Capítulo 2:
Projeto de Operadores pela Aprendizagem

Resumo e nossas contribuições

Um operador restrito à janela (W-operador) é uma transformação de imagem onde a cor de um pixel da imagem de saída é escolhida em função das cores da vizinhança desse pixel na imagem de entrada. Os W-operadores desempenham funções essenciais em diversas áreas do Processamento e Análise de Imagens. A maioria dos filtros utilizados no Processamento e Análise de Imagens são W-operadores (por exemplo, a convolução espacial, o filtro mediano, e os operadores morfológicos). A escolha de um W-operador adequado para uma dada aplicação normalmente é feita manualmente, o que é uma tarefa trabalhosa e tediosa. Temos pesquisado o uso da aprendizagem de máquina para automatizar esta tarefa, isto é, projetar um W-operador automaticamente a partir das imagens amostras entrada-saída. Este capítulo descreve as teorias que embasam o projeto automático de W-operadores (a aprendizagem provavelmente aproximadamente correta e a estimação estatística) e as nossas contribuições científicas nesta área. Nesta tese, abordamos somente o projeto de W-operadores de uma imagem binária para outra binária, e de uma imagem binária para outra em níveis de cinza, pois são as áreas onde encontramos as aplicações mais interessantes.

Formalizamos o problema de aprendizagem de W-operadores binários usando a teoria de aprendizagem PAC (provavelmente aproximadamente correta). Descrevemos como a estimação estatística pode ser utilizada para estimar os erros dos operadores projetado e ótimo. Também utilizamos a estimação estatística para comparar os dife-

rentes métodos de aprendizagem de máquina quanto a acurácia esperada do operador projetado e para escolher uma janela conveniente. Depois, aplicamos as teorias PAC e estimação estatística no problema de aumento da resolução espacial de imagens binárias e meio-tom. Por fim, aplicamos a aprendizagem no problema de meio-tom inverso.

Diferentemente das outras áreas descritas nesta tese, tenho realizado praticamente sozinho as pesquisas nesta área, com pouca ajuda dos meus orientandos e de outros pesquisadores. As nossas principais contribuições científicas na área de projeto automático de W-operadores pela aprendizagem de máquina são:

1) *Aumento de resolução de imagens binárias:* Esta contribuição científica foi publicada em [Ri03; Ci02; Cn10]. Nesta tese, ela está documentada na seção 2.3. Eu fui o principal responsável por esta contribuição, contando com alguma colaboração do meu ex-orientando de doutorado Paulo S. L. M. Barreto.
   Resumo: Num ambiente de escritório típico, equipamentos e softwares heterogêneos, cada um trabalhando numa resolução espacial diferente, devem interagir juntos. Assim, freqüentemente aparece o problema de conversão de resolução. Esta contribuição trata do problema de aumento de resolução espacial (ou ampliação) de documentos e imagens binárias (por exemplo, a conversão de uma imagem 300 dpi em 600 dpi). Uma solução nova, acurada e eficiente para este problema é proposta. Ela utiliza a aprendizagem *k*-NN (*k* vizinhos mais próximos) para projetar automaticamente os operadores de ampliação restritos à janela a partir dos pares de imagens entrada-saída de treinamento. O operador resultante é armazenado numa look-up-table, que é extremamente rápida computacionalmente. É útil conhecer, *a priori*, a complexidade de amostra (a quantidade de amostras de treinamento necessária para obter, com probabilidade 1-δ, um operador com a acurácia ε). Utilizamos a teoria de aprendizagem PAC (provavelmente aproximadamente correta) para calculá-la, nos casos sem ruído e ruidoso. Como a teoria PAC geralmente superestima a complexidade de amostra, a estimação estatística é utilizada para

estimar, *a posteriori*, um intervalo estreito para o erro. A estimação estatística também é usada para mostrar que a aprendizagem *k*-NN possui um bom viés indutivo que permite reduzir o tamanho necessário das imagens amostras.

2) *Aumento de resolução de imagens meio-tom:* Esta contribuição científica foi publicada em [Ri05; Ci05]. Nesta tese, ela está documentada na seção 2.4. Eu fui o principal responsável por esta contribuição.

