UNIVERSIDADE ESTADUAL DO CEARÁ

CENTRO DE CIÊNCIAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MESTRADO ACADÊMICO EM CIÊNCIA DA COMPUTAÇÃO

FELIPE JOSÉ AGUIAR MAIA

DETECÇÃO DA PRESENÇA DO CAPACETE DE SEGURANÇA DO MOTOCICLISTA EM IMAGENS DE VIAS DO TRÂNSITO

FORTALEZA – CEARÁ

2015

FELIPE JOSÉ AGUIAR MAIA

DETECÇÃO DA PRESENÇA DO CAPACETE DE SEGURANÇA DO MOTOCICLISTA EM IMAGENS DE VIAS DO TRÂNSITO

Dissertação apresentada ao Curso de Mestrado Acadêmico em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação

Orientadores: José Everardo Bessa Maia

Thelmo Pontes de Araujo

FORTALEZA – CEARÁ

2015

Deve ser gerada através do preenchimento do Formulário Eletrônico de Elaboração da Ficha Catalográfica, disponível no link: http://www.uece.br/biblioteca/index.php/entrega-de-trabalho.

X000x Sobrenome, Nome do 1º autor. (citado na folha de rosto)
Título principal: subtítulo./Nome completo do 1º autor, Nome completo do 2º autor, Nome completo do 3º autor; orientação [de]. – Local: ano.
Nº de folhas.: il.(se houver ilustração); 30 cm.

Inclui bibliografias: f.(nº da folha em que se encontra)
Trabalho de Conclusão de Curso (Graduação em) – Universidade Estadual do Ceará – (UECE).

1. Assunto. 2. Assunto. 3. Assunto. I. Sobrenome, Nome do 2º autor. II. Sobrenome, Nome do 3º autor. III. Sobrenome, Nome do orientador (orient.). IV. Universidade Estadual do Ceará – UECE. V. Título.

CDU

FELIPE JOSÉ AGUIAR MAIA

DETECÇÃO DA PRESENÇA DO CAPACETE DE SEGURANÇA DO MOTOCICLISTA EM IMAGENS DE VIAS DO TRÂNSITO

Dissertação apresentada ao Curso de Mestrado Acadêmico em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação

Aprovada em: 29 de julho de 2015

BANCA EXAMINADORA

Prof. Dr. José Everardo Bessa Maia (Orientador)
Universidade Estadual do Ceará – UECE

Prof. Thelmo Pontes de Araujo, PhD. (Co-Orientador)
Universidade Estadual do Ceará – UECE

Profa. Dra. Fátima Nelsizeuma Sombra de Medeiros
Universidade Federal de Fortaleza - UFC

Prof. Francisco Carlos de Mattos Brito Oliveira, PhD.
Universidade Estadual do Ceará – UECE

À minha família pelo apoio incondicional e à Cristina por me mostrar o que é o Amor.

AGRADECIMENTOS

Aos meus pais Regina e Júnior e à minha irmã Isabelle pelo apoio, paciência e bons conselhos ao longo de toda a minha jornada.

Aos meus amigos com quem criei uma forte ligação ao longo do curso. Em especial Robson Oliveira, Igor Brasil, Robert Marinho, Anderson Couto e Israel Santos.

À minha namorada Cristina que me trouxe inspiração e não saiu do meu lado durante toda a realização deste trabalho, me dando forças quando eu pensava estar esgotado.

À Capes pelo apoio no início do curso e à empresa Atlanta Tecnologia e seus colaboradores por apoiarem a minha pesquisa e fornecerem parte da base de dados utilizada.

A todos os professores que me fizeram crescer ao longo do curso. Em especial, aos professores Mariela Inés Cortés, Thelmo Pontes de Araujo e José Everardo Bessa Maia pelas oportunidades.

A todos aqueles que de alguma forma fizeram parte da minha vida e me transfomaram no que sou hoje.

“Reality is frequently inaccurate. ”

(Douglas Adams, The Restaurant at the End of the Universe)

RESUMO

Este trabalho é uma contribuição para a fiscalização do uso de capacetes por motociclistas através de um sistema capaz de detectar um motociclista em uma imagem de via de tráfego e em seguida determinar se este condutor está utilizando ou não o capacete de segurança. O sistema proposto realiza essa tarefa por meio da aplicação de Modelos de Aparência Ativa para extrair características da imagem do condutor a serem utilizadas por um classificador supervisionado. A acurácia satisfatória obtida na discriminação do uso do capacete de segurança recomenda a utilização de características dos Modelos de Aparência Ativa para esta tarefa.

Palavras-chave: Detecção de Capacetes. Modelos de Aparência Ativa. Modelos de Forma Ativa. Reconhecimento de Padrões.

ABSTRACT

This work is a contribution to the monitoring of the use of helmets by motorcyclists with a system capable of detecting a bike rider in an image and then determining whether he is wearing or not the safety helmet. The system performs this task by using Active Appearance Models to collect image features of a motorcyclist to be used in a supervised classifier. The satisfactory accuracy obtained in the discrimination of the use of the helmet indicates the suitability of the Active Appearance Models to this task.

Keywords: Helmet Detection. Active Appearance Models. Active Shape Models. Pattern Recognition.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ALGORITMOS

SUMÁRIO

1 INTRODUÇÃO

O desenvolvimento de algoritmos para detectar múltiplas instâncias de uma entidade em uma imagem e reconhecer alguma propriedade específica nestas instâncias, em situações muito gerais, continua sendo um problema desafiador para a comunidade de pesquisa. As condições de captura da imagem (ruído, iluminação), as propriedades amorfas dos objetos, o comportamento dinâmico do *background* estão entre os fatores complicadores do problema. Este trabalho aborda uma instância deste problema, qual seja, detectar a presença de motociclistas em imagens de vias de trânsito e reconhecer o porte ou não do capacete de segurança.

Nos últimos anos, tem-se observado um aumento considerável na frota de motocicletas do Brasil. De acordo com o Departamento Nacional do Trânsito (DENATRAN, 2015), em maio de 2015 o Brasil possuía uma frota de 19.667.658 motocicletas, 3.707.360 motonetas e 167.018 veículos ciclomotores. O que representa $26,63\%$ de toda a frota nacional.

Esse grande número de veículos possui forte impacto nas estatísticas de acidentes, principalmente levando-se em consideração que os condutores de veículos de duas rodas são mais suscetíveis à lesões em acidentes do que os condutores de veículos maiores. De acordo com o Boletim Estatístico do Seguro de Danos Pessoais Causados por Veículos Automotores de Vias Terrestres (DPVAT, 2014), entre janeiro e dezembro de 2014 foram pagas 580.063 indenizações por conta de acidentes com motociclistas, 76% de todas as indenizações. Dessas, 82% foram indenizações causadas por invalidez e 4% causadas por morte.

Com o objetivo de reduzir o efeito causado pelo grande número de acidentes, considerado um problema de saúde pública, pesquisas recentes da área de Processamento de Imagens propõem sistemas para auxiliar na fiscalização dos motociclistas.

Sendo o capacete o principal item de segurança do motociclista, a detecção automática da presença do mesmo pode ser utilizada pelos órgãos fiscalizadores, juntamente com ações educativas tais como multas ou advertências, como uma medida para diminuição dos casos de acidentes graves.

Alguns trabalhos já abordaram o problema da localização do capacete utilizando monitoramento por vídeo, como em (SILVA et al., 2013b), (SILVA et al., 2013a), (KU et al., 2008) e (CHIVERTON, 2012). Entretanto, para ganhar sinergia com outros sistemas de fiscalização baseados em fotografia, como por exemplo a detecção do avanço de sinal vermelho, é importante abordar a detecção de motociclistas e a presença do capacete em uma única imagem. Nesse caso, o problema se apresenta bastante diferente e, que seja do conhecimento dos autores,

não foi ainda abordado na literatura.

Dessa forma, este trabalho se propõe a realizar a detecção da presença de capacetes a partir de fotos frontais de motociclistas. Através da utilização de fotos frontais, pode-se aproveitar caraterísticas de textura presentes na face dos motociclistas. Além disso, são utilizadas fotos estáticas em vez de vídeo devido a possíveis limitações nos equipamentos em que o sistema pode vir a ser executado. Normalmente, nesses equipamentos, a câmera é acionada apenas quando o veículo é detectado por meio de outros sensores, como um laço indutivo sob o asfalto.

Para a aquisição de características a partir da imagens dos motociclistas, este trabalho utiliza os Modelos de Aparência Ativa (AAM - *Active Appearence Models*) propostos em (COOTES et al., 1998a) que possuem a capacidade de modelar a forma e a aparência de objetos. Dessa maneira os parâmetros de descrição dos modelos podem ser usados para discriminar imagens de motociclistas com e sem o capacete de segurança.

Na Figura 1 é possível observar exemplos de imagens que o sistema de detecção de capacetes deve receber como entrada.

Figura 1 – Imagens de motociclistas sem (acima) e com (abaixo) capacetes após serem detectados na imagem da via de trânsito.

Fonte: Elaborado pelo autor

Sendo assim, o sistema deve ser capaz de localizar o motociclista na imagem e em seguida inferir se esse motociclista está utilizando o capacete ou não. Embora neste trabalho seja também implementada a detecção do motociclista, a ênfase é a identificação da presença ou ausência do capacete de segurança.

1.1 OBJETIVOS

Objetivo Geral

O objetivo geral deste trabalho é construir um algoritmo para detectar a presença de motociclistas em imagens de vias de trânsito e reconhecer o uso ou não do capacete de segurança.

Objetivos Específicos

Mais especificamente, este trabalho visa a:

a) Detectar o motociclista e localizar a região da cabeça em imagens de vias de trânsito;
b) Utilizar os Modelos de Aparência Ativa (AAM) para extrair características da região detectada;
c) Determinar a presença do capacete de segurança;
d) Aplicar e avaliar o uso de Modelos de Aparência Ativa (AAM) em problemas de classificação.

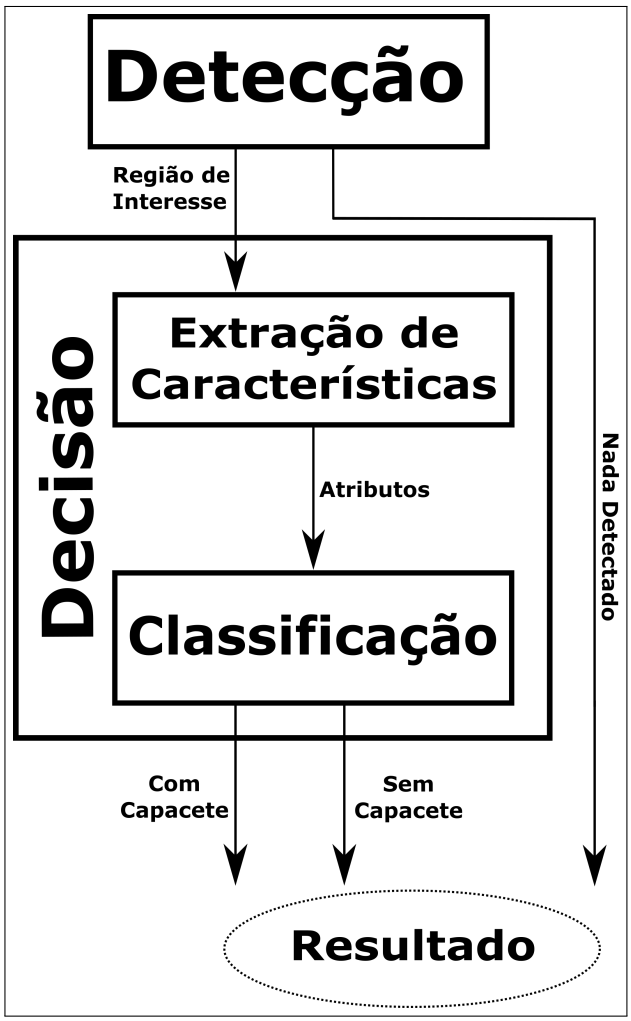
1.2 METODOLOGIA

Para resolver o problema de determinar se o motociclista está utilizando o capacete no momento da captura da imagem, elaborou-se um procedimento dividido em duas etapas principais: detecção e decisão, sendo que a etapa de decisão é dividida em extração de características e classificação. Na Figura 2 é possível observar o diagrama do procedimento. Assim, após a localização da região onde se encontra a cabeça do motociclista, é possível iniciar a aquisição de características que serão usadas pelo algoritmo que determina se este motociclista está ou não utilizando o capacete.

Durante a etapa de detecção, um algoritmo de localização realiza a busca pela região de interesse contendo a cabeça do motociclista. Neste trabalho, o algoritmo de detecção de objetos Viola-Jones (VIOLA; JONES, 2001a) realiza essa tarefa. Essa etapa pode ou não detectar cabeças de motociclistas na imagem. Caso regiões de interesse sejam detectadas, essas regiões são então fornecidas para o módulo de decisão. Caso não haja detecções, as etapas seguintes não são executadas e o procedimento proposto é finalizado.

O módulo de decisão realiza a extração de características em uma região de interesse

Figura 2 – Diagrama contendo as etapas do procedimento.



Fonte: Elaborado pelo autor

com o objetivo de obter dados que possam ser usados pelo classificador para determinar se o condutor está usando capacete. A extração de características é feita através da aplicação dos Modelos de Aparência Ativa (COOTES et al., 1998a). A execução do algoritmo de aproximação do AAM resulta em 4 parâmetros de pose (posição em x e y, escala e rotação) e um vetor c contendo os parâmetros do modelo combinando informações de forma e de textura.

Os parâmetros de pose são pouco relevantes em termos de descrição da imagem, já que eles informam apenas o posicionamento da região aproximada nas coordenadas da imagem. Porém, o vetor de parâmetros c possui excelentes descritores da região onde o modelo foi aplicado, pois seus valores tendem a variar significativamente entre exemplos das duas classes possíveis. Para a etapa de classificação, foram feitos testes com Máquinas de Vetores de Suporte (CORTES; VAPNIK, 1995) além da rede neural Perceptron de Múltiplas Camadas (RUMELHART et al., 1986).

No procedimento proposto nesta dissertação são utilizadas imagens estáticas em vez de uma sequência de vídeo. Por esse motivo, é necessário utilizar um outro detector para realizar a primeira etapa. Dessa forma, foi escolhida a técnica de localização de objetos proposta por Viola e Jones em (VIOLA; JONES, 2001a), utilizando um classificador em cascata baseado em AdaBoost (FREUND; SCHAPIRE, 1997) e características Haar-*like* (PAPAGEORGIOU et al., 1998).

O detector de Viola-Jones é um dos mais utilizado na literatura e pode ser treinado para detectar qualquer objeto. Este detector deve ser treinado para realizar detecções tanto com quanto sem capacetes. A partir de testes, percebeu-se que é possível realizar esse treinamento de maneira que o classificador seja genérico o bastante para realizar os dois tipos de detecção, mas ainda assim manter uma taxa de falso negativo aceitável. A Figura 3 mostra exemplos da saída do detector de Viola-Jones para imagens com e sem o capacete. No Apêndice A o leitor pode encontrar uma revisão da fundamentação teórica do detector de Viola-Jones.

Figura 3 – Exemplo de execução de um único detector para os casos com e sem capacete.

Fonte: Elaborado pelo autor

Após a determinação da região de interesse, inicia-se o processo de aquisição de caraterísticas. A utilização de modelos deformáveis funciona muito bem para essa aplicação. Em (COOTES et al., 1998a) é mostrado como a matriz de conhecimento a priori consegue realizar os ajustes necessários apenas quando o deslocamento é muito pequeno, de poucos pixels. Isso ocorre porque o ajuste é feito através de uma regressão linear e foi observado que a relação entre o erro e o ajuste só se mantém linear quando o deslocamento dos pixels é pequeno.

Após a execução do algoritmo de aproximação do AAM, como pode ser observado nas Figuras 4 e 5, o modelo consegue reproduzir com sucesso a imagem original do capacete e da cabeça sem o mesmo. Como o modelo sintetiza essas imagens utilizando um número bastante reduzido de atributos, esses atributos podem ser utilizados como características dessas imagens que foram sintetizadas.

Para evitar problemas com possíveis falsos positivos detectados pelo classificador em

Figura 4 – Exemplo de convergência para o caso com capacete.

Fonte: Elaborado pelo autor

Figura 5 – Exemplo de convergência para o caso sem capacete.

Fonte: Elaborado pelo autor

cascata, uma verificação pode feita com os parâmetros do modelo logo após a sua aproximação. Esse passo ocorre quando mais de um objeto é detectado em uma única imagem, como pode ser visto na Figura 6. Nesse caso é feita uma comparação dos dois vetores de parâmetros encontrados com o objetivo de identificar qual possui maior diferença em relação à média do modelo. Aquele vetor que se encontra mais próximo da aparência média do modelo é escolhido como o vetor de atributos dessa imagem.

Figura 6 – Exemplo de detecção dupla.

Fonte: Elaborado pelo autor

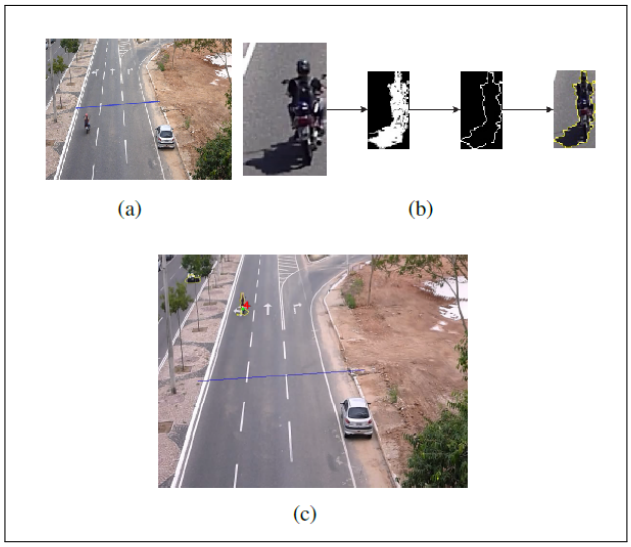
Como as imagens geradas para os casos com ou sem capacete são extremamente diferentes, o esperado é que seja fácil realizar a separação entre os dois casos, pois os atributos do modelo devem ser distintos. Para realizar essa classificação foram testados quatro classificadores: SVM com três funções Kernel diferentes e a rede neural MLP. Um referencial teórico a respeito desses dois algoritmos de reconhecimento de padrões pode ser encontrado nos Apêndices B e C.

Independente do classificador utilizado, a entrada desse classificador é o vetor de atributos resultante da aproximação do Modelo de Aparência Ativa, e a sua saída é a classificação entre duas classes: com e sem capacete.

1.3 BREVE REVISÃO DA LITERATURA RELACIONADA

Alguns trabalhos já abordaram este problema utilizando monitoramento por vídeo. Em (SILVA et al., 2013b) foi realizado um trabalho cujo objetivo é a localização e o rastreio de motociclistas em vias públicas. Para isso foram utilizados descritores, como as características de Haar, histogramas de gradientes e padrões locais binários, juntamente com vários classificadores, como as Redes de Base Radial, Perceptron de Múltiplas Camadas e Máquina de Vetores de Suporte. Essas características foram extraídas a partir de imagens com o fundo subtraído para diminuir o processamento, sendo que a fonte dessas imagens são vídeos de câmeras de segurança, como é possível observar na Figura 7.

Figura 7 – Passos para a detecção do motociclista em (SILVA et al., 2013b).

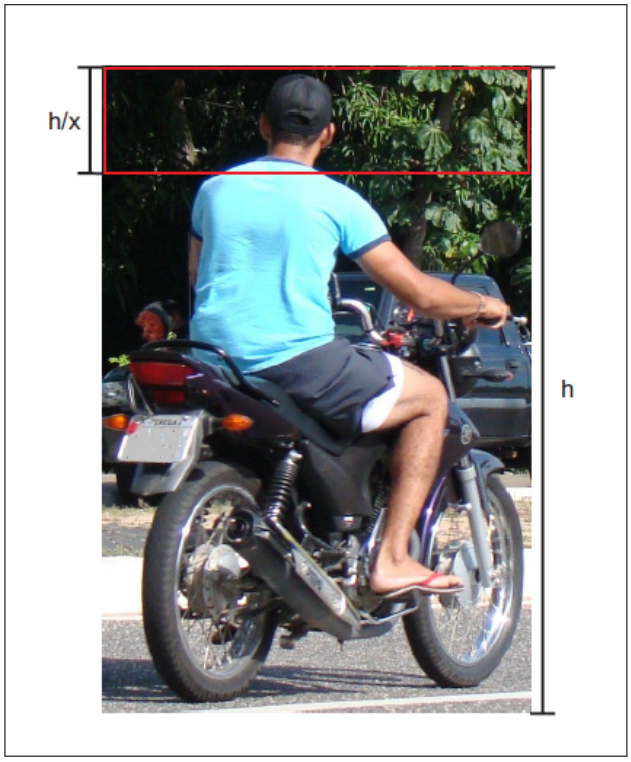


Fonte: Imagem retirada de (SILVA et al., 2013b)

Uma extensão desse trabalho pode ser encontrada em (SILVA et al., 2013a). Nesse

caso, após a localização de uma motocicleta, são extraídas características da região onde há maior probabilidade de encontrar a cabeça do motociclista, com o objetivo de determinar se o mesmo está ou não utilizando o capacete de segurança.

Figura 8 – Região onde é realizada a busca pelo capacete.



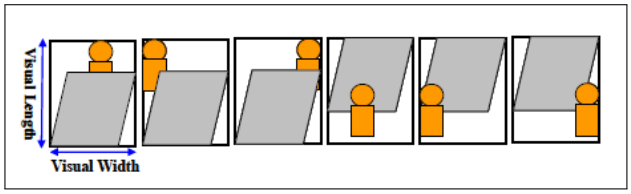
Fonte: Imagem retirada de (SILVA et al., 2013a)

Como é mostrado na Figura 8, a região correspondente ao topo da imagem do motociclista é tida como a região de maior probabilidade de se encontrar o capacete. A partir dessa região é feita uma combinação de transformada circular de Hough, padrões binários locais, histogramas de gradiente e *wavelet* Haar com o objetivo de obter descritores para essa região. Nesse trabalho são feitos diversos testes contendo combinações desses descritores e os classificadores Perceptron de Múltiplas Camadas, Máquinas de Vetores de Suporte e rede neural de Funções de Bases Radiais a fim de determinar a presença do capacete.

Em (KU et al., 2008) é possível encontrar outro trabalho com o objetivo de detectar motocicletas. Nesse estudo é feita uma detecção de oclusão com o objetivo de realizar a detecção mesmo em imagens mais confusas. Nesse caso são utilizadas projeções horizontais e verticais das imagens além da proporção entre altura e largura para classificar a oclusão como mostrado nas Figuras 9 e 10. Além disso o trabalho realiza uma detecção de capacete com o objetivo de validar um motociclista localizado. Isso é feito através de uma busca por contornos que se

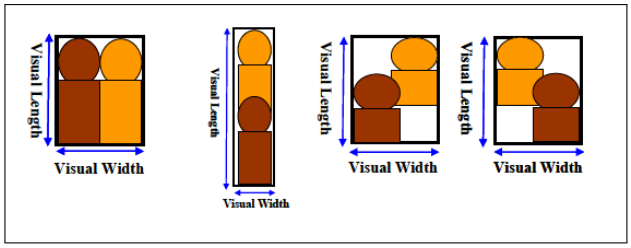
assemelhem a círculos. É feito um cálculo das dimensões do motociclista encontrado através da extração de borda na região de interesse. Isso juntamente com a proporção do pixel é utilizado para determinar se o motociclista está usando o capacete.

Figura 9 – Tipos de oclusão causados entre automóveis e motocicletas.



Fonte: Imagem retirada de (KU et al., 2008)

Figura 10 – Tipos de oclusão causados entre motocicletas



Fonte: Imagem retirada de (KU et al., 2008)

Em (CHIVERTON, 2012) é feita a subtração do *background* com o objetivo de encontrar veículos em movimento. Em seguida é feita a classificação do veículo entre motocicleta e automóvel com base em suas proporções. Por fim são aplicadas algumas operações na região onde se espera encontrar a cabeça do motociclista com o objetivo de extrair histogramas para detectar a presença ou não do capacete.

Figura 11 – Imagens de capacetes sob iluminação variada mostrando que o topo tende a permanecer mais brilhoso que a base.

Fonte: Imagem retirada de (CHIVERTON, 2012)

Como é possível observar na Figura 11, o algoritmo se baseia no fato que o topo

da região com capacete tende a ser mais brilhoso que a parte inferior. Assim, após determinar a região de interesse, essa região é dividida em quatro quadrantes de onde são calculados os histogramas dos níveis de cinza, histogramas de gradientes utilizando uma máscara de diferença finita e núcleos Sobel além de histogramas de gradientes utilizando o gradiente máximo para cada cor tanto com a mascara de diferença finita quanto com o núcleo Sobel. Para realização da classificação é utilizado uma Máquina de Vetores de Suporte com núcleo linear.

Em (KU et al., 2008), (SILVA et al., 2013b), (SILVA et al., 2013a) e (CHIVERTON, 2012) a etapa de localização da região de interesse é feita através da extração do *background*, que contém apenas a rodovia, do *foreground* que contém as regiões de interesse na imagem. Isso é possível, porque nesses trabalhos a determinação da presença do capacete é feita a partir de uma sequência de vídeo. Essa técnica é extremamente eficaz em reduzir a região de busca, pois assim é possível destacar tudo que se move no vídeo, como os motociclistas por exemplo. Após a extração do fundo é feita uma aglutinação dos pixels, gerando bolhas para cada objeto que se move no vídeo. A área de cada uma dessas bolhas é então utilizada para determinar se a região corresponde a um automóvel, um motociclista, um pedestre ou a um ruído. No caso de corresponder a uma motocicleta, a região mais provável de se encontrar a cabeça é então estimada.

Porém, é interessante notar que a extração de *background* pode gerar alguns problemas. Como o algoritmo destaca tudo que se move, pode ser difícil segmentar o motociclista de um automóvel que esteja passando muito próximo. Além disso, como a técnica utiliza vídeo, é necessário realizar o rastreamento do motociclista enquanto ele estiver sendo enquadrado, para que ele não seja detectado novamente.

