3
Técnica Baseada em Treinamento e Cascata de Classificadores

3.1.
Introdução

A utilização de algoritmos para a extração de características de objetos e a geração de classificadores em cascata capazes de reconhecer padrões de objetos em uma imagem tem demonstrado resultados robustos, eficientes e extremamente rápidos para a detecção de objetos em tempo real.

Entre os trabalhos que abordam o problema de detecção de objetos em tempo real, destaca-se a técnica proposta inicialmente por Viola e Jones (2001) por apresentar uma solução genérica quanto à detecção de qualquer tipo de objeto, além de possuir uma taxa muito pequena de ocorrência de falsos positivos.

Neste capítulo apresenta-se detalhadamente a teoria por trás da definição de como são formadas as características de um objeto, o algoritmo de aprendizado utilizado para a seleção de suas principais características, a formação da cascata de classificadores e como são realizadas as comparações para descartar ou não regiões na imagem durante a busca pelo objeto de interesse.

3.2.
Formato das Características de um Objeto

A rotina para detecção de objetos é baseada nas principais características (feições) de um objeto previamente extraídas a partir de um algoritmo de aprendizado de máquina. Tais características são responsáveis por diferenciar objetos uns dos outros, pois cada conjunto de características encontradas em um dado objeto é altamente distintivo em relação aos conjuntos de características encontradas também em outros objetos.

A maior motivação para o uso de características de um objeto ao invés do uso de seus pixels, é que a velocidade da análise de uma imagem baseada no conjunto de suas principais características é muito maior do que a análise

baseada sobre seus pixels, devido ao fato do número de características ser substancialmente inferior em relação ao número de pixels de um objeto.

Sabendo-se isto, Viola e Jones (2001) propõem a utilização de três **formatos** de características: 1 – Característica de “dois retângulos”, onde o seu valor é a diferença entre a soma dos pixels dentro de duas regiões retangulares adjacentes de mesmo tamanho; 2 - Característica de “três retângulos”, cujo valor é a soma em um retângulo central menos a soma de dois retângulos externos; 3 - Por fim, as características de “quatro retângulos” que possuem seu valor calculado através da diferença entre pares diagonais de retângulos. Tais formatos de características possíveis são exemplificados na figura 12. Os tipos de características representados nas figuras 12-A e 12-B são do tipo de “dois retângulos”, só que a primeira encontra-se no sentido vertical e a segunda no sentido horizontal. As figuras 12-C e 12-D ilustram as características “três retângulos” e “quatro retângulos” respectivamente.

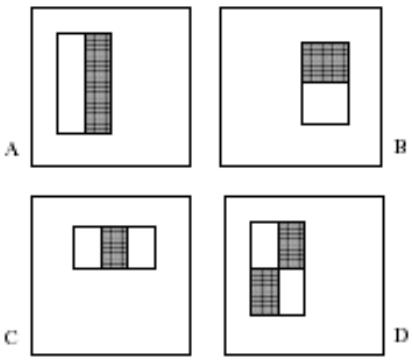

Figura 12. Formas possíveis de uma característica retangular (extraído de Viola e Jones (2001))

A motivação para a definição de tais características no formato retangular, também conhecidas como “características Haar”, é baseada no estudo realizado previamente por Papageorgiou e Poggio (2000), no qual é proposto o uso de Haar Wavelets (Mallat, 1989) para facilitar o treinamento de classificadores. Tal estudo demonstrou que os usos destas características retangulares são suficientes para a extração de informações relevantes das formas de um objeto.

Com o intuito de otimizar o cálculo de tais características retangulares, Viola e Jones (2001) propõem a utilização de uma representação intermediária da imagem chamada de ***Imagem Integral***. Nesta forma de representação, cada ponto *x,y* da imagem contém o somatório de pixels da origem da imagem até a

sua localização (Figura 13). Tal forma de representação pode ser calculada com uma única passada na imagem original, de acordo com a equação (3-1).

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x', y'), \tag{3-1}$$

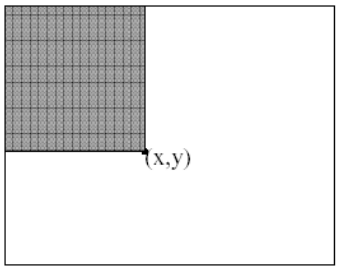

Figura 13. Exemplo de área retangular a ser calculada na Imagem Integral (extraído de Viola e Jones (2001))

Tendo-se calculado a imagem integral, é possível encontrar o valor de uma área retangular qualquer utilizando apenas os quatro pontos dos vértices da área desejada. Supondo que temos o valor total da área de origem da imagem até cada um dos quatro vértices, para se encontrar a área desejada basta realizar uma subtração simples de retângulos. Por exemplo, na figura 14 para encontrar a área da região D, seria necessário fazer o simples cálculo *4 + 1 – (2 + 3)*. Desta maneira, o cálculo de qualquer região de uma imagem integral pode ser realizado em tempo constante.