   Resumo: Esta contribuição trata-se de uma técnica nova, acurada e eficiente para aumentar a resolução espacial de imagens meio-tom. Essa técnica faz uso de um processo de aprendizagem de máquina para projetar automaticamente um operador de ampliação a partir das imagens amostras de entrada-saída. Para ampliar com acurácia uma imagem meio-tom, uma ampla janela e grandes imagens amostras devem ser usadas. Infelizmente, neste caso, o tempo de execução da maioria das técnicas anteriores torna-se proibitivo. A nova solução supera esta dificuldade utilizando a aprendizagem pela árvore de decisão (decision tree, abreviado como DT). A aprendizagem DT original é alterada para obter uma técnica mais eficiente denominada aprendizagem WZDT. É útil conhecer, *a priori*, a complexidade de amostra (o número de amostras de treinamento necessário para obter, com probabilidade 1-δ, um operador com acurácia ε): usamos a aprendizagem provavelmente aproximadamente correta (PAC) para calculá-la. Como a teoria PAC normalmente superestima a complexidade de amostra, a estimação estatística é usada para avaliar, *a posteriori*, um intervalo estreito para o erro. A estimação estatística é também usada para escolher uma janela apropriada e para mostrar que a aprendizagem DT tem um bom viés indutivo. A nova técnica é mais acurada que a ampliação baseada em técnicas de meio-tom inverso simples. A qualidade da solução proposta está muito próxima da qualidade ótima possível de ser obtida, para um processo de ampliação baseada em vizinhança e usando a distância de Hamming para quantificar o erro.

3) *Meio-tom inverso pela aprendizagem.* Esta contribuição científica foi publicada em [Ci11]. Nesta tese, ela está documentada na seção 2.5. Eu fui o principal responsável por esta contribuição, com a colaboração do prof. Ricardo de Queiroz da UnB.

   Resumo: O meio-tom inverso (inverse halftoning, abreviado como IH) é o processo usado para obter uma imagem em níveis de cinza a partir da imagem meio-tom correspondente. Recentemente, as técnicas de IH baseadas na aprendizagem de máquina foram propostas. A aprendizagem por árvore de decisão tem sido aplicada com sucesso em várias tarefas de aprendizagem de máquina durante bastante tempo. Nesta pesquisa, propomos usar a árvore de decisão para resolver o problema de IH. Isto permite-nos reusar alguns algoritmos já desenvolvidos e testados. Especialmente, a maximização do ganho de entropia é uma idéia poderosa que faz com que o algoritmo de aprendizagem selecione automaticamente a janela ideal à medida que a árvore de decisão é construída. A nova técnica gerou imagens em níveis de cinza com PSNR vários dB acima daqueles previamente reportados na literatura. Além disso, ela possui uma implementação muito rápida, possibilitando usá-la em aplicações de tempo real.

4) Temos também utilizado o projeto automático de W-operadores por aprendizagem de máquina em outras aplicações, tais como emulação de operadores em níveis de cinza ou coloridos [Ri01; T02], reconhecimento de texturas [Ci01] e reconhecimento de caracteres (OCR) sem segmentação [Cn06], mas essas aplicações não estão documentadas nesta tese.

2.1 Introdução

Em Processamento e Análise de Imagens, os operadores restritos à janela (W-operadores) desempenham um papel fundamental. Um W-operador é uma transformação de imagem onde a cor de um pixel da imagem de saída é decidida em função

das cores do pixel na imagem de entrada correspondente e seus vizinhos (veja a figura 2.1).

Muitos operadores clássicos de diferentes ramos do Processamento e Análise de Imagens são W-operadores (convolução espacial, mediana, filtro de pilha, erosão, dilatação, abertura, fechamento, hit-miss, etc.). As transformações mais complexas de imagens (emagrecimento, esqueletonização, reconstrução, divisor d'água, etc.) costumam utilizar os W-operadores como seus blocos construtores.

Um W-operador, que desempenha um determinado papel numa aplicação de Processamento e Análise de Imagens, é tradicionalmente projetado manualmente, e esta tarefa é muitas vezes laboriosa e tediosa. Muitas técnicas diferentes têm sido propostas para facilitá-la. Temos trabalhado com a aprendizagem de máquina no projeto automático de W-operadores a partir das imagens exemplos.

Nesta abordagem, um W-operador Ψ é projetado automaticamente a partir da distribuição da probabilidade P responsável pela geração das imagens de entrada $Q^{\mathbf{x}}$ e de saída $Q^{\mathbf{y}}$. Por exemplo, suponha que $Q^{\mathbf{x}}$ seja uma imagem ruidosa e $Q^{\mathbf{y}}$ a imagem limpa correspondente. Supondo totalmente conhecido o processo estatístico P de corrupção da imagem $Q^{\mathbf{y}}$, é possível construir o operador Ψ de forma que a imagem processada $\hat{Q}^{\mathbf{y}} = \Psi(Q^{\mathbf{x}})$ seja "semelhante" à imagem ideal $Q^{\mathbf{y}}$. Isto é, Ψ é projetado para minimizar a esperança da diferença entre $\hat{Q}^{\mathbf{y}}$ e $Q^{\mathbf{y}}$. Por exemplo, os livros clássicos de processamento de imagens como [Gonzalez and Woods, 1992] trazem as técnicas lineares para a restauração de imagens, baseadas na transformada de Fourier bidimensional. Os trabalhos [Coyle and Lin, 1988] e [Lee et al., 1997] projetam o "filtro de pilha" que minimiza o erro médio absoluto e os trabalhos [Dougherty, 1992a] e [Dougherty, 1992b] projetam o operador morfológico que minimiza o erro médio quadrático.