1.4 CONTRIBUIÇÕES DESTA DISSERTAÇÃO

Como contribuições desta dissertação, podemos citar:

a) Um algoritmo para localizar motociclistas e detectar o uso ou não do capacete de segurança utilizando uma única imagem da via de trânsito.
b) Aplicação e avaliação de parâmetros extraídos do AAM como características (*features*) para problemas de classificação.

1.5 ORGANIZAÇÃO DO TEXTO

Nesta introdução foram apresentados o problema em foco e os objetivos do trabalho, além de uma visão geral dos algoritmos e da metodologia utilizados para abordá-lo.

O restante deste texto está organizado da seguinte maneira. No Capítulo 2 são apresentados modelos deformáveis e a fundamentação teórica dos algoritmos ASM e AAM. O Capítulo 3 apresenta os resultados deste trabalho e a discussão sobre os mesmos. O Capítulo 4 conclui esta dissertação e aponta alguns caminhos para melhorar os resultados obtidos em trabalhos futuros.

Optou-se por colocar em apêndices a fundamentação teórica dos métodos auxilares utilizados neste trabalho. Dessa forma cada leitor decide a conveniência de consultá-los ou não, sendo facultativo para o entendimento da estrutura geral do texto principal. São expostos nos apêndices a Análise de Componentes Principais, técnicas de reconhecimento de padrões, como as Máquinas de Vetores de Suporte e a rede neural Perceptron Multicamadas, além da aplicação de análise de sensibilidade na avaliação desses classificadores

2 MODELOS DE APARÊNCIA ATIVA

A interpretação de imagens através de modelos deformáveis tem sido uma abordagem bastante utilizada devido a sua robustez à presença de ruídos, oclusão e confusão no ambiente da imagem. Os modelos deformáveis são modelos que se ajustam de maneira a melhor se encaixar nos dados fornecidos. Exemplo de modelos deformáveis são os Modelos de Contornos Ativos (KASS et al., 1988), também chamados de *Snakes*, Modelos de Forma Ativa (COOTES et al., 1995) e Modelos de Aparência Ativa (COOTES et al., 1998a). A principal diferença entre *Snakes* e Modelos de Aparência/Forma Ativa é que os primeiros não impõem restrições sobre as formas que serão geradas durante o processo de ajuste enquanto que os últimos utilizam um Modelo de Distribuição de Pontos (PDM - *Point Distribution Model*), garantindo que novas formas geradas permaneçam razoavelmente similares àquelas observadas no conjunto de treinamento. Em comum, todos requerem uma boa inicialização para apresentar bom desempenho. É comum a estratégia de utilizar algum outro algoritmo para fazer uma localização grosseira inicial na imagem e em seguida aplicar os modelos deformáveis no refinamento. A localização e a forma iniciais também podem ser feitas manualmente, como ocorre em algumas aplicações médicas.

Em (MCINERNEY; TERZOPOULOS, 1996) é dito que a fundação matemática dos modelos deformáveis representa a junção da geometria, física e teoria da aproximação. A geometria serve para representar a forma do objeto, a física impõe as restrições de como a forma pode variar sobre o tempo e o espaço, enquanto a teoria da aproximação fornece os mecanismos formais para o ajuste do modelo em relação aos dados fornecidos. A continuidade e suavidade herdada pelos modelos são responsáveis por compensar ruídos, falhas e irregularidades nas bordas de objetos. Segundo (TERZOPOULOS et al., 1987), modelos elasticamente deformáveis, como é o caso por exemplo dos Modelos de Contorno Ativo, são ditos ativos porque respondem de maneira natural a aplicação de forças, restrições, meio ambiente e obstáculos impenetráveis.

Muitos modelos deformáveis aplicados a imagens são ajustados através de uma medida de energia. Essa medida, normalmente leva em consideração a energia de deformação do modelo e uma função potencial definida no plano da imagem. Através da medida de deformação do modelo é possível controlar a tensão e a rigidez do modelo. A função potencial definida na imagem pode ser escolhida de maneira que os seus mínimos locais coincidam com bordas ou outras características de interesse na imagem.

A minimização da energia de um modelo deformável pode ser vista como um problema de otimização. Diversas técnicas podem ser empregadas para resolver esse problema.

Em (HILL; TAYLOR, 1992) por exemplo, um algoritmo genético é utilizado para encontrar o melhor modelo. Outra forma muito utilizada de minimizar a energia é deformar o modelo de acordo com os gradientes da imagem. Nas *Snakes* é utilizada a energia $E = -|\nabla I(x,y)|^2$ de forma que o modelo se move em direção aos contorno de maior gradiente. *Snakes* não são utilizados neste trabalho e foram aqui referidos apenas para uma visão mais completa dos modelos deformáveis mais conhecidos e para diferenciá-los.

2.1 MODELOS DE FORMA ATIVA - ASM

Os Modelos de Forma Ativa (ASM - *Active Shape Models*) foram propostos em (COOTES et al., 1995). Os ASM são modelos deformáveis que podem ser utilizados para localizar estruturas em imagens. Levando-se em conta que a forma dessas estruturas pode variar levemente de exemplo para exemplo, a principal característica do ASM é a capacidade de capturar a variabilidade natural presente nos diversos exemplos que compõem uma certa classe de estruturas, porém continuando específico para a classe que esse modelo representa.

A utilização de um ASM pode ser dividido em duas etapas: a construção do modelo e a utilização desse para encontrar a forma modelada em uma nova imagem. Na primeira etapa, um conjunto de treinamento contendo exemplos da estrutura alvo é utilizado para criar um Modelo de Distribuição de Pontos. Esse PDM contém informações a respeito da forma típica e os modos de variação comuns dessa forma. Na segunda etapa, quando uma nova imagem é apresentada, as informações do PDM são utilizadas para que o modelo possa ser ajustado na nova imagem da melhor maneira, obedecendo as formas possíveis que a estrutura pode assumir.

Criando o Modelo de Distribuição de Pontos

Um PDM é um modelo capaz de descrever tanto a forma quanto a variabilidade típica de uma determinada estrutura a partir de um conjunto de exemplos.

Para gerar o modelo de uma determinada forma, é necessário representar tal forma por meio de um conjunto previamente determinado de pontos rotulados. É importante que cada ponto rotulado represente a mesma parte ou fronteira em todos os exemplos do conjunto de treinamento. Por exemplo, quando o objetivo é marcar pontos em um conjunto de imagens de faces, se um ponto de rótulo x_i representa o canto extremo direito do olho esquerdo, esse ponto x_i deve representar o canto extremo direito do olho esquerdo em todas a imagens do conjunto de

treinamento.

A rotulação correta dos pontos de interesse é importante pois assim a técnica consegue capturar como esses pontos, ou agrupamentos deles, tendem a se mover na medida que a forma varia nos exemplos do conjunto de treinamento.

Figura 12 – Exemplo de imagem rotulada.

Fonte: Elaborada pelo autor

Na Figura 12 é possível observar um exemplo de rotulação de pontos em imagens de condutores com e sem capacete. Os pontos de interesse devem ser marcados em uma ordem que deve ser repetida em toda a base de dados. Assim, observando a variação na posição relativa de cada ponto ao longo das imagens da base, o modelo captura as variações permitidas à forma.

Alinhando os exemplos do PDM

Após rotular o conjunto de treinamento, é necessário alinhar todos os seus pontos para em seguida obter as estatísticas dos pontos rotulados. O alinhamento é alcançado através de rotação, translação e redimensionamento das formas do conjunto de treinamento de maneira que os pontos correspondentes em todos os exemplos fiquem o mais próximo possível, sem que haja alteração na forma. O objetivo é então minimizar a soma ponderada do quadrado das distâncias entre pontos equivalentes nos exemplos de forma do conjunto de treinamento.

Considerando x_i e x_j dois vetores coluna descrevendo os n pontos da i-ésima e da j-ésima forma do conjunto de treinamento:

$$x_i = (x_{i0}, y_{i0}, x_{i1}, y_{i1}, ..., x_{ik}, y_{ik}, ..., x_{i,n-1}, y_{i,n-1})^T e$$
$$x_j = (x_{j0}, y_{j0}, x_{j1}, y_{j1}, ..., x_{jk}, y_{jk}, ..., x_{j,n-1}, y_{j,n-1})^T,$$

o alinhamento desses dois exemplos pode ser obtido através da minimização da função objetivo:

$$E_j = (x_i - M(s_j, \theta_j)[x_j] - t_j)^T W (x_i - M(s_j, \theta_j)[x_j] - t_j), \tag{2.1}$$

em que $M(s,\theta)[x]$ é uma operação de rotação por θ e redimensionamento por s, t é um vetor $(t_x,t_y,t_x,t_y,...,t_x,t_y)$ que realiza uma translação (t_x,t_y) em todas os pontos e W é uma matriz que carrega o peso de cada ponto.

Assim, estamos escolhendo a rotação θ, o redimensionamento s e a translação t de maneira a mapear x_i em $M(s_j,\theta_j)[x_j]+t_j$ minimizando a Equação 2.1. No Apêndice D é possível encontrar detalhes de como essa minimização pode ser resolvida.

A matriz W é utilizada para dar pesos diferentes aos pontos. Ela pode ser acertada de maneira que pontos que variam pouco entre si recebam pesos maiores e portanto tenham maior importância.

O alinhamento entre todas as formas do conjunto de treinamento é realizado aplicando-se o alinhamento 2-a-2 nos exemplos do conjunto de treinamento seguindo o Algoritmo 1:

Algoritmo 1: Alinhamento do conjunto de treinamento ((COOTES et al., 1995)).

Entrada: Exemplos de forma do conjunto de treinamento.

Saída: Exemplos de forma do conjunto de treinamento alinhados.

início

    Rotaciona, redimensiona e translada cada forma de maneira a essa se alinhar com um *template* previamente selecionado;

    repita

        Calcula a forma média dos exemplos alinhados;

        Encontra a rotação, escala e translação que devem ser aplicadas à forma média de maneira que seus pontos fiquem o mais próximo possível dos pontos do *template*;

        Realinha cada forma com a nova média;

    até *convergir*;

fim

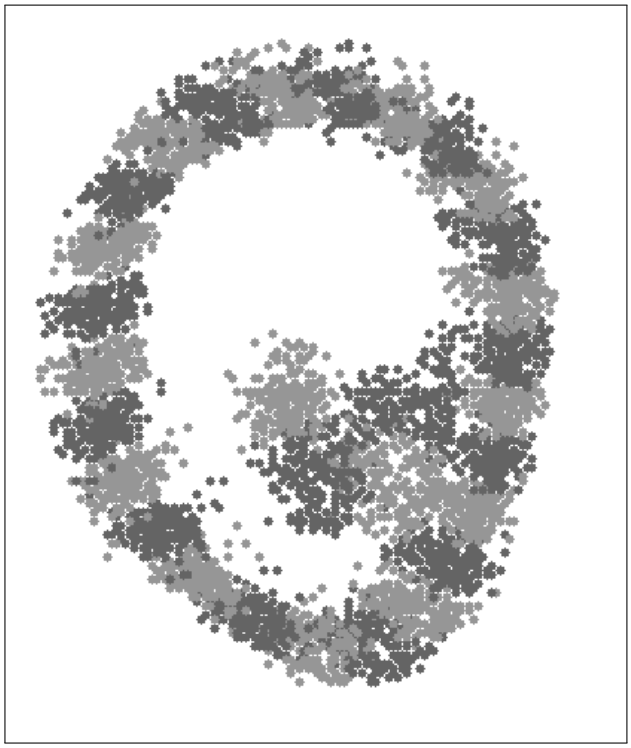
O *template* previamente selecionado serve apenas para definir uma escala, orientação (rotação) e posição (translação) para serem usadas como base para o alinhamento. Essa forma pode ser previamente desenhada ou é possível escolher um dos exemplos do conjunto de treinamento.

Após o alinhamento dos pontos do conjunto de treinamento de uma base de dados, é possível observar na Figura 13 que esses pontos formam nuvens onde pode-se notar a variabilidade de cada um desses pontos de interesse.

Essa variabilidade de cada ponto de interesse é então utilizada para extrair do PDM

os modos de variação das imagens do conjunto de treinamento.

Figura 13 – Pontos do conjunto de treinamento após serem alinhados.



Fonte: Elaborada pelo autor

Informações extraídas do PDM

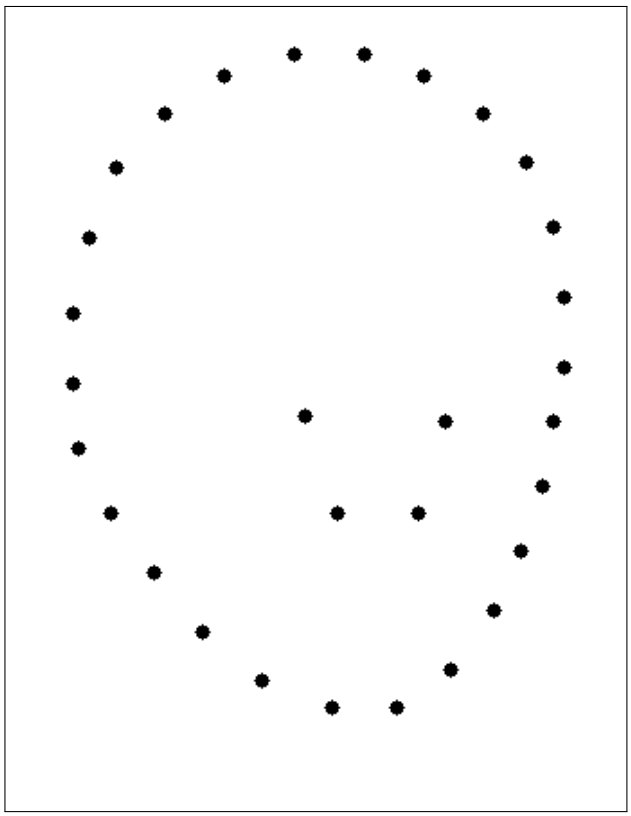
Uma primeira informação estatística que podemos extrair dos pontos após eles serem alinhados é a sua forma média:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i,$$

onde N é o número de exemplos do conjunto de treinamento, resulta em um vetor contendo a forma média dos exemplos do conjunto de treinamento. A Figura 14 mostra um exemplo de forma média.

Além da forma média, o Modelo de Distribuição de Pontos busca encontrar os principais modos de variação das coordenadas nessas nuvens. Uma característica importante dos pontos demarcados é o fato deles não variarem de maneira independente ao longo dos exemplos do conjunto de treinamento. Isso quer dizer que as suas posições estão parcialmente

Figura 14 – Forma média do conjunto de treinamento.



Fonte: Elaborada pelo autor

correlacionadas. Com essa informação, podemos aplicar uma Análise de Componentes Principais no conjunto de dados com o objetivo de extrair esses principais modos de variação.

Segundo (JOLLIFFE, 2002) a Análise de Componentes Principais (PCA - *Principal Component Analysis*) é uma técnica utilizada para reduzir a dimensionalidade de um conjunto de dados que possui um grande número de variáveis inter-relacionadas. Isso é alcançado transformando os dados para um novo conjunto de variáveis, as componentes principais, que são não correlacionadas e estão organizadas de maneira que as primeiras componentes contêm a maior parte da variância presente nos dados originais, como pode ser visto no Apêndice E. Essa descrição do PCA combina bem com o PDM, pois essas componentes principais são justamente os modos de variação mais significativos que se deseja encontrar.

O primeiro passo para a aplicação da Análise de Componentes Principais é organizar os dados em uma matriz:

$$X = \begin{pmatrix} | & | & | & \dots & | \\ x_1 & x_2 & x_3 & \dots & x_N \\ | & | & | & \dots & | \end{pmatrix}.$$

É então criada uma matriz X, com N colunas e $2n$ linhas, em que cada coluna é um vetor x_i que descreve os n pontos da i-ésima forma do conjunto de treinamento. Para um conjunto de treinamento com N exemplos de forma, temos uma matriz com N colunas.

Pode-se provar que os autovetores da matriz XX^T, que é a matriz de covariância de X, descrevem os modos de variação das variáveis originais $(x_0, y_0, x_1, y_1, ..., x_k, y_k, ..., x_{n-1}, y_{n-1})$ do conjunto de treinamento. O PDM precisa encontrar os modos de variação mais significativos, isso pode ser feito através da Decomposição de Valores Singulares dessa matriz:

$$X = U\Sigma V^T.$$

A matriz U, que é uma matriz $2n \times 2n$, resultante da Decomposição de Valores Singulares, contém os autovetores matriz XX^T, a matriz V, que é uma matriz $N \times N$, contém os autovetores da matriz X^TX, e a diagonal principal da matriz Σ, que é uma matriz $2n \times N$, contém os autovalores correspondentes a esses autovetores.

Sendo assim, as colunas da matriz U descrevem os modos de variação das variáveis originais do conjunto de treinamento. Além disso, os autovalores que podem ser encontrados na diagonal principal da matriz Σ informam a variância de cada um desses modos.

A maior parte da variância do conjunto de dados pode ser explicada por um pequeno número de modos de variação t. Assim, é possível escolher t modos de variação que podem gerar um conjunto de dados que é uma aproximação do conjunto de dados original que possui dimensão $2n$.

É possível então criar uma matriz P contendo os t principais modos de variação, ou componentes principais, do conjunto original:

$$P = \begin{pmatrix} | & | & ... & | \\ p_1 & p_2 & ... & p_t \\ | & | & ... & | \end{pmatrix}.$$

Qualquer exemplo de forma do conjunto de treinamento pode ser gerada aproximadamente a partir de uma combinação linear dos t componentes principais descritos acima adicionando-se a forma média $\bar{x}$ como pode ser visto em:

$$x = \bar{x} + Pb, \tag{2.2}$$

onde x é um exemplo do conjunto de treinamento e $b = (b_1, b_2, ..., b_t)$ é um vetor contendo os pesos de cada componente da matriz P, ou seja os parâmetros do modelo.

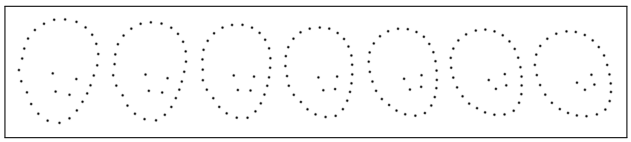
A Equação 2.2 nos permite também gerar novos exemplos de forma variando os valores do vetor b. Ao manter os valores de b dentro de um limite adequado, os novos exemplos gerados devem permanecer similares aos exemplos do conjunto de treinamento. Sendo que a variância correspondente a cada componente principal p_k é igual a λ_k, um exemplo de limite adequado utilizado em (COOTES et al., 1995) é:

$$-3\sqrt{\lambda_k} \leq b_k \leq 3\sqrt{\lambda_k}.$$

Essa restrição do valor das componentes principais do PDM é uma característica importante dos Modelos de Forma Ativa pois ela garante que as novas formas geradas permaneçam razoavelmente similares àquelas do conjunto de treinamento. Em (COOTES et al., 1995) é dito que isso diferencia o ASM de outros modelos deformáveis, como por exemplo os Modelos de Contorno Ativo, que não impõem esse tipo de restrição.

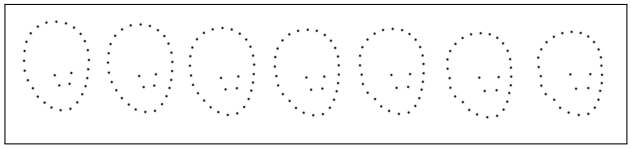
As Figuras 15, 16 e 17 mostram exemplos que podem ser sintetizados variando os 3 primeiros parâmetros de b. Para geração desses exemplos, os valores de cada b_k foram mantidos dentro dos limites sugeridos anteriormente.

Figura 15 – Exemplos de formas geradas variando o primeiro parâmetro de b.



Fonte: Elaborado pelo autor

Figura 16 – Exemplos de formas geradas variando o segundo parâmetro de b.

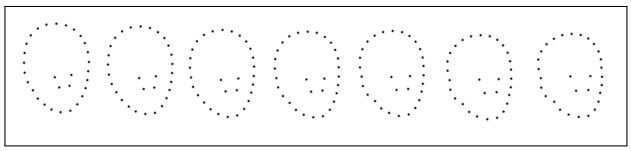


Fonte: Elaborado pelo autor

Usando o PDM na busca por estruturas na imagem - Os Modelos de Forma Ativa

Como foi mostrado, é possível utilizar o PDM para gerar novos exemplos de formas, similares àqueles encontrados no conjunto de treinamento. Isso pode ser utilizado para encontrar,

Figura 17 – Exemplos de formas geradas variando o terceiro parâmetro de b.



Fonte: Elaborado pelo autor

em novas imagens, objetos similares à forma modelada.

O ASM realiza ajustes nos parâmetros de forma do PDM de maneira iterativa com o objetivo de encontrar em uma nova imagem uma estrutura similar a forma modelada pelo PDM. Também são ajustados parâmetros de pose do modelo. Ao todo, esses parâmetros são:

a) b que é um vetor contendo os parâmetros da variação da forma, ou seja os parâmetros do PDM;
b) s que é um fator de redimensionamento;
c) θ que informa a rotação;
d) e t que contém o deslocamento.

Dessa maneira, uma instância do PDM pode ser obtida aplicando-se todos esses parâmetros sobre a forma média e aplicando os parâmetros de pose como pode ser visto nas equações:

$$x = \bar{x} + Pb, \tag{2.3}$$

$$X = M(s,\theta)[x] + t, \tag{2.4}$$

sendo que $M(s,\theta)[x]$ é uma operação de rotação por θ e redimensionamento por s e $t = (x_t, y_t, x_t, y_t, ..., x_t, y_t,)$ é um vetor de dimensão $2n$ que desloca todos os pontos em (x_t, y_t).

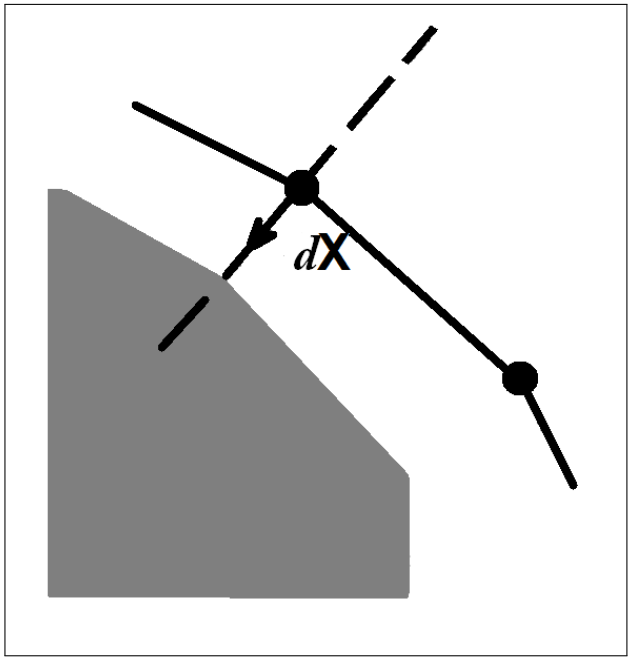
A Equação 2.3 realiza as deformações na forma média como foi demonstrado nos exemplos da Seção 2.1. Já a Equação 2.4 realiza o deslocamento, a rotação e o ajuste do tamanho da forma já deformada, ajustando a posição, o ângulo e a escala da instância do modelo na nova imagem apresentada. Após a aplicação dessas equações, o que se obtém é um vetor X de dimensão $2n$ que informa a localização de cada ponto do PDM na nova imagem apresentada.

Os ajustes nos parâmetros do modelo de distribuição de pontos são então iniciados a partir de uma estimativa inicial, mesmo que grosseira, da posição da imagem sobre a qual se deseja extrair a forma. Durante essa estimativa, a forma de X é mantida igual à forma média,

com um vetor b composto apenas de zeros. Assim, são estimados apenas a posição (t), o ângulo (θ) e a escala (s) inicial do modelo para então ser executado o algoritmo de ajuste. Existem diversas maneiras de realizar essa estimativa. Porém, mesmo a estimativa não sendo muito boa, o algoritmo se propõe a ajustar os parâmetros para encontrar a estrutura na imagem.

Após a estimativa da posição inicial, deve-se encontrar os melhores ajustes que movam cada ponto em direção a uma melhor posição. Isso pode ser feito de diversas maneiras, dependendo das características dos pontos modelados. Quando os pontos representam as fronteiras de um objeto, uma boa estratégia é mover os pontos em direção as bordas da imagem (como pode ser visto na Figura 18), que podem ser detectadas por exemplo passando-se um filtro passa-altas, que preserva as altas frequências da imagem.

Figura 18 – Exemplo de procedimento para encontrar os ajustes de cada ponto.



Fonte: Imagem adaptada de (COOTES et al., 1995).

Independente de como esses ajustes forem realizados, o que se obtém é um vetor $dX = (dX_0, dY_0, ..., dX_{n-1}, dY_{n-1})$ que desloca cada ponto de X em direção a uma posição melhor.

O objetivo então é encontrar os melhores ajustes nos parâmetros de pose (ds, $d\theta$, dt) e de forma (db) que movam os pontos da sua posição atual em X, para as suas novas posições sugeridas ($X + dX$), mantendo as restrições do modelo.

Os ajustes de pose podem ser feitos aplicando-se o mesma técnica utilizada para alinhar dois exemplos do conjunto de treinamento que foi mostrada na Seção 2.1, sendo que

nesse caso, deseja-se alinhar os pontos de X com os pontos de $(X + dX)$.

O que resta então é encontrar os ajustes de forma (db). Para isso é necessário descobrir primeiro os ajustes residuais dx que podem ser vistos em:

$$M(s(1+ds),(\theta+d\theta))[x+dx]+(t+dt)=(X+dX).$$

Esses ajustes são responsáveis pela mudança na forma de X após terem sido feitos todos os ajustes de pose. Portanto eles podem ser utilizados para se descobrir db.

O valor de dx pode encontrado isolando-o, como pode ser visto em:

$$dx = M((s(1+ds))^{-1}, -(\theta+d\theta))[y]-x,$$

onde $y = M(s,\theta)[x]+dX-dX$.

A Equação 2.3 nos diz que $x = \bar{x}+Pb$, então queremos encontrar db de acordo com:

$$x+dx \approx \bar{x}+P(b+db), \tag{2.5}$$

que é uma aproximação já que P contém apenas t modos de variação enquanto dx pode mover seus pontos em $2n$ dimensões diferentes.