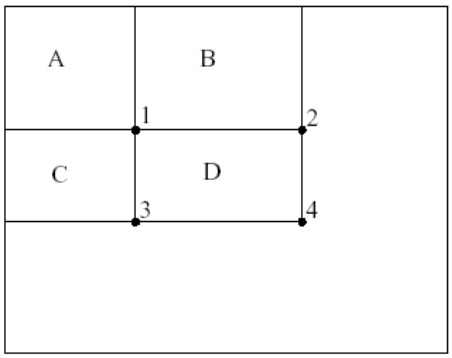

Figura 14. Cálculo da área de região retangular de uma Imagem Integral

Com o intuito de melhorar ainda mais a etapa de detecção de objetos, Lienhart e Maydt (2002) estenderam a quantidade de tipos possíveis de

características retangulares, propondo características novas às previamente sugeridas, mas desta vez rotacionadas em 45 graus. Com estas novas adições ao conjunto, consegue-se reduzir o número de falsos positivos em torno de 10%, aumentando ainda mais a eficiência do processo de detecção. Na figura 15 é apresentado o conjunto completo com as novas características retangulares propostas.

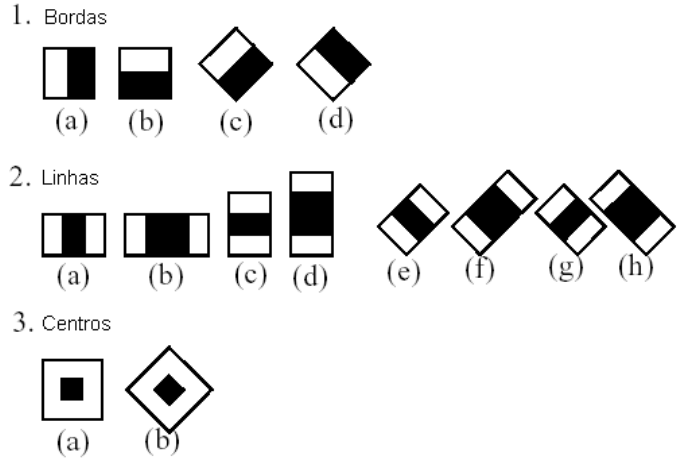

Figura 15. Características Estendidas (extraído de Lienhart e Maydt (2002))

3.3. Funções de Classificação

O objetivo principal a ser alcançado por uma função de classificação é o de ser capaz de classificar corretamente um dado objeto a partir do conjunto de suas principais características. Para descobrir quais são tais características, podem ser utilizados vários tipos de algoritmos de aprendizado, tais como, por exemplo, redes neurais e SVM (Support Vector Machine) (Meyer et al. 2003). O algoritmo escolhido por Viola e Jones (2001) é uma variante do algoritmo de aprendizado AdaBoost (Freund e Schapire, 1995).

Um classificador construído sobre um bom conjunto de características passa a avaliar regiões da imagem de forma correta e precisa. No exemplo da figura 16 o classificador percorre a imagem procurando regiões com os mesmos padrões que as características do objeto desejado, no exemplo da figura em questão este objeto é uma face de um ser humano. Na primeira linha da figura 16 estão exemplos da primeira e segunda características selecionadas pelo AdaBoost durante o treinamento do tipo de objeto “face humana”. Na segunda linha da figura as características estão sobrepostas sobre uma face qualquer, podendo-se observar que a primeira característica mais marcante mostra que a

região dos olhos é geralmente mais escura que a região das bochechas. Nota-se também que a segunda característica selecionada mostra a diferença entre as intensidades da região dos olhos e do nariz.