Na prática, a distribuição P é normalmente desconhecida. Assim, uma abordagem mais pragmática emprega as imagens de treinamento $A^{\mathbf{x}}$ (entrada) e $A^{\mathbf{y}}$ (saída), que

são as realizações da distribuição P, ao invés da própria distribuição P, para projetar o operador Ψ automaticamente por um processo de aprendizagem de máquina.

Muitas abordagens diferentes de aprendizagem de máquina podem ser utilizadas para projetar W-operadores: algoritmos genéticos, redes neurais, aprendizagem bayesiana, etc. Mas, para o problema presente, o desempenho computacional é a pedra de toque que distingue os métodos úteis daqueles que são impraticáveis, pois as imagens e as janelas envolvidas são normalmente muito grandes, e assim uma técnica inadequada poderia levar meses ou anos para processar uma única imagem. Ousaríamos dizer que provavelmente a aprendizagem de máquina ainda não é mais amplamente utilizada para projetar W-operadores devido ao fraco desempenho computacional dos algoritmos de aprendizagem, quando estes são escolhidos sem uma preocupação criteriosa pelo seu desempenho.

O desempenho de um algoritmo de aprendizagem deve ser medida analisando três parâmetros: tempo para aprender um W-operador (tempo de treinamento), tempo para aplicar um W-operador previamente construído a uma imagem (tempo de aplicação) e a quantidade de memória de computador necessária (espaço necessário). Como uma propriedade essencial, o tempo de aplicação deve ser curto, pois de outro modo o método nunca poderá ser utilizado em qualquer aplicação prática, notavelmente nas aplicações de tempo real. Embora não seja tão essencial, é muito conveniente que o tempo de treinamento também seja curto, para não aborrecer o usuário. Finalmente, o requerimento do espaço usualmente não é muito preocupante, desde que o W-operador caiba dentro da memória de um computador comum.

Para atingir o desempenho computacional necessário, temos utilizado a aprendizagem k vizinhos mais próximos (k-NN) [Cover and Hart, 1967; Mitchell, 1997] e a aprendizagem por árvore de decisão (DT) [Mitchell, 1997]. Conforme descrevemos mais abaixo, estes dois métodos podem se tornar extremamente rápidos se as estruturas de dados convenientes forem utilizadas.

O viés indutivo (inductive bias) é um assunto bastante discutido na aprendizagem de máquina. O viés indutivo é o conjunto de suposições *a priori* pelas quais o aprendiz generaliza além dos dados observados para inferir a classificação de novas instâncias. Um algoritmo de aprendizagem que não fizesse suposições *a priori* no que diz respeito ao conceito alvo, não possuiria nenhuma base racional para classificar qualquer instância ainda não vista. Se um viés indutivo confiável for usado, a imagem processada será semelhante à imagem de saída ideal, mesmo usando somente uma pequena quantidade de amostras de treinamento. Ambas as técnicas k-NN e DT têm vieses indutivos sólidos. O viés indutivo da aprendizagem k-NN corresponde à suposição de que a classificação de uma instância será mais semelhante às classificações de outras instâncias que estão próximas em distância. Isto é especialmente verdadeiro para o problema que estamos tratando, pois é muito natural e intuitivo atribuir uma cor de saída semelhante aos padrões visualmente semelhantes. O viés indutivo da aprendizagem de árvore de decisão é conhecido como "a navalha de Occam" [Mitchell, 1997, cap. 3], que diz: "Prefira a hipótese mais simples que se ajusta aos dados". O algoritmo de construção de DT coloca os atributos de alto ganho de informação mais próximos da raiz. Esta prática corresponde a adotar o viés indutivo que prefere as árvores mais baixas às mais altas, isto é, a navalha de Occam.

Utilizando a teoria de aprendizagem computacional PAC [Mitchell, 1997; Anthony and Biggs, 1992; Haussler, 1992], é possível pré-calcular o tamanho necessário da amostra para que o W-operador aprendido atinja uma precisão ε com a probabilidade $1-\delta$, independentemente do método particular de aprendizagem de máquina adotado (basta que o método seja consistente, isto é, produza W-operador que concorde com os exemplos de treinamento). Porém, os resultados fornecidos por esta teoria costumam superestimar o tamanho da amostra, pois não considera o viés indutivo do particular método de aprendizagem. Este problema não pode ser contornado mesmo utilizando as teorias mais fortes, como a dimensão Vapnik-Chervonenkis [Vapnik, 1995; Mitchell, 1997]. Para superar esta dificuldade, além de utilizar a teoria PAC *a priori* (isto é, antes de realizar a aprendizagem), temos também utilizado os métodos

de estimação estatística *a posteriori*. A estimação estatística permite estimar precisamente a taxa de erro real do W-operador projetado.