Subtraindo a Equação 2.3 da Equação 2.5, obtém-se: $dx \approx P(db)$. Como as colunas de P são vetores ortonormais entre si, temos que $P^T P = I$. Assim, é possível chegar à:

$$db = P^T dx.$$

Assim temos todos os elementos necessários para atualizar os valores dos parâmetros do modelo de distribuição de pontos. Essa atualização pode ser feita como descrito no Algoritmo 2.

Sendo que w_t, w_θ e w_s são pesos escalares e W_b é uma matriz diagonal de pesos, um para cada modo de variação.

Como foi dito, o processo de ajuste dos pesos é iterativo. A cada iteração, o ASM ajusta os parâmetros do PDM de maneira que os pontos de X tentem casar com as regiões da nova imagem correspondente aos pontos modelados pelo PDM. O procedimento deve se repetir até que X fique estável.

Porém essa técnica não garante que o valor final encontrado para X seja realmente o valor ótimo. O algoritmo está sujeito a ficar preso em mínimos locais. Além disso a técnica depende de procedimentos que variam de problema para problema, como é o caso da determinação

Algoritmo 2: Ajuste dos parâmetros de pose do modelo de forma ((COOTES et al., 1995)).

Entrada: Parâmetros de pose a serem ajustados ($t(x)_i$, $t(y)_i$, θ_i, s_i, b_i);

Valor dos ajustes calculado como descrito ($dt(x)$, $dt(y)$, $d\theta$, ds, db);

Pesos aplicado a cada ajuste (w_t, w_θ, w_s, w_b).

Saída: Novos valores para os parâmetros de pose ($t(x)_{i+1}$, $t(y)_{i+1}$, θ_{i+1}, s_{i+1}, b_{i+1}).

início

$t(x)_{i+1} = t(x)_i + w_t * dt(x)$;

$t(y)_{i+1} = t(y)_i + w_t * dt(y)$;

$\theta_{i+1} = \theta_i + w_\theta * d\theta$;

$s_{i+1} = s_(1 + w_s ds)$;

$b_{i+1} = b_i + W_b * db$;

fim

de dX. Quando os pontos se encontram em regiões de bordas, é muito comum utilizar algoritmos de detecção de bordas para determinar o melhor deslocamento dX, porém quando esses pontos não estão em região de borda, é necessário utilizar outros procedimentos como a construção de um perfil de textura (de uma ou duas dimensões) para cada ponto com o objetivo de encontrar o melhor deslocamento para esse ponto. Assim, a escolha do melhor procedimento é importante pois quando dX é mal determinado, não se pode esperar que o resultado final seja satisfatório.

Outro procedimento que deve ser muito bem escolhido é a estimativa inicial da posição do modelo. Apesar de em (COOTES et al., 1995) ser dito que essa estimativa inicial não influencia muito no resultado final do algoritmo, experimentos preliminares mostraram que pequenas variações na estimativa inicial resultam em formas completamente diferentes umas das outras.

2.2 MODELOS DE APARÊNCIA ATIVA - AAM

Os Modelos de Aparência Ativa (AAM - Active Appearence Models) foram propostos em (COOTES et al., 1998a) como uma extensão do ASM com a capacidade de modelar além da forma, a aparência do objeto alvo. Por aparência, o que se entende são informações a respeito dos valores de intensidade dos pixels.

Assim os modelos são gerados combinando um modelo de variação de forma com um modelo de variação de aparência em um quadro com a forma normalizada.

A seguir é mostrado como essa informação de aparência é modelada e em seguida

como a técnica realiza a busca em novas imagens.

Modelando a Aparência

Assim como ocorre no ASM, o AAM também modela a forma do objeto. Portanto é necessário um conjunto de treinamento de imagens rotuladas, onde pontos chaves são marcados em cada exemplo da mesma maneira que acontece na criação do modelo do ASM.

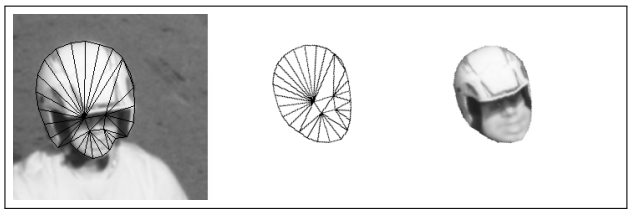
Dado um conjunto de treinamento corretamente rotulado, em que cada forma é representada por um vetor x, é possível gerar um modelo estatístico da variação da forma assim como foi explicado na Seção 2.1. Todos os exemplos do conjunto de treinamento são alinhados e então é aplicada a Análise de Componentes Principais nos dados. Assim um exemplo pode ser aproximado utilizando:

$$x = \bar{x} + P_s b_s,$$

em que $\bar{x}$ é a forma média, P_s é um conjunto de modos de variação e b_s é um conjunto de parâmetros de forma.

Em seguida é construído um modelo estatístico da aparência do nível de cinza. Para isso cada imagem é distorcida de maneira que todos os pontos de controle combinem com aqueles da forma média. Essa distorção é feita utilizando um algoritmo de triangulação.

Figura 19 – Exemplo de distorção de uma imagem para a forma média.



Fonte: Elaborado pelo autor

A Figura 19 mostra uma exemplo do procedimento de triangulação. Inicialmente o é obtida a triangulação de Delaunay dos pontos da forma média (centro). Em seguida esses mesmos triângulos são reproduzidos na imagem que se deseja distorcer (esquerda). Por fim cada um dos pixels internos de cada triângulo da imagem alvo é mapeado de acordo com os respectivos triângulos na forma média (direita).

Então a informação do nível de cinza das imagens com a forma distorcida é amostrada na região coberta pela forma média. A aparência de um exemplo passa então a ser representado

por um vetor g. Para minimizar o efeito da variação de iluminação, é feita uma normalização nos exemplos amostrados aplicando:

$$g = (g_{im} - \beta)/\alpha, \tag{2.6}$$

sendo que os valores de α e β são escolhidos de maneira que a média dos dados normalizados $\bar{g}$ seja zero e a variância seja a unidade. Os valores de α e β são dados por:

$$\alpha = g_{im}.\bar{g},$$
$$\beta = (g_{im})/n.$$

Como pode se observar, esses parâmetros de normalização são definidos em termos da média $\bar{g}$. Portanto α e β são definidos de maneira iterativa.

Então, aplicando a Análise de Componentes Principais nos dados normalizados, obtém-se um modelo linear:

$$g = \bar{g} + P_g b_g$$

em que $\bar{g}$ é o vetor de nível de cinza médio normalizado, P_g é uma matriz contendo os modos de variação ortogonais e b_g é um conjunto de parâmetros para a aparência.

Assim, a forma e a aparência de cada exemplo podem ser representadas pelos vetores b_s e b_g. Como é possível que haja correlação entre as variações da forma e dos níveis de cinza, é aplicado novamente a Análise de Componentes Principais nos dados como pode ser visto na equação:

$$b = \begin{pmatrix} W_s b_s \\ b_g \end{pmatrix} = \begin{pmatrix} W_s P_s^T (x - \bar{x}) \\ P_g^T (g - \bar{g}) \end{pmatrix},$$

onde W_s é uma matriz diagonal de pesos para cada parâmetro de forma que compensa a diferença entres as unidades de medidas da forma e dos níveis de cinza.

Então é aplicado novamente o PCA com o objetivo de evitar correlações entre os parâmetros de forma e textura. Isso resulta na construção de um novo modelo:

$$b = Qc,$$

onde Q são os autovetores e c é o vetor contendo os parâmetros da aparência que controlam tanto forma como os níveis de cinza do modelo. Como os parâmetros de forma e nível de cinza já possuem média zero, c também possui.

Como esse modelo é linear, é possível expressar a forma e os níveis de cinza diretamente a partir de c, como pode ser visto em:

$$x = \bar{x} + P_s W_s Q_s c \quad \text{e} \quad g = \bar{g} + P_g Q_g c, \tag{2.7}$$

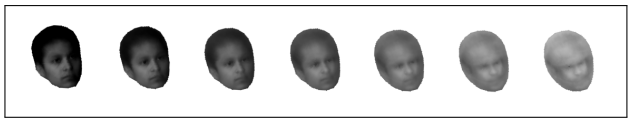
sendo que Q_s e Q_g são:

$$Q = \begin{pmatrix} Q_s \\ Q_g \end{pmatrix}.$$

Assim uma imagem exemplo pode ser sintetizada para um determinado c simplesmente gerando a imagem livre de forma dos níveis de cinza g e distorcendo essa imagem de maneira que os seus pontos de controle se acertem com aqueles descritos por x.

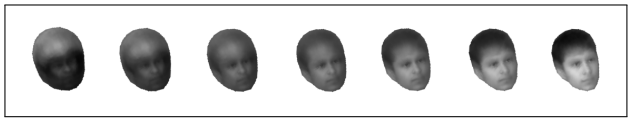
As Figuras 20, 21 e 22 mostram exemplos que podem ser sintetizados variando os 3 primeiros parâmetros de c.

Figura 20 – Exemplos de aparência geradas variando o primeiro parâmetro de c.



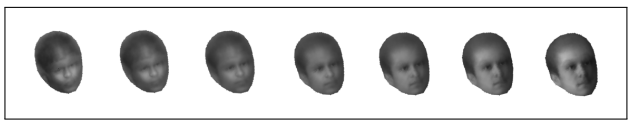
Fonte: Elaborado pelo autor

Figura 21 – Exemplos de aparência geradas variando o segundo parâmetro de c.



Fonte: Elaborado pelo autor

Figura 22 – Exemplos de aparência geradas variando o terceiro parâmetro de c.



Fonte: Elaborado pelo autor

Portanto, assim como ocorre no ASM, o procedimento de busca no AAM simplesmente encontra os valores dos parâmetros, nesse caso o vetor c, que melhor representa uma nova imagem apresentada.

Aproximando um novo Exemplo

Dada uma nova imagem, rotulada com o conjunto de pontos de controle, é possível gerar uma aproximação com o modelo.

O vetor b pode ser obtido como está descrito na seção anterior combinando os o parâmetros de forma e de nível de cinza. Como Q é ortogonal, o vetor c, que possui os parâmetros de aparência e nível de cinza combinados, pode ser encontrado usando:

$$c = Q^T b.$$

A partir dos parâmetros no vetor c, é possível realizar uma reconstrução aproximada da imagem utilizando as Equações 2.7, em seguida invertendo a normalização dos níveis de cinza e aplicando as distorções na imagem livre de forma para que ela se acerte com a forma x obtida.

Usando o Modelo de Aparência Ativa para Buscas

A partir de um modelo de aparência como descrito acima e uma nova imagem, o problema passa a ser o de ajustar os parâmetros do modelo de maneira que a imagem sintetizada por ele case com a nova imagem o melhor possível. Esse problema pode ser visto como um problema de otimização em que se deseja minimizar a diferença entre a nova imagem e aquela sintetizada pelo modelo de aparência.

Assim, pode-se definir um vetor de diferenças δI de acordo com a equação:

$$\delta I = I_i - I_m,$$

sendo que I_i é o vetor com os valores de níveis de cinza na imagem e I_m é o vetor contendo os valores do nível de cinza para os parâmetros atuais do modelo.

É necessário então minimizar a magnitude do vetor de diferença, $\Delta = |\delta I|^2$, variando os parâmetros do modelo, o vetor c. Isso pode ser feito armazenando com antecedência informações a respeito de como resolver esse problema. Uma maneira simples de fazer isso é encontrar o relacionamento entre δI e o erro nos parâmetros do modelo:

$$\delta c = A\delta.I$$

Para encontrar A, é realizada uma regressão linear em exemplos que se conhece o deslocamento δc dos parâmetros do modelo e as correspondentes diferenças nas imagens δI. É possível gerar deslocamentos aleatórios perturbando os valores dos parâmetros de imagens em que eles são conhecidos. Pode-se utilizar o conjunto de treinamento original ou então imagens sintetizadas com o modelo de aparência.

Assim como as perturbações no modelo de aparência, também deve-se realizar pequenos deslocamentos na posição, escala e orientação. Isso é feito incluindo quatro parâmetros extras na regressão. Esses são similares aos parâmetros de pose presentes no ASM: (s_x, s_y, t_x, t_y) onde $s_x = s\cos(\theta)$ e $s_y = s\sin(\theta)$

Seja c_0 os parâmetros conhecidos de uma dada imagem, a diferença é então calculada de acordo com o Algoritmo 3.

Algoritmo 3: Cálculo do vetor de perturbações e efeitos (COOTES et al., 1998a).

Entrada: Exemplos de aparência com parâmetros conhecidos.

Saída: Vetores δc e δg contendo as pertubações nos parâmetros e o efeitos dessas perturbações no vetor de aparência.

início

δc recebe um vetor aleatório contendo deslocamentos para os parâmetros;
são aplicados esses deslocamentos nos parâmetros do modelo: $c = \delta c + c_0$;
é gerada uma forma x de acordo com os novos parâmetros contidos em c;
é gerada uma aparência g_m a partir de x e c;
a aparência original é distorcida de acordo com x, obtendo g_s;
o efeito dos deslocamentos é então calculado: $\delta g = g_s - g_m$;

fim

Dessa forma o algoritmo de treinamento simplesmente realiza deslocamentos aleatórios nos parâmetros do modelo em cada imagem de treinamento, armazenando δc e δg. Em seguida é feita uma regressão linear para obter o relacionamento entre δc e δg como pode ser visto em:

$$\delta c = A\delta g.$$

A matriz A que guarda esse relacionamento é então utilizada durante a etapa de ajuste do modelo quando uma imagem desconhecida é apresentada e se deseja encontrar os

parâmetros que melhor sintetizem essa imagem.

Ajustando os parâmetros do AAM em Buscas

Encontrada a matriz A, é possível utilizá-la para encontrar os melhores parâmetros do modelo para sintetizar uma imagem que case com uma nova imagem fornecida.

Sendo c_0 a estimativa atual dos parâmetros do modelo, x a sua forma gerada e g_m os níveis de cinza normalizados e g_s a nova imagem fornecida deformada por x. Isso pode ser feito seguindo o Algoritmo 4.

Algoritmo 4: Alinhamento do conjunto de treinamento (COOTES et al., 1998a).

Entrada: Uma imagem para a qual se deseja ajustar os parâmetros do modelo.
Saída: Parâmetros do modelo ajustados para sintetizar o objeto na imagem.
início
  repita
    calcule o vetor de erro: $\delta g_0 = g_s - g_m$;
    calcule o erro atual: $E_0 = |\delta g_0|^2$;
    calcule o deslocamento previsto: $\delta c = A\delta g_0$;
    estabeleça k = 1;
    faça $c_1 = c_0 - k\delta c$;
    sintetize uma nova imagem a partir de c_1 e calcule o novo erro δg_1;
    se $|\delta g_1|^2 < E_0$ **então**
      aceite a nova estimativa c_1;
    fim
    senão
      tente com $k = 0.5$, $k = 0.25$, $k = 0.125$,...;
    fim
  até *não haja mais melhorias significantes no erro*;
fim

Ao final da execução, c deve conter os parâmetros para sintetizar uma aproximação da imagem fornecida.

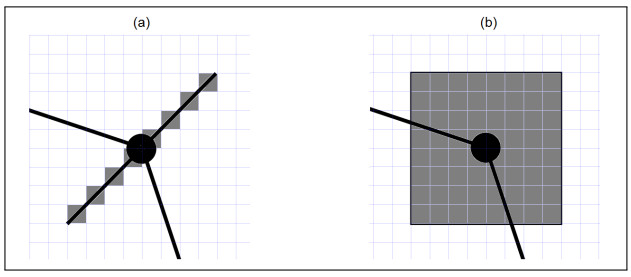
2.3 APLICAÇÕES DOS AAMS

A literatura apresenta diversos exemplos de aplicações de técnicas que utilizam modelos deformáveis.

Em (COOTES et al., 1993) é mostrado como os Modelos de Forma Ativa podem ser utilizados em aplicações médicas como o objetivo de facilitar a análise de imagens resultantes de ressonância magnética, como segmentação de ecocardiograma, localização da próstata além de outras aplicações médicas e na industria. O trabalho também sugere a utilização de perfis de níveis de cinza (similar a modelagem da aparência) no ajuste dos pontos do PDM.

Na Figura 23 é possível observar que esses perfis são construídos a partir dos níveis de cinza de pixels próximos ao ponto de controle, sendo que pode ser construído um perfil 1D (Figura 23.a) a partir de pixels contidos em uma reta perpendicular à borda, ou um perfil 2D (Figura 23.b) a partir de uma submatriz contendo os valores dos pixels no entorno do ponto de controle. O trabalho também indica a utilização de algoritmo genético na inicialização do modelo.

Figura 23 – Exemplos de perfis 1D e 2D que podem ser usados para descrever pontos de controle no ASM.



Fonte: Imagem produzida pelo Autor

Em (MITCHELL et al., 2001) podemos encontrar uma abordagem híbrida que utiliza uma mistura de AAM e ASM para segmentar os ventrículos direito e esquerdo em imagens cardiológicas. O autor utiliza um algoritmo híbrido para realizar o ajuste do modelo porque o algoritmo do AAM tende a priorizar a aparência durante o seu ajuste e falha em encontrar certas estruturas locais. Tendo o ASM sucesso nessa tarefa, o trabalho realiza o ajuste do modelo de forma mesclada. Cada iteração do ajuste do modelo acontece como descrito no Algoritmo 5. A inicialização do posicionamento do modelo ocorre através da aplicação da transformada de Hough. O objetivo da técnica é realizar a segmentação das imagens e sua performance é medida em termos de erro de posicionamento das bordas.

Algoritmo 5: Ajuste dos parâmetros Híbrido de ASM e AAM (MITCHELL et al., 2001).

Entrada: Imagem onde o modelo será ajustado, Modelo de Aparência Ativa devidamente inicializado.

Saída: Modelo de Aparência Ativa ajustado para a imagem.

início

  repita

    os parâmetros de forma do AAM são obtidos do modelo e ajustados de acordo com o ASM;

    os parâmetros combinados do AAM são ajustados de acordo com o algoritmo de ajuste do AAM;

    os dois ajustes realizados simultaneamente resultam em dois vetores de parâmetros de forma, esses vetores são combinados ponderadamente em um único vetor que passa a conter os parâmetros de forma do AAM;

  até *convergir*;

fim

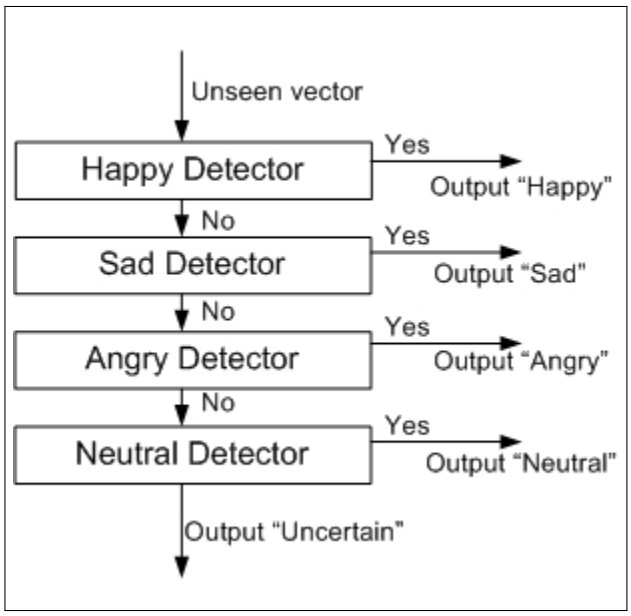
Em (SAATCI; TOWN, 2006) são utilizadas características extraídas a partir de Modelos de Aparência Ativa para determinar expressões características de emoções e o gênero em imagens faciais. Para realizar a classificação dos exemplos foram construídos classificadores para cada emoção (raiva, alegria, tristeza e indiferença) e para o gênero (masculino e feminino), sendo que cada classificador funciona como um classificador binário decidindo apenas se o exemplo apresenta ou não tal emoção ou a qual gênero ele pertence, como pode ser visto na Figura 24.

O trabalho também realiza um estudo a respeito da utilização de uma hierarquia na construção dos classificadores. São realizados testes tanto no caso em que o gênero é utilizado para facilitar a classificação da emoção, quanto no caso em que a emoção é utilizado para facilitar a classificação do gênero, como pode ser visto na Figura 25.

Em (AHLBERG; FORCHHEIMER, 2003) é construído um Modelo de Aparência Ativa com o objetivo de se obter um modelo de deformação e rastreio estatístico. Esse modelo é utilizado então para sintetizar imagens de expressões faciais com base nos parâmetros de ajuste do modelo no rosto de um indivíduo. O trabalho também apresenta uma série de sugestões com o objetivo de alcançar a utilização da técnica em tempo real, permitindo a redução do tempo de cada iteração e do número de iterações além de sugerir a utilização de estimativa de movimento.

Em (COOTES et al., 2000), os Modelos de Aparência Ativa são utilizados com o

Figura 24 – Classificadores binários para determinar a emoção.



Fonte: Imagem retirada de (AHLBERG; FORCHHEIMER, 2003)

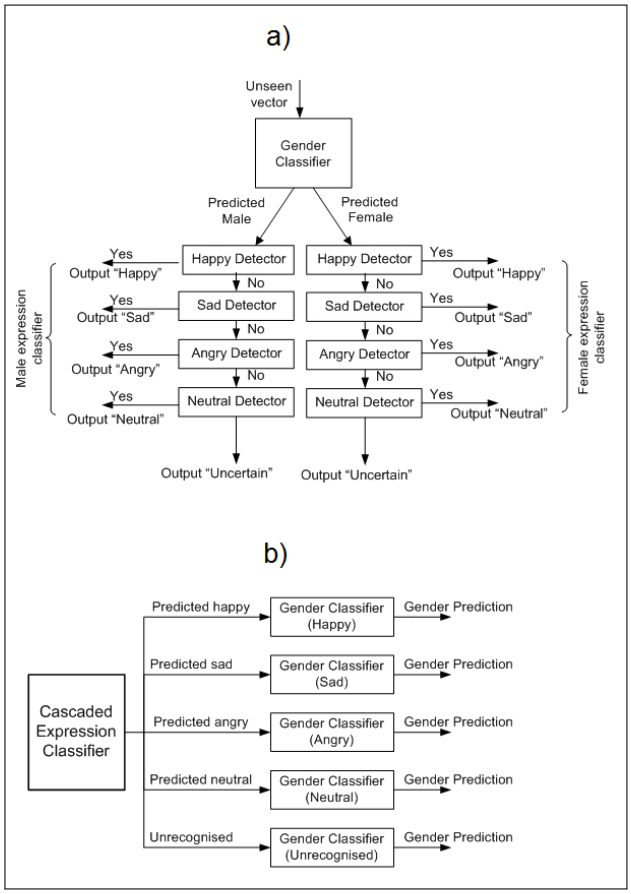
objetivo de capturar a forma e a aparência de uma face em uma variedade de pontos de vista. A Figura 26 mostra como uma face pode ser capturada de frente e de perfil simultaneamente através do uso de espelhos. A construção de um modelo casado dessas múltiplas imagens de uma mesma face, permite a descrição do relacionamento entre a aparência frontal e do perfil da face. Esse relacionamento pode ser usado para prever a aparência de um ponto de vista a partir de uma imagem de outro ponto de vista. Uma das aplicações dessa técnica é permitir que o modelo casado seja usado como restrição em algoritmos de busca que procuram a localização de uma face a partir de múltiplos pontos de vista simultaneamente.

O procedimento constrói modelos a partir de exemplos que variam dentro de uma diversidade de ângulos como pode ser visto na Figura 27. Em seguida é determinada a relação entre os parâmetros dos modelos e o ângulo das faces através de uma regressão. Isso permite a síntese de imagens em novos ângulos.

Uma vez realizado o treinamento, são obtidas as relações entre os parâmetros dos diversos modelos, permitindo a construção do modelo de aparência de visão casada. Isso permite que se obtenha os parâmetros de um modelo a partir da aproximação dos parâmetros de outro modelo que foi treinado com uma faixa de ângulos diferente.

Em (THEOBALD et al., 2007) é demonstrada uma técnica cujo objetivo é reproduzir os gestos faciais de um indivíduo para outro usando Modelos de Aparência Ativa. Isso é feito

Figura 25 – Classificadores em cascata para determinar gênero e emoção. (a) informação de gênero é considerado quando determina a emoção. (b) informação de emoção é considerado quando determina o gênero.



Fonte: Imagem retirada de (AHLBERG; FORCHHEIMER, 2003)

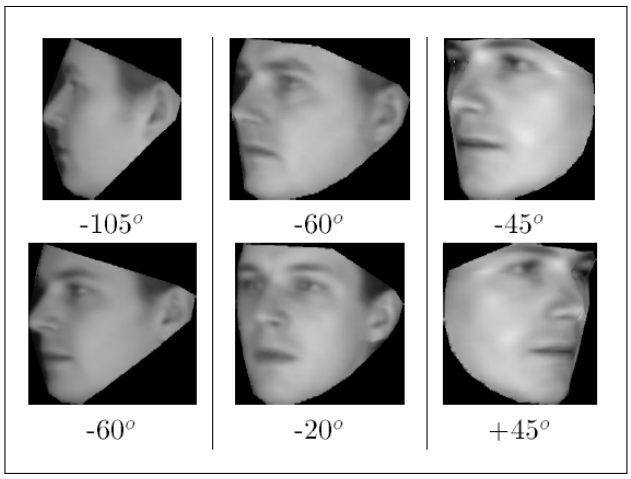
Figura 26 – Exemplo de face sendo capturada de frente e de perfil simultaneamente.

Fonte: Imagem retirada de (COOTES et al., 2000)

através da construção de modelos específicos para cada pessoa e em seguida a elaboração dos mapas dos parâmetros de um modelo para outro, permitindo a transferências dessas expressões em tempo real.

O mapeamento do parâmetros entre modelos, considera que o alinhamento entre

Figura 27 – Faixas de ângulo que podem ser usadas para construção dos modelos.



