles são inclusos no processo.

Ao longo da apresentação dos resultados obtidos pela RNA são demonstrados gráficos sobre o comportamento deste classificador durante o processo de treinamento, com a configuração adotada durante a execução apresentada no canto superior direito de cada gráfico. Devido ao fato de o número de neurônios nas camadas de entrada e saída da rede neural não variar, toda a referência ao "número de neurônios" realizada nestes gráficos indica o número de neurônios utilizados na camada intermediária durante o treinamento apresentado.

5.1 Resultados desconsiderando-se os objetos descritos como truncados

Durante a avaliação de desempenho dos classificadores, desconsiderando-se os objetos descritos como trucados, todos obtiveram os mesmos resultados ao classificar os dois grupos de objetos de interesse, errando apenas ao classificar uma TV/Monitor e uma garrafa, os mesmos em todos os casos. No gráfico da figura 19 é demonstrada a posição que estes objetos, em

vermelho e rosa, ocupavam no espaço de características em relação as amostras utilizadas durante o treinamento, tons de azul, de todos os classificadores.

O erro cometido pelos classificadores ao atribuir o rótulo de garrafa a uma TV/Monitor pode ser melhor entendido observando-se novamente as imagens apresentadas nas figuras 11 e 12, onde pode ser visualizada a falha no processo de segmentação que levou a alteração nos padrões de área ocupada pela TV/Monitor dentro do menor retângulo capaz de envolvê-la. Já o erro cometido ao classificar uma garrafa como TV/Monitor deve-se ao ângulo em que a garrafa encontrava-se durante o processo de aquisição da imagem, conforme pode ser observado nas figuras 20 e 21, onde a garrafa encontra-se destacada por um retângulo verde.

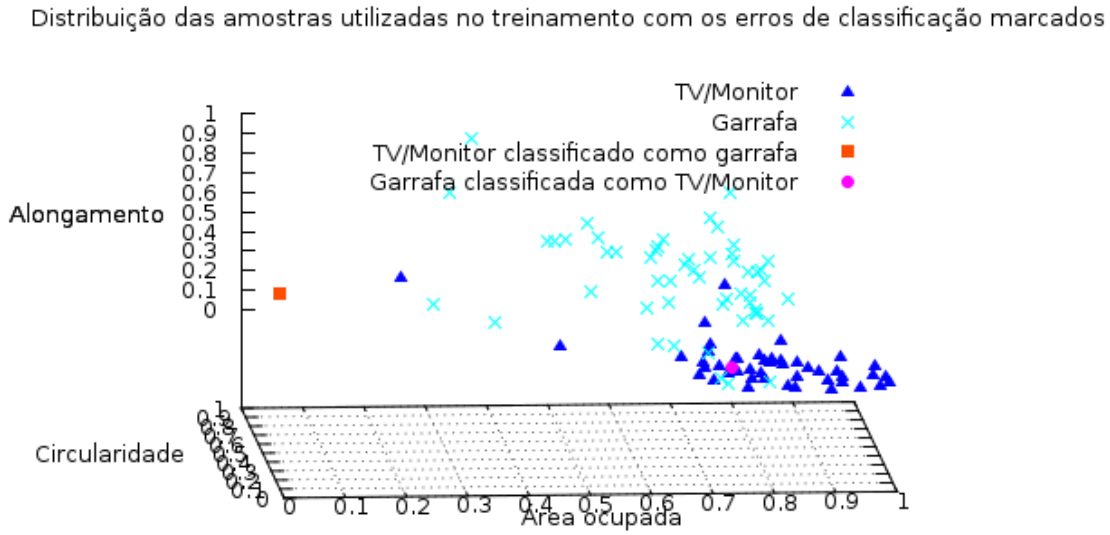


FIGURA 19 - Distribuição das amostras de TV/Monitor e garrafa utilizadas durante o treinamento dos classificadores com os erros destacados em vermelho e rosa.

FIGURA 20 - Imagem original da garrafa classificada como TV/Monitor pelos três classificadores.

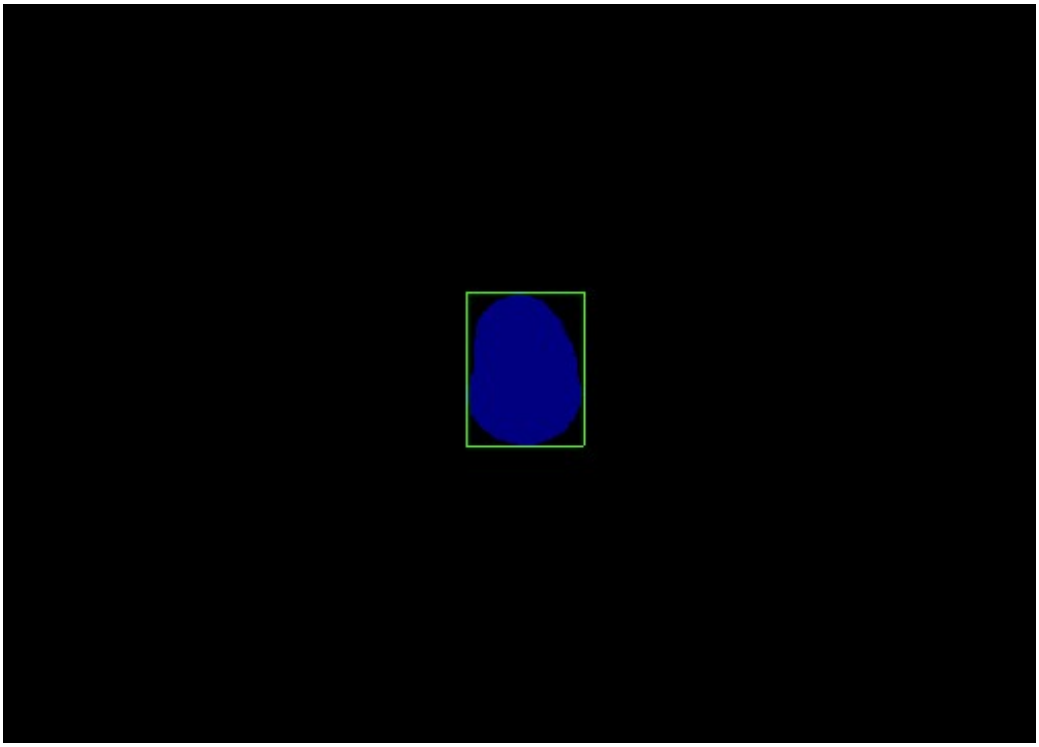


FIGURA 21 - Imagem segmentada da garrafa classificada como TV/Monitor pelos três classificadores.

Conforme o observado na figura 20, a posição em que a garrafa encontrava-se dentro da cena fez com que a relação entre largura e altura do menor retângulo capaz de envolvê-la (Alongamento) se aproximasse da relação entre largura e altura gerada pela imagem de uma TV/Monitor, como consequência, a posição deste objeto dentro do espaço de características foi deslocada para uma região ocupada por monitores, levando os classificadores ao erro. A seguir, nas seções de 5.1.1 a 5.1.3 , os resultados obtidos pelos classificadores nesta etapa da avaliação são apresentados de forma mais detalhada.

5.1.1 KNN (K-Nearest Neighbors)

O desempenho do classificador KNN ao atribuir rótulos para carros e bicicletas pode ser observado no gráfico da figura 22, onde cada barra representa a taxa de acertos "y" do classificador quando o número "x" de vizinhos são considerados durante o processamento do conjunto de testes. Ao observar este gráfico é possível notar que o desempenho do KNN manteve-se estável a partir de vinte e quatro vizinhos, classificando corretamente 100% do conjunto de testes.

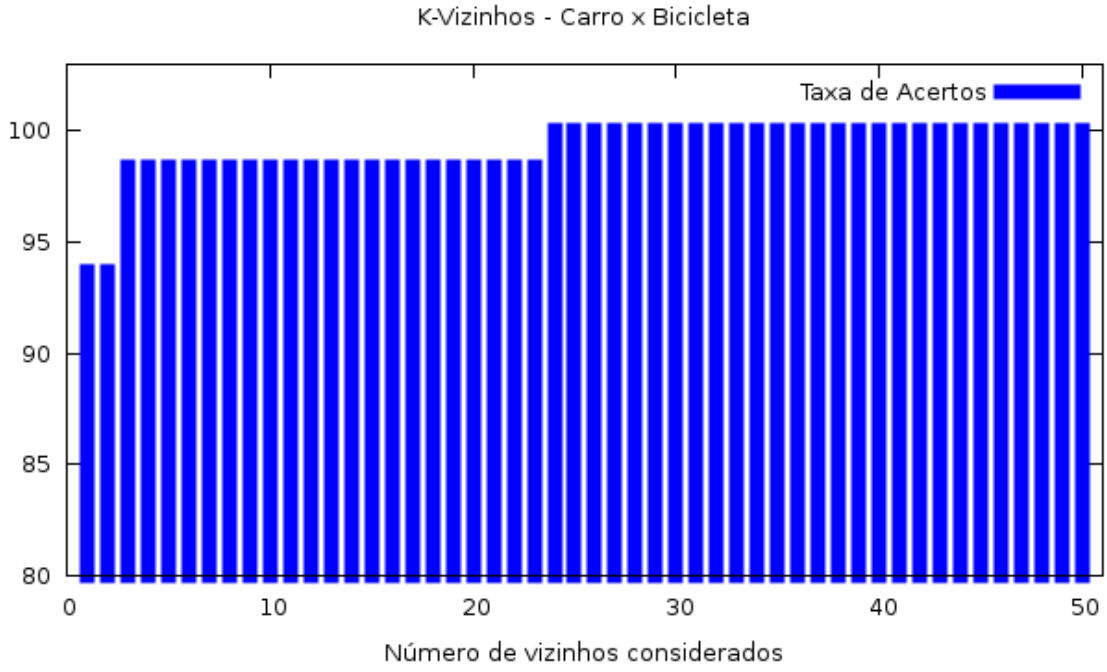


FIGURA 22 - Taxa de acertos do KNN ao classificar carros e bicicletas.