O problema torna-se um pouco mais complexo quando há ruídos nas amostras de treinamento. Ou, equivalentemente, se o *professor* pode cometer alguns erros ao ensinar o *aprendiz*. Neste caso, o W-operador ótimo possui uma taxa de erro maior que zero. Esta taxa mínima de erro pode ser medida empiricamente e é possível construir um intervalo de confiança para essa medida. Além disso, utilizando a estimação estatística, dois métodos de aprendizagem diferentes podem ser comparados entre si quanto à acurácia, o que nos permite decidir, por exemplo, se o método k-NN é superior ou inferior à aprendizagem DT para uma determinada aplicação.

O algoritmo força-bruta para a aprendizagem k-NN é extremamente lento, pois para cada pixel da imagem a ser processada, deve-se fazer uma busca exaustiva na imagem de treinamento. Os trabalhos [Ci01] e [Ci02; Ri03] propõe duas soluções para o problema: o uso da kd-árvore (árvore binária multidimensional [Bentley, 1975; Friedman et al., 1977; Preparata and Shamos, 1985]) e look-up-table (LUT). A velocidade de treinamento da kd-árvore é bastante boa, porém a sua velocidade de aplicação somente é satisfatória para dimensões pequenas, piorando rapidamente com o aumento da dimensão. Por outro lado, a velocidade de aplicação da LUT é ótima em qualquer dimensão, porém a velocidade de treinamento e a memória necessária crescem exponencialmente com o aumento da dimensão.

A aprendizagem DT pode ser vista como uma kd-árvore sem o processo de backtracking. O uso da aprendizagem DT é especialmente recomendado para situações onde muitos atributos são irrelevantes para o conceito que está sendo aprendido. As experiências empíricas têm mostrado que o viés indutivo de DT é ligeiramente pior que k-NN para o problema de aprendizagem de W-operador. Porém, a árvore de decisão é rápida tanto no treinamento quanto na aplicação, propriedade que torna o seu uso na prática extremamente atraente.

O projeto de operadores pela aprendizagem computacional tem sido aplicado com sucesso em diferentes áreas, como na emulação de filtros desconhecidos [T02; Ri01; Ri03; Cn10], para atenuar ruídos [T02; Ri03], na segmentação da imagem de acordo com a textura [T02; Ci01], em OCR [Cn06], para aumentar a resolução de imagens binárias pela aprendizagem k-NN [Ci02; Ri03; Cn10], para aumentar a resolução de imagens meio-tom pela aprendizagem DT [Ci05; Ri05] e no problema de meio-tom inverso [Ci11]. Esta tese descreve detalhadamente somente as três últimas aplicações.

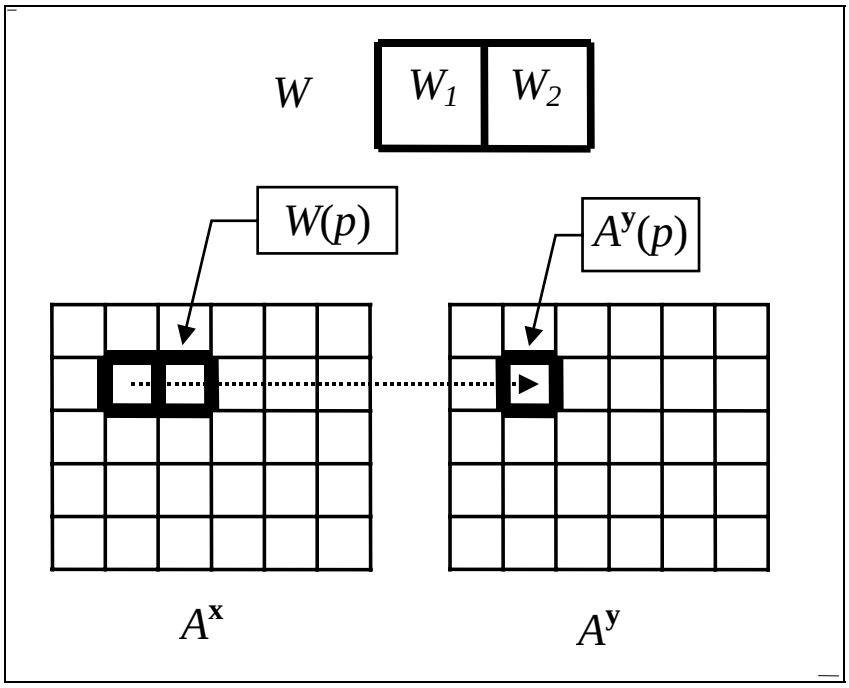


Fig. 2.1: Um W-operador decide a cor de um pixel p na imagem de saída $A^{\mathbf{y}}$ analisando uma vizinhança $W(p)$ do pixel p na imagem de entrada $A^{\mathbf{x}}$.