Fonte: Imagem retirada de (COOTES et al., 2000)

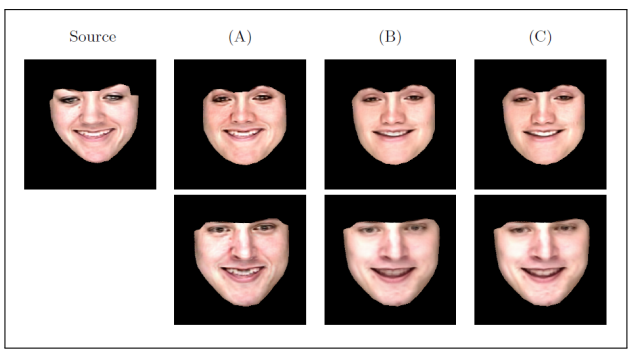
vetores base que produzem um movimento correspondente em modelos distintos é dado pelo produto interno desses vetores $< s_i^s, s_j^* >$. Assim é possível mapear os parâmetros do modelo de acordo com a equação:

$$s^* = s_0^* + S^*(Rp_s), \tag{2.8}$$

em que S^* são os vetores base geradores do espaço alvo, R é uma matriz $q \times r$ de produtos internos, p_s são os parâmetros no espaço original, q é a dimensão do espaço alvo, r é a dimensão no espaço original e s_0^* é a média do modelo.

Além disso, o trabalho também sugere a recuperação de componentes que podem ser perdidas durante o mapeamento. Isso é feito aplicando deslocamentos aleatórios dos vetores do espaço original e observando o resíduos que esses deslocamentos causam no espaço alvo. Assim, é possível perceber quais componentes do espaço original não podem ser reconstruídos no espaço alvo.

Figura 28 – Exemplo de aplicação do procedimento proposto em (THEOBALD et al., 2007)



Fonte: Imagem retirada de (THEOBALD et al., 2007)

A Figura 28 mostra um exemplo de aplicação do procedimento. Nela podemos observar uma imagem original que foi aproximada usando AAM e que seus parâmetros foram transformados para outros dois espaços. Nas imagens superiores temos um modelo gerado a partir de fotos de um indivíduo do sexo feminino, e nas imagens inferiores temos imagens geradas a partir de um indivíduo do sexo masculino. As colunas representam as técnicas empregadas, sendo que em (a) a forma e aparência média do espaço alvo são substituídas no espaço original, em (b) ocorre o mapeamento dos parâmetros de acordo com a Equação 2.8 e em (c) ocorre a compensação através dos resíduos.

Em (EDWARDS et al., 1998) os Modelos de Aparência Ativa são usados em uma aplicação de reconhecimento facial. Essa tarefa é realizada utilizando os parâmetros extraídos do modelo. Uma Análise de Discriminante Linear é aplicada nos parâmetros dos exemplos do conjunto de dados com o objetivo de maximizar a distância entre as classes. Assim, cada exemplo de vetor de parâmetros extraído utilizando o modelo é projetado para um subespaço que é definido como o subespaço identidade.

Um conjunto de treinamento é usado para encontrar o centroide, no subespaço identidade, das classes possíveis. A classificação de uma nova face de teste é então realizada de acordo com o centroide mais próximo.

3 RESULTADOS E DISCUSSÃO

3.1 DADOS E PROCEDIMENTOS EXPERIMENTAIS

Com a finalidade de avaliar o desempenho do procedimento descrito na seção anterior, foram implementados testes utilizando a linguagem de programação C++ e a biblioteca de processamento de imagens OpenCV.

Para a realização dos testes foram reunidas 551 imagens reais de motociclistas com e sem capacete. Desse total, 530 eram imagens de motociclistas com capacete e apenas 21 eram de motociclistas sem capacete. Essa desproporção ocorre porque é raro encontrar motociclistas sem capacete trafegando dentro da zona urbana de uma cidade grande.

Para solucionar essa disparidade foram utilizadas imagens da base de imagens Yaleface database (GEORGHIADES et al., 2001). Essa base foi escolhida por conter imagens de cabeças em diversas poses e ângulos, inclusive em ângulos semelhante as imagens que seriam capturadas por uma câmera de trânsito. Assim, foram selecionadas aleatoriamente 509 imagens nas poses 4, 6 e 7, totalizando 1060 imagens sendo 530 de cada classe. Como será visto, esse balanceamento irá facilitar a análise de sensibilidade dos classificadores.

Os experimentos foram realizados em um computador com processador Intel® Core™ i7 3610QM com 8 GB de memória RAM DDR3 1600 MHz executando o sistema operacional Linux Ubuntu versão 14.04.

Treinamento Classificador Cascata

O primeiro passo da realização dos testes foi a criação do conjunto de treinamento do detector de objetos Viola-Jones. Para esse conjunto foram selecionados 800 imagens, sendo 400 de cada classe. Entre essas imagens, propositalmente, não foi selecionada nenhuma das 21 imagens sem capacete do conjunto original, com o objetivo de reservar essas imagens para os testes.

As imagens escolhidas foram devidamente rotuladas e o treinamento foi realizado utilizando a ferramenta de treinamento de classificador em cascata fornecida com o OpenCV.

Além dessas imagens foram escolhidas 1400 imagens contendo exemplos negativos. Esses exemplos foram retirados do plano de fundo dos exemplos do conjunto de treinamento e de imagens aleatórias obtidas na internet contendo qualquer objeto exceto faces e capacetes.

As imagens utilizadas nos exemplos positivos foram retiradas da base de imagens

e não foram reutilizadas em nenhuma outra etapa do procedimento. Isso foi feito porque não seria possível realizar um grande número de testes sendo necessário executar o treinamento do classificador em cascata diversas vezes, já que observou-se que o treinamento desse classificador pode chegar a mais de 150 horas de execução.

O treinamento foi realizado utilizando uma taxa máxima de falso positivo de $0,01$, um total de 8 estágios de treinamento, profundidade máxima de 4 camadas, um máximo de 5 classificadores fracos e uma janela de 24x24. Para realizar a classificação foram escolhidas as características Haar-*like*. O significado preciso desses parâmetros do treinamento do Viola-Jones pode ser encontrado na revisão apresentada no Apêndice A.

Geração do Modelo e Treinamento do Classificador

Os treinamentos e testes das etapas seguintes do sistema foram realizados utilizando as 260 imagens restantes. Para estimar a performance do método foi utilizada validação cruzada com 10 *folds*. Assim, foram feitas 10 repetições de treinamento e teste. Em cada repetição, 9 partes do conjunto foram utilizados para gerar o Modelo de Aparência Ativa e treinar o classificador e 1 parte para realizar os testes.

A geração do Modelo de Aparência Ativa foi realizada utilizando 30 pontos manualmente marcados em cada imagem como pode ser observado na Figura 29. As Análises de Componentes Principais aplicadas na geração dos modelos de forma e de tons de cinza e na geração do modelo duplo de aparência foram aplicados de maneira a manter 0.99% da variância. Por esse motivo o número de parâmetros dos modelos não é fixo, variando de acordo com o conjunto de treinamento.

Figura 29 – Exemplo de imagem rotulada.

Fonte: Elaborada pelo autor

Para a geração da matriz de regressão utilizada na aproximação do modelo, foram realizadas 50 perturbações aleatórias nos parâmetros de cada exemplo do conjunto de treinamento.

As perturbações aplicadas foram de 2 vezes a variância do parâmetro correspondente.

O treinamento dos classificadores SVM foi realizado utilizando a função de treinamento baseado em validação cruzada disponibilizado pelo OpenCV.

No caso da rede neural MLP, utilizou-se uma topologia com apenas uma camada interna. Levando-se em consideração que o número de parâmetros do modelo pode variar em cada *fold*, pois esse valor é restringido apenas pela quantidade de variância mantida, não foi possível realizar um ajuste fino da topologia que fosse utilizável em todos os *folds* da validação cruzada. Empiricamente, a topologia que apresentou o melhor resultado possui o número de neurônios da camada interna igual ao dobro do número de parâmetros do modelo. A taxa de aprendizagem utilizada foi de $0,1$ e o *momentum* de $0,1$.

O tempo de treinamento do sistema é tomado quase que totalmente pela geração do Modelo de Aparência Ativa. O treinamento do classificador tem pouca influência na medição desse período. Em média o tempo de treinamento para um conjunto de 234 imagens, que correspondem a $9/10$ do conjunto de dados, é de 25 minutos.

3.2 RESULTADOS

Com o objetivo de realizar uma análise de sensibilidade sobre os resultados dos classificadores empregados, os resultados da fase de detecção foram separados dos resultados dos classificadores. O Apêndice F apresenta sobre a construção das matrizes de confusão, dos índices que podem ser extraídos e da construção do gráfico no espaço de Característica de Operação do Receptor (ROC - *Receiver Operating Characteristic*).

Nas matrizes de confusão obtidas de cada avaliação, podemos ver além da acurácia, a taxa de verdadeiros positivos (TVP), também chamada de *recall*; taxa de falso negativo (TFN); taxa de falsos positivos (TFP); taxa de verdadeiro negativo (TVN), também chamado de especificidade; valor previsto positivo (VPP); valor previsto negativo (VPN); taxa de omissão falsa (TOF); taxa de descoberta falsa (TDF).

Localização do Motociclista

Localizar motociclistas na imagem ruidosa da via do trânsito não foi o foco principal desta pesquisa embora seja essencial para o desempenho final de uma software de detecção motociclistas trafegando sem capacete. Também foi necessário para a realização dos testes do

algoritmo de detecção do capacete baseado em AAM. Essa fase requer maior investimento em pesquisa.

Como já foi descrito, para realizar a detecção da localização da cabeça do motociclista, foi utilizado um detector de objetos Viola-Jones. As Tabelas 1 e 2 mostram o acerto e a matriz de confusão do detector.

Tabela 1 – Matriz de confusão do detector Viola-Jones.

<table>
<tr><td></td><td></td><td colspan="2">Condição (“padrão ouro”)</td><td></td><td></td></tr>
<tr><td></td><td>População Total
388</td><td>Positivo
260</td><td>Negativo
128</td><td></td><td></td></tr>
<tr><td rowspan="2">Resultado do Teste</td><td>Positivo
325</td><td>197</td><td>128</td><td>Precision
0,6061</td><td>TDF
0,3938</td></tr>
<tr><td>Negativo
63</td><td>63</td><td>*</td><td>TOF
*</td><td>VPN
*</td></tr>
<tr><td></td><td></td><td>Recall
0,7576</td><td>TFP
*</td><td colspan="2">Fmeasure
0,6735</td></tr>
<tr><td></td><td></td><td>TFN
0,2423</td><td>Specificity
*</td><td colspan="2">Acurácia
0,50773</td></tr>
</table>

Tabela 2 – Taxa de acerto do detector Viola-Jones.

	Média	Desvio Padrão
Taxa de Acerto	0,50773	0,09508

A taxa de acerto foi obtida a partir da execução do detector nos 10 *folds* originalmente separados para o procedimento de validação cruzada. Observou-se uma taxa de acerto média de $50,773\%$ com uma variância de $0,009041$.

Apesar de sugerir que o detector tem desempenho similar a um classificador aleatório, a Tabela 1 mostra que isso não é o caso. Por ser um classificador do tipo unário, o conjunto dos exemplos negativos é composto na verdade por todas as possíveis submatrizes das imagens testadas. Assim, não é interessante medir o conjunto dos exemplos verdadeiros negativos, uma vez que esse número seria esmagadoramente superior aos demais.

Isso também explica o motivo da população total ser superior a quantidade de imagens testadas, uma vez que a população total engloba o conjunto de falsos positivos.

Por não se obter uma taxa de falso positivo não é possível desenhar o espaço ROC dos detectores de objetos, assim está exposto aqui apenas o detector que obteve melhor resultado.

As imagens de motociclista localizadas nesta fase alimentam a fase de classificação. É de destacar que a Taxa de Verdadeiros Positivos foi de 75,76%, significando que quase 76% dos motociclista são localizados, simultaneamente com uma Taxa de Falsos Positivos igual a 0 (zero), significando que não serão processadas imagens sem a presença de motociclista.

Apesar do resultado ser satisfatório, ainda é abaixo de esperado para um detector Viola-Jones. O principal motivo para esse desempenho é a quantidade pequena de exemplos de treinamento a qual o classificador foi exposto. Em (VIOLA et al., 2005), (LIENHART et al., 2003), (KHAMMARI et al., 2005), (KÖLSCH; TURK, 2004), é possível observar que são usados conjuntos de treinamento com números bem superiores ao desse trabalho para a construção de detectores. O esperado é que com a aquisição de mais exemplos os resultados melhorem.

Fase de Classificação

Nesta fase de classificação foram comparados os desempenhos dos classificadores SVM Linear, SVM Polinomial, SVM com núcleo de base radial, e MLP, utilizando as mesmas características. Assim, a Tabela 7 contem as taxa de acerto e variância de cada um dos classificadores. Nas Tabelas 3, 4, 5 e 6 é possível observar as matrizes de confusão de cada classificador assim como o cálculo das métricas de sensibilidade desses classificadores.

Tabela 3 – Matriz de confusão do classificador SVM Linear.

<table>
<tr><td colspan="2" rowspan="2"></td><td colspan="2">Condição (“padrão ouro”)</td><td colspan="2" rowspan="2"></td></tr>
<tr><td>População Total
197</td><td>Positivo
103</td><td>Negativo
94</td></tr>
<tr><td rowspan="2">Resultado do Teste</td><td>Positivo
148</td><td>97</td><td>51</td><td>Precision
0,6554</td><td>TDF
0,3445</td></tr>
<tr><td>Negativo
49</td><td>6</td><td>43</td><td>TOF
0,1224</td><td>VPN
0,8775</td></tr>
<tr><td colspan="2"></td><td>Recall
<u>0,9417</u></td><td>TFP
0,5426</td><td colspan="2">Fmeasure
0,7729</td></tr>
<tr><td colspan="2"></td><td>TFN
0,0583</td><td>Specificity
0,4574</td><td colspan="2">Acurácia
0,7107</td></tr>
</table>

Observando a Tabela 3, contendo a matriz de confusão do classificador SVM com núcleo linear, encontramos o melhor valor negativo positivo entre os classificadores testados além da segunda melhor taxa de positivo verdadeiro. Isso significa que em operação, esse classificador produz a menor quantidade de alertas errados, já que de um total de 49 alertas de condutor sem capacete apenas 6 são incorretos.

Em contrapartida, a taxa de falso positivo é consideravelmente alta, sendo superior a $0,5$. Dessa forma, o esperado é que mais da metade dos alertas de motociclista sem capacete sejam perdidos. Como é possível observar, esses índices resultam na pior taxa de acerto entre os classificadores testados.

Tabela 4 – Matriz de confusão do classificador SVM Polinomial.

<table>
<tr><td></td><td></td><td colspan="2">Condição (“padrão ouro”)</td><td colspan="2"></td></tr>
<tr><td></td><td>População Total
197</td><td>Positivo
103</td><td>Negativo
94</td><td colspan="2"></td></tr>
<tr><td rowspan="2">Resultado do Teste</td><td>Positivo
116</td><td rowspan="2">91</td><td rowspan="2">25</td><td>Precision
0,7845</td><td>TDF
0,2155</td></tr>
<tr><td>Negativo
81</td><td>TOF
0,1481</td><td>VPN
0,8519</td></tr>
<tr><td></td><td></td><td>Recall
0,8835</td><td>TFP
0,266</td><td colspan="2">Fmeasure
<u>0,8310</u></td></tr>
<tr><td></td><td></td><td>TFN
0,1165</td><td>Specificity
0,734</td><td colspan="2">Acurácia
<u>0,8122</u></td></tr>
</table>

Como é possível observar na Tabela 4, o classificador SVM com núcleo polinomial possui a melhor taxa de acerto entre os demais. Além disso, a taxa de verdadeiro negativo é a maior alcançada durante os testes, significando que esse classificador perde a menor quantidade de alerta de motociclistas sem capacete.

Isso combinado com os outros índices bem balanceados demonstra que o SVM com núcleo Polinomial é o melhor classificador para uso geral.

Na Tabela 5 podemos perceber que o classificador SVM no núcleo de base radial possuiu o maior valor positivo previsto. Dessa forma, esse classificador é o que apresenta a menor quantidade de perdas de ocorrências de motociclistas sem capacete. Porém, essa matriz de confusão também nos mostra que esse classificador tende a afirmar que o condutor está sem capacete, mesmo quando isso não é verdade. Isso impacta na taxa de verdadeiro positivo, que é a

Tabela 5 – Matriz de confusão do classificador SVM RBF.

		Condição (“padrão ouro”)			
	População Total **197**	Positivo **103**	Negativo **94**		
Resultado do Teste	Positivo **85**	**73**	**12**	Precision **0,8588**	TDF **0,1412**
	Negativo **112**	**30**	**82**	TOF **0,2679**	VPN **0,7321**
		Recall **0,7087**	TFP **0,1277**	Fmeasure **0,7766**	
		TFN **0,2913**	Specificity **0,8723**	Acurácia **0,7868**	

pior encontrada nos testes.

Tabela 6 – Matriz de confusão do classificador MLP.

		Condição (“padrão ouro”)			
	População Total **197**	Positivo **103**	Negativo **94**		
Resultado do Teste	Positivo **124**	**94**	**30**	VPP **0,7581**	TDF **0,2419**
	Negativo **73**	**9**	**64**	TOF **0,1233**	VPN **0,8767**
		TVP **0,9126**	TFP **0,3191**	Acurácia **0,802**	
		TFN **0,0874**	TVN **0,6809**		

No caso do classificador MLP, como é possível observar pela Tabela 6, encontramos uma taxa de acerto quase tão boa quanto a do classificador SVM com núcleo Polinomial, porém, alcançando uma melhor taxa de positivo verdadeiro e de valor negativo previsto. Isso torna esse classificador uma boa escolha quando se busca uma menor quantidade de falsos alarmes de motociclistas sem capacete.

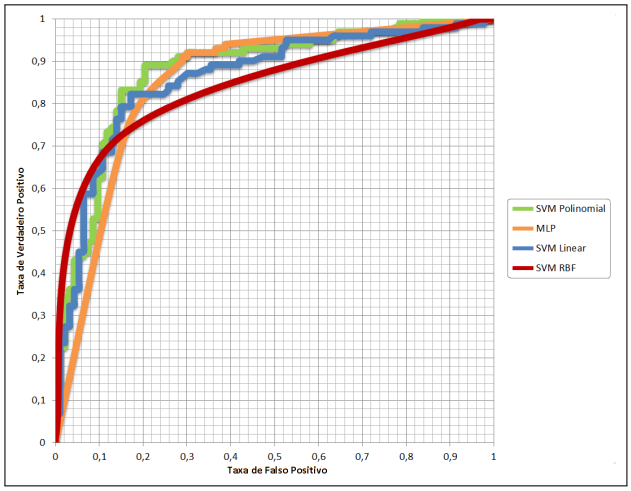
A Tabela 7 compara os resultados médios dos classificadores, mostrando que o classificador SVM com núcleo Polinomial obteve resultado superior. Também é possível observar o *p-value* obtido através de um *t-test*. Para todos os testes o classificador SVM Polinomial foi

Tabela 7 – Taxa de acerto dos classificadores.

	Média	Desvio Padrão	*p-value*
SVM Polinomial	**0,81218**	**0,11199**	
MLP	0,80203	0,08244	✓ 0,61592
SVM RBF	0,7868	0,03917	✓ 0,44565
SVM Linear	0,71066	0,09766	X 0,02059

utilizado como referência. Um critério de $0,05$ foi tomado com o objetivo de determinar se a diferença entre o SVM Polinomial e os demais é significativa. Assim, as hipóteses nulas $\mu_{svmpoly} = \mu_{svmrbf}$ e $\mu_{svmpoly} = \mu_{mlp}$ foram confirmadas (✓) enquando que a hipótese $\mu_{svmpoly} = \mu_{svmlinear}$ foi rejeitada (X). Esse teste mostra que a técnica de extração de características é pouco sensível ao classificador. Apenas no caso de classificador linear, o SVM Linear, é que o resultado tornou-se sensível ao classificador.

Figura 30 – Localização dos classificadores na curva ROC.



Fonte: Elaborado pelo autor

Reconhecidamente, a Curva Característica de Operação (ROC) é uma representação mais rica para a comparação de classificadores em problemas específicos permitindo a visualização, organização e seleção de classificadores com base em suas performances (FAWCETT, 2006). Isso acontece porque os classificadores são comparados não apenas por um número índice mas em todo o seu espectro de operação. Na Figura 30 podemos observar um comparativo das curvas ROC produzidas a partir dos classificadores testados. Este gráfico foi elaborado variando sistematicamente o *threshold* dos classificadores e observando os valores da taxa de verdadeiro

positivo e da taxa de falso positivo em cada uma dessas variações. Essa figura evidencia como os resultados dos classificadores MLP e SVM com núcleo Polinomial são próximos, sendo que o uso do classificador MLP pode ser mais interessante dependendo o objetivo da aplicação.

Tabela 8 – Área sob a Curva ROC dos classificadores.

	AUC
SVM Polinomial	**0,87855**
MLP	0,84935
SVM RBF	0,79380
SVM Linear	0,86169

Em análise baseada em Curvas ROC o parâmetro mais importante para comparar desempenhos é a Área sob a Curva ROC (AUC - *Area under the curve*). Na Tabela 8 é possível observar a AUC dos classificadores testados. Novamente o classificador SVM Polinomial manteve o melhor resultado. Um fato curioso é que o classificador SVM Linear apresentou uma AUC acima do esperado mostrando que ao longo do seu espectro de operação há ocasiões em que esse classicador supera a rede MLP e o SVM RBF.

3.3 DISCUSSÃO

Além dos testes realizados individualmente em cada componente do procedimento, foram feitos testes combinados com o objetivo de observar os resultados reais da técnica.

Tabela 9 – Taxa de acerto combinada do detector de objetos e classificadores.

	Média	Desvio Padrão
SVM Polinomial	**0,65**	**0,10328**
MLP	0,59615	0,09289
SVM RBF	0,62307	0,06486
SVM Linear	0,56153	0,07519

Como é possível observar na Tabela 9, os resultados pioram sensivelmente em decorrência dos erros causados durante a fase de detecção.

O principal motivo dessa diminuição na taxa de acerto é a baixa taxa de positivos

verdadeiros observada na matriz de confusão do detector. Isso porque as imagens onde não é feita detecção são rotuladas como indeterminado e o classificador não chega a ser executado.

Nos trabalhos citados em que a fase de deteção da presença do motocilista foi realizada por vídeo, o algoritmo dispõe de uma sequência de imagens para localizar a região de interesse. Isso cria uma certa redundância, diminuindo o efeito de imagens em que a detecção não é favorável, e permite a remoção do *background*, diminindo a área de busca significativamente. Dessa forma a detecção por vídeo possui uma chance maior de alcançar uma alta taxa de deteção além ser mais eficiente durante a busca em relação à deteção em uma única imagem, que não conta com esses benefícios, como foi feito neste trabalho.

O alto número de falsos positivos encontrado na detecção não chegam a ser problemáticos para o classificador, pois como foi exemplificado na Figura 6, a fase de classificação consegue tratar bem esse tipo de problema.

O tempo de execução do sistema não apresentou uma grande diferença entre os classificadores. A execução de todos os passos do algoritmo utilizando o classificador SVM Linear tomou um tempo médio $0,56661$s por imagem. Com o classificador SVM RBF esse tempo foi de $0,56252$s. No caso do classificador SVM Polinomial o tempo foi de $0,52837$s. E o classificador MLP utilizou em média $0,52296$s por imagem.

Esse tempo de execução é prejudicado pela alta taxa de descoberta falsa apresentada pelo detector ($0,3938\%$). Por esse motivo o ajuste dos parâmetros do AAM comumente é aplicado mais de uma fez em cada imagem. Sendo esse passo o que demanda a maior quantidade de processamento do algoritmo, o esperado é que melhorias no passo de detecção também resultem em melhorias no tempo de execução.

4 CONCLUSÕES E TRABALHOS FUTUROS

4.1 PRINCIPAIS CONCLUSÕES

Imagens de vias de trânsito são imagens ruidosas com grande variação de iluminação, oclusão e presença de confusão no ambiente. Isto torna o problema de localizar o motociclista e discriminar o uso de capacetes um problema desafiador. Os trabalhos anteriores investiram na decisão baseados em vídeo a qual utiliza informações de múltiplas imagens. Entretanto, essa solução dificulta sua integração com outros sistemas que são baseados em uma única imagem. A utilização de um sistema que utiliza uma única imagem para realizar a sua decisão permite que, em momentos de tráfego intenso, seja criado um lote de imagens para posterior processamento. Isso é mais complexo de ser feito em vídeo tendo em vista o fluxo continuo de dados.

As principais conclusões deste trabalho são as seguintes:

a) A identificação do capacete de segurança do motociclista a partir de uma única imagem (e não em um vídeo) é uma necessidade para a sua integração com outros sistemas de vigilância do trânsito que utilizam uma só imagem como, por exemplo, os sistemas de detecção de ultrapassagem do sinal vermelho. Este trabalho mostrou que essa abordagem por uma única imagem torna esse um problema significativamente mais difícil que a abordagem utilizando vídeo.

b) A tarefa foi dividida em dois subproblemas: a detecção e localização da região da cabeça e a discriminação do porte do capacete de segurança. Esta pesquisa focou na discriminação do capacete de segurança, mas constatou que o problema da detecção e localização, nas condições de imagens de vias do trânsito, é um problema de difícil solução por um algoritmo de visão computacional.

c) AAM é frequentemente utilizado em tarefas de segmentação com muitas aplicações em imagens médicas já publicadas. Características extraídas de modelos AAM para classificação são menos frequentes tendo sido encontrados poucos trabalhos na literatura. Características extraídas de modelos AAM se mostraram promissoras na tarefa de discriminação automática do capacete de segurança atingindo acurácia média de 81% sendo este resultado próximo daqueles alcançados em abordagens baseadas em vídeo.

d) A localização da região da cabeça do motociclista foi abordada utilizando o algoritmo de Viola-Jones da biblioteca OpenCV. Essa abordagem alcançou acurácia

máxima de $50,77\%$ o que reduziu substancialmente a acurácia final do sistema todo. Sendo o algoritmo de Viola-Jones reconhecido pelos seus bons resultados em localização de faces e outros objetos, pode-se concluir que este subproblema requer uma nova abordagem para se conseguir alcançar melhores resultados.

Do estudo detalhado do problema e dos resultados obtidos, algumas direções ficaram bem delineadas para a continuidade desta pesquisa. Elas são apresentadas na próxima seção.