A matriz de confusão do KNN ao atribuir rótulos para carros e bicicletas pode ser observada na tabela 7, onde a interseção linhas e colunas identificadas pelo mesmo objeto indica o número de classificações corretas o objeto em questão, enquanto a interseção entre linhas e colunas identificadas por objetos diferentes indica o número de erros de classificação.

TABELA 7

Matriz de confusão do KNN ao classificar carros e bicicletas.

Matriz de confusão do KNN – Carro x Bicicleta		
	Carro	**Bicicleta**
Carro	37	0
Bicicleta	0	26

Quando o segundo grupo de objetos, composto por TV/Monitor e garrafas, foi apresentado a este classificador o seu desempenho atingiu 97.14 % de acerto, conforme pode ser observado no gráfico apresentado na figura 23, onde é demonstrada a taxa de acertos que o KNN obteve para cada quantia de vizinhos considerados durante a classificação.

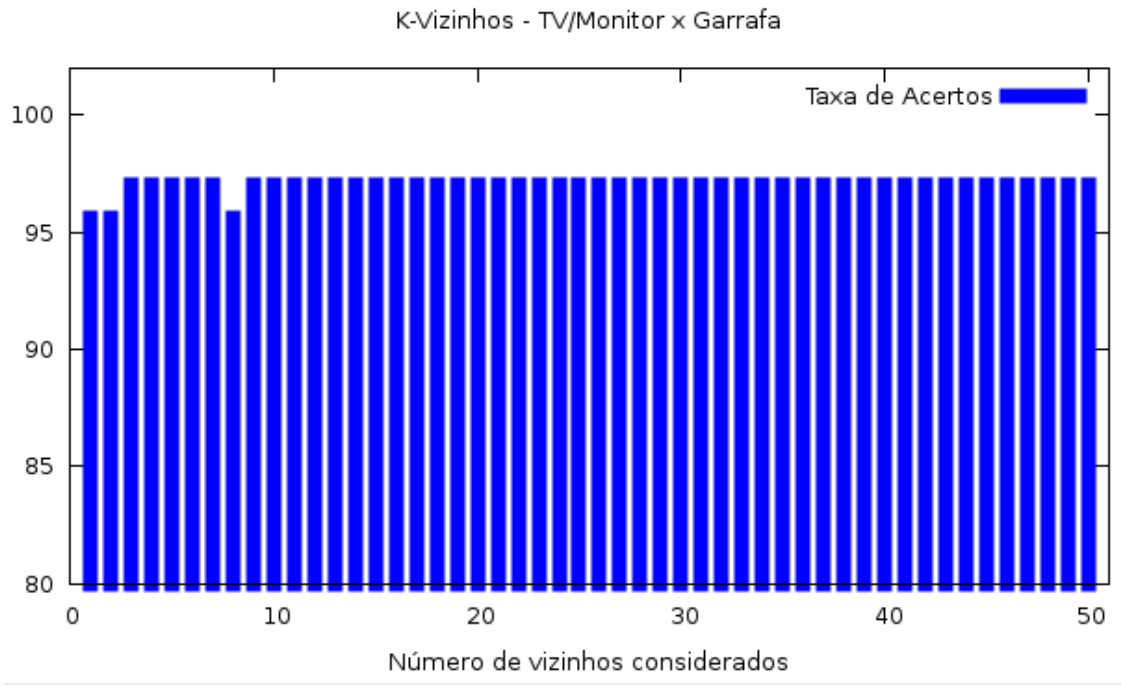


FIGURA 23 - Taxa de acertos do KNN ao classificar TV/Monitor e garrafas.

Na tabela 8 é possível visualizar os erros cometidos pelo KNN ao classificar TV/Monitor e garrafa, através de uma matriz de confusão.

TABELA 8

Matriz de confusão do KNN ao classificar TV/Monitor e garrafas.

Matriz de confusão do KNN – TV/Monitor x Garrafa		
	TV/Monitor	**Garrafa**
TV/Monitor	32	1
Garrafa	1	36

5.1.2 RNA (Rede Neural Artificial)

A principal variante apresentada pela RNA ao classificar o grupo de objetos composto por carros e bicicletas foi o tempo necessário para realizar o treinamento em cada uma das duas etapas da avaliação deste classificador, porém, apesar de em ambas as avaliações a RNA ter apresentado resultados de 100% de acertos durante o teste, em apenas 2,88% das execuções com a taxa de aprendizado abaixo do intervalo recomendado os resultados deste classificador atingiram 100% de acertos. Quando os valores utilizados para a taxa de aprendizado estavam dentro do intervalo recomendado, a RNA atingiu 100% de acertos em 55,13% das 945 execuções da avaliação para o grupo composto por carros e bicicletas.

O comportamento apresentado pela RNA durante dois dos treinamentos em que ela obteve 100% de acertos nos teste, ao classificar carros e bicicletas, pode ser observado nos gráficos das figuras 24 e 25, onde são apresentados o erro de validação e o somatório do erro quadrático da rede ao término de cada época. Lembrando que cada época representa uma apresentação completa do conjunto de amostras utilizadas durante o treinamento.

No treinamento apresentado no gráfico da figura 24 foi utilizado um valor baixo para a taxa de aprendizado, como consequência foram necessárias quase 1200 apresentações do conjunto de treinamento para que a RNA fosse capaz de atingir 100% de acertos durante a validação. Já no treinamento apresentado no gráfico da figura 25, quando o valor utilizado para a taxa de aprendizado estava dentro do intervalo recomendado, o número de épocas necessárias

para atingir 100% de acertos durante a validação foi inferior a 100. Conforme o descrito no início do capítulo 5 , as configurações adotadas durante o treinamento da RNA são apresentadas no canto superior direito de cada gráfico.

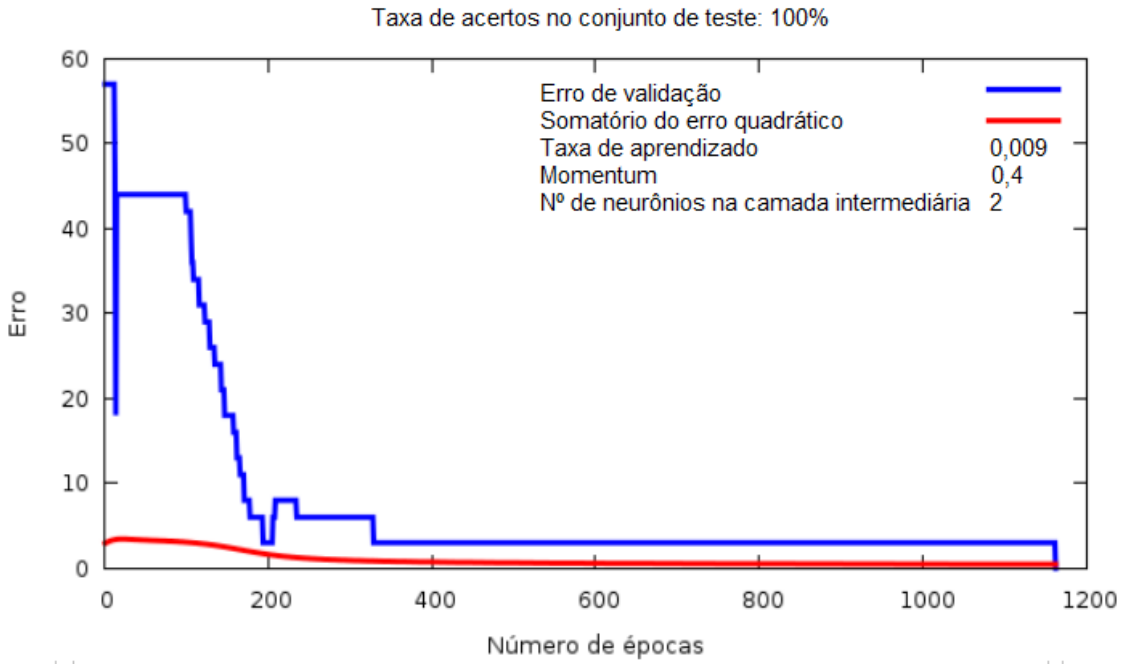


FIGURA 24 - Comportamento da RNA durante o treinamento para reconhecer carros e bicicletas ao utilizar-se um valor abaixo do intervalo recomendado para a taxa de aprendizado.

O motivo da variação no número de épocas necessárias para concluir o treinamento em cada uma das etapas da avaliação da RNA deve-se a velocidade com que os pesos de suas ligações ajustam-se, sendo a taxa de aprendizado a principal influência sobre a variação na velocidade de ajuste da rede. Este comportamento pode ser melhor observado nos gráficos nas figuras 26 e 27, os quais contém somente o somatório do erro quadrático da RNA ao longo dos treinamentos apresentados nos gráficos das figuras 24 e 25 respectivamente.

Ao observar-se os gráficos das figuras 26 e 27 é possível verificar a diferença entre o número de épocas necessárias em cada caso para fazer com que o somatório do erro quadrático da rede caia abaixo de 1. Enquanto são necessárias mais de 200 épocas quando a taxa de aprendizado assume o valor de 0,009, gráfico da figura 26, são necessárias aproximadamente 40 épocas quando o valor utilizado para a taxa de aprendizado é de 0,1, gráfico da figura 27.