Organização deste capítulo

O restante deste capítulo está organizado como segue. A seção 2.2 apresenta as teorias que embasam o projeto automático de W-operadores por aprendizagem de máquina. O problema é formalizado como um processo de aprendizagem computacional PAC e analisamos os casos sem ruído e ruidoso. Depois, descrevemos a teoria da estimação estatística. Em seguida, expomos o algoritmo de aprendizagem *k*-NN e sugerimos uma pequena alteração nele para torná-lo empiricamente ótimo. Este algoritmo alterado será denotado como ek-NN. Explicamos também a aprendizagem por árvore de decisão (DT). Terminamos a seção explicando como a estimação estatística pode ser usada para comparar diferentes algoritmos de aprendizagem ou diferentes janelas. A seção 2.3 trata do problema de aumento de resolução de imagens binárias (documentos impressos ou manuscritos, ortográficos ou escaneados), usando a aprendizagem ek-NN. A seção 2.4 trata do problema de aumento de resolução de imagens meio-tom usando o algoritmo DT. Sugerimos uma alteração no algoritmo DT para torná-lo mais eficiente no problema de ampliação de imagens meio-tom. O algoritmo modificado é chamado de aprendizagem WZDT. A seção 2.5 trata do problema de meio-tom inverso usando a árvore de decisão. Finalmente, a seção 2.6 apresenta as nossas conclusões.

2.2 Aprendizagem de W-Operadores Binários

Introdução

Nesta seção, analisaremos o caso binário do problema do projeto automático de W-operadores pela aprendizagem de máquina. Faremos uso da teoria de aprendizagem PAC clássica [Anthony and Biggs, 1992] e generalizada [Haussler, 1992] para calcular a complexidade de amostra do problema de aprendizagem de operadores binários. Infelizmente, com freqüência, somente uma complexidade de amostra superestimada pode ser obtida utilizando esta teoria. Mesmo assim, ela será útil como um limite superior para a quantidade de amostras necessárias, e para mostrar a convergência do processo de aprendizagem. Além disso, a teoria de aprendizagem PAC irá nos permitir expressar rigorosamente o problema de aprendizagem do W-operador, e pode clarificar consideravelmente a compreensão do problema. Para superar o problema de superestimação da complexidade de amostra, além de utilizar a teoria PAC, temos também utilizado os métodos de estimação estatística. A estimação estatística permite estimar precisamente a taxa de erro real do W-operador projetado.

O problema

Vamos definir uma imagem binária como uma função $Q:\mathbb{Z}^2 \to \{0,1\}$. O suporte de uma imagem binária Q é um subconjunto finito de $\mathbb{Z}^2$ onde a imagem está de fato definida. O tamanho do suporte é o número de pixels da imagem e uma imagem é considerada estar preenchida com uma cor-de-fundo fora do seu suporte.

Um W-operador binário Ψ é uma função que mapeia uma imagem binária numa outra, definida através de um conjunto de w pontos chamado janela

$$W = \{W_1,\ldots,W_w\},\ W_i \in \mathbb{Z}^2$$

e um conceito ou uma função característica $\psi:\{0,1\}^w \to \{0,1\}$ como segue:

$$\Psi(Q)(p) = \psi(Q(W_1 + p),\ldots,Q(W_w + p)),$$

onde $p \in \mathbb{Z}^2$. Cada ponto W_i da janela é chamado *peephole* ou furo-de-espiar.

Sejam as imagens $A^{\mathbf{x}}$, $A^{\mathbf{y}}$, $Q^{\mathbf{x}}$ e $Q^{\mathbf{y}}$ respectivamente a imagem de entrada de treinamento, imagem de saída de treinamento, a imagem a ser processada e a imagem de saída ideal (supostamente desconhecida). Podemos supor que existe um único par de imagens de treinamento ($A^{\mathbf{x}}$ e $A^{\mathbf{y}}$), porque se existirem muitos pares, elas podem ser "coladas" para formarem um único par. A fim de projetar um W-operador $\hat{\Psi}$, o usuário deve escolher manualmente uma janela apropriada *W*.