4.2 DIREÇÕES DE PESQUISA

Do conhecimento acumulado neste trabalho e das linhas já investigadas, as seguintes direções de pesquisa são consideradas promissoras:

a) O procedimento aplicado não utilizou técnicas de extração e manutenção de modelo de *background*. Explorar o *background* da cena é uma ajuda eficiente em outros algoritmos que tratam de problemas semelhantes. Sendo que o *background* no problema aqui tratado não é estático isso requer algum mecanismo de manutenção do modelo do fundo. Pesquisar uma forma de incluir no algoritmo um mecanismo de extração e manutenção do modelo de *background* é uma direção promissora para melhorar os resultados atuais.

b) Para melhorar a taxa de detecção e localização do motociclista em imagens do trânsito, o algoritmo deve investir na construção de um detector mais vigoroso. O detector de Viola-Jones usado nos testes apresentou excelentes resultados na detecção das imagens de face, porém teve desempenho sofrível na localização de imagens dos motociclistas, quando treinado com o conjunto de dados adquirido.

c) Para melhorar o desempenho na discriminação sobre a presença do capacete novas características de aparência devem ser avaliadas. Foram feitos estudos para o uso de *color tensors*, no entanto estes estudos não chegaram a ser concluídos.

d) Além disso, espera-se que seja possível melhorar ainda mais a fases de extração de características através do emprego de evoluções do algoritmo AAM. Em (COOTES; KITTIPANYA-NGAM, 2002) são expostas algumas variações do algoritmo de busca do AAM, incluindo um AAM baseado em Forma, proposto em (COOTES et al., 1998b), um abordagem composicional na atualização dos parâmetros proposta em (BAKER; MATTHEWS, 2001) e uma maneira de predizer a forma diretamente a partir da textura proposta em (HOU et al., 2001).

REFERÊNCIAS

AHLBERG, J.; FORCHHEIMER, R. Face tracking for model-based coding and face animation. **International journal of imaging systems and technology**, Wiley Online Library, New Jersey, USA, v. 13, n. 1, p. 8–22, 2003.

BAKER, S.; MATTHEWS, I. Equivalence and efficiency of image alignment algorithms. In: IEEE. **Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on**. Kauai, HI, USA, 2001. v. 1, p. I–1090.

CHIVERTON, J. Helmet presence classification with motorcycle detection and tracking. **Intelligent Transport Systems, IET**, IET, Stevenage,Hertfordshire, England, v. 6, n. 3, p. 259–269, 2012.

COOTES, T. F.; EDWARDS, G. J.; TAYLOR, C. J. Active appearance models. In: **Computer Vision, 1998. European Conference on**. Freiburg, Germany: Springer, 1998. p. 484–498.

COOTES, T. F.; EDWARDS, G. J.; TAYLOR, C. J. A comparative evaluation of active appearance model algorithms. In: BRITISH MACHINE VISION ASSOCIATION (BMVA). **British Machine Vision Conference**. Southampton, Hampshire, UK, 1998. v. 98, p. 680–689.

COOTES, T. F.; HILL, A.; TAYLOR, C. J.; HASLAM, J. The use of active shape models for locating structures in medical images. In: SPRINGER. **Information Processing in Medical Imaging**. Flagstaff, Arizona, USA, 1993. p. 33–47.

COOTES, T. F.; KITTIPANYA-NGAM, P. Comparing variations on the active appearance model algorithm. In: BRITISH MACHINE VISION ASSOCIATION (BMVA). **British Machine Vision Conference**. Cardiff, UK, 2002. p. 1–10.

COOTES, T. F.; TAYLOR, C. J.; COOPER, D. H.; GRAHAM, J. Active shape models-their training and application. **Computer vision and image understanding**, Elsevier, Amsterdam, Netherlands, v. 61, n. 1, p. 38–59, 1995.

COOTES, T. F.; WHEELER, G. V.; WALKER, K. N.; TAYLOR, C. J. Coupled-view active appearance models. In: BRITISH MACHINE VISION ASSOCIATION (BMVA). **British Machine Vision Conference**. Bristol, UK, 2000. v. 1, n. 2000, p. 52–61.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine learning**, Springer, Boston, Massachusetts, US, v. 20, n. 3, p. 273–297, 1995.

DENATRAN. **Frota Nacional (Maio de 2015)**. 2015. [Online; acessado em 27 de junho de 2015]. Disponível em: <http://www.denatran.gov.br/frota2015.htm>.

DPVAT. **Boletim Estatístico - Ano 04 - Volume 04**. 2014. [Online; acessado em 27 de junho de 2015]. Disponível em: <http://www.seguradoralider.com.br/SitePages/boletim-estatistico.aspx>.

EDWARDS, G. J.; COOTES, T. F.; TAYLOR, C. J. Face recognition using active appearance models. In: **Computer Vision, 1998. European Conference on**. Freiburg, Germany: Springer, 1998. p. 581–595.

FAWCETT, T. An introduction to roc analysis. **Pattern recognition letters**, Elsevier, Amsterdam, Netherlands, v. 27, n. 8, p. 861–874, 2006.

FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. **Journal of computer and system sciences**, Elsevier, Amsterdam, Netherlands, v. 55, n. 1, p. 119–139, 1997.

GAVRILA, D. M.; PHILOMIN, V. Real-time object detection for "smart" vehicles. In: IEEE. **Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on**. Kerkyra, Greece, 1999. v. 1, p. 87–93.

GEORGHIADES, A.; BELHUMEUR, P.; KRIEGMAN, D. From few to many: Illumination cone models for face recognition under variable lighting and pose. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, LOS ALAMITOS, CA, USA, v. 23, n. 6, p. 643–660, 2001.

HILL, A.; TAYLOR, C. J. Model-based image interpretation using genetic algorithms. **Image and Vision Computing**, Elsevier, Genova, Italy, v. 10, n. 5, p. 295–300, 1992.

HOU, X. W.; LI, S. Z.; ZHANG, H.; CHENG, Q. Direct appearance models. In: IEEE. **Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on**. Kauai, HI, USA, 2001. v. 1, p. I–828.

JOLLIFFE, I. **Principal component analysis**. New Jersey, USA: Wiley Online Library, 2002.

KASS, M.; WITKIN, A.; TERZOPOULOS, D. Snakes: Active contour models. **International journal of computer vision**, Springer, New York, NY, USA, v. 1, n. 4, p. 321–331, 1988.

KHAMMARI, A.; NASHASHIBI, F.; ABRAMSON, Y.; LAURGEAU, C. Vehicle detection combining gradient analysis and adaboost classification. In: IEEE. **Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE**. Vienna, Austria, 2005. p. 66–71.

KÖLSCH, M.; TURK, M. Robust hand detection. In: IEEE. **Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on**. Seoul, South Korea, 2004. p. 614–619.

KOVÁCS, Z. L. **Redes Neurais Artificiais Fundamentos e Aplicações: um texto básico**. São Paulo, SP, Brasil: Editora Livraria da Física, 2006.

KU, M.-Y.; CHIU, C.-C.; CHEN, H.-T.; HONG, S.-H. Visual motorcycle detection and tracking algorithm. **WSEAS Trans. on Electronics**, Athens, GREECE, v. 5, n. 4, p. 121–131, 2008.

LIENHART, R.; KURANOV, A.; PISAREVSKY, V. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In: **Pattern Recognition**. Magdeburg, Germany: Springer, 2003. p. 297–304.

LIENHART, R.; MAYDT, J. An extended set of haar-like features for rapid object detection. In: IEEE. **Image Processing. 2002. Proceedings. 2002 International Conference on**. Rochester, New York, USA, 2002. v. 1, p. I–900.

MCCULLOCH, W. S.; PITTS, W. H. A Logical Calculus of Ideas Immanent in Nervous Activity. **Bullentin of Mathematical Biophysics**, Springer, New York, NY, USA, v. 5, p. 115–133, 1943.

MCINERNEY, T.; TERZOPOULOS, D. Deformable models in medical image analysis: a survey. **Medical image analysis**, Elsevier, v. 1, n. 2, p. 91–108, 1996.

MITCHELL, S. C.; LELIEVELDT, B. P.; GEEST, R. J. Van der; BOSCH, H. G.; REIVER, J.; SONKA, M. Multistage hybrid active appearance model matching: segmentation of left and right ventricles in cardiac mr images. **Medical Imaging, IEEE Transactions on**, IEEE, New York, NY, USA, v. 20, n. 5, p. 415–423, 2001.

MOGHADDAM, B.; PENTLAND, A. Probabilistic visual learning for object detection. In: IEEE. **Computer Vision, 1995. Proceedings., Fifth International Conference on**. Boston, Massachusetts, USA, 1995. p. 786–793.

MOHAN, A.; PAPAGEORGIOU, C.; POGGIO, T. Example-based object detection in images by components. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, IEEE, New York, NY, USA, v. 23, n. 4, p. 349–361, 2001.

MÜLLER, K.-R.; MIKA, S.; RÄTSCH, G.; TSUDA, K.; SCHÖLKOPF, B. An introduction to kernel-based learning algorithms. **Neural Networks, IEEE Transactions on**, IEEE, New York, NY, USA, v. 12, n. 2, p. 181–201, 2001.

PAPAGEORGIOU, C.; POGGIO, T. A trainable system for object detection. **International Journal of Computer Vision**, Springer, New York, NY, USA, v. 38, n. 1, p. 15–33, 2000.

PAPAGEORGIOU, C. P.; OREN, M.; POGGIO, T. A general framework for object detection. In: IEEE. **Computer vision, 1998. sixth international conference on**. Mumbai, India, 1998. p. 555–562.

RUMELHART, D. E.; HINTON, G. E.; J., W. R. Learning representations by back-propagating errors. **Nature**, Nature Publishing Group, London, United Kingdom, v. 323, p. 533–536, 1986.

SAATCI, Y.; TOWN, C. Cascaded classification of gender and facial expression using active appearance models. In: IEEE. **Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on**. Southampton, United Kingdom, 2006. p. 393–398.

SILVA, I. N.; SPATTI, D. H.; FLAUZINO, R. A. **Redes Neurais Artificiais para Engenharia e Ciências Aplicadas**. São Paulo, SP, Brasil: Artliber, 2010.

SILVA, R.; AIRES, K.; SANTOS, T.; ABDALA, K.; VERAS, R.; SOARES, A. Automatic detection of motorcyclists without helmet. In: IEEE. **Computing Conference (CLEI), 2013 XXXIX Latin American**. Naiguatá, Venezuela, 2013. p. 1–7.

SILVA, R.; AIRES, R. V. K.; SANTOS, T.; LIMA, K.; SOARES, A. Automatic motorcycle detection on public roads. **CLEI ELECTRONIC JOURNAL**, Montevideo, Uruguay, v. 16, n. 3, 2013.

TERZOPOULOS, D.; PLATT, J.; BARR, A.; FLEISCHER, K. Elastically deformable models. In: ACM. **ACM Siggraph Computer Graphics**. New York, NY, USA, 1987. v. 21, n. 4, p. 205–214.

THEOBALD, B.-J.; MATTHEWS, I. A.; COHN, J. F.; BOKER, S. M. Real-time expression cloning using appearance models. In: ACM. **Proceedings of the 9th international conference on Multimodal interfaces**. Nagoya, JAPAN, 2007. p. 134–139.

VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: IEEE. **Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on**. Kauai, HI, USA, 2001. v. 1, p. I–511.

VIOLA, P.; JONES, M. Robust real-time object detection. **International Journal of Computer Vision**, Springer, New York, NY, USA, v. 4, p. 51–52, 2001.

VIOLA, P.; JONES, M. J.; SNOW, D. Detecting pedestrians using patterns of motion and appearance. **International Journal of Computer Vision**, Springer, New York, NY, USA, v. 63, n. 2, p. 153–161, 2005.

WEBB, A. R.; D., C. K. **Statistical pattern recognition**. Terceira edição. Chichester, West Sussex, United Kingdom: John Wiley & Sons, 2011.

APÊNDICES

APÊNDICE A – Detecção de Objetos usando Viola-Jones

O problema de detecção de objetos tem sido amplamente pesquisado ao longo dos anos. Diversas técnicas foram desenvolvida com o objetivo de resolver esse problema.

Em (MOGHADDAM; PENTLAND, 1995) é aplicada uma técnica probabilística para realizar a detecção de objetos com base em estimativa de densidades. O espaço de alta dimensão em que se encontram as imagens é transformado para um espaço contendo apenas as suas componentes principais através da técnica PCA. Dois modelos de estimativa de densidade são construídos a partir do conjunto de treinamento: gaussiana multivariada e mistura de gaussianas multivariada. Para realizar a decisão é utilizada um regra de máxima verossimilhança.

Em (GAVRILA; PHILOMIN, 1999) é exposta uma técnica com o objetivo de realizar detecção de objetos (pedestres e sinais de trânsito) em tempo real para auxiliar veículos inteligentes. A técnica descreve o objeto alvo a partir de sua forma, baseando-se em transformada de distância. Para realizar a decisão é criada uma hierarquia de modelos de formas a partir do conjunto de treinamento. As camadas dessa hierarquia são construídas utilizando uma variante do algoritmo de clusterização *k-means*.

Em (PAPAGEORGIOU; POGGIO, 2000) é proposto um sistema para detecção de objetos cujo objetivo é funcionar em cenas confusas e livre de restrições. A técnica descreve uma classe de objetos a partir de características locais construídas a partir da diferença entre regiões adjacentes que são calculadas de maneira similar às *wavelets* de Haar. Para realizar a classificação, um SVM é treinado a partir de um conjunto contendo uma grande quantidade de exemplos positivos e negativos.

Em (MOHAN et al., 2001) é mostrada uma técnica para detectar objetos a partir de componentes. Nesse trabalho são construídos detectores distintos que são treinados para identificar determinados componentes do objeto principal. Depois de determinar se os componentes do objeto realmente estão na imagem, é avaliada a configuração geométrica desses componentes com o objetivo de verificar se o objeto foi realmente encontrado. *Wavelets* de Haar são usadas para representar as características dos componentes que são buscados nas imagens. Para realizar as decisões são treinados classificadores SVM a partir de um conjunto de treinamento.

Para resolver o problema da localização da região do capacete em uma imagem estática, optou-se por utilizar o procedimento proposto em (VIOLA; JONES, 2001a). O principal ponto do detector Viola-Jones é a utilização de características Haar-*like*, vistas anteriormente em (PAPAGEORGIOU; POGGIO, 2000), para descrever regiões das imagens. A extração

dessas características pode ser acelerada através da computação da imagem integral. Outra particularidade da estratégia utilizada é o uso de uma sequência em cascata de classificadores fracos em vez de um único classificador forte, o que diminui consideravelmente o tempo de construção do classificador.

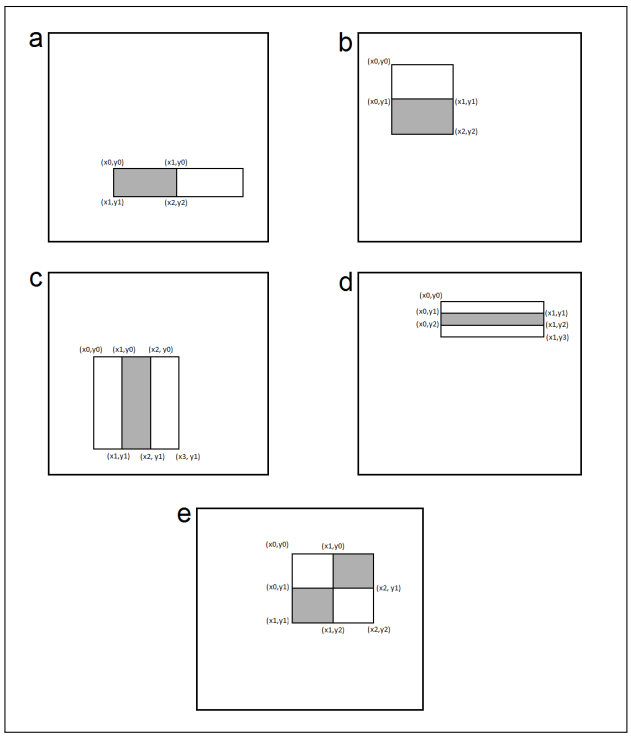
Nas seções a seguir é possível observar um breve referencial teórico esclarecendo os procedimentos do detector Viola-Jones.

Características tipo Haar

Para realizar o julgamento de uma determinada região de uma imagem, o detector Viola-Jones utiliza características similares às *wavelets* de Haar usadas em (PAPAGEORGIOU et al., 1998). Essa características tipo Haar (Haar like features) são compostas de operações de soma e subtração dos valores dos pixels aplicadas em uma determinada região da imagem.

Para a realização dessas operações, a região da imagem é subdividida em retângulos de tamanho igual. É feito então um somatório com os valores das intensidades dos pixels de cada subdivisão. O valor total de cada sub-região é então adicionado ou diminuído dependendo da característica que se deseja obter. O valor resultante dessa soma corresponde a essa característica.

Figura 31 – Exemplos de características tipo Haar que podem ser extraídas de uma imagem.



Fonte: Elaborado pelo autor.

$$HaarLike_A = -\sum_{i=x0}^{x1}\sum_{j=y0}^{y1} I(i,j) + \sum_{i=x1}^{x2}\sum_{j=y0}^{y1} I(i,j). \quad \text{(A.1)}$$

$$HaarLike_B = \sum_{i=x0}^{x1}\sum_{j=y0}^{y1} I(i,j) - \sum_{i=x0}^{x1}\sum_{j=y1}^{y2} I(i,j). \quad \text{(A.2)}$$

$$HaarLike_C = \sum_{i=x0}^{x1}\sum_{j=y0}^{y1} I(i,j) - \sum_{i=x1}^{x2}\sum_{j=y0}^{y1} I(i,j) + \sum_{i=x2}^{x3}\sum_{j=y0}^{y1} I(i,j). \quad \text{(A.3)}$$

$$HaarLike_D = \sum_{i=x0}^{x1}\sum_{j=y0}^{y1} I(i,j) - \sum_{i=x0}^{x1}\sum_{j=y1}^{y2} I(i,j) + \sum_{i=x0}^{x1}\sum_{j=y2}^{y3} I(i,j). \quad \text{(A.4)}$$

$$HaarLike_E = \sum_{i=x0}^{x1}\sum_{j=y0}^{y1} I(i,j) - \sum_{i=x0}^{x1}\sum_{j=y1}^{y2} I(i,j) - \sum_{i=x1}^{x2}\sum_{j=y0}^{y1} I(i,j) + \sum_{i=x1}^{x2}\sum_{j=y1}^{y2} I(i,j). \quad \text{(A.5)}$$

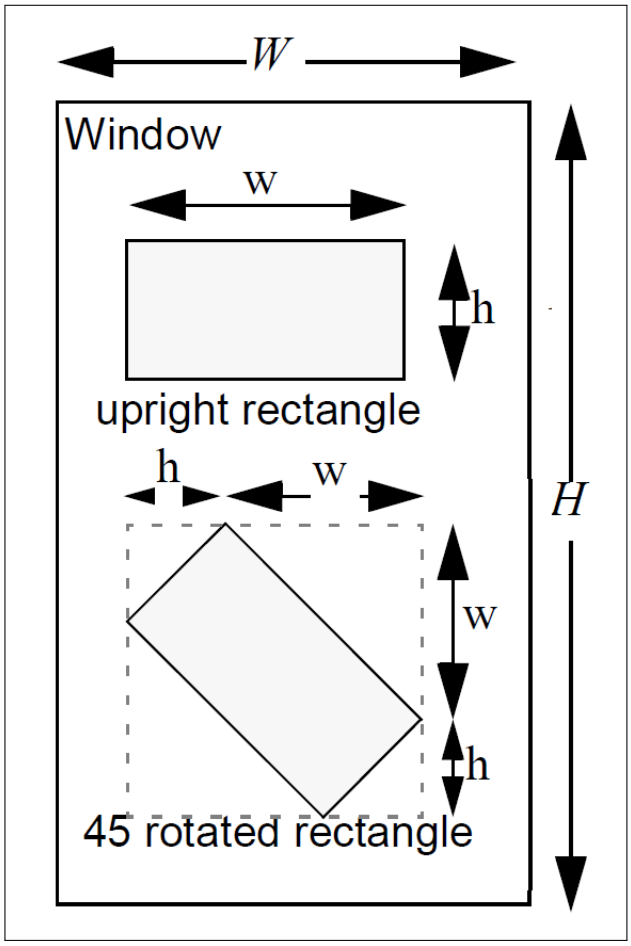
A Figura 31 mostra exemplos de características tipo Haar que podem ser obtidas de uma região de imagem. Em 31.a e 31.b são mostradas duas caraterísticas que realizam duas subdivisões sendo que os valores da subdivisão em branco devem ser somados enquanto que os da região escura devem ser subtraídos, como é possível observar nas Equações A.1 e A.2. Em 31.c e 31.d são mostradas duas caraterística que contém três subdivisões, novamente os valores das regiões em branco devem ser somados enquanto que os das regiões escuras devem ser subtraídos, como é possível observar nas Equações A.3 e A.4. Em 31.e é mostrado um exemplo contendo 4 subdivisões, a regra das somas e subtrações se mantém, como é possível observar na Equação A.5.

Originalmente essas características foram utilizadas apenas vertical e horizontalmente com o objetivo de manter o seu cálculo facilitado pela construção da imagem integral. Porém, em (LIENHART; MAYDT, 2002) foram propostas extensões para o conjunto de possíveis características tipo Haar, incluindo a utilização de retângulos dispostos diagonalmente na imagem. A Figura 32 mostra um exemplo desse tipo de caraterística.

Construção da Imagem Integral

O cálculo dessas características é extremamente agilizado durante o treinamento e execução do detector através da construção de uma imagem integral. Essa imagem é calculada

Figura 32 – Exemplo de retângulo disposto diagonalmente para a extração de uma característica tipo Haar.



Fonte: Imagem retirada de (LIENHART; MAYDT, 2002)

de maneira que o valor de cada pixel (x,y) passa a ser a soma de todos os pixels (x_i,y_i) que satisfazem $x_i \leq x$ e $y_i \leq y$, como pode ser observado na Figura 33.

Assim, o valor da intensidade integral (II) de cada pixel passa a ser:

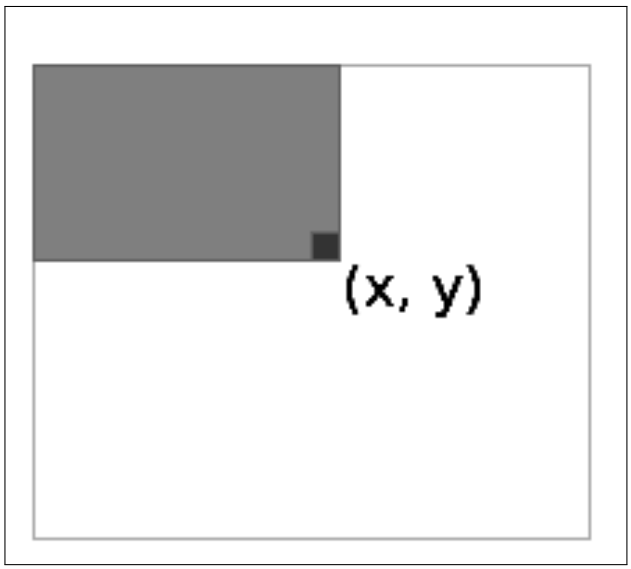
$$II(x,y) = \sum_{i=0}^{x} \sum_{j=0}^{y} I(i,j) \tag{A.6}$$

A partir da Equação A.6 obtém-se a imagem integral que será usada a seguir para agilizar o cálculo das características tipo Haar.

Na Figura 34 é possível observar como o cálculo do somatório de uma sub-região pode ser realizado em tempo constante o cálculo da imagem integral. Assim, a soma das intensidades dos pixels da região D fica igual a $II(D) - II(B) - II(C) + II(A)$.

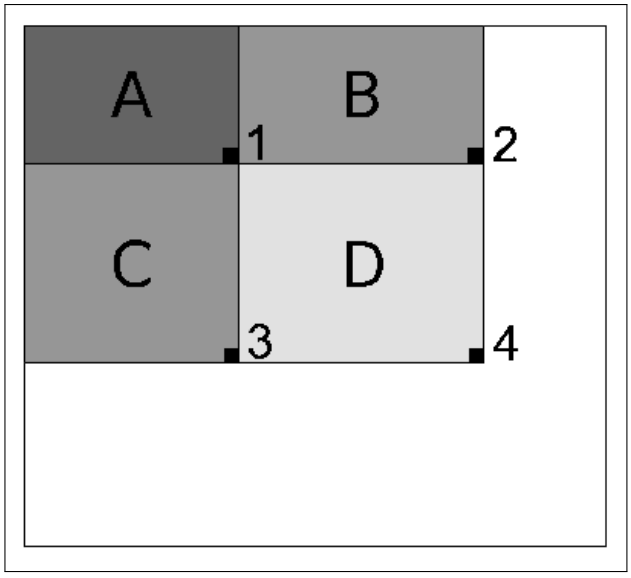
Dessa forma o tempo do cálculo das características tipo Haar fica constante, podendo

Figura 33 – Cálculo da imagem integral.



Fonte: Imagem adaptada de (VIOLA; JONES, 2001a).

Figura 34 – Cálculo do somatório de um retângulo da imagem integral.



Fonte: Imagem adaptada de (VIOLA; JONES, 2001a).

ser realizado para diversas posições e escalas sem que haja grande custo computacional.

Para agilizar a computação de características rotacionadas em 45^o, pode-se construir uma imagem integral rotacionada. Nesse caso, a construção pode ser realizada de maneira iterativa através de duas passagens pela imagem, sendo que os valores iniciais dos pixels deve ser igual aos da imagem original.