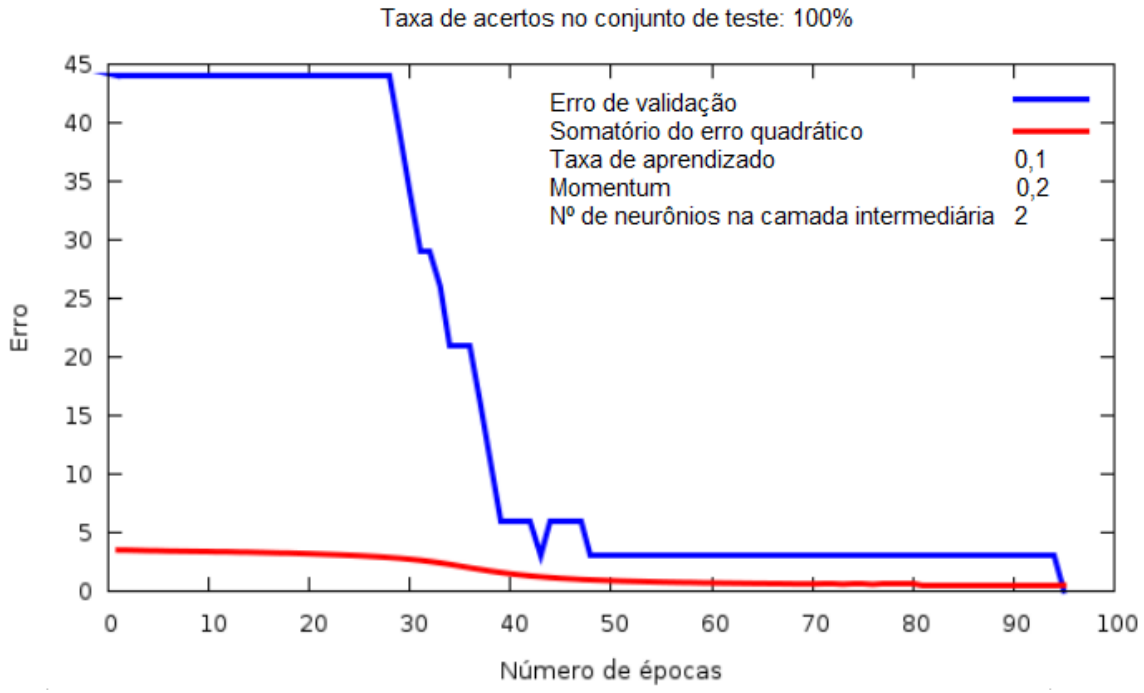


FIGURA 25 - Comportamento da RNA durante o treinamento para reconhecer carros e bicicletas ao utilizar-se um valor dentro do intervalo recomendado para a taxa de aprendizado.

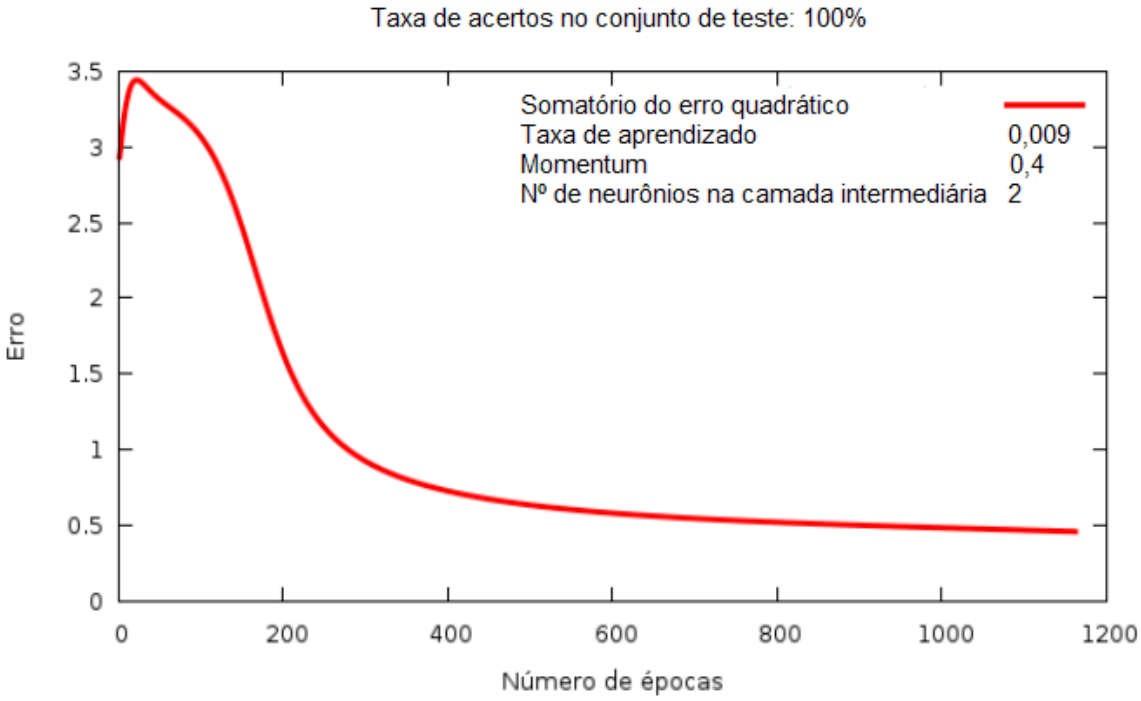


FIGURA 26 - Variação do somatório do erro quadrático apresentada pela RNA ao utilizar-se um valor abaixo do intervalo recomendado para a taxa de aprendizado.

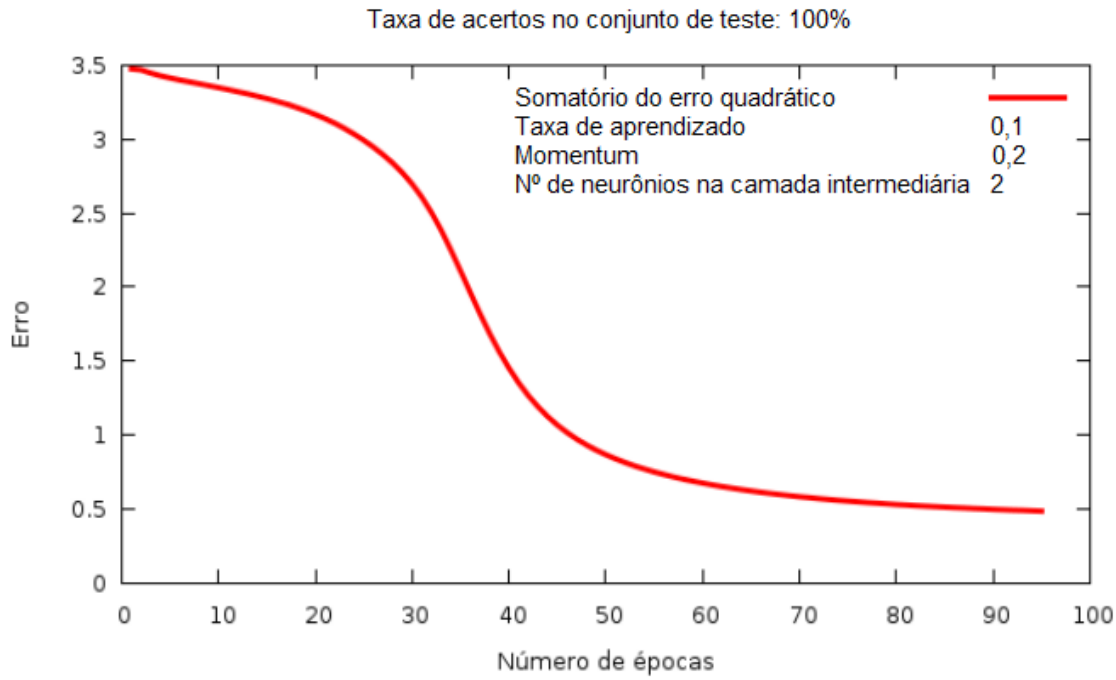


FIGURA 27 - Variação do somatório do erro quadrático apresentada pela RNA ao utilizar-se um valor dentro do intervalo recomendado para a taxa de aprendizado.

Os resultados obtidos com a RNA ao classificar carros e bicicletas também podem ser verificados na tabela 7, onde é apresentada a matriz de confusão para classificador KNN, uma vez que os resultados foram os mesmos.

Quando o segundo grupo de objetos de interesse, composto por TVs/Monitores e garrafas, foi apresentado a RNA sua taxa de acertos igualou-se à obtido pelo KNN, permanecendo como melhor resultado 97.14 % de acertos quando o valor atribuído para a taxa de aprendizado estava dentro do intervalo recomendado. A configuração e o comportamento da RNA durante o treinamento, para uma das execuções em que ela obteve 97,14% de acertos no teste, podem ser verificados no gráfico da figura 28. Utilizando valores abaixo do recomendado para a taxa de aprendizado o melhor resultado obtido pela RNA durante o avaliação foi de 95.58% de acertos. A matriz de confusão deste classificador ao atribuir rótulos para TVs/Monitores e garrafas pode ser verificada na tabela 8, apresentada na seção 5.1.1 .

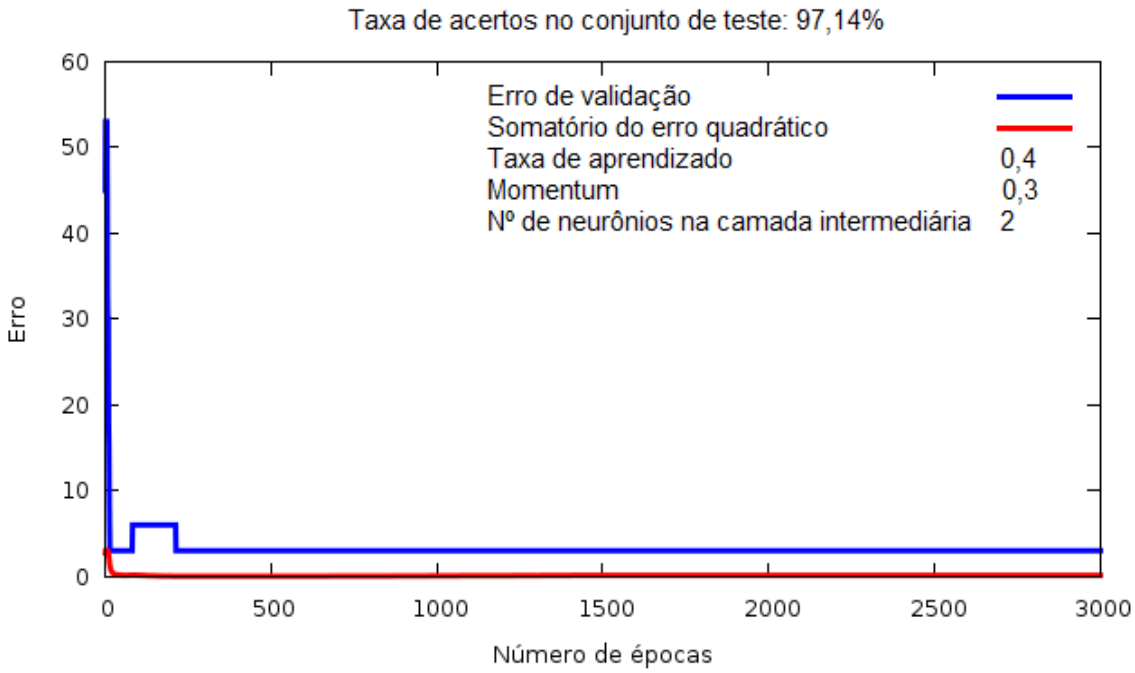


FIGURA 28 - Comportamento da RNA durante um dos treinamentos em que ela obteve a maior taxa de acertos durante o teste, ao reconhecer TVs/Monitores e garrafas.