Vamos denotar o conteúdo em $A^{\mathbf{x}}$, da janela *W* deslocada para $p \in \mathbb{Z}^2$, como $a_p^{\mathbf{x}}$ e denominá-lo uma instância de treinamento ou um padrão de entrada em torno do pixel *p*:

$$a_p^{\mathbf{x}} = [A^{\mathbf{x}}(W_1 + p), A^{\mathbf{x}}(W_2 + p),\ldots,A^{\mathbf{x}}(W_w + p)] \in \{0,1\}^w.$$

Cada padrão $a_p^{\mathbf{x}}$ está associado com uma cor de saída ou classificação $A^{\mathbf{y}}(p) \in \{0,1\}$. Vamos denotar os dados obtidos quando todos os pixels de $A^{\mathbf{x}}$ e $A^{\mathbf{y}}$ são varridos como uma seqüência

$$\vec{a} = \left((a_{p_1}^{\mathbf{x}}, A^{\mathbf{y}}(p_1)),\ldots,(a_{p_m}^{\mathbf{x}}, A^{\mathbf{y}}(p_m))\right)$$

e denominá-la seqüência de amostras (*m* é a quantidade dos pixels das imagens $A^{\mathbf{x}}$ e $A^{\mathbf{y}}$). Cada elemento $(a_{p_i}^{\mathbf{x}}, A^{\mathbf{y}}(p_i)) \in \vec{a}$ é chamado um exemplo ou uma amostra de treinamento. Vamos construir de forma semelhante a seqüência

$$\vec{q} = \left((q_{p_1}^{\mathbf{x}}, Q^{\mathbf{y}}(p_1)),\ldots,(q_{p_n}^{\mathbf{x}}, Q^{\mathbf{y}}(p_n))\right)$$

a partir de $Q^{\mathbf{x}}$ e $Q^{\mathbf{y}}$ (*n* é a quantidade de pixels de $Q^{\mathbf{x}}$ e $Q^{\mathbf{y}}$). Cada $q_{p_i}^{\mathbf{x}}$ é chamado um padrão de busca ou uma instância a-ser-processada, e a saída $Q^{\mathbf{y}}(p_i) \in \{0,1\}$ é chamada a cor de saída ideal ou a classificação ideal.

O aprendiz ou o algoritmo de aprendizagem **A** é requisitado para construir, baseado em $A^{\mathbf{x}}$ e $A^{\mathbf{y}}$, um W-operador $\hat{\Psi}$ tal que, quando $\hat{\Psi}$ é aplicado à $Q^{\mathbf{x}}$, espera-se que a

imagem resultante $\hat{Q}^{\mathbf{y}} = \hat{\Psi}(Q^{\mathbf{x}})$ seja semelhante à imagem de saída ideal $Q^{\mathbf{y}}$. Mais precisamente, o aprendiz **A** deve construir uma função característica ou hipótese $\hat{\psi}$ baseado em seqüência de amostras $\vec{a}$ de forma que, quando $\hat{\psi}$ é aplicado a um padrão a-ser-processado $q^{\mathbf{x}}_{p_i}$, espera-se que a sua classificação $\hat{Q}^{\mathbf{y}}(p_i) = \hat{\psi}(q^{\mathbf{x}}_{p_i})$ seja igual a $Q^{\mathbf{y}}(p_i)$ com alta probabilidade. A função $\hat{\psi}$ e a janela *W* juntas representam o W-operador $\hat{\Psi}$.

Caso sem ruído

Vamos estudar em primeiro lugar o caso sem ruído. Pois, embora a maioria dos problemas práticos seja ruidosa, o estudo do caso sem ruído irá nos ajudar a compreender melhor os casos ruidosos.

Num ambiente sem ruído, existe um conceito alvo claramente definido $\psi : \{0,1\}^W \to \{0,1\}$ que o aprendiz deve aprender. Em tal ambiente, podemos supor que as instâncias de treinamento $a^{\mathbf{x}}_{p_i}$ são geradas aleatória e independentemente no espaço $\{0,1\}^W$ por uma distribuição de probabilidade *P*. Além disso, as cores de saída $A^{\mathbf{y}}(p_i)$ são obtidas aplicando a função alvo ψ em cada $a^{\mathbf{x}}_{p_i}$, isto é, $A^{\mathbf{y}}(p_i) = \psi(a^{\mathbf{x}}_{p_i})$ para todos os pares $(a^{\mathbf{x}}_{p_i}, A^{\mathbf{y}}(p_i)) \in \vec{a}$.