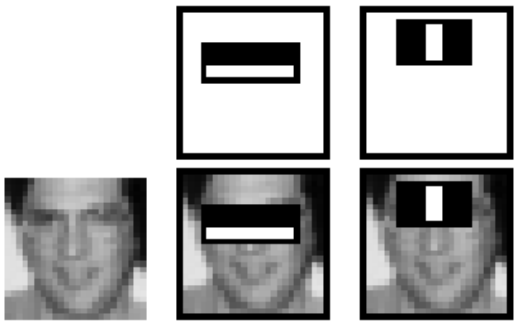

Figura 16. Características retangulares mais marcantes de uma face (extraído de Viola e Jones (2001))

Conforme já mencionado, para que o algoritmo de aprendizado seja executado é necessário um grupo de imagens do objeto de interesse, chamado de **imagens positivas**, e um segundo grupo de imagens que não contenha o objeto, chamado de **imagens negativas**. O AdaBoost é utilizado tanto para selecionar as características quanto para treinar o classificador, mostrando-se ser um algoritmo eficiente para encontrar um pequeno número de características altamente relevantes no objeto comparando as imagens nos conjuntos positivos e negativos. Em seguida é demonstrado o pseudocódigo do algoritmo para o treinamento dos classificadores:

- Dado os exemplos de imagens (x_1,y_1), ..., (x_n,y_n) onde $y_i = 0,1$ para imagens negativas e positivas respectivamente;
- Inicializar os pesos $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ para $y_0 = 0,1$, respectivamente, onde m e l são o número de imagens negativas e positivas;
- De $t = 1$ até T repetir:
    - Normalizar os pesos, $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$ para que w_t seja uma distribuição de probabilidade.
    - Para cada característica, j, treinar um classificador h_j que seja restrito a utilizar apenas uma característica. O erro é calculado respeitando-se $w_t, \epsilon_j = \sum_i w_i |h_j(x_i) - y_i|$.
    - Escolher o classificador, h_t, com o menor erro ϵ_t.

- Atualizar os pesos: $w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$
  onde e_i=0 se o exemplo x_i for classificado corretamente, e e_i = 1 caso contrário, e $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.

- Ao término, o classificador forte é:

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{Caso contrário} \end{cases}$$

,

onde $\alpha_t = \log \frac{1}{\beta_t}$.

Com a utilização do algoritmo de treinamento apresentado, Viola e Jones (2001) conseguiram obter ótimos resultados, onde um classificador de teste criado com 200 características retangulares e uma de taxa de detecção desejada de 95%, o número de falsos positivos obtidos num total de 14.804 imagens testadas foi de apenas 1.

3.4. Cascata de Classificadores

A idéia de usar uma cascata (árvore) de classificadores com N estágios tem como objetivo otimizar o reconhecimento de objetos. Estágios dentro de uma cascata são criados através da combinação de funções de classificação previamente montadas pelo uso do algoritmo de aprendizado AdaBoost. O principal objetivo da cascata de classificadores é fazer com que seus estágios iniciais descartem um grande número de regiões que contém o objeto desejado, e estágios mais avançados sejam cada vez mais precisos para evitar um falso positivo na região sendo analisada. Caso uma área da imagem passe pelo último estágio da cascata, então esta área contém o objeto desejado.

Na figura 17 são representados os N estágios de uma cascata de classificadores. Cada um destes estágios (C_0, C_1,..., C_N) deve descartar ao máximo o número regiões da imagem que não contém o objeto, a fim de diminuir a quantidade de processamento de outras sub-áreas da imagem original.

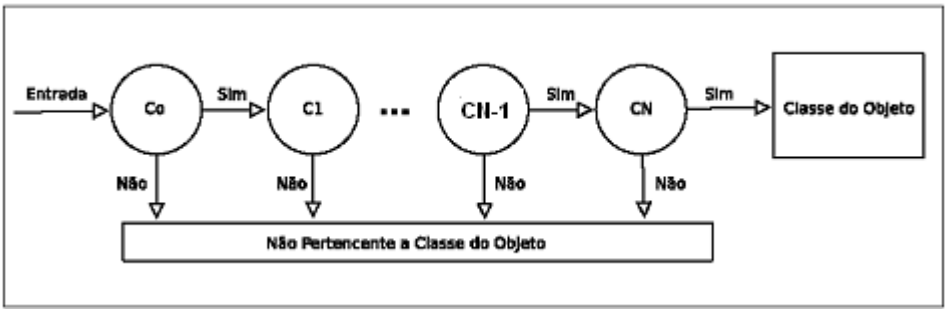

Figura 17. Cascata de Classificadores

O processo para geração da cascata é guiado por um conjunto de metas de detecção e de desempenho. O número de estágios na cascata devem ser o suficiente para garantir uma taxa elevada de detecção de objetos, uma baixa ocorrência de falsos positivos e a minimização do tempo de processamento durante o teste de uma região da imagem.