5.1.3 SVM (Support Vector Machine)

Os resultados obtidos pelo classificador SVM ao reconhecer carros e bicicletas foram de 100% de acertos para todas funções de kernel disponíveis na LIBSVM, conforme o demonstrado no gráfico da figura 29. No entanto, ao classificar TV/Monitor e garrafa, os resultados não ultrapassaram 97,14% de acerto durante o teste, igualando-se aos outros classificadores e apresentando uma variação de menos de 2% entre os melhores resultados obtidos com cada uma das funções de kernel testadas. As configurações adotadas nesta avaliação podem ser verificadas na tabela 6, apresentada na seção 4.2.2. O gráfico da figura 30 apresenta os melhores resultados para cada uma das funções de kernel utilizadas pelo SVM ao classificar TV/Monitor e garrafa, sendo que a posição espacial dos erros de classificação do SVM pode ser verificada no gráfico da figura 19. As configurações para as quais cada função de kernel gerou o melhor resultado podem ser verificadas na tabela 9, nos casos em que mais de uma configura-

ção gerou o melhor resultado, somente a primeira melhor configuração (em ordem crescente dos valores de cada argumento) de cada função é apresentada.

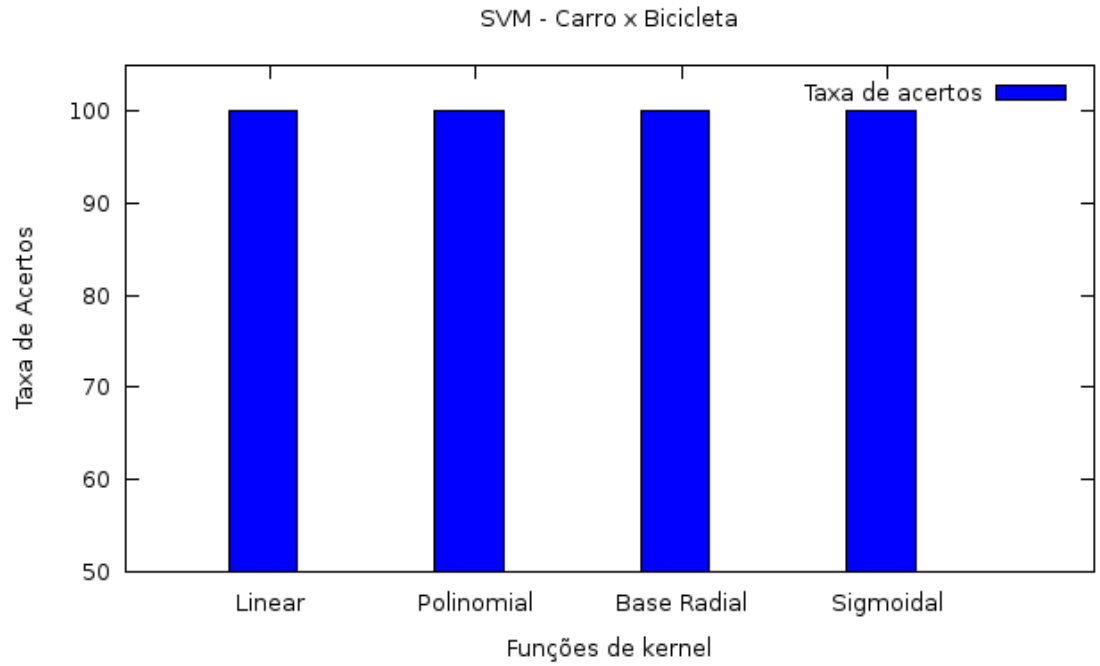


FIGURA 29 - Desempenho do SVM ao classificar carros e bicicletas.

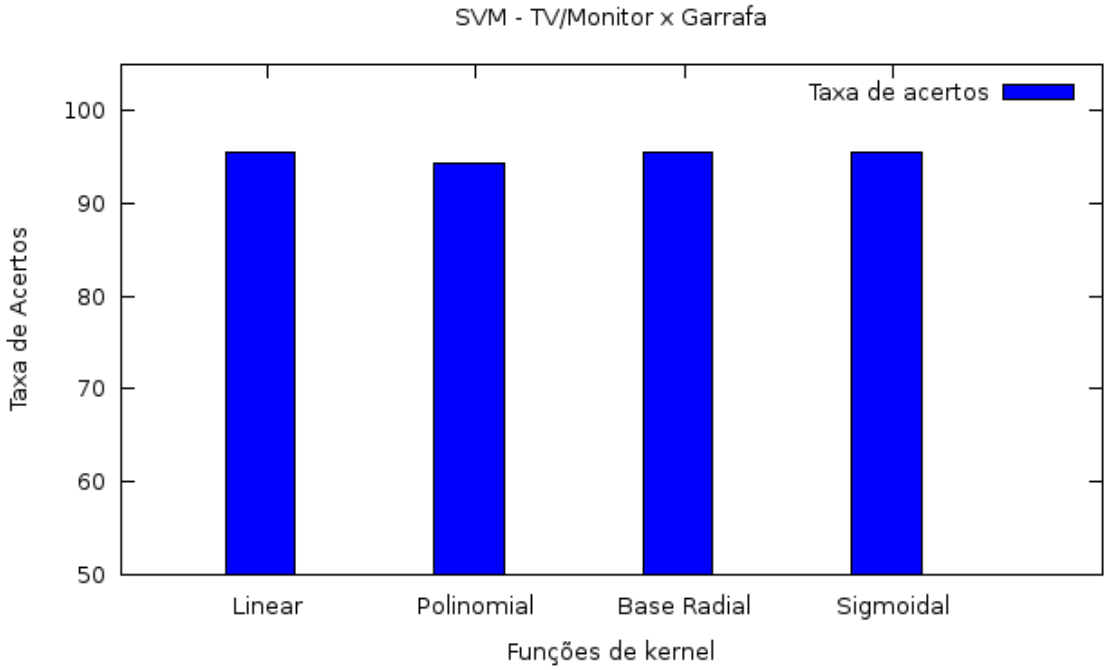


FIGURA 30 - Desempenho do SVM ao classificar TV/Monitor e garrafas.

TABELA 9

Configurações em que o SVM obteve o melhor resultado com cada função de kernel.

Configurações do SVM			
kernel	**Grau**	**Gama**	**Taxa de acertos**
Carro x Bicicleta			
Linear	-	-	100%
Polinomial	2	0,4	100%
Base Radial	-	0,2	100%
Sigmoidal	-	0,2	100%
TV/Monitor x Garrafa			
Linear	-	-	97,14%
Polinomial	2	0,5	97,14%
Base Radial	-	0,3	97,14%
Sigmoidal	-	0,3	95,58%

5.2 Resultados considerando-se os objetos descritos como truncados

Com o aumento na dificuldade de resolução do problema, causado pela inserção dos objetos truncado a base de dados, houve um distanciamento entre os resultados obtidos com cada um dos classificadores, reflexo do aumento no número de ruídos presentes na base de conhecimento e da maior ocorrência de objetos na região de fronteira entre as regiões ocupadas por cada uma das classes. A distribuição das amostras com inclusão das descritas como truncadas pode ser observada nos gráficos apresentados nas figuras 31 e 32, que quando comparados aos gráficos apresentados nas figuras 14 e 15, demonstram de forma mais clara o aumento na dificuldade enfrentada pelos classificadores ao atribuir rótulos aos objetos de interesse.

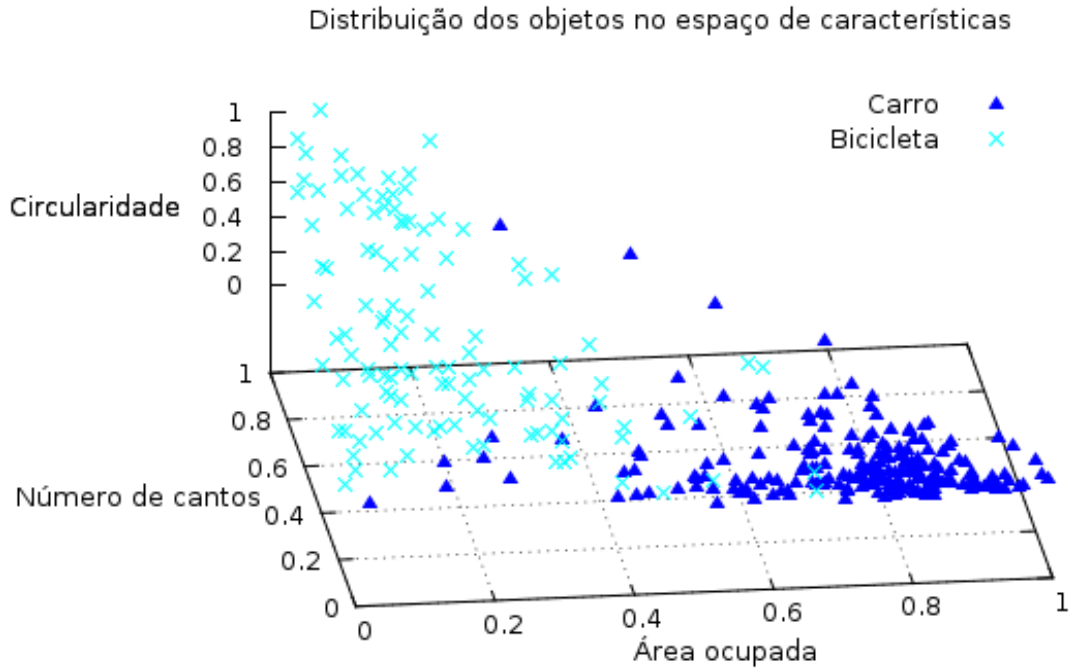


FIGURA 31 - Distribuição da base de conhecimento de carros e bicicletas incluindo os objetos descritos como truncados.