A primeira passagem ocorre da esquerda para direita e cima para baixo determinando o valor dos pixel de acordo com:

$$RII(x,y) = RII(x-1,y-1) + RII(x-1,y) + I(x,y) - RII(x-2,y-1),$$

onde:

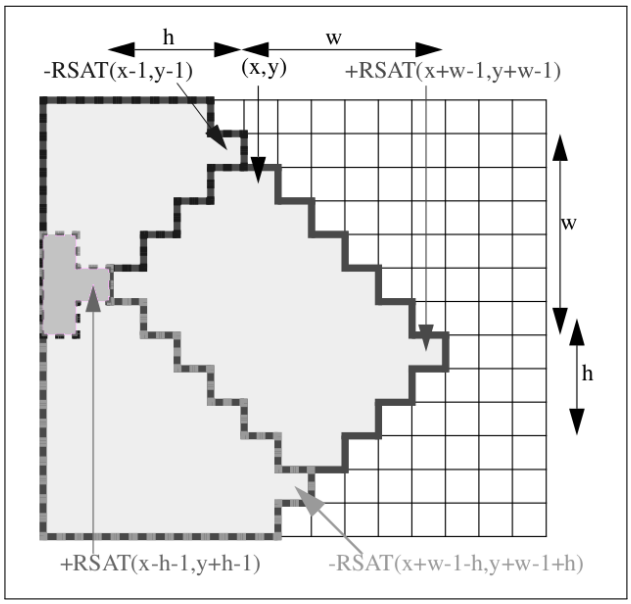
$$RII(-1,y) = RII(-2,y) = RII(x,-1) = 0.$$

Já a segunda passagem ocorre da direita para esquerda e de baixo para cima sendo que o valor dos pixels é calculado de acordo com:

$$RII(x,y) = RII(x,y) + RII(x-1,y+1) - RII(x-2,y).$$

A Figura 35 mostra como a imagem integral pode ser usada para calcular o somatório de região retangular rotacionada em 45^o.

Figura 35 – Cálculo das características tipo-Haar rotacionadas em 45^o.



Fonte: Imagem adaptada de (LIENHART; MAYDT, 2002)

Nesse caso o cálculo do somatório dos pixels dessa região é feito de acordo com:

$$RecSum(r) = RII(x+w,y+w) + RII(x-h,y+h) - RII(x,y) - RII(x+w-h,y+w-h).$$

Classificador Adaboost

Para realização da decisão, o detector Viola-Jones utiliza o algoritmo Adaboost (*Adaptative Boost*). O Adaboost constrói uma série classificadores simples com o objetivo de

uni-los em um único classificador.

Para a construção dos classificadores simples, o Adaboost é rigoroso em selecionar um pequeno conjunto de regras de decisão bem simples. Dessa forma a aprendizagem de um classificador simples (h) é composta da seleção de uma característica (f) e um limiar (θ) que melhor separa corretamente as classes. Assim, uma sub-janela (x) pode ser classificada de acordo com a regra:

$$h(x,f,p,\theta) = \begin{cases} 1 \text{ se } pf(x) < p\theta \\ 0 \text{ se } pf \geq p\theta \end{cases},$$

sendo que p é um fator de polaridade que indica o sinal.

O procedimento para a construção de um classificador Adaboost a partir de várias regras simples pode ser visto no Algoritmo 6.

Algoritmo 6: Treinamento de um classificador Adaboost (VIOLA; JONES, 2001a).

Entrada:

Imagens de treinamento devidamente rotuladas: $(x_0,y_0),(x_1,y_1),...,(x_n,y_n)$ em que y_i é 0 ou 1 dependendo da classe que pertence x_i;

Conjuntos de treinamento contendo exemplos positivos (P) e negativos (N);

Saída: Classificador Adaboost contendo a combinação ponderada de regras simples adicionada de um limiar.

início

Inicializar os pesos $w_{1,i} = \frac{1}{N_n}\frac{1}{N_p}$ em que N_n e N_p são os números de exemplos positivos e negativos;

para $t = 1,..,T$ **faça**

Normalize os pesos fazendo: $w_{t,i} = \frac{w_{t,i}}{\sum_n^{j=1} w_{t,j}}$;

Selecione o melhor classificador fraco utilizando o erro:

$\varepsilon_t = min_{f,p,\theta}(\sum_i w_i |h(x_i,f,p,\theta) - y_i|)$;

Defina $h_t(x) = h(x,f_t,p_t,\theta_t)$ em que f_t, p_t e θ_t são minimizadores de ε_t;

Atualize os pesos: $w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$ onde $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$, e $e_i = 0$ se x_i for classificado corretamente e $e_i = 1$ se x_i for classificado erroneamente;

fim

O classificador forte resultante é: $C(x) = \begin{cases} 1 \text{ se } \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2}\sum_{t=1}^{T}\alpha_t \\ 0 \text{ caso contrário} \end{cases}$ onde $\alpha_t = log\frac{1}{\beta_t}$

fim

Dessa forma, é feita a ordenação de todos os exemplos para cada característica, juntamente com o cálculo do limiar ótimo. Também são calculadas:

1. a soma total dos pesos dos exemplos positivos;
2. a soma total dos pesos dos exemplos negativos;
3. a soma dos pesos positivos abaixo do exemplo atual;
4. a soma dos pesos negativos abaixo do exemplo atual.

Tais somas têm a finalidade de calcular o erro para o limiar localizado entre o exemplo anterior e o atual na lista ordenada.

Apesar de promissor, o resultado obtido com esse classificador ainda é abaixo do esperado, uma vez que com o aumento no número de características, também cresce consideravelmente o seu custo computacional (VIOLA; JONES, 2001a).

Aninhando Classificadores Adaboost em Cascata

Para evitar a necessidade de aumentar a quantidade de características de um classificador, adotou-se a estratégia de construir um classificador maior composto de vários classificadores Adaboost aninhados em cascata. Assim, durante o processo de aprendizagem, são construídos classificadores Adaboost mais simples e portanto mais fracos, porém eficientes em classificar a maioria dos exemplos. Nas etapas posteriores, a construção dos classificadores pode ficar mais rigorosa, buscando assim evitar a classificação de exemplos como falso positivo.

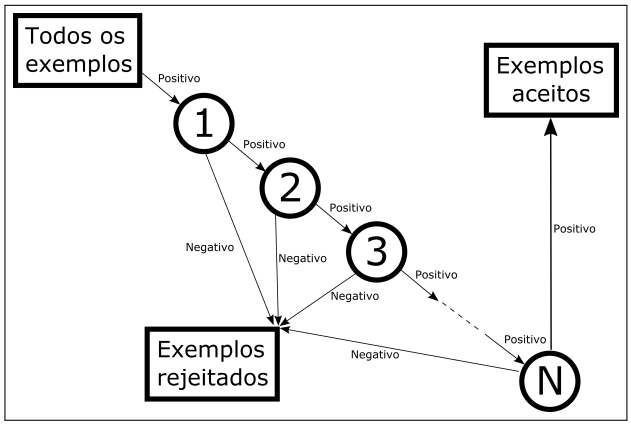
Aplicando essa estratégia é possível obter um desempenho superior e ainda diminuir o custo computacional, uma vez que esses classificadores, por serem mais simples, são mais fáceis de serem construídos.

O resultado final é um classificador formado a partir de uma sequência de classificadores Adaboost aninhados em cascata. Cada classificador da cascata é considerado um estágio do classificador final.

A cascata de classificadores funciona como uma árvore de decisão, onde cada nó pai tem apenas um nó filho. Nessa árvore de decisão, apenas os exemplos considerados positivos são encaminhados para o nó seguinte. Em caso de rejeição, em qualquer estágio, o exemplo é imediatamente considerado negativo e não segue na cascata. Assim, um exemplo só é aceito quando considerado positivo pelo último estágio do classificador, porém qualquer estágio tem poder de eliminação. A Figura 36 exibe esse procedimento. Por esse motivo os estágio, principalmente os iniciais, são treinados de forma que tenham uma taxa de verdadeiro positivo

próxima de 100% porém é aceitável que sejam fracos tendo uma taxa de falso positivo também elevada.

Figura 36 – Fluxo de execução de um classificador em cascata.



Fonte: Figura adaptada de (VIOLA; JONES, 2001a).

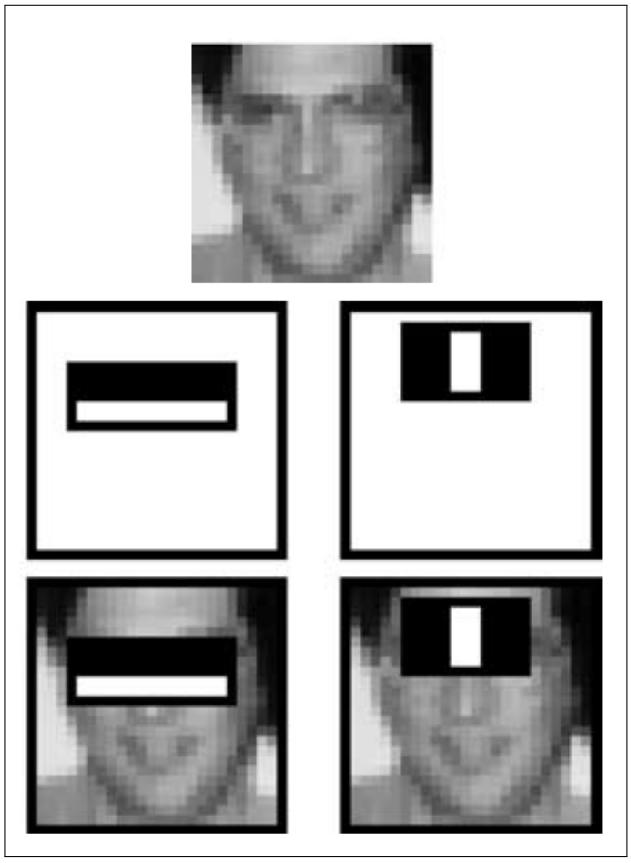
Na Figura 37 é possível observar o primeiro estágio de um classificador treinado em (VIOLA; JONES, 2001a). O classificador desse estágio é relativamente simples sendo composto de apenas duas características. Ele foi projetado para detectar corretamente 100% dos exemplos positivos do conjunto de treinamento mantendo uma taxa de falso positivo de 50%. Dessa forma, apesar de ser considerado um classificador fraco por manter uma taxa de falso positivo elevada, ele é eficiente para a cascata de classificadores pois elimina metade dos exemplos negativos.

Durante o treinamento do classificador em cascata, cada novo estágio acrescentado é treinado a partir dos exemplos que passaram pelos estágios anteriores. Dessa forma, os classificadores mais profundos têm que lidar com uma tarefa mais difícil e o esperado é que, dada uma taxa de acerto designada, esses classificadores apresentem taxas de falso positivo mais elevadas (VIOLA; JONES, 2001a).

A construção dos estágios do classificador em cascata é feita seguindo metas de desempenho. Esse desempenho do classificador é medido em termos de taxa de verdadeiro positivo (taxa de detecção) e taxa de falso positivo. As metas de desempenho do classificador dependem da aplicação. Uma vez que elas sejam estabelecidas, cada estágio pode ser construído de forma que seja o mais simples possível, porém garantindo que a meta final seja cumprida.

A taxa de detecção de um classificador em cascata, é definida em termo das taxas de detecção de cada um de seus estágios como pode ser visto na Equação A.7. De forma semelhante

Figura 37 – Características tipo Haar usadas no primeiro estágio de treinamento de um classificador em cascata.



Fonte: Figura retirada de (VIOLA; JONES, 2001a).

a Equação A.8 mostra a definição da taxa de falso positivo.

$$D = \prod_{i=1}^{K} d_i. \tag{A.7}$$

$$F = \prod_{i=1}^{K} f_i. \tag{A.8}$$

Sendo que na Equação A.7, D é a taxa de detecção do classificador em cascata, K é o número de estágios desse classificador e d_i é a taxa de detecção correspondente ao i-ésimo estágio de classificação. No caso da Equação A.8, F é a taxa de falso positivo do classificador em cascata, K é o número de estágios do classificador e f_i é a taxa de falso positivo referente ao i-ésimo estágio.

Um vez definidas as metas do classificador em cascata, é fácil determinar as metas individuais de cada classificador. Se um classificador, por exemplo, precisar alcançar a taxa de

detecção de 90% (0,9), cada um de seus estágios deve conseguir detectar 99% dos exemplos ($0.99^{10} \approx 0.9$).

Parece que a construção da cascata não faz sentido já que os classificadores que a compõem precisão atingir uma taxa de detecção tão alta, porém não é o caso. É fácil construir um classificador com taxa de detecção de 99% quando sua taxa de falso positivo é relativamente alta. No exemplo dado anteriormente, o classificador relativamente simples atinge uma taxa de detecção de 100% com uma taxa de falso positivo de 50%. Levando em consideração que $0,5^{10} = 0.009765625$ um classificador final construído a partir de classificadores que mantém a mesma meta do exemplo citado, teria uma taxa de falso positivo final de aproximadamente 0,1% o que pode ser uma taxa aceitável dependendo da aplicação e é alcançável a partir da construção de classificadores relativamente simples.

O treinamento de um classificador em cascata alcança bons resultados usando classificadores fáceis de serem construídos que buscam alcançar altas taxa de detecção às custas de também altas taxas de falso positivo. Essa estratégia é mais eficiente que a construção de um único classificador Adaboost cujo objetivo é minimizar o erro, o que só pode ser feito através do aumento da sua complexidade.

A meta de desempenho de cada classificador pode ser alcançada ajustando o limiar que o mesmo possui. Limiares maiores resultam em uma menor taxa de detecção além de uma menor taxa de falso positivo. A diminuição do valor do limiar, resulta em no aumento da taxa de detecção e por consequência também se aumenta a taxa de falsos positivos.

Dada uma meta de desempenho geral para o classificador em cascata, as metas individuais de cada um de seus estágios devem ser balanceadas de maneira que a meta geral seja alcançada. Porém essa é um tarefa de alta complexidade. O que normalmente é feito é permitir ao usuário determinar as metas de desempenho mínimas de cada estágio e a partir dessas metas determinar quantos estágios o classificador em cascata deverá possuir.

Em cada adição de estágio durante o treinamento do classificador em cascata, um classificador Adaboost é treinado como descrito no Algoritmo 6. Nesse classificador Adaboost vão sendo adicionadas características até que suas metas de desempenho (taxa de detecção e de falso positivo) sejam alcançadas.

Quando o treinamento de um estágio é concluído, o conjunto de exemplos negativos da próxima etapa do classificador é construído a partir de todas a falsas detecções realizadas pelo classificador parcialmente montado. Dessa forma, a medida que os estágios vão progredindo, o

classificador se torna mais resistente a essas falsas detecções.

Novos estágios são adicionados ao classificador até que suas metas globais sejam alcançadas. O Algoritmo 7 mostra esse procedimento.

Algoritmo 7: Treinamento do classificador em cascata (VIOLA; JONES, 2001b).

Entrada:

Valores de d e f que são os valores mínimos de taxa de detecção e de falso positivo para cada estágio do conjunto de treinamento;

Valor de falso positivo global F_t;

Conjuntos de treinamento contendo exemplos positivos (P) e negativos (N);

Saída: Classificador em cascata final contendo a sequência de classificadores fracos.

início

- $F_0 = 1,0$;
- $D_0 = 1,0$;
- $i = 0$;
- **enquanto** $F_i > F_t$ **faça**
  - $i = i + 1$;
  - $n_i = 0$;
  - $F_I = F_{i-1}$;
  - **enquanto** $F_i > f \times F_{i-1}$ **faça**
    - $n_i = n_i + 1$;
    - Os conjunto P e N são usados para treinar um classificador Adaboost contendo n_i características;
    - O classificador em cascata é avaliado usando um conjunto de validação para determinar os seus valores de F_i e D_i;
    - O limiar do i-ésimo estágio é diminuído até que o classificador em cascata atual até que sua taxa de detecção seja de no mínimo $d \times D_{i-1}$, afetando também F_i ;
  - **fim**
  - $N = \varnothing$;
  - **se** $F_i > F_t$ **então**
    - O classificador em cascata atual avalia todos os exemplos do conjunto de imagens negativas e os casos de falso positivo são inseridos em N;
  - **fim**
- **fim**

fim

Em (VIOLA; JONES, 2001a) é observado que durante o procedimento de treinamento do classificador em cascata, ao passo que os estágios se sucedem, a construção do conjunto de exemplos negativos para o treinamento do próximo estágio passa a consumir mais e mais tempo, se tornando a etapa mais custosa do treinamento. Isso ocorre porque a medida que o classificador vai alcançando a sua meta geral, ele passa a ter que realizar testes em um número gigantesco de janelas de imagens em busca de atingir as metas das etapas sucessivas.

APÊNDICE B – Máquinas de Vetores de Suporte

A ideia básica das Máquinas de Vetores de Suporte consiste de mapear vetores de padrões de múltiplas classes em um espaço de características de alta dimensão, onde é possível construir o melhor hiperplano que separe esses padrões de acordo com suas classes (WEBB; D., 2011).

A construção do hiperplano que separa as classes pode ser aplicada diretamente sobre o espaço original dos padrões, ou é possível que esses padrões possam ser projetados para um novo espaço de características como mostrado em (CORTES; VAPNIK, 1995).

Quando a separação é feita no espaço original, não ocorre projeção dos dados e apenas é possível separar completamente classes de padrões que por natureza já sejam linearmente separáveis.

Quando as classes não são linearmente separáveis, não é possível a construção de um hiperplano que apresente uma boa separação entre as mesmas. Nesse caso é possível projetar essas classes em um novo espaço que facilite a construção desse hiperplano.

Nas subseções a seguir é explicado como é feita a construção do hiperplano quando as classes são linearmente separáveis e como podem ser feitas projeções desses dados para novos espaços que facilitem a construção do hiperplano.

Construção do Hiperplano entre Classes Linearmente Separáveis

Quando duas classes são linearmente separáveis, é possível construir diversos hiperplanos que discriminem esses dois agrupamentos. O SVM busca construir esse hiperplano de maneira que maximize essa separação entre as classes.

No caso em que temos um conjunto de padrões de treinamento linearmente separáveis $x_1, x_2, ..., x_n$ em que cada padrão pode ser associado a uma entre duas classes, ω_1 e ω_2, onde o rótulo de cada padrão pode ser $y_i = \pm 1$, pode-se separar essas classes através da função discriminante:

$$g(x_1) = w^T x + w_0,$$

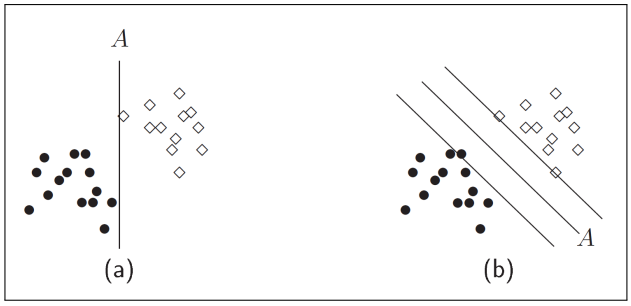
onde a regra de decisão é:

$$g(x_i) = \begin{cases} > 0 \\ < 0 \end{cases} \Longrightarrow x_i \in \begin{cases} \omega_1 \text{ que corresponde a } y_i = -1 \\ \omega_2 \text{ que corresponde a } y_i = 1 \end{cases} .$$

Dessa forma temos que a função discriminante deve assumir um valor negativo quando o padrão pertence a classe ω_1 ou um valor positivo quando o padrão pertence a classes ω_2. Observe que, como o conjunto de treinamento é linearmente separável, não é esperado que nenhum dos padrões possua regra de decisão $g(x_i) = 0$. Isso significaria que algum dos exemplos do conjunto se encontra sobre o limiar de separação e, portanto, o conjunto não é linearmente separável. Dessa forma, também é possível descrever o conjunto de acordo com a inequação:

$$y_i(w^T x_i + x_0) > 0 \quad \text{para todo } i.$$

Figura 38 – Conjunto de dados composto por duas classes linearmente separáveis com dois exemplos de hiperplanos de separação.



Fonte: Figura retirada de (WEBB; D., 2011)

Como pode ser observado na Figura 38, existem diversas maneiras de construir o hiperplano que realize a separação entre as classes. Um classificador de margem máxima busca construir um hiperplano tal que a soma das suas distâncias para o exemplo mais próximo da classe ω_1 e o exemplo mais próximo da classe ω_2 seja a maior possível.

Assim podemos determinar uma margem $b > 0$ que garanta uma distância mínima entre os exemplos e o hiperplano separador, como pode ser observado na inequação:

$$y_i(w^T x_i + x_0) > b.$$

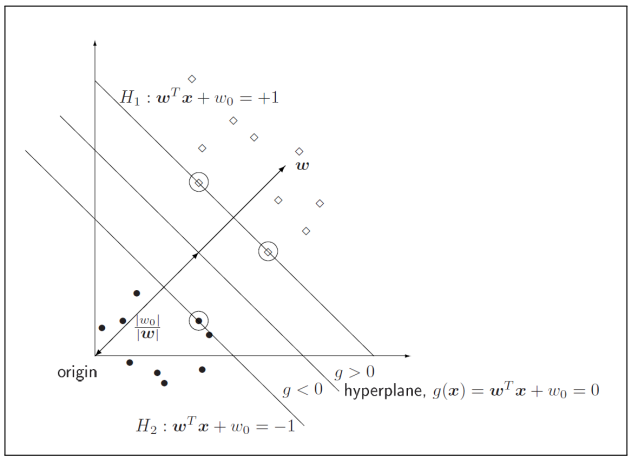
Assim fica garantido que todos os exemplos do conjunto de treinamento estejam a uma distância mínima de $b/|w|$ do hiperplano separador.

É possível escalar os valores de w, w_0 e b de maneira que os pontos mais próximos do hiperplano separador satisfaçam a equação:

$$|w^T x_i + w_0| = 1$$

obtendo assim os hiperplanos canônicos (MÜLLER et al., 2001).

Figura 39 – Hiperplanos canônicos extraídos de um conjunto cujas classes são linearmente separáveis.



Fonte: Figura retirada de (WEBB; D., 2011)

A Figura 39 mostra como função $g(x) = w^T x + w_0$ projeta os exemplos do conjunto de treinamento em uma reta perpendicular ao hiperplano separador. Assim é possível determinar a classe que o exemplo pertence dependendo do sinal da função $g(x)$. É notável também que os hiperplanos canônicos ($H_1 : w^R + w_0 = +1$ e $H_1 : w^R + w_0 = +1$) são construídos a partir dos exemplos mais próximos do hiperplano separador. Esse exemplos que descrevem os hiperplanos separadores são os chamados vetores de suporte.

A distância entre cada hiperplano canônico e o hiperplano separador ($g(x) = 0$) é $1/|w|$, já a margem é a distância entre os hiperplanos canônicos, ou seja $2/|w|$.

$$\begin{aligned} w^T x_i + w_0 &\geq +1 \qquad \text{para } y_i = +1 \\ w^T x_i + w_0 &\geq -1 \qquad \text{para } y_i = -1 \end{aligned} \quad . \tag{B.1}$$

Nas Inequações B.1 temos a regra de classificação dos exemplos do conjunto de treinamento levando em consideração a margem estabelecida. Como a meta durante a construção

do classificador é maximizar essa margem, o objetivo passa a ser encontrar uma solução que minimize $|w|$ sujeito as restrições:

$$R : y_i(w^T x_i + w_0) \geq 1 \quad i = 1, ..., n. \tag{B.2}$$

Esse problema de otimização que envolve tanto igualdades quanto desigualdades, pode ser expresso utilizando o formalismo de Lagrange. Assim, o problema de minimizar $|w|$ é equivalente ao de encontrar o ponto de sela da função L_p definida na equação:

$$L_p = \frac{1}{2} w^T w - \sum_{i=1}^{n} \alpha_i (y_i(w^T x_i + w_0) - 1), \tag{B.3}$$

onde L_p é minimizado em relação a w e w_0 e maximizado em relação a α_i.

No contexto desse problema de otimização, as condições *Karush-Kuhn-Tucker* (KKT) garantem condições necessárias e suficientes que devem ser satisfeitas quando o problema é sujeito a condições de igualdade e desigualdade. Em particular a condição complementar KKT (Equação B.4) implica que em restrições ativas (quando $y_i(w^T x_i + w_0) = 1$) temos que $\alpha_i \geq 0$ enquanto que em restrições inativas temos que $\alpha_i = 0$.

$$\alpha_i (y_i(x_i^T w + w_0) - 1) = 0 \tag{B.4}$$

Dessa maneira, nos casos em temos o multiplicador de Lagrange diferente de zero, os pontos correspondentes se posicionam sobre os hiperplanos canônicos. Sendo esses os chamados vetores de suporte.

Diferenciando L_p em relação a w_0 e w, é possível obter as equações:

$$\sum_{i=1}^{n} \alpha_i y_i = 0, \tag{B.5}$$

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i. \tag{B.6}$$

A substituição de B.5 e B.6 na Equação B.3 resulta na forma dual do problema que pode ser observada na equação:

$$L_D = \sum_{i=1}^{n} \alpha - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j. \tag{B.7}$$

A forma dual do problema permite a sua resolução através de algoritmos de programação quadrática. Uma vez que os multiplicadores de Lagrange tenham sido obtidos, o valor de w_0 pode ser obtido através da condição complementar KKT (Equação B.4) usando qualquer um dos vetores de suporte. A solução de w pode ser obtida através da Equação B.6.

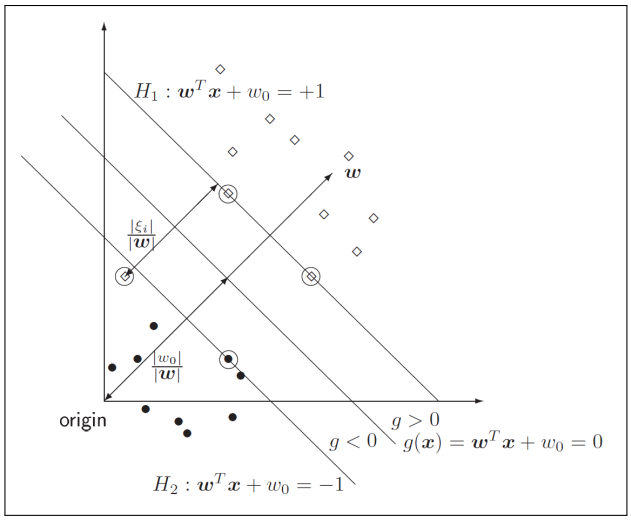
Dessa forma, um novo padrão x pode ser classificado de acordo com o sinal de $x^T x + w_0$, onde substituindo w e w_0, temos x pertence a classe ω_1 se for verdadeira a condição:

$$\sum_{i \in SV} \alpha_i y_i x_i^T x - \frac{1}{n_{SV}} \sum_{i \in SV} \sum_{j \in SV} \alpha_i y_i x_i^T x_j + \frac{1}{n_{SV}} \sum_{i \in SV} y_i > 0.$$

Construção do Hiperplano entre Classes não Separáveis Linearmente

Nos casos em que os dados do conjunto de treinamento não são linearmente separáveis, como está exposto na Figura 40 não é possível encontrar uma hiperplano que realize com sucesso a discriminação entre os exemplos.