Dado uma cascata de classificadores treinada, a sua taxa de falsos positivos é de

$$F = \prod_{i=1}^{K} f_i, \tag{3-2}$$

onde F é a taxa de falsos positivos do classificador, K é o número de classificadores e f_i é a taxa de falso positivo do *i-ésimo* classificador dos exemplos que conseguem passar por ele. A taxa de detecção é de

$$D = \prod_{i=1}^{K} d_i, \tag{3-3}$$

onde D é a taxa de detecção do classificador, K é o número de classificadores e d_i é a taxa de detecção do *i-ésimo* classificador dos exemplos que conseguem passar por ele. Definindo-se taxas gerais de detecção e de falso positivo permitido na cascata, é possível também definir metas para cada estágio que a compõe. Por exemplo, uma taxa de detecção de 0,9 pode ser alcançada pelo décimo estágio de classificador se cada um dos estágios possuírem uma taxa de 0,99, sabendo-se que $0,9 \approx 0,99^{10}$.

Na prática, uma arquitetura muito simples pode ser usada para produzir uma eficiente cascata de classificadores. O usuário pode informar as taxas mínimas aceitáveis para f_i e d_i. Cada estágio da cascata é então treinada pelo AdaBoost, conforme mostrado na seção anterior, com o número de

características utilizadas sendo aumentadas até que se atinjam as taxas de detecção e falsos positivos previstos. As taxas são encontradas testando-se o detector corrente sobre um conjunto de validação. Se a taxa de falsos positivos ainda não é atendida, então um novo estágio é adicionado à cascata. O conjunto de imagens negativas para treinar os estágios subseqüentes é obtido coletando todas as falsas detecções encontradas ao processar o detector corrente sobre as imagens que não contêm alguma instância do objeto desejado. Este algoritmo é demonstrado abaixo:

- O usuário informa os valores para f, o máximo de falsos positivos aceitável, e d, o mínimo de detecções aceitáveis por estágio.
- O usuário também informa a taxa geral de máximo de falsos positivos, F_{geral}.
- P = conjunto de imagens positivas.
- N = conjunto de imagens negativas.
- $F_0 = 1{,}0$. $D_0 = 1{,}0$
- $i = 0$
- Enquanto $F_i > F_{\text{geral}}$
    - $i = i + 1$
    - $n_i = 0$; $F_i = F_{i-1}$
    - Enquanto $F_i > f \times F_{i-1}$
        - $n_i = n_i + 1$
        - Usar P e N para treinar um classificador com n_i características usando AdaBoost
        - Processar a cascata corrente sobre o conjunto de validação a fim de determinar F_i e D_i
        - Decrementar o limiar para o *i-ésimo* classificador até que a cascata corrente tenha uma taxa de detecção de pelo menos $d \times D_{i-1}$
    - N = vazio
    - Se $F_i > F_{geral}$ então processar a cascata corrente sobre o conjunto de imagens que não contém o objeto e colocar quaisquer falsos positivos dentro do conjunto N.

Apesar dos estudos realizados demonstrarem que a utilização de uma cascata de classificadores bem treinados ser extremamente eficiente quanto à detecção de objetos em velocidades muito rápidas, o processo de treinamento

em si para geração da cascata é extremamente demorado. No trabalho apresentado por Viola e Jones (2001), foi realizado o treinamento de uma cascata de classificadores para detecção de faces humanas, com 32 estágios, 4.297 características, 5.000 imagens de faces (24 x 24 pixels) e 10.000 imagens que não continham faces. O tempo total de treinamento segundo o trabalho publicado por eles foi de semanas numa máquina de processador simples de 466 MHz. Este tempo elevado de treinamento para novos objetos é a principal desvantagem da técnica apresentada como um todo.

3.5. A Busca pelo Objeto

Para que o algoritmo possa detectar o objeto de interesse numa imagem, é necessário que a imagem seja percorrida diversas vezes em vários tamanhos, fazendo com que cada vez seja analisada uma região de tamanho diferente, o que é chamado de **janela de busca**. O usuário define o tamanho mínimo que esta janela de busca pode ter e que será considerado como o tamanho inicial para o processamento de cada região. O tamanho mínimo desta janela de busca deve ser maior ou igual ao tamanho com o qual a cascata foi treinada, ou seja, se a cascata foi treinada com imagens positivas de 24 x 24 pixels, o tamanho mínimo da janela de busca deve seguir tal resolução.