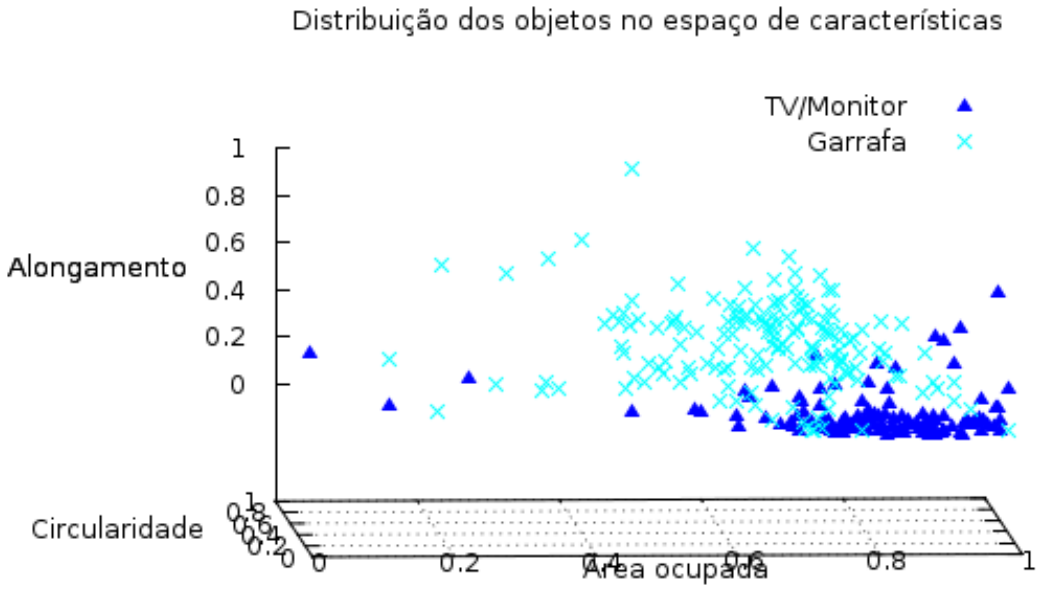


FIGURA 32 - Distribuição da base de conhecimento de TVs/Monitores e garrafas incluindo os objetos descritos como truncados.

5.2.1 KNN (K-Nearest Neighbors)

Mesmo com a inclusão dos objetos truncados ao problema, o classificador KNN manteve suas taxas de acertos acima dos 90% para os dois grupos de objetos, chegando a atingir 95,48% de acertos ao classificar carros e bicicletas. Este resultado foi obtido quando foram considerados oito vizinhos durante o processo de votação, o que pode ser observado na figura 33. A matriz de confusão e a posição espacial dos objetos classificados de forma errada pelo KNN para este caso de teste podem ser observadas na tabela 10 e no gráfico apresentado na figura 34 respectivamente.

TABELA 10

Matriz de confusão do KNN ao classificar carros e bicicletas.

Matriz de confusão do KNN – Carro x Bicicleta		
	Carro	**Bicicleta**
Carro	82	4
Bicicleta	2	45

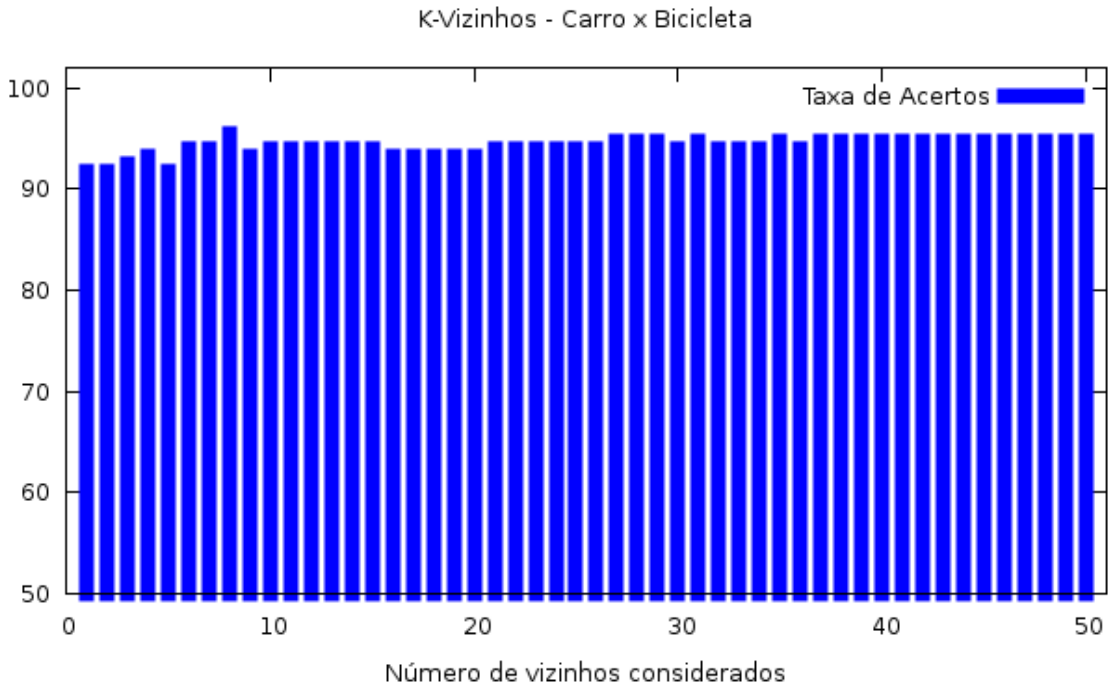


FIGURA 33 - Taxa de acertos do KNN ao classificar carros e bicicletas.

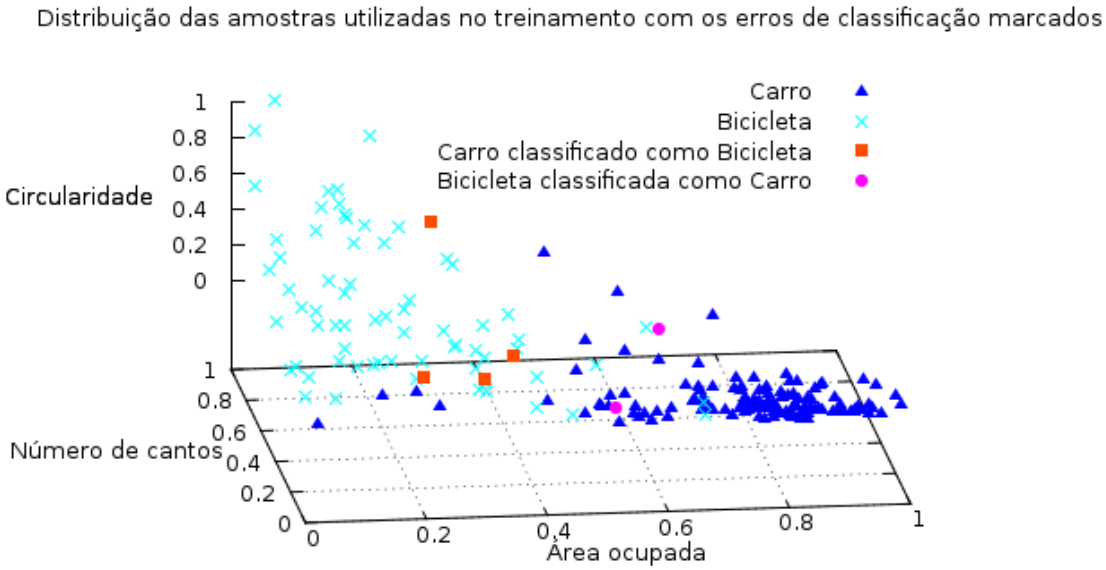


FIGURA 34 - Distribuição das amostras de carros e bicicletas utilizadas durante o treinamento com os erros de classificação do KNN marcados.

Quando o KNN foi utilizado para classificar o segundo grupo de objetos, composto por TV/Monitor e garrafas, os resultados obtidos atingiram 91,37% de acertos quando foram utilizados três, quatro e cinco vizinhos durante o processo de votação, sendo que os resultados obtidos podem ser observados em sua totalidade no gráfico apresentado na figura 35.

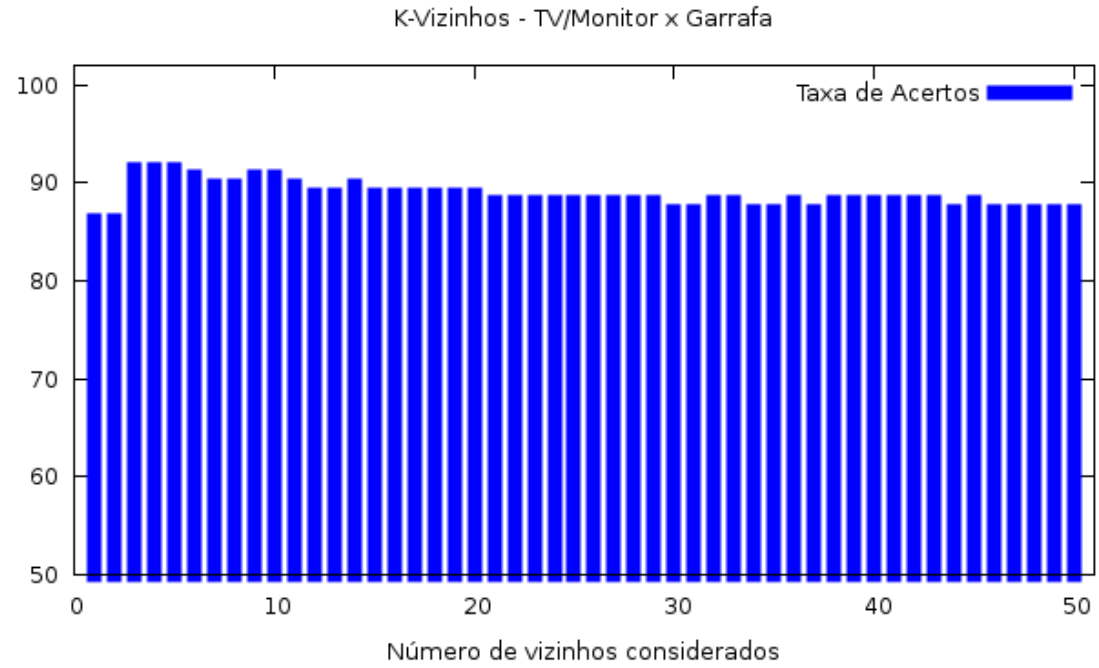


FIGURA 35 - Taxa de acertos do KNN ao classificar TV/Monitor e garrafa.