Figura 40 – Hiperplanos canônicos extraídos de um conjunto cujas classes não são linearmente separáveis.



Fonte: Figura retirada de (WEBB; D., 2011)

Entretanto, é possível adaptar as ideias descritas anteriormente relaxando as restrições através da introdução das variáveis de folga ξ_i, $i = 1, ..., n$ como pode ser visto nas inequações:

$$\begin{aligned} w^T x_i + w_0 &\geq +1 - \xi_i \qquad \text{para } y_i = +1 \\ w^T x_i + w_0 &\geq -1 + \xi_i \qquad \text{para } y_i = -1 \\ \xi_i &\geq 0 \end{aligned} \quad .$$

Dessa forma, para que um exemplo i seja classificado incorretamente, é necessário que $\xi_i > 1$.

Assim, como acontece em B.2, as restrições do problema podem ser escritas como exposto em:

$$\begin{matrix} y_i(w^T x_i + w_0) \geq 1 - \xi_i \quad i = 1, ..., n \\ \xi_i \geq 0 \end{matrix} . \tag{B.8}$$

Pensando nas variáveis de folga como um tipo de penalidade causada por erros na classificação, podemos adicionar essa penalidade na função objetiva do problema acrescentando o termo $C\sum_i \xi_i$. Sendo que C é um parâmetro que quanto menor o seu valor, menor a penalidade para os exemplos fora do comum.

Assim, temos que objetivo do problema é minimizar B.9 de acordo com as restrições B.8.

$$\frac{1}{2} w^T w + C \sum_i \xi_i. \tag{B.9}$$

Novamente podemos escrever a forma primal desse problema usando formalismo Lagrangiano como pode ser visto na equação:

$$L_p = \frac{1}{2} w^T w + C \sum_i \xi_i - \sum_{i=1}^{n} \alpha_i (y_i (w^T x_i w_0) - 1 + \xi_i) - \sum_{i=1}^{n} r_i \xi_i, \tag{B.10}$$

onde $\alpha_1 \geq 0$ e $r_1 \geq 0$ são os multiplicadores de Lagrange e r_i é introduzido para garantir positividade de ξ_i.

Nesse caso condições complementares KKT são como mostradas nas equações:

$$\begin{matrix} \alpha_i (y_i (x_i^T w + w_0) - 1 + \xi_i) = 0 \\ r_i \xi_i = (C - \alpha_i) \xi_i = 0 \end{matrix} , \tag{B.11}$$

onde os padrões onde $\alpha_i > 0$ correspondem aos vetores de suporte e satisfazem $y_i(x_i^w + w_0) - 1 + \xi_i = 0$.

Aqueles padrões onde $0 < \alpha_i < C$ se posicionam sobre um do hiperplanos canônicos (esses recebem a denominação de vetores de suporte de margem), e portanto possuem $\xi_i = 0$.

Variáveis de folga diferentes de zero ocorrem quando $\alpha_i = C$. Nesse caso, os padrões são classificados incorretamente se $\xi_i > 1$. Quando $\xi_i > 1$ o padrão é classificado corretamente, porém se encontra mais próximo do hiperplano separador do que os vetores de suporte.

Diferenciando B.10 em relação a w e w_0, novamente se obtém os resultados de B.5 e B.6.

$$\sum_{i=1}^{n} \alpha_i y_i = 0. \tag{B.5}$$

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i. \tag{B.6}$$

Realizando a diferenciação em relação a ξ_i obtemos a equação:

$$C - \alpha_i - r_i = 0. \tag{B.12}$$

Aplicando B.5, B.6 e B.12 na forma primal (Equação B.10), obtemos a sua forma dual:

$$L_D = \sum_{i=1}^{n} \alpha - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j. \tag{B.13}$$

Sendo que a forma dual B.13 é igual a B.7 porém sendo sujeita as restrições:

$$\sum_{i=1}^{n} a_i y_i = 0$$
$$0 \leq a_i \leq C$$

Assim o que muda no problema de maximização é apenas o limite máximo de α_i. Novamente algoritmos de programação quadrática podem ser utilizados para resolver o problema.

Utilizando Transformações para Mudança de Espaço

Em grande parte das aplicações do mundo real, a construção de hiperplanos para separar classes de padrões não costuma apresentar bons resultados porque raramente essas classes são linearmente separáveis.

Uma forma de resolver essa dificuldade é realizar a separação dos padrões após eles serem projetados em um novo espaço onde as classes podem ser linearmente separáveis.

Nesse caso o classificador utiliza uma função discriminante exposta na equação:

$$g(x_i) = w^T \phi(x_i) + w_0$$

Onde a sua regra de decisão pode ser vista em:

$$w^T\phi(x_i)+w_0=\begin{cases}>0\\<0\end{cases}\Longrightarrow x_i\in\begin{cases}\omega_1 \text{ que corresponde a } y_i=-1\\\omega_2 \text{ que corresponde a } y_i=1\end{cases},$$

onde ϕ é uma função não linear que transforma x em um novo espaço que facilite a sua separação.

Levando em consideração essa transformação, temos que a forma dual do problema de otimização pode ser observada na equação:

$$L_D=\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j\phi^T(x_i)\phi(x_j). \tag{B.14}$$

Nesse caso, $y_i=\pm 1$, $i=1,...,n$ são os rótulos das classes correspondentes e $\alpha_i=\pm 1$, $i=1,...,n$ são os multiplicadores de Lagrange, que satisfazem as restrições:

$$\sum_{i-1}^{n}\alpha_i y_i=0, \tag{B.15}$$

$$0\leq\alpha_i\leq C, \tag{B.16}$$

sendo que C é um parâmetro de regularização.

Novamente, maximizar B.14 sujeito as restrições B.15 e B.16 resulta na identificação dos vetores de suporte através de valores de α_i diferentes de zero.

A solução de w passa a ser na equação:

$$w=\sum_{i\in SV}\alpha_i y_i\phi(x_i).$$

Para descobrir w_0 podemos utilizar:

$$w_0=\frac{1}{n_{\widetilde{SV}}}\left\{\sum_{i\in\widetilde{SV}}y_i-\sum_{i\in SV,j\in\widetilde{SV}}\alpha_i y_i\phi^T(x_i)\phi(x_j)\right\}, \tag{B.17}$$

onde SV é o conjunto do vetores de suporte associados aos valores de α_i que satisfazem $0<\alpha_i\leq C$, enquanto que $\widetilde{SV}$ é o conjunto de vetores de suporte que satisfazem $0<\alpha_i<C$ e $n_{\widetilde{SV}}$ é a quantidade de elementos nesse conjunto.

A classificação de um novo exemplo x, é possível através do sinal de:

$$g(x)=\sum_{i\in SV}\alpha_i y_i\phi^T(x_i)\phi(x)+w_0. \tag{B.18}$$

O produto escalar entre vetores transformados para um novo espaço através de ϕ é usado em B.17 e B.18 e por questão de simplicidade pode ser substituído por K (Equação B.19).

Dessa forma é possível evitar calcular a transformação $\phi(x)$ explicitamente e, em vez do produto escalar, utilizar $K(x,y)$.

$$K(x,y) = \phi^T(x)\phi(y). \tag{B.19}$$

A função discriminante passa a ser a descrita em:

$$g(x) \sum_{i \in SV} \alpha_i y_i K(x_i,x) + w_0. \tag{B.20}$$

A principal vantagem de utilizar essa substituição é que durante o algoritmo de aprendizagem pode-se utilizar apenas K. Assim não é necessário saber explicitamente a função ϕ, desde que esse núcleo possa ser escrito como um produto interno.

Assim como ocorre no caso linear, é possível aplicar programação quadrática para resolver o problema de otimização e obter os valores de α_i, porém nesse caso os produtos escalares entre os vetores devem ser substituídos pelas avaliações desse núcleo. Para isso é possível utilizar o algoritmo de decomposição de Osuna ou o algoritmo de Otimização Sequencial Mínima (SMO - Sequential Minimal Optimization).

Escolha do Núcleo e ajustes de seus Parâmetros

Diversos tipos de núcleo podem ser utilizados juntamente com uma máquina de vetores de suporte. Nesse trabalho são utilizados os núcleos polinomial como pode ser visto em B.21 e de função base radial (gaussiano) como pode ser visto em B.22.

$$K(x,y) = (r + \lambda x^T y)^p, \quad onde \lambda > 0. \tag{B.21}$$

$$K(x,y) = exp(-|x-y|^2/\sigma^2). \tag{B.22}$$

A capacidade de generalização de uma máquina de vetores de suporte, depende da escolha de seu tipo de núcleo, dos parâmetros desse núcleo e do parâmetro de regularização C, que penaliza os erro no treinamento.

Em geral, para a maioria do núcleos, é possível encontrar valores de parâmetros em que as classes sejam completamente separáveis. Porém isso pode levar ao super-ajuste do modelo ao conjunto de treinamento, deixando-o fraco ao tentar generalizar dados não vistos.

Uma forma de tratar esse problema é dividir o conjunto de treinamento, reservando um subconjunto de validação para verificar a performance do classificador a medida que seus parâmetros variam. Isso pode ser feito através de validação cruzada.

Para encontrar os parâmetros do núcleo, diversas técnicas podem ser aplicadas, desde o uso de grades de busca que variam os valores dos parâmetros seguindo uma determinada estratégia, ou a utilização de técnicas bio-inspiradas, como algorítimos genéticos ou otimização por enxame de partículas.

APÊNDICE C – Perceptron de Múltiplas Camadas

As RNAs (Redes Neurais Artificiais) são modelos computacionais inspirados no sistema nervoso de seres vivos que possuem capacidade de aquisição e manutenção do conhecimento. Elas podem ser definidas como um conjunto de unidades de processamento, que representam os neurônios, interligadas por um grande número de interconexões e representadas por matrizes de pesos sinápticos (SILVA et al., 2010).

A rede MLP proposta em (RUMELHART et al., 1986) foi uma das responsáveis pelo ressurgimento do interesse em RNAs, principalmente graças à sua gama de aplicações.

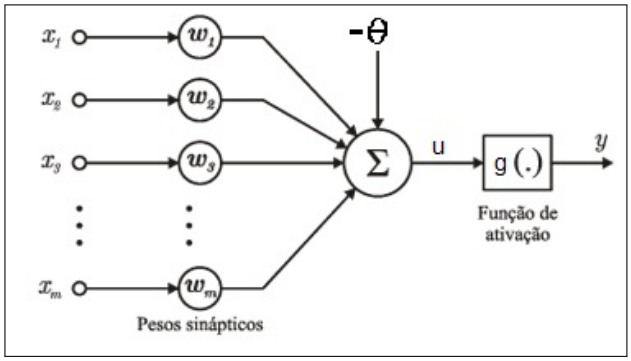
A rede MLP possui sua arquitetura em várias camadas, isso permite que ela possa realizar classificações em conjuntos de dados não linearmente separáveis. Para permitir o ajuste de pesos nas diversas camadas, a rede MLP utiliza o algoritmo *Backpropagation* que é baseado na regra delta generalizada.

Neurônio Artificial

Quando surgiu o primeiro modelo de RNA em 1943, proposto em (MCCULLOCH; PITTS, 1943), a rede neural consistia basicamente de um dispositivo binário com valores arbitrários como entrada, podendo ser excitatórios ou inibitórios, sendo a saída pulso ou não pulso (KOVÁCS, 2006).

Os neurônios das Redes Neurais Artificiais são os elementos de processamento básico da formação das mesmas. O neurônio artificial possui uma entrada de dados, que corresponde aos dendritos no neurônio biológico, um centro onde o processamento é realizado, que corresponde ao núcleo celular, e uma saída de dados que faz o papel das terminações sinápticas.

Figura 41 – Representação de um neurônio artificial.



Fonte: Elaborado pelo autor

Como é possível observar na Figura 41, cada entrada do neurônio artificial possui um peso sináptico. Os valores que chegam no neurônio, que na figura são representados pelas componentes do vetor $[x_1, x_2, x_3, \ldots, x_m]$, são multiplicados pelos correspondentes pesos sinápticos, que são representados pelas componentes do vetor $[w_1, w_2, \ldots, w_m]$ e em seguida são somados dentro do núcleo. Por fim, o valor encontrado é diminuídos de θ, o limiar de ativação. Essa soma ponderada resulta em u como pode ser visto na equação:

$$u = \sum_{i=1}^{n} w_i x_i - \theta$$

Após ser calculado, aplica-se a u uma função de ativação g cujo principal objetivo é limitar a saída do neurônio para uma faixa de valores que melhor convier. Existem diversas funções de ativação, algumas delas serão apresentadas na próxima seção.

O valor resultante da função de ativação $y = g(u)$ é a saída do neurônio artificial. Esse valor pode então ser enviado para outros neurônios, no caso deste neurônio pertencer a uma camada intermediária, ou pode ser a saída final da rede, caso esse neurônio esteja na última camada da rede. Existem diversas formas de organizar a arquitetura das redes neurais. Algumas dessas formas serão mostradas a seguir.

Os valores do vetor de pesos sinápticos, assim como o valor de θ, são o que definem como será a saída do neurônio. São eles que serão ajustados durante a fase de treinamento para que o neurônio possa responder de acordo com o esperado. A forma como esses valores são ajustados variam de acordo com o tipo de rede neural, a seguir é explicado como a rede MLP realiza o seu ajuste.

Funções de Ativação

As funções de ativação têm a finalidade de limitar a saída os neurônios para um intervalo onde é mais confortável de se trabalhar na rede neural.

Algumas redes utilizam funções parcialmente diferenciáveis, como é o caso da função degrau (Equação C.1) na qual a saída é 1 se a entrada for não-negativa, ou 0 se a entrada for negativa.

$$g(u) = \begin{cases} 1 & \text{se } u \geq 0 \\ 0 & \text{se } u < 0 \end{cases} . \tag{C.1}$$

Há também a função degrau bipolar (Equação C.2) na qual a saída é 1 se a entrada for positiva, 0 se entrada for 0 e -1 se a entrada for negativa.

$$g(u) = \begin{cases} 1 & \text{se } u > 0 \\ 0 & \text{se } u = 0 \\ -1 & \text{se } u < 0 \end{cases} . \tag{C.2}$$

Porém, dependendo da regra de aprendizagem, funções precisam ser totalmente diferenciáveis, como é o caso da função logística (Equação C.3) na qual a saída tende para 1 ($g(u) \to 1$) à medida que a entrada cresce ($u \to \infty$) e para 0 ($g(u) \to 0$) à medida que a entrada diminui ($u \to -\infty$), sendo que β indica a suavização dessa função. Assim, é possível alcançar um efeito similar a da função degrau em uma regra que exija uma função de ativação completamente diferenciável.

$$g(u) = \frac{1}{1+e^{-\beta u}} . \tag{C.3}$$

Há também a função tangente hiperbólica (Equação C.4) na qual a saída tende para 1 ($g(u) \to 1$) a medida que a entrada cresce ($u \to \infty$), e para -1 ($g(u) \to -1$) a medida que a entrada diminui ($u \to -\infty$), sendo que β também indica a suavização dessa função. Novamente, alcançando um efeito similar a da função degrau bipolar em uma regra que exija uma função de ativação completamente diferenciável.

$$g(u) = \frac{1-e^{-\beta u}}{1+e^{-\beta u}} . \tag{C.4}$$

Outro exemplo de função de ativação é função gaussiana (Equação C.5) em que $g(u)$ diminui a medida que a entrada se afasta do centro c ($g(u) \to 0$, quando $\|u-c\| \to \infty$). O parâmetro σ define o raio da gaussiana.

$$g(u) = \exp\left(-\frac{(u-c)^2}{2\sigma^2}\right) . \tag{C.5}$$

Também é possível utilizar a função linear (Equação C.6) na qual a saída é a própria entrada.

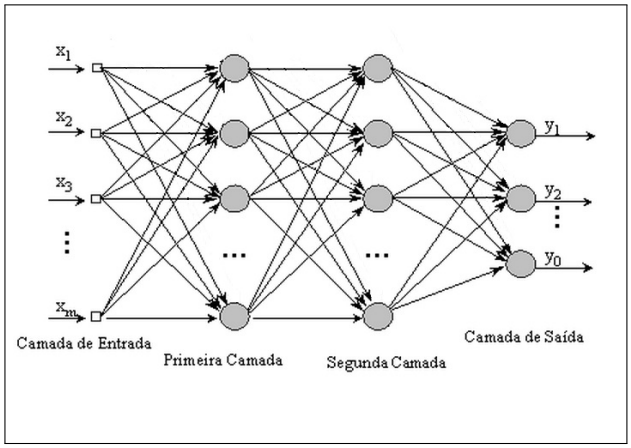
$$g(u) = u. \tag{C.6}$$

Apesar das funções parcialmente diferenciáveis parecerem mais simples e portanto mais fáceis de se trabalhar, as funções totalmente diferenciáveis são necessárias, pois algumas técnicas de aprendizagem exigem encontrar a derivada dessa função.

Arquitetura *feedforward* de camadas múltiplas

As redes que utilizam arquiteturas em múltiplas camadas são semelhantes à rede observada na Figura 42. Nesse caso, é possível observar que a saída de um neurônio de uma camada intermediária será a entrada de um neurônio da camada seguinte. Nesse tipo de rede, os dados seguem apenas em uma única direção, da camada de entrada para a camada de saída, sem retorno.

Figura 42 – Representação de uma rede de múltiplas camadas contendo os neurônios e as ligações sinápticas entre os mesmos.



Fonte: Elaborado pelo autor

A presença de várias camadas na rede permite que, à medida em que os dados avançam para uma nova camada, eles passam a ser representados em um novo espaço, o que permite a distinção entre padrões cuja separação não é trivial.

Aprendizagem *Backpropagation*

O algoritmo *Backpropagation* tem esse nome porque ele realiza o ajuste dos pesos propagando o erro da última camada em direção a primeira de acordo com a regra mostrada ne equação:

$$W_{ji}(t+1) = W_{ji}(t) + \eta.\delta_j.Y_i. \tag{C.7}$$

Sendo W_{ji} o peso da ligação entre o j-ésimo neurônio da última camada e o i-ésimo neurônio da penúltima camada, o ajuste desse peso é realizado de acordo com a equação (C.7), sendo que $W_{ji}(t+1)$ é o novo peso, $W_{ji}(t)$ é o peso antes do ajuste, η é a taxa de aprendizagem definida previamente, Y_i é a saída da função de ativação do i-ésimo neurônio da penúltima camada e δ_j é definido de acordo com a Equação C.8, onde d_j é a saída desejada do j-ésimo neurônio da última camada, Y_j é a saída obtida deste neurônio, g' é a derivada da função de ativação que a rede utiliza e I_j é a saída do neurônio antes de ser aplicada a função de ativação.

$$\delta_j = (d_j - Y_j).g'(I_j). \tag{C.8}$$

Para o j-ésimo neurônio da camada escondida, o novo peso também é calculado segundo a equação C.7. Porém, neste caso o δ_j é calculado de acordo com a equação C.9, onde n é o número de neurônios da camada seguinte, δ_k é o delta calculado pelo k-ésimo neurônio da camada seguinte, W_{kj} é o peso da ligação entre o j-ésimo neurônio da camada atual com o k-ésimo neurônio da camada seguinte, e $g'(I_j)$ continua sendo a saída do neurônio que passa pela derivada da função de ativação em vez da própria função de ativação.

$$\delta_j = (\sum_{k=1}^{n} \delta_k.W_{kj}).g'(I_j). \tag{C.9}$$

Através dessas equações de ajuste de peso, é possível perceber algumas características da rede MLP. Como é possível observar na equação C.8, o algoritmo *Backpropagation* utiliza a saída esperada de um determinado padrão para realizar o ajuste dos pesos, portanto a rede MLP é considerada uma rede com treinamento supervisionado.

Outro ponto a se observar é o fato do algoritmo *Backpropagation* utilizar a derivada da função de ativação durante o ajuste dos pesos, isso serve para que o algoritmo saiba em que direção o peso deve ir com o objetivo de minimizar o erro. Por esse motivo a rede MLP precisa utilizar uma função de ativação que seja totalmente diferenciável.

APÊNDICE D – Modelando o Alinhamento de Dois Exemplos de Forma do PDM

A partir de um conjunto X de k exemplos (x_1, x_2, ..., x_k) de uma determinada forma é possível construir um modelo de distribuição de pontos (PDM - *Point Distribution Model*) que descreve os principais modos de variação dessa forma ao longo de X.

Um dos passos da construção do PDM é e alinhar esses exemplos dois-a-dois. Sendo cada exemplo um vetor de tamanho $2 \times n$ que contém as coordenadas vertical e horizontal de cada um dos n pontos de marcação que descrevem forma que se deseja modelar. É possível aplicar uma função de transformação T que modifique a rotação, escala e translação de um determinado exemplo mantendo a sua forma inalterada. Essa função M pode ser descrita pela equação:

$$M(s,\theta,t)[x_i] = M(s,\theta,t)\begin{bmatrix} x_{i1} \\ y_{i1} \\ \vdots \\ x_{in} \\ y_{in} \end{bmatrix} = \begin{pmatrix} ((s\cos(\theta))x_{i1} - (s\sin(\theta))y_{i1}) - t_x \\ ((s\sin(\theta))x_{i1} + (s\cos(\theta))y_{i1}) - t_y \\ \vdots \\ ((s\cos(\theta))x_{in} - (s\sin(\theta))y_{in}) - t_x \\ ((s\sin(\theta))x_{in} + (s\cos(\theta))y_{in}) - t_y \end{pmatrix}, \tag{D.1}$$

onde s é o fator de escala, θ é a rotação e $t = (t_x, t_y)^T$ é a translação a ser aplicada nos eixos vertical e horizontal.

O procedimento de alinhar um determinado exemplo de forma x_i de acordo com os pontos de marcação de outro exemplo x_j, corresponde a encontrar os valores de s, θ e t que minimizem a função objetivo:

$$E = (x_j - M(s,\theta,t)[x_i])^T W (x_j - M(s,\theta,t)[x_i]), \tag{D.2}$$

sendo que W é uma matriz diagonal de pesos que pode dar mais ou menos importância para cada ponto da forma.

Tomando $a_x = s * cos(\theta)$ e $a_y = s * sin(\theta)$, podemos encontrar quatro equações lineares a partir da diferenciação em relação às variáveis a_x, a_y, t_x, t_y. Essas equações podem ser arranjadas da seguinte maneira:

$$\begin{pmatrix} X_i & -Y_i & W & 0 \\ Y_i & X_i & 0 & W \\ Z & 0 & X_i & Y_i \\ 0 & Z & -Y_i & X_i \end{pmatrix} \begin{pmatrix} a_x \\ a_y \\ t_x \\ t_y \end{pmatrix} = \begin{pmatrix} X_j \\ Y_j \\ C_1 \\ C_2 \end{pmatrix}, \tag{D.3}$$

onde temos que:

$$X_u = \sum_{k=0}^{n-1} w_k x_{uk}, \tag{D.4}$$

$$Y_u = \sum_{k=0}^{n-1} w_k y_{uk}, \tag{D.5}$$

$$Z = \sum_{k=0}^{n=1} w_k (x_{ik}^2 + y_{ik}^2), \tag{D.6}$$

$$W = \sum_{k=0}^{n=1} w_k, \tag{D.7}$$

$$C_1 = \sum_{k=0}^{n=1} w_k (x_{jk} x_{ik} + y_{jk} y_{ik}), \tag{D.8}$$

$$C_2 = \sum_{k=0}^{n=1} w_k (y_{jk} x_{ik} - x_{jk} y_{ik}). \tag{D.9}$$

Assim, podemos encontrar a_x, a_y, t_x e t_y simplesmente resolvendo o sistema.

APÊNDICE E – Análise de Componentes Principais

Segundo (JOLLIFFE, 2002) a Análise de Componentes Principais (PCA - Principal Component Analysis) consiste da redução da dimensionalidade de um conjunto de dados que contém um grande número de variáveis inter-relacionadas, mantendo o máximo possível a variação presente nos dados.

Dessa forma, uma das principais utilidades do PCA é permitir a visualização, em um espaço reduzido, de um conjunto de dados que possui um número elevado de variáveis. Isso porque quando o número de variáveis é muito grande, não é uma tarefa simples, nem muito útil, visualizar o conjunto original. Através dessa técnica é possível visualizar esse conjunto mais facilmente em um espaço mais simples de se observar, mantendo a maior parte da sua variabilidade.

O PCA realiza a redução de dimensionalidade em um conjunto de dados buscando pelas suas componentes principais. O primeiro passo é encontrar uma transformação linear $\alpha_1^T x$ que possui variância máxima quando aplicada a um vetor x, contendo p variáveis aleatórias.

$$\alpha_1^T x = \alpha_{11} x_1 + \alpha_{12} x_1 + ... + \alpha_{1p} x_p = \sum_{j=1}^{p} \alpha_{1j} x_j. \tag{E.1}$$

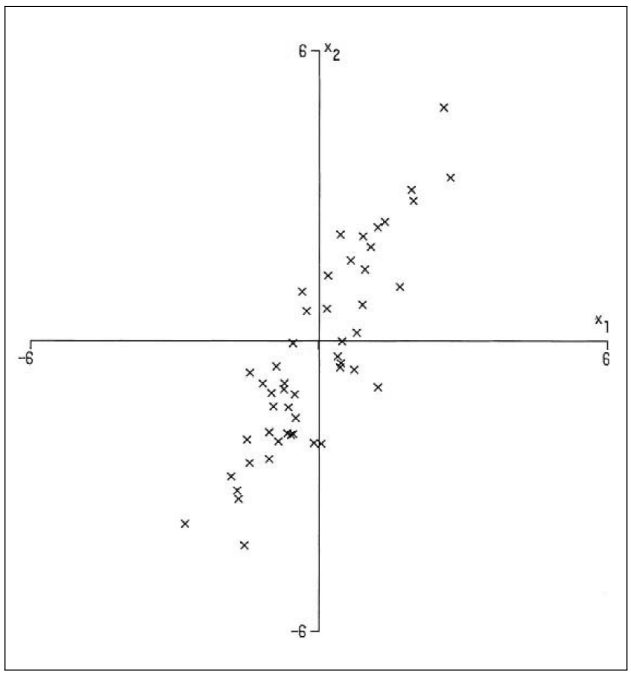
Na equação E.1 temos que $\alpha_1^T x$ é uma combinação linear ponderada das variáveis aleatórias de x, além disso α_1 deve ser escolhido de tal forma que a variância dessa transformação resultante seja igual a variação máxima de x.