O aprendiz **A** deve considerar algum conjunto $H \subset \left(\{0,1\}^W \to \{0,1\}\right)$ de possíveis hipóteses quando tenta aprender o conceito alvo ψ. Se nenhuma informação sobre ψ estiver disponível, o aprendiz deve assumir que $H = \left(\{0,1\}^W \to \{0,1\}\right)$. Porém, uma informação *a priori* pode simplificar bastante o processo de aprendizagem, pois ela pode reduzir substancialmente a cardinalidade do espaço das hipóteses *H*. Por exemplo, emular uma erosão Ψ com a informação de que Ψ é uma erosão é muito mais fácil do que emulá-la sem nenhuma informação *a priori* (exemplos 2.2 e 2.3). Uma erosão é um operador elementar de morfologia matemática e a sua definição encontra-se, por exemplo, em [Gonzalez and Woods, 1992]. No estágio de treinamento do

W-operador, o aprendiz **A** recebe uma seqüência de amostras $\vec{a}$ e procura uma hipótese $\hat{\psi} = \mathbf{A}(\vec{a})$ no espaço *H*.

Vamos definir o erro verdadeiro (t-erro) da hipótese $\hat{\psi}$ como a probabilidade de que $\hat{\psi}$ irá classificar incorretamente uma instância $q_{p_i}^{\mathbf{x}}$ escolhida aleatoriamente por *P*:

$$\text{t\_error}_P(\hat{\psi}) = P\left\{q_{p_i}^{\mathbf{x}} \in \{0,1\}^w \middle| \psi(q_{p_i}^{\mathbf{x}}) \neq \hat{\psi}(q_{p_i}^{\mathbf{x}})\right\}$$

De acordo com a teoria PAC [Mitchell, 1997; Anthony and Biggs, 1992], qualquer aprendiz consistente utilizando um espaço de hipótese finito *H* com uma função alvo $\psi \in H$ irá, com probabilidade maior que (1-δ), gerar uma hipótese $\hat{\psi}$ com erro menor que ε, depois de observar *m* exemplos escolhidos aleatoriamente pelo *P*, desde que

$$m \geq \frac{1}{\varepsilon}\left[\ln\left(\frac{1}{\delta}\right) + \ln\left(|H|\right)\right]. \tag{2.1}$$

Um aprendiz é consistente se, sempre que possível, gerar uma hipótese que se adapte perfeitamente aos dados de treinamento. O limite (2.1) freqüentemente está substancialmente superestimado, principalmente porque nenhuma suposição foi feita sobre o aprendiz exceto a consistência. Alguns exemplos de uso desta equação seguem.

Exemplo 2.1: Na figura 2.2, uma imagem de impressão digital $A^{\mathbf{x}}$ (2.2a) foi processada por W-operador Ψ, gerando a imagem $A^{\mathbf{y}}$ (2.2b). Este operador consistiu em união de 8 operadores hit-or-miss definidos dentro da janela 3×3. O operador hit-or-miss é um dos operadores elementares da morfologia matemática e a sua definição encontra-se, por exemplo, em [Gonzalez and Woods, 1992]. Vamos supor que de alguma forma conhecemos que Ψ está definida na janela 3×3. Utilizando esta informação e as imagens $A^{\mathbf{x}}$ e $A^{\mathbf{y}}$, um W-operador $\hat{\Psi}$ foi construído por um aprendiz consistente. De acordo com a equação (2.1), com probabilidade maior que 99%, o erro verdadeiro de $\hat{\Psi}$ será menor que 1%, desde que as imagens de treinamento tenham uma quantidade de pixels

$$m \geq \frac{1}{0{,}01}\left[\ln\left(\frac{1}{0{,}01}\right) + \ln\left(2^{2^9}\right)\right] \cong 35950\,.$$

Como as imagens $A^{\mathbf{x}}$ e $A^{\mathbf{y}}$ têm 200×200=40000 pixels, quase certamente $\hat{\Psi}$ irá apresentar uma taxa de erro menor que 1%. De fato, quando $\hat{\Psi}$ foi aplicado a uma outra imagem de impressão digital (figura 2.2c), uma imagem $\hat{Q}^{\mathbf{y}} = \hat{\Psi}(Q^{\mathbf{x}})$ (figura 2.2d) exatamente igual à saída ideal $Q^{\mathbf{y}} = \Psi(Q^{\mathbf{x}})$ foi produzida. Isto é, $\hat{\Psi}$ apresentou erro zero. Este teste foi repetido algumas vezes e as taxas de erro sempre foram zero. ■

Note que a análise acima somente é válida quando se pode supor que as imagens $A^{\mathbf{x}}$ e $Q^{\mathbf{x}}$ foram geradas por uma mesma distribuição de probabilidade. Isto é, $A^{\mathbf{x}}$ e $Q^{\mathbf{x}}$ devem ser do mesmo tipo: imagens de impressões digitais, documentos manuscritos, documentos impressos, etc.