A matriz de confusão e a posição no espaço de características dos erros cometidos por este classificador ao atribuir rótulos a TVs/Monitores e garrafas podem ser observados na tabela 11 e no gráfico apresentado na figura 36 respectivamente.

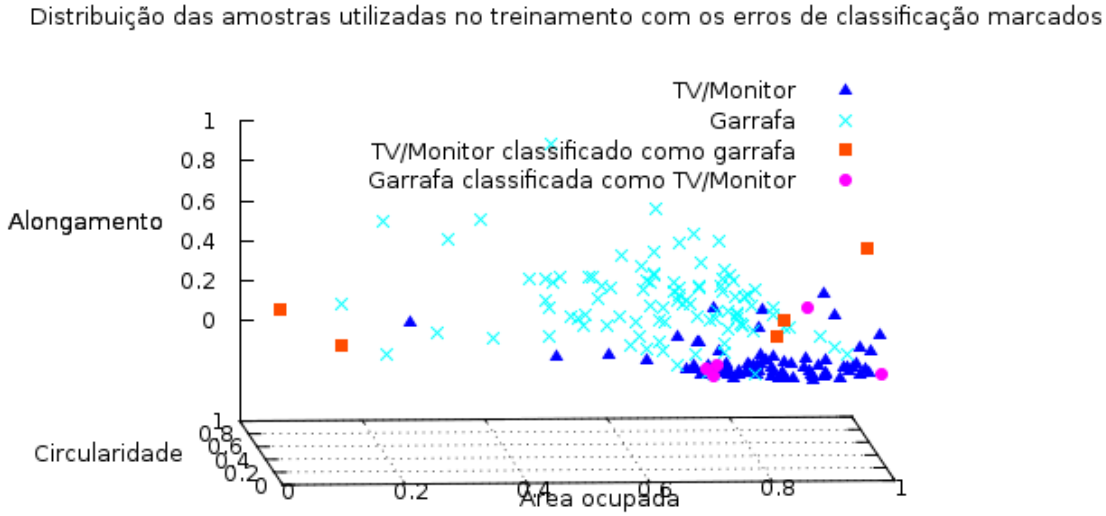


FIGURA 36 - Distribuição das amostras de TV/Monitor e garrafas utilizadas durante o treinamento com os erros de classificação do KNN marcados.

TABELA 11

Matriz de confusão do KNN ao classificar TVs/Monitores e garrafas.

Matriz de confusão do KNN – TV/Monitor x Garrafa		
	TV/Monitor	**Garrafa**
TV/Monitor	46	5
Garrafa	5	60

5.2.2 RNA (Rede Neural Artificial)

Os resultados obtidos com o uso da RNA, tanto ao classificar os objetos do grupo um como os do grupo dois, foram os melhores entre os classificadores analisados quando os objetos truncados foram considerados durante a avaliação, chegando a atingir 97,01% de acertos ao classificar carros e bicicletas e 93.1% de acertos ao classificar TVs/Monitores e garrafas. Em ambos os casos, os resultados foram obtidos durante os testes em que os valores atribuí-

dos para a taxa de aprendizado estavam dentro do intervalo recomendado na literatura, ao passo que quando estes valores estavam abaixo do recomendado os melhores resultados obtidos foram de 95.52% de acetos para carros e bicicletas, e 88.79% de acertos para TV/Monitor e garrafas.

Durante a avaliação da RNA, classificando carros e bicicletas, o melhor resultado ocorreu em 0,74% das execuções com a taxa de aprendizado dentro do intervalo recomendado e em nenhuma das execuções com o valor da taxa de aprendizado abaixo do recomendado. Já quando o segundo grupo de objetos estava sendo avaliado, o melhor resultado foi obtido em 7,08% das execuções com a taxa de aprendizado dentro do intervalo recomendado, contra nenhuma quando o valor estava abaixo do recomendado.

Os erros de classificação cometidos pela RNA estão destacados nos gráficos apresentados nas figuras 37 e 38, sendo que o primeiro gráfico é referente ao grupo de objetos composto por carros e bicicletas, e o segundo referente ao grupos compostos por TVs/Monitor e garrafas. Seguindo esta mesma ordem, as matrizes de confusão da RNA ao classificar os objetos dos dois grupos estão apresentadas nas tabelas 12 e 13, concluindo com a tabela 14, onde são apresentadas as configurações utilizadas pela RNA durante uma das execuções em que ela obteve o melhor resultado, para cada grupo de objetos de interesse, as mesma execuções que geraram os resultados apresentados nos gráficos das figuras 37 e 38.

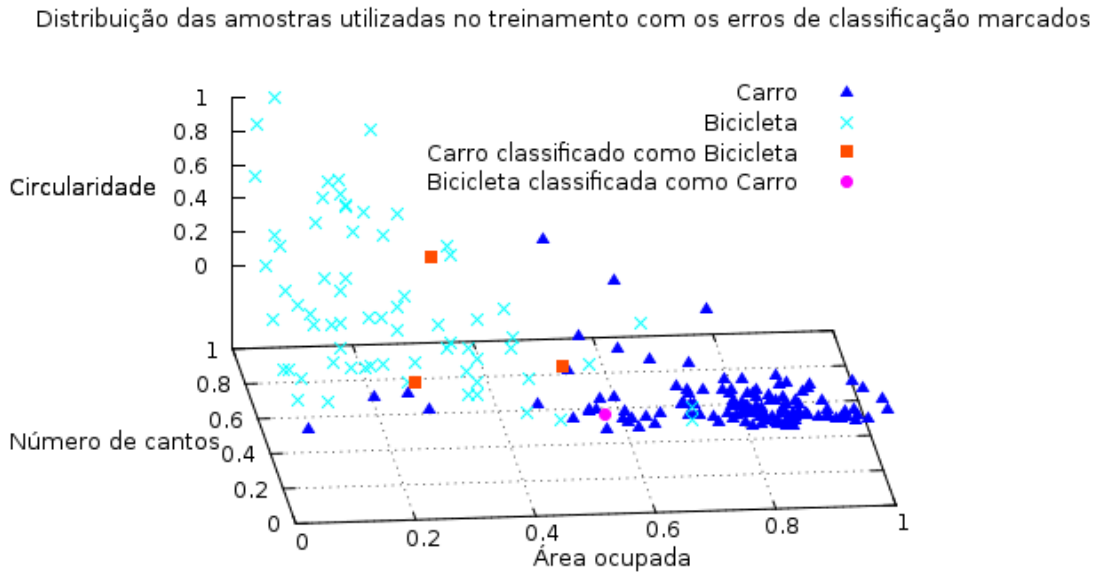


FIGURA 37 - Distribuição das amostras de carros e bicicletas utilizadas durante o treinamento com os erros de classificação do RNA marcados.

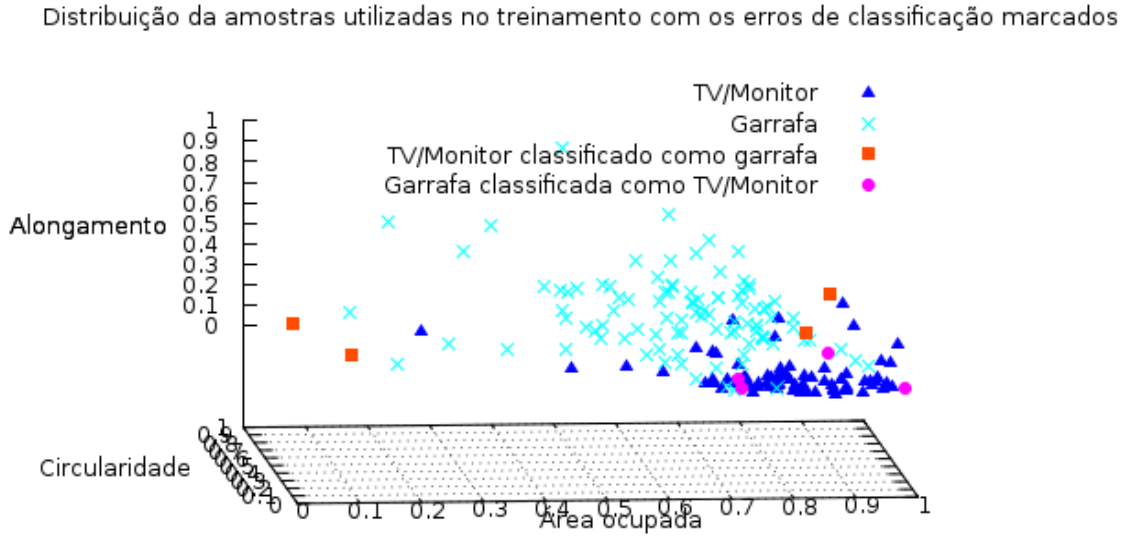


FIGURA 38 - Distribuição das amostras de TVs/Monitores e garrafas utilizadas durante o treinamento com os erros de classificação da RNA marcados.

TABELA 12

Matriz de confusão da RNA ao classificar carros e bicicletas.

Matriz de confusão da RNA – Carro x Bicicleta		
	Carro	**Bicicleta**
Carro	83	3
Bicicleta	1	46

TABELA 13

Matriz de confusão da RNA ao classificar TVs/Monitores e garrafas.