Em seguida, inicia-se a busca por uma outra função $\alpha_2^T x$ que seja completamente não correlacionada com $\alpha_1^T x$ e que, quando aplicada aos elementos do vetor x, gere uma nova variável aleatória que seja completamente não correlacionada com $\alpha_1^T x$ e que apresente a segunda maior variância possível, ficando atrás apenas da primeira transformação encontrada.

O processo se repete para $\alpha_3^T x$, $\alpha_4^T x$,..., $\alpha_k^T x$, sendo que a i-ésima variável derivada $\alpha_i^T x$ é a i-ésima componente principal. Até no máximo p componentes principais podem ser encontradas, mas o esperado é que a maior parte da variância das variáveis presentes em x possa ser representada por um total de m componentes principais, sendo que $m \ll p$.

Na Figura 43 podemos observar um conjunto de dados com $p = 2$ contendo 50 exemplos. É visível como as duas variáveis x_1 e x_2 são altamente correlacionadas. Esses dados podem ser transformados para um espaço contendo as suas componentes principais z_1 e z_2. Essa transformação é desenhada na Figura 44. É visível que os exemplos do conjunto de dados

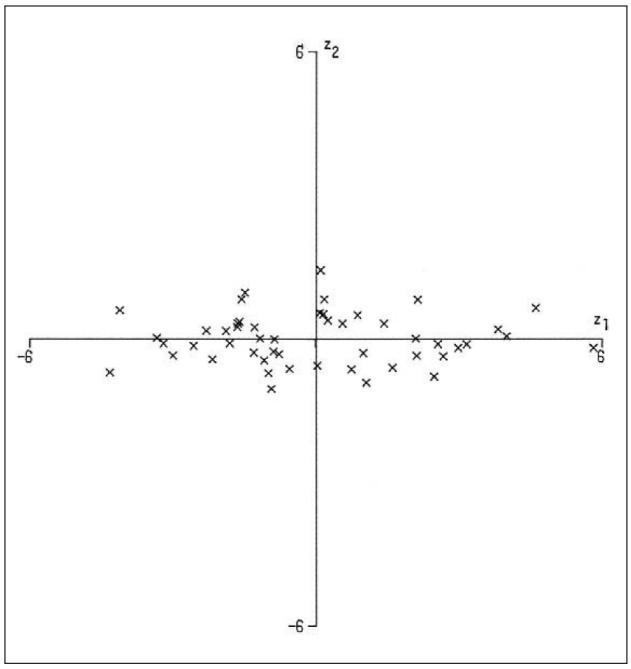
Figura 43 – Conjunto de dados contendo duas variáveis: x_1 e x_2.



Fonte: Figura retirada de (JOLLIFFE, 2002)

original apresentam maior variância na variável x_2 do que na variável x_1. Porém, observando o espaço das componentes principais (z_1 e z_2), podemos ver que z_1 apresenta uma variância maior que qualquer uma das variáveis originais, uma vez que essa direção é a que possui a maior variância de todo o conjunto.

Figura 44 – Conjunto de dados representado no espaço de componentes principais.



Fonte: Figura retirada de (JOLLIFFE, 2002)

Encontrando as Componentes Principais

Para encontrar as componentes principais de um vetor x contendo p variáveis aleatórias, é necessário conhecer primeiro a matriz de covariância dessas variáveis.

Uma matriz de covariância Σ é tal que seu (i, j)-ésimo elemento informa a covariância entre as variáveis i e j do vetor x. Quando $i = j$, temos que este elemento mostra a variância da i-ésima variável.

A partir dessa matriz de covariância, a i-ésima componente principal $z_i = \alpha_i^T x$ corresponde ao autovetor α_i de Σ associado ao seu i-ésimo maior autovalor λ_i. Além disso se α_i for escolhido de maneira que tenha como norma a unidade, então a variância de z_i é $var(z_i) = \lambda_i$.

Para encontrar as componentes principais de um conjunto de dados, devemos primeiro considerar a busca pela primeira componente $\alpha_1^T x$. Dessa forma devemos encontrar um vetor α_1 que maximize a sua variância (E.2), sujeito a restrição (E.3).

$$\alpha_1^T \Sigma \alpha_1. \tag{E.2}$$

$$\alpha_1^T \alpha_1 = 1. \tag{E.3}$$

Uma abordagem para a resolução desse problema de otimização é a utilização da técnica dos multiplicadores de Lagrange. Dessa forma o problema passa a ser o de maximizar:

$$\alpha_1^T \Sigma \alpha_1 - \lambda(\alpha_1^T \alpha_1 - 1),$$

onde λ é um multiplicador da Lagrange.

Assim, aplicando a diferenciação em relação a α_1, obtemos a equação:

$$\Sigma \alpha_1 - \lambda \alpha_1 = 0. \tag{E.4}$$

Isolando α_1 de E.4, obtemos a equação:

$$(\Sigma - \lambda I_p)\alpha_1 = 0,$$

onde I_p é a matriz identidade de tamanho $(p \times p)$. Além disso, λ é um autovalor de Σ e α_1 é o seu autovetor correspondente.

A decisão de qual dos p autovetores fornece a $\alpha_1^T x$ a maior variância possível leva em consideração que λ_1 deve ser o maior possível, levando-se em consideração que:

$$\alpha_1^T \Sigma \alpha_1 = \alpha_1^T \lambda_1 \alpha_1 = \lambda_1 \alpha_1^T \alpha_1 = \lambda_1.$$

Assim, α_1 é o autovetor correspondente ao maior autovalor de Σ e a $var(\alpha_1^T x) = \alpha_1^T \Sigma \alpha_1 = \lambda_1$ é esse maior autovalor.

Em geral, a i-ésima componente principal de x é $\alpha_i^T x$ e $var(\alpha_i^T x) = \lambda_i$ onde λ_i é o i-ésimo maior autovalor de Σ e α_i é o seu autovetor correspondente.

A segunda componente principal, $\alpha_2^T x$ maximiza:

$$\alpha_2^T \Sigma \alpha_2, \tag{E.5}$$

sujeito a ser completamente não correlacionado com $\alpha_1^T x$:

$$cov(\alpha_1^T x, \alpha_2^T x) = 0. \tag{E.6}$$

Sendo que $cov(x, y)$ denota a covariância entre as variáveis, a restrição E.6 pode ser reescrita como exposto na em:

$$cov(\alpha_1^T x, \alpha_2^T x) = \alpha_1^T \Sigma \alpha_2 = \alpha_2^T \Sigma \alpha_1 = \alpha_2^T \lambda_1 \alpha_1 = \lambda_1 \alpha_2^T \alpha_1 = \lambda_1 \alpha_1^T \alpha_2.$$

Assim, qualquer das Equações E.7, E.8, E.9 ou E.10 pode ser usada para especificar a não existência de correlação entre $\alpha_1^T x$ e $\alpha_2^T x$.

$$\alpha_1^T \Sigma \alpha_2 = 0. \tag{E.7}$$

$$\alpha_2^T \Sigma \alpha_1 = 0. \tag{E.8}$$

$$\alpha_2^T \alpha_1 = 0. \tag{E.9}$$

$$\alpha_2^T \alpha_1 = 0. \tag{E.10}$$

Dessa forma, para encontrar o segundo maior autovalor e seu autovetor correspondente, usando formalismo de Lagrange, devemos maximizar a expressão:

$$\alpha_2^T \Sigma \alpha_2 - \lambda_2(\alpha_2^T \alpha_2 - 1) - \phi \alpha_2^T \alpha_1,$$

onde λ_2 e ϕ são multiplicadores de Lagrange.

Diferenciando em relação a α_2, obtemos a equação:

$$\Sigma\alpha_2 - \lambda_2\alpha_2 - \phi\alpha_1 = 0.$$

Multiplicando na esquerda por α_1^T, resulta na equação:

$$\alpha_1^T\Sigma\alpha_2 - \lambda_2\alpha_1^T\alpha_2 - \phi\alpha_1^T\alpha_1 = 0.$$

Como os dois primeiros termos são zero e $\alpha_1^T\alpha_1 = 1$, podemos concluir que $\phi = 0$. Dessa forma, $\Sigma\alpha_2 - \lambda_2\alpha_2 = 0$ o que equivale a $(\Sigma\lambda_2 I_p)\alpha_2 = 0$, assim, λ_2 é novamente um autovalor de Σ sendo que α_2 é o seu autovetor correspondente.

Novamente, $\lambda_2 = \alpha_2^T\Sigma\alpha_2$, logo λ_2 deve ser o maior possível. Assumindo que Σ não apresenta autovalores repetidos λ_2 não pode ser igual a λ_1. Se fosse possível, significaria que $\alpha_2 = \alpha_1$. Isso viola a restrição $\alpha_1^T\alpha_2 = 0$. Sendo λ_2 o segundo maior autovalor, α_2 é o seu autovetor correspondente.

Assim como foi feito para o primeiro e segundo autovetores, pode ser mostrado que os coeficientes α_3, α_4, ..., α_p são os autovetores de Σ correspondentes aos autovalores λ_3, λ_4,...,λ_p.

Implicações estatísticas

Algumas implicações estatísticas podem ser extraídas a partir da Analise de Componentes Principais. Para isso, considere o vetor z, onde seu k-ésimo, z_k, elemento corresponde à k-ésima componente principal.

Esse vetor z pode ser escrito como mostrado na equação:

$$z = A^T x,$$

onde A é a matriz ortogonal em que a sua k-ésima coluna (a_k) é o autovetor correspondente ao k-ésimo autovalor de Σ.

Assim as componentes principais podem ser escritos como uma transformação ortogonal linear de x.

Além disso, também temos que é verdade a equação:

$$\Sigma A = A\Lambda, \tag{E.11}$$

em que Λ é a matriz diagonal em que o k-ésimo elemento é λ_k, o k-ésimo autovalor de Σ.

Também podemos expressar E.11 como mostrado nas Equações E.12 e E.13.

$$A^T \Sigma A = \Lambda. \tag{E.12}$$

$$\Sigma = A\Lambda A^T. \tag{E.13}$$

Essa transformação linear ortogonal de x possui algumas propriedade interessantes que serão discutidas as seguir.

Para qualquer inteiro q, $1 \leq q \leq p$ considere a transformação linear mostrada na equação:

$$y = B^T x, \tag{E.14}$$

onde y é um vetor de q elementos e B^T é uma matriz $(q \times p)$.

Sendo $\Sigma_y = B^T \Sigma B$ a matriz de covariância para y, temos que o traço de Σ_y $(tr(\Sigma_y))$ é maximizado quando $B = A_q$. Essa propriedade implica que as primeiras colunas de A contém o subconjunto de colunas que permitem a construção de uma matriz de transformação B que maximiza a variância dos elementos de y.

Além disso temos que $tr(\Sigma_y)$ é minimizado tomando $B = A_q$, sendo que A_q consistem das últimas colunas de A. Nesse caso, o que se percebe é que a variância dos elementos de y vão sendo minimizadas a medida a matriz de transformação B é constituída das últimas colunas de A.

Sendo $det(\Sigma_y)$ o determinante da matriz de covariância y, temos que $det(\Sigma_y)$ é maximizado quando $B = A_q$. Essa propriedade implica que a variância generalizada, que pode ser medida a partir do determinante da matriz de covariância, é maior quando a matriz de transformação é formada a partir da primeiras colunas de A.

Também podemos definir a matriz Σ como mostrado na equação:

$$\Sigma = \lambda_1 \alpha_1 \alpha_1^T + \lambda_2 \alpha_2 \alpha_2^T + ... + \lambda_p \alpha_p \alpha_p^T. \tag{E.15}$$

A Equação E.15 pode ser obtida a partir da Equação E.13. Por ela podemos perceber que a matriz de covariância Σ pode ser obtida a partir de contribuições $\lambda_k \alpha_k \alpha_k^T$ de cada componente principal. Além disso, a contribuição de cada componente principal para a construção da matriz de covariância vai diminuindo uma vez que o valor de λ_k diminui ao passo que k cresce.

Supondo que se deseja prever cada variável aleatória, x_j de x através de uma função de y, onde $y = B^T x$. Se considerarmos σ^2 como sendo a variância residual da previsão de x_i a partir de y, então o valor de $\sum_{j=1}^{p}$ é minimizado se $B = A_q$. Isso implica que em um preditor de x em um espaço de q dimensões, o subespaço ótimo que pode ser utilizado consiste dos q primeiros componentes principais.

Essas propriedades que podem ser observadas durante a transformação de elementos do espaço de x para o espaço de y, nos mostram que as primeiras componentes da matriz de transformação A são as mais importantes e que as demais componentes vão perdendo importância a medida que a variância das mesmas diminui.

Isso permite a transformação dos dados para um espaço de dimensão menor, selecionando por ordem de importância as componentes que farão a transformação, de maneira não haja perda significativa entre os dados. Um vez nesse espaço de dimensão inferior, eles pode ser manipulados e quando necessário é possível aplicar a transformação inversa que retorna os dados para o seu espaço original sem que haja uma perda significativa.

Componentes Principais de um conjunto de dados discreto

Nem sempre o valor da matriz de covariância Σ de uma população x é conhecida, nesse caso é comum estimar essa matriz a partir de um conjunto de observações dessa população.

Assim, a partir de n observações independentes de x (x_1, x_2,...,x_n), devemos buscar um vetor a_1 que maximiza a variância das amostras que pode ser calculada como mostrado na equação:

$$\frac{1}{n-1} \sum_{i=1}^{n} (z_{i1} - \bar{z}_1)^2, \tag{E.16}$$

sendo que $z_{i1} = a_1^T x_i$ e $\bar{z}_1$ é o valor médio dessas amostras transformadas e tal procura deve ser sujeita a restrição $a_1^T a_1 = 1$.

De forma semelhante ao que foi exposto anteriormente, podemos procurar um vetor a_2 que maximiza a variância de z_{i2} estando sujeito a $a_2^T a_2 = 1$ e também de maneira que z_{i2} seja completamente não correlacionado com z_{i1}. Se continuarmos procurando por todas as melhores

transformações possíveis para as amostras que maximizem a variância e se mantenham não correlacionadas entre si, podemos encontrar todas as componente principais dessas amostras. Tal que $a_k^T x$ é definida com sendo a k-ésima componente principal das amostras e z_{i1} nos informa o valor da i-ésima amostra em relação a essa k-ésima componente principal.

De forma semelhante ao que foi exposto anteriormente, podemos encontrar que a variância da k-ésima componente principal das amostras é l_k, que corresponde ao k-ésimo autovalor de maior valor absoluto da matriz de covariância S calculada a partir das amostras. Além disso, a_k são os seus autovalores correspondentes.

Se considerarmos a matriz $n \times p$ X cujo (i,k)-ésimo elemento é igual ao k-ésimo elemento (x_{ik}) de x_i e a matriz também $n \times p$ Z cujo (i,k)-ésimo elemento é corresponde a z_{ik}, temos que essas matrizes são relacionadas como mostrado na equação:

$$Z = XA, \tag{E.17}$$

onde A é uma matriz $p \times p$ ortogonal em que a h-ésima coluna é a_k.

Para calcular o (j,k)-ésimo elemento da matriz de covariância S, podemos usar a equação:

$$s_{jk} = \frac{1}{n-1} \sum_{i=1}^{n} (x_{ij} - \overline{x}_j)(x_{ik} - \overline{x}_k),$$

sendo que $\overline{x}$ é a média, como pode ser visto na equação:

$$\overline{x}_j = \frac{1}{n} \sum_{i=1}^{n} x_{ij}.$$

Se considerarmos uma matriz $n \times p$ $\mathbf{X}$ em que o seu (i,j)-ésimo elemento é $(x_{ij} - \overline{x}_j)$, portanto as amostras contidas em $\mathbf{X}$ são iguais as amostras contidas em X subtraídas da média , podemos calcular S através da equação:

$$S = \frac{1}{n-1} \mathbf{X}^T \mathbf{X}. \tag{E.18}$$

Usando $\mathbf{X}$ no lugar de X na Equação E.17, temos a equação:

$$\mathbf{Z} = \mathbf{XA}.$$

Nesse caso os valores das amostras nas componentes principais em $\mathbf{Z}$ terão as mesmas variâncias e covariâncias que em Z, mudando apenas a sua média para zero.

Também pode ser provado que os autovetores de $\frac{1}{n-1}\mathbf{X}^T\mathbf{X}$ e $\mathbf{X}^T\mathbf{X}$ são exatamente os mesmos. Além disso, os autovalores de $\frac{1}{n-1}\mathbf{X}^T\mathbf{X}$ são iguais aos autovalores de $\mathbf{X}^T\mathbf{X}$ divididos

por $(n-1)$. Por esse motivo é muito comum calcular diretamente os autovalores de $\mathbf{X}^T\mathbf{X}$ em vez dos autovalores de S.

As propriedade estatísticas descritas na seção anterior ainda se mantêm para o caso em que as componentes são extraídas a partir de amostras feitas as devidas adaptações.

Decomposição de Valores Singulares

A Decomposição de Valores Singulares (SVD - Singular Value Decomposition) é relevante para a aplicação da Analise de Componentes Principais de diversas maneiras.

Dada uma matriz X de dimensões $n \times p$, na qual podemos encontrar n observações de p atributos cada, X pode ser escrito de acordo com a equação:

$$X = ULA^T,$$

onde U é uma matriz $n \times r$ tal que $U^TU = I_r$, A é uma matriz $p \times r$ tal que $A^TA = I_r$, L é uma matriz diagonal $r \times r$ e r é o posto de X.

Sendo a_k a k-ésima coluna de A, pode ser provado (JOLLIFFE, 2002), que a_k é o autovetor correspondente ao k-ésimo autovalor de X^TX.

Também é verdade que u_k, a k-ésima coluna de U, é o autovetor correspondente ao k-ésimo autovalor de XX^T, o que pode ser útil quando na matriz X as amostras estão localizadas nas colunas e o atributos na linhas. Dessa forma, matematicamente, não importa como os dados estejam organizados, desde que se entenda as consequências dessa organização.

Assim, o cálculo da decomposição SVD provê um método computacionalmente eficiente para encontrar as componentes principais de um conjunto de dados. Além disso, os elementos da diagonal principal da matriz L correspondem à raiz quadrada dos autovalores diferentes de zero de X^TX, que são os mesmos autovalores diferentes de zero de XX^T.

Outro fato importante é que, os elementos de X pode ser escritos cada um na forma mostrada na equação:

$$x_{i,j} = \sum_{k=1}^{r} u_{ik} l_k^{1/2} a_{jk},$$

onde u_{ik} e a_{jk} são respectivamente os (i,k)-ésimos e (j,k)-ésimos elementos de U e A, $l_k^{1/2}$ é o k-ésimo elemento da diagonal de L.

É possível obter uma aproximação de $x_{i,j}$ através das m primeiras componentes

principais como exposto na equação:

$$\tilde{x}_{i,j} = \sum_{k=1}^{m} u_{ik} l_k^{1/2} a_{jk},$$

em que $\tilde{x}_{i,j}$ é uma aproximação de $x_{i,j}$ que tende a ficar mais confiável a medida que m se aproxima de r.

Isso é possível porque as primeiras colunas de A e U possuem a maior parte da capacidade de representar X, uma vez que essas possuem as componentes principais em ordem de importância.

APÊNDICE F – Análise de Sensibilidade

Em (FAWCETT, 2006) é dito que um gráfico de Característica de Operação do Receptor (ROC - Receiver Operating Characteristic) é uma técnica que permite a visualização, organização e seleção de classificadores com base em suas performances.

Para a produção de um gráfico ROC é necessário obter algumas outras medidas que devem ser colhidas de um classificador além de sua taxa de acerto. Essas métricas podem ser obtidas através da análise de uma matriz de confusão produzida durante os testes.

Tabela 10 – Matriz de confusão simples.

		Condição ("padrão ouro")	
		Positivo	Negativo
Resultado do Teste	Positivo	**Verdadeiro Positivo**	**Falso Positivo**
	Negative	**Falso Negativo**	**Verdadeiro Negativo**

Fonte: Adaptado de (FAWCETT, 2006)

A Tabela 10 mostra o conteúdo presente em uma matriz de confusão para um problema de discriminação entre duas classes, positiva e negativa. Nesse caso, a matriz possui duas colunas e duas linhas, sendo que as colunas separam os exemplos do conjunto de testes de acordo com a sua verdadeira natureza, enquanto que as linhas separam os exemplos testados de acordo com a classe atribuída pelo classificador.

Em analogia ao problema da detecção do capacete, podemos considerar que os exemplos positivos são aqueles em que o condutor está utilizando o capacete enquanto que os casos negativos são aqueles em que o condutor não utiliza capacete.

Dessa forma cada exemplo testado pode ser inserido em um de 4 possíveis conjuntos. Ele pode ser considerado verdadeiro positivo, quando o classificador determinar que o motociclista utiliza capacete e isso for a sua verdadeira natureza; verdadeiro negativo, quando o classificador determinar que o motociclista não utiliza capacete e isso for verdade; falso positivo, quando o classificador determina que o motociclista utiliza capacete e porém ele não utiliza; e falso negativo, quando o classificador determina que o motociclista não utiliza capacete porém isso é falso.

A partir da quantidade de exemplos presentes em cada um desses conjuntos, é possível determinar várias métricas que são utilizadas para realizar a análise de sensibilidade do classificador.

Tabela 11 – Métricas calculadas a partir da matriz de confusão.

	Condição ("padrão ouro")			
Resultado do Teste	**Verdadeiro Positivo** (VP)	**Falso Positivo** (FP)	Valor Previsto Positivo (precision) = $\frac{VP}{VP+FP}$	Taxa de Descoberta Falsa = $\frac{FP}{VP+FP}$
	Falso Negativo (FN)	**Verdadeiro Negativo** (VN)	Taxa de Omissão Falsa = $\frac{FN}{FP+VN}$	Valor Previsto Negativo = $\frac{VN}{FN+VN}$
	Taxa de Verdadeiro Positivo (recall) = $\frac{VP}{VP+FN}$	Taxa de Falso Positivo (fall-out) = $\frac{FP}{FP+VN}$	F-measure = $2 \times \frac{precision.recall}{precision+recall}$	
	Taxa de Falso Negativo (miss-rate) = $\frac{FN}{VP+FN}$	Taxa de Verdadeiro Negativo (specificity) = $\frac{VN}{FP+VN}$	Acurácia (ACC) = $\frac{VP+VN}{VP+FP+FN+VN}$	

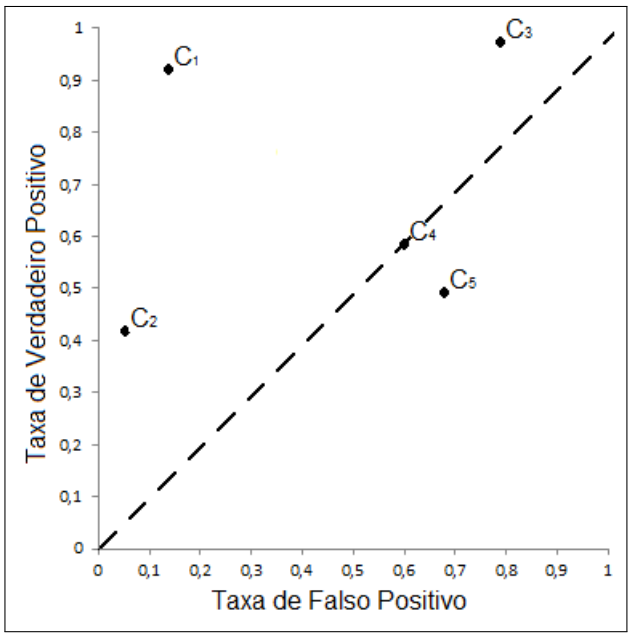
Fonte: Adaptado de (FAWCETT, 2006)

Na Tabela 11 podemos ver que além da acurácia, essas métricas incluem: taxa de verdadeiros positivos, também chamada de *recall*; taxa de falso negativo; taxa de falsos positivos; taxa de verdadeiro negativo, também chamado de especificidade; valor previsto positivo; valor previsto negativo; taxa de omissão falsa; taxa de descoberta falsa.

Dentre essas métricas podemos utilizar a taxa de verdadeiros positivos e a taxa de falso negativo para inserir esse classificador em um gráfico ROC. Na Figura 45 podemos observar o espaço descrito por um gráfico ROC. Neste espaço, os classificadores que ficam mais próximos do ponto $(0,1)$, como é o caso do classificador C_1, são considerados os melhores classificadores, pois sua taxa de taxa de verdadeiros positivos e sua taxa de falso negativo são mais próximas do ideal. Quando um classificador se encontra mais próximo do ponto $(0,0)$, como o classificador C_2, podemos concluir que o mesmo possui uma tendência para realizar classificações negativas, uma vez que tanto sua Taxa de Falso Negativo, quanto sua Taxa de Verdadeiro Negativo são altas. Nos casos em que o classificador se encontra mais próximo do ponto $(1,1)$, como o classificador C_3, podemos concluir que o mesmo possui uma tendencia para realizar classificações Positivas, uma vez que tanto sua Taxa de Verdadeiro Positivo, quanto sua Taxa de Falso Positivo são altas.

Classificadores localizados na linha tracejada, como é o caso de C_4, utilizam uma

Figura 45 – Representação do espaço ROC.



Fonte: Elaborado pelo autor

estratégia aleatória para realizar a sua decisão. No caso de um classificador se localizar abaixo da linha tracejada, como C_5, sua decisão é pior que aleatória. Nesse caso, sua saída pode ser invertida, transformando-o em um bom classificador. Por esse motivo não se costumam encontrar classificadores abaixo da reta $x = y$.

Também é comum a construção de um gráfico ROC a partir de um único classificador. Nessa caso é necessário realizar sucessivos testes alterando certos parâmetros desse classificador, normalmente o seu limiar de classificação, a fim de explorar a performance desse classificador quando o mesmo é enviesado para uma das classes.