A imagem deve ser percorrida em múltiplas locações. Para tanto, a janela de busca deve ser deslocada de um valor Δ de pixels na horizontal enquanto houver janelas a serem analisadas no eixo *x* da imagem, e do mesmo valor Δ de pixels na vertical, enquanto houver tamanho suficiente para novas janelas no eixo *y*. O usuário deve ter cuidado no valor escolhido para Δ, pois apesar de um número alto fazer com que menos janelas de busca sejam processadas, diminuirá o número de detecções que deveriam ser feitas, e um número pequeno para Δ acarretará o efeito contrário. Experimentos têm demonstrado que Δ igual a 1,0 pixel é um bom valor para o deslocamento horizontal/vertical da janela de busca, apesar de ser um pouco lento para imagens de resoluções maiores.

A imagem também deve ser percorrida em várias escalas. Portanto, da mesma forma como se tem um valor Δ para o deslocamento da janela de busca, também é necessário existir um fator δ de escalonamento da janela, ou seja, um δ que aumente o tamanho da janela como um todo. Desta forma, cada vez que a janela de busca terminar de percorrer toda a imagem na horizontal e na vertical, a imagem deve ser percorrida novamente com um novo tamanho para a janela

de busca. Baseado em seus próprios experimentos, Viola e Jones (2001) sugerem 1,25 como sendo o valor ideal para δ. O mesmo cuidado sobre o valor utilizado para o fator de Δ deve ser tomado na decisão do fator δ, pois, apesar de deixar a busca com uma velocidade maior, o caso de valores maiores também acarreta o aumento de falsos positivos indesejados. O pseudocódigo é apresentado abaixo:

- Dado $J_{(wj,hj)}$ o tamanho inicial da janela de busca
- Dado $I_{(w,h)}$ a imagem original
- Dado δ o fator de escala
- Dado Δ o fator de deslocamento da Janela
- Enquanto $J_{(wj,hj)} * \delta < I_{(w,h)}$ faça
    - $Y = 0; X = 0;$
    - Enquanto $Y + \Delta < I_{(h)}$ faça
        - Enquanto $X + \Delta < I_{(w)}$ faça
            - Verificar se a área da janela J na imagem contém ou não o objeto procurado.
            - Caso positivo, armazenar as informações da localização atual da janela de busca.
- Marcar na imagem as localizações de objetos encontradas.

O fato da janela de busca ter que ser redimensionada várias vezes durante a busca por regiões que contenham o objeto de interesse não implica em novos cálculos para redimensionar o classificador corrente da cascata. Isto ocorre porque, com o uso da técnica de imagem integral citada em seções anteriores, o custo do re-escalonamento do detector é feito em tempo constante, através de um cálculo algébrico direto. Desta forma, a análise da região da imagem é feita de forma muito rápida e eficiente.

3.6. Conclusão

Viola e Jones (2001) apresentam um novo conjunto de soluções que abordam o problema de detecção de objetos e que minimizam o custo de processamento, enquanto atingem altos níveis de detecção. No trabalho destes autores são demonstrados novos algoritmos, representações, e visões bastante

genéricas que podem ter aplicabilidades em outras áreas da visão computacional e no processamento de imagens.

A primeira contribuição de Viola e Jones (2001) é a introdução de um novo conceito chamado de Imagem Integral. A sua utilização tem como grande vantagem o fato dela ser invariável a redimensionamentos e deslocamentos de janelas. Portanto, como para se detectar um objeto numa imagem ela deve ser percorrida várias vezes e em diferentes tamanhos, através do uso da imagem integral é possível realizar a mudança de tamanho e locação sem perdas substanciais de desempenho, pois o cálculo é realizado em tempo constante não havendo a necessidade de se re-dimensionar a imagem original.

A segunda contribuição é a técnica utilizada para a seleção e escolha de características baseado no algoritmo de aprendizado AdaBoost (Freund e Schapire, 1995). Esta técnica é capaz de detectar as características mais importantes de um objeto e criar um classificador extremamente eficiente.

Por fim, é demonstrado um método para combinar os classificadores entre si com o objetivo de se formar uma cascata de classificadores. Através dos vários estágios da cascata, é possível descartar rapidamente áreas que não possuem o objeto desejado, e concentrar o processamento apenas em áreas promissoras.

Apesar de todo o trabalho proposto por Viola e Jones (2001) ser muito eficiente, ele possui alguns pontos negativos. Por exemplo, ao se utilizar o algoritmo de detecção em vídeos de alta resolução, a taxa de detecção deixa de ocorrer em tempo real. O mesmo problema ocorre com a busca simultânea de vários objetos na cena, pois a taxa de *frames* por segundo cai de forma considerável. As melhorias propostas no algoritmo de detecção são discutidas no próximo capítulo, no qual são abordadas questões como segmentação de vídeo e paralelismo.