Matriz de confusão da RNA – TV/Monitor x Garrafa		
	TV/Monitor	**Garrafa**
TV/Monitor	47	4
Garrafa	4	61

TABELA 14

Configurações da RNA durante uma das execuções em que ela obteve o melhor resultado para cada grupo de objetos.

Configurações da Rede Neural Artificial			
	Taxa de aprendizado	**Momentum**	**Nº de neurônios**
TV/Monitor x Garrafa	0,35	0,2	3
Carro x Bicicleta	0,2	0,3	3

5.2.3 SVM (Support Vector Machine)

Dentro das configurações adotadas nesta avaliação, o SVM demonstrou-se o classificador mais inadequado a base de dados gerada para o desenvolvimento desta análise, apresentando os piores resultados entre os classificadores aqui abordados. Durante a avaliação de seu desempenho, a taxa de acertos máxima obtida com o SVM ao classificar carros e bicicletas ficou em 94,02% e em 87,06% ao classificar TVs/Monitores e garrafas.

Os resultados obtidos com cada função de kernel ao classificar cada um dos grupo de objetos de interesse podem ser verificados nos gráficos apresentados nas figuras 39 e 41, sendo que a configuração para a qual cada função de kernel obteve o melhor resultado é demonstrada na tabela 17. A posição espacial dos erros em relação aos exemplos utilizados durante o treinamento estão demonstradas nos gráficos das figuras 40 (com a função de kernel de base radial) e 42 (com a função de kernel linear). Na sequência é apresentada a matriz de confusão dos melhores resultados obtidos com este classificador para cada um dos grupos de objetos de interesse, nas tabelas 15 e 16.

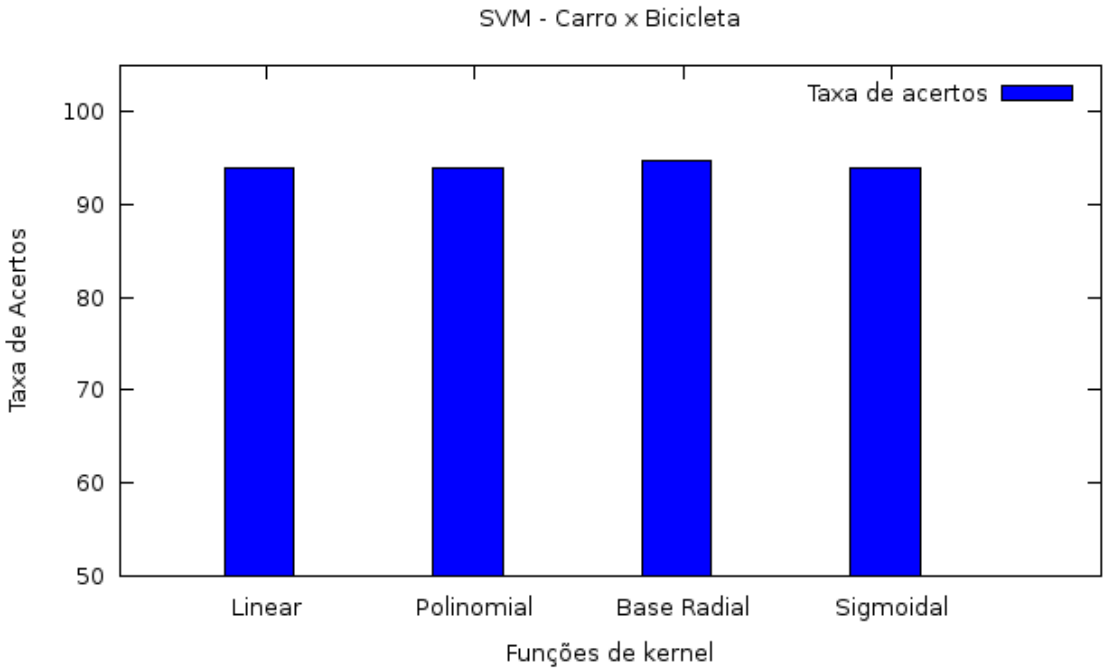


FIGURA 39 - Desempenho do SVM ao classificar carros e bicicletas.

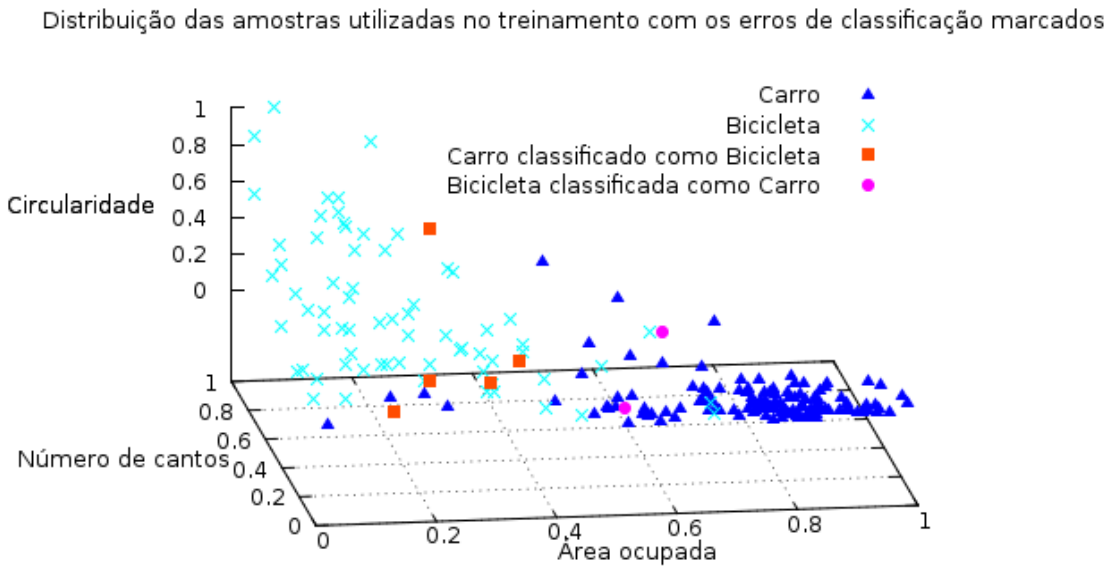


FIGURA 40 - Distribuição das amostras de carros e bicicletas utilizadas durante o treinamento com os erros de classificação do SVM marcados.

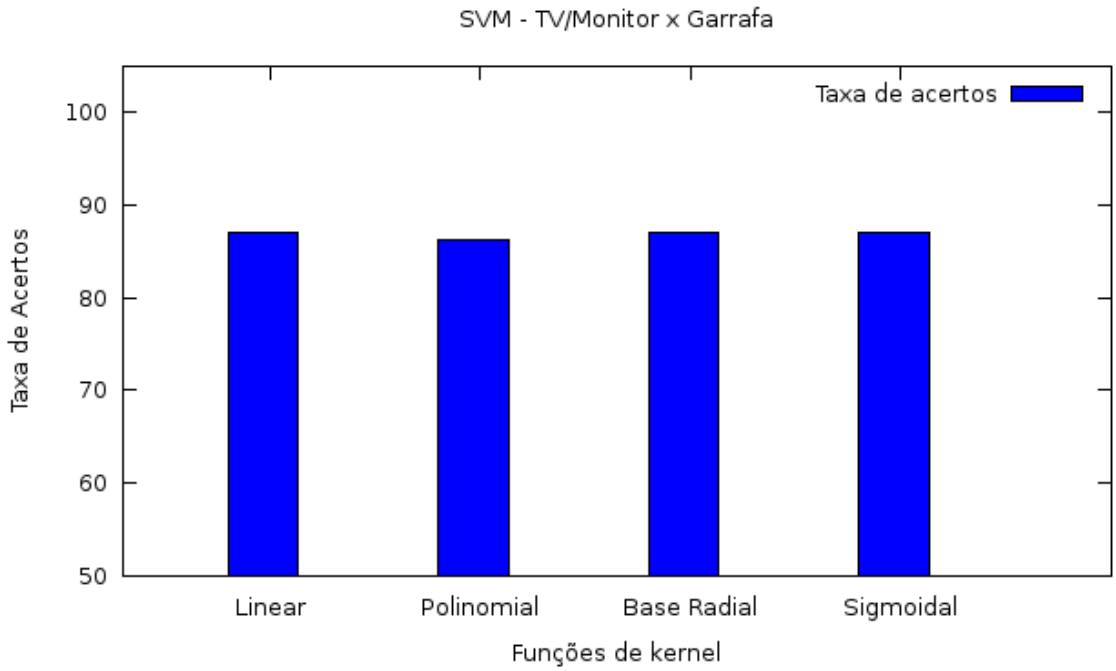


FIGURA 41 - Desempenho do SVM ao classificar TV/Monitor e garrafa.

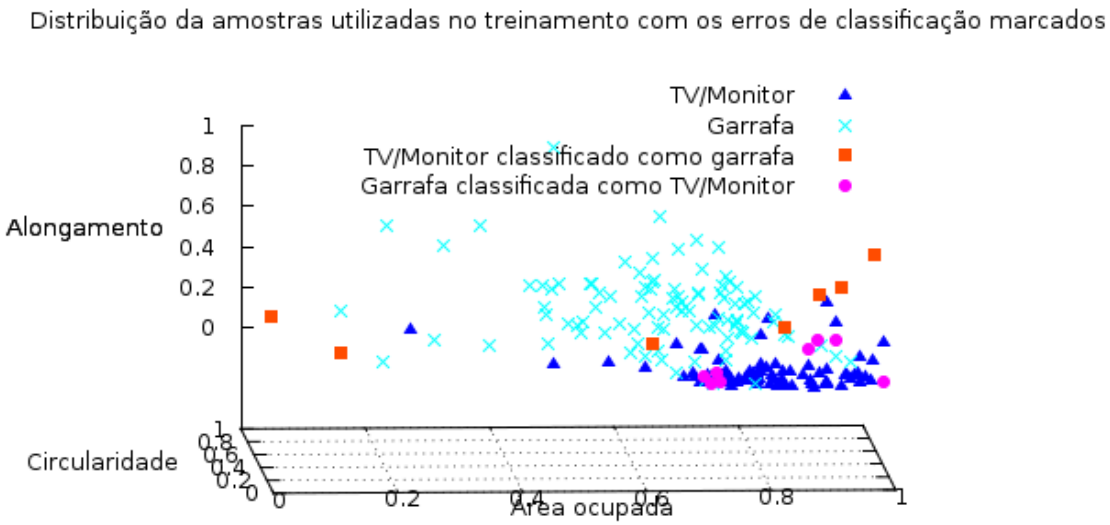


FIGURA 42 - Distribuição das amostras de TVs/Monitores e garrafas utilizadas durante o treinamento com os erros de classificação do SVM marcados.