Exemplo 2.2: Vamos resolver novamente o exemplo 2.1, desta vez supondo que o operador alvo é mais complexo e está definido dentro de uma janela 7×7. Neste caso:

$$m \geq \frac{1}{0{,}01}\left[\ln\left(\frac{1}{0{,}01}\right) + \ln\left(2^{2^{49}}\right)\right] \cong 3{,}9\times 10^{16}\,.$$

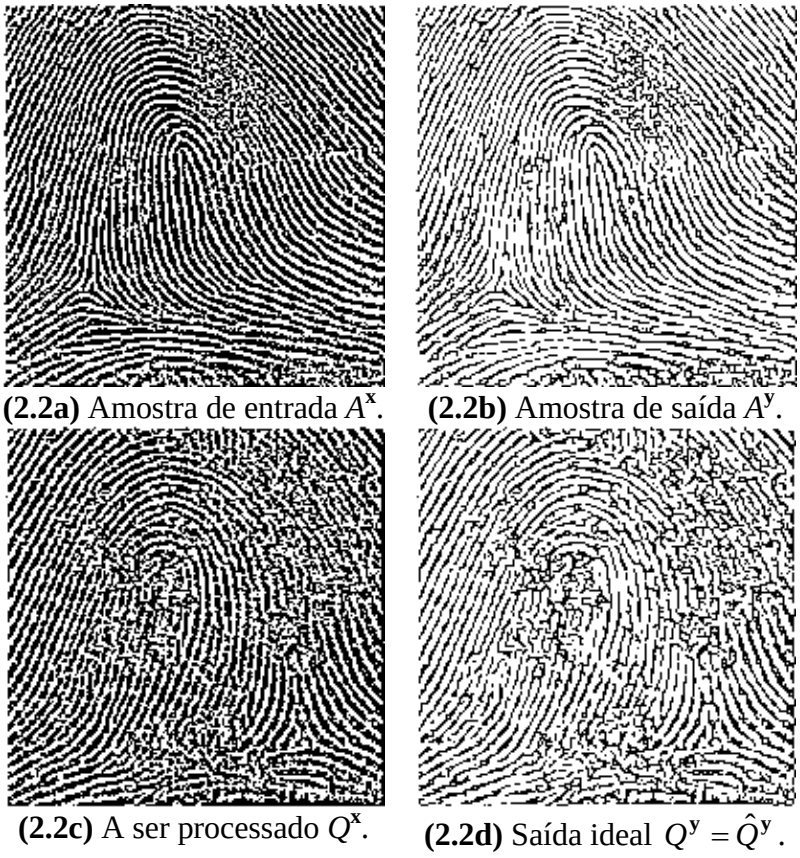
Isto é, as imagens amostras devem ser maiores que $(2\times10^8)\times(2\times10^8)$! Claramente, uma imagem tão grande não pode ser obtida na prática. ■

Exemplo 2.3: Vamos resolver novamente o exemplo 2.2, desta vez supondo que temos conhecimento de que o operador alvo é uma erosão cujo elemento estruturante cabe dentro de uma janela 7×7. Como cada um dos 49 furos pode pertencer ou não ao elemento estruturante, o operador alvo tem de ser uma das 2^{49} erosões. Assim, $|H| = 2^{49}$ e:

$$m \geq \frac{1}{0{,}01}\left[\ln\left(\frac{1}{0{,}01}\right) + \ln\left(2^{49}\right)\right] \cong 3857\,.$$

Isto é, qualquer par de imagens de treinamento maiores que 63×63 será suficiente. Compare com o tamanho das imagens $(2\times10^8)\times(2\times10^8)$ do exemplo 2.2. ■

A simplificação acima somente é válida quando se utiliza um algoritmo de aprendizagem projetado especialmente para as erosões. Resultados semelhantes podem ser obtidos para outros operadores elementares tais como dilatação, hit-or-miss, união de k erosões, e assim por diante.



(2.2a) Amostra de entrada $A^{\mathbf{x}}$. **(2.2b)** Amostra de saída $A^{\mathbf{y}}$.

(2.2c) A ser processado $Q^{\mathbf{x}}$. **(2.2d)** Saída ideal $Q^{\mathbf{y}} = \hat{Q}^{\mathbf{y}}$.

Fig. 2.2: Aprendizagem de W-operador num ambiente sem ruído.

Caso ruidoso

Para modelar o caso ruidoso, vamos supor que cada exemplo $(a_p^{\mathbf{x}}, A^{\mathbf{y}}(p)) \in \vec{a}$ tenha sido gerado independentemente por uma distribuição de probabilidade conjunta *P* desconhecida no espaço $\{0,1\}^w \times \{0,1\}$. Vamos também supor que cada elemento $(q_{p_i}^{\mathbf{x}}, Q^{\mathbf{y}}(p_i)) \in \vec{q}$ tenha sido gerado pela