TABELA 15

Matriz de confusão do SVM ao classificar carros e bicicletas.

Matriz de confusão do SVM – Carro x Bicicleta		
	Carro	**Bicicleta**
Carro	81	5
Bicicleta	2	45

TABELA 16

Matriz de confusão do SVM ao classificar TVs/Monitores e garrafas.

Matriz de confusão do SVM – TV/Monitor x Garrafa		
	TV/Monitor	**Garrafa**
TV/Monitor	44	7
Garrafa	8	57

TABELA 17

Configurações em que o SVM obteve o melhor resultado com cada função de kernel.

Configurações do SVM			
kernel	**Grau**	**Gama**	**Taxa de acertos**
Carro x Bicicleta			
Linear	-	-	93,98%
Polinomial	2	0,4	93,98%
Base Radial	-	0,3	94,73%
Sigmoidal	-	0,2	93,98%
TV/Monitor x Garrafa			
Linear	-	-	87,06%
Polinomial	2	0,5	86,20%
Base Radial	-	0,3	87,06%
Sigmoidal	-	0,5	87,06%

6 CONSIDERAÇÕES FINAIS

Ao longo do desenvolvimento deste trabalho foram analisados experimentalmente os resultados de três métodos clássicos de aprendizado de máquina quando aplicados a um problema de visão computacional voltado ao reconhecimento e classificação de objetos, contribuindo com uma revisão bibliográfica sobre algoritmos de aprendizado de máquina e a análise do desempenho destes algoritmos quando aplicados a um problema de visão computacional voltado ao reconhecimento de objetos. Dentro do contexto analisado, foi possível verificar o impacto causado na distribuição espacial dos objetos de interesse pela seleção de diferentes grupos de informações para caracterizá-los, distribuição que está diretamente ligada aos resultados obtidos pelos classificadores e demonstra a importância da seleção destas informações para o desempenho final deste tipo de sistema. Outra questão interessante que foi levantada ao longo deste trabalho diz respeito as falhas decorrentes de etapas anteriores a extração de características, as quais podem ter impactos negativos nos resultados de cada classificador.

Com a análise dos erros cometidos pelos classificadores também foram expostas as limitações que este tipo de sistema enfrenta ao tentar identificar objetos através da representação em 2D de um mundo 3D, presente nas imagens digitalizadas. Um exemplo dessas limitações foi demonstrado nas figuras 18 e 19, onde devido a posição em que o objeto de interesse encontrava-se durante o processo de aquisição da imagem, os classificadores não foram capazes de identificar corretamente a garrafa através das informações utilizadas para caraterizá-la. Ainda com relação a análise dos erros cometidos pelos classificadores, foi possível verificar como o desempenho de cada um deles pode ser influenciado de forma significativa ao assumirem diferentes configurações.

De modo geral, os resultados alcançados pelos classificadores no contexto deste trabalho foram bons, falhando apenas ao classificar alguns dos objetos que não estavam representados

em sua totalidade na imagem ou quando estes objetos tiveram suas características fortemente alteradas, devido ao ângulo que encontravam-se em relação a câmera durante a aquisição da imagem ou a falhas na etapa de segmentação. Porém, não é possível afirmar que não existam configurações para as quais qualquer um deles seja capaz de atingir maiores taxas de acertos para o problema em estudo. Além disso, é possível que existam no conjunto de amostras utilizas exemplos mais representativos do que os utilizados durante o treinamento, visto que os conjuntos de treinamento e testes foram selecionados de forma aleatória. O que os resultados demonstram é uma aproximação do comportamento de cada uma das técnicas abordadas quando aplicadas a este problema, onde, devido aos resultados obtidos, o MLP (Multi Layer Perceptron) demonstrou-se como melhor alternativa à resolução do problema utilizado para o caso em estudo.

Tendo em vista a complexidade do tema abordado e vasta gama de alternativas a serem seguidas visando a otimização de um sistema de visão computacional, várias linhas de pesquisa podem vir a ser foco de trabalhos futuros, entre elas estão:

- Verificar o comportamento de outras técnicas de aprendizado de máquina quando aplicadas a mesma base de dados, onde pode ser interessante utilizar-se uma técnica capaz de atribuir um grau de certeza a classificação de cada objeto.

- Pesquisar, incluir e combinar outras informações capazes de caracterizar os objetos de interesse.

- Baseando-se nos resultados aqui obtidos, selecionar apenas uma das técnicas e otimizá-la em busca de uma configuração que possibilite uma maior taxa de acertos durante a classificação.

- Aumentar a dificuldade do problema de classificação com a inclusão de novos objetos de interesse. Onde uma alternativa seria tentar reconhecer os sinais de trânsito brasileiros.

7 REFERÊNCIAS BIBLIOGRÁFICAS

AMORIM, Dinani Gomes; **Redes ART com Categorias Internas de Geometria Irregular**. Santiago de Compostela: USC, 2007. 320p. Tese (Doutorado) - Faculdade de Física, Universidade de Santiago de Compostela, Santiago de Compostela, 2007.

BOSER, Bernhard E.; GUYON, Isabelle M.; VAPNIK, Vladimir N. **A Training Algorithm for Optimal Margin Classifiers**. AT&T Bell Laboratories, 1992. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.103.1189&rep=rep1&type=pdf> Acesso em: 10 out. 2010.

BOSER, Bernhard E.; GUYON, Isabelle M.; VAPNIK, Vladimir N. **Automatic Capacity Tuning os Very Large VC-dimension Classifiers**. AT&T Bell Laboratories, 1993. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.17.7215&rep=rep1&type=pdf> Acesso em: 10 out. 2010.

CARDOSO, Fátima. A Inteligência do CHIP: softwares que imitam o raciocínio humano. **Super Interessante**. São Paulo, v. 78, a. 8, n. 3, mar./94.

CASTRO, Leandro Nunes de; **Fundamentals of Natural Computing**: Basic Concepts, Algorithms, and Applications. 1. ed. Chapman & Hall/CRC, 2006.

CARVALHO, Bernardo. **O estado da arte em métodos para reconhecimento de padrões: Support Vector Machine**. Universidade Federal de Minas Gerais - UFMG, 2005. Disponível em: <http://www.cpdee.ufmg.br/~bpenna/bpenna_SUCESU_2005.pdf> Acesso em: 09 out. 2010

CONCI, Aura; AZEVEDO, Eduardo; LETA, Fabiana R.. **Computação Gráfica: Teoria e Prática**. 2. ed. CAMPUS, 2008.

GONZALEZ, Rafael C.; WOODS, Richard E.. **Processamento de Imagens Digitais**. 1.ed. Blucher, 2000.

GRAVES, Mark; BATCHELOR, Bruce G.. **Machine Vision For The Inspection Of Natural Products**. 1. ed. SPRINGER VERLAG NY, 2003.

HAYKIN, Simon S.; **Redes Neurais**: Princípios e prática. 2. ed. Bookman, 2001.

HARRIS, C.; STEPHENS, M. **A combined corner and edge detector**. Proceedings of Fourth Alvey Vision Conference, 1988. 189-192p.

KECMAN, Vojislau. **Learning And Soft Computing With Support Vector Machines, Neural Networks, and Fuzzy Logic Models**. 1. ed. MIT PRESS, 2001.

LORENA, Ana Carolina; CARVALHO, André C. P. L. F.. **Uma Introdução às *Support Vector Machines***. RITA, Porto Alegre, v. 14, a. 18, n. 2, 2007.

MARSLAND, Stephen; **Machine Learning**: An Algorithmic Perspective. 1. ed. Chapman & Hall/CRC Press, 2008.

MASSAD, Eduardo; ORTEGA, Neli Regina S.; SLIVEIRA, Paulo S. P.; MENEZES, Renée X. de; **Métodos Quantitativos em Medicina**. 1. ed. Manole, 2004.

MITCHELL, Tom M.. **Machine Learning**. 1.ed. McGraw-Hill, 1997.

MORAVEC, Hans P. **Obstacle avoidance and navigation in the real world by a seeing a robot rover.** Stanford, 1980, 170p. Tese (Tese Ph.D). Carnegie-Mellon University, Robotics Institute, 1980.

REZENDE, Solange Oliveira. **Sistemas Inteligentes:** Fundamentos e Aplicações. 1. ed. Manole, 2002.

ROCHA, Anderson de Rezende; **Classificadores e Aprendizado em Processamento de Imagens e Visão Computacional**. Campinas: Unicamp, 2009. 182p. Tese (Doutorado) - Programa de Pós-Graduação: Doutorado em Ciência da Computação, Universidade Estadual de Campinas, Campinas, 2009.

SANTOS, Roosevelt de Lara Junior; APARECIDO, Edson. **Extração automática de pontos de apoio para integração de imagens aéreas digitais e dados de perfilamento laser aerotransportado**. Curitiba: UFPR, 2007, 132p. Tese (Doutorado) – Programa de Pós-Graduação em Ciências Geodésicas, Universidade Federal do Paraná, Curitiba, 2007.

SAMPAIO, José Eduardo Dias. **Detecção e Reconhecimento de Sinais de Trânsito em Tempo Real**. Portugal: UMinho, 2010, 111p. Tese (Mestrado) – Escola de Engenharia, Universidade do Minho, Portugal, 2010.

SONKA, Milan; HLAVAC, Vaclav; BOYLE, Roger. **Image Processing, Analysis, and Machine Vision**. 2. ed. Thomson Learning, 1998.

WVC, 4., 2008, Bauru. **IV Workshop De Visão Computacional**. Bauru: UNESP, 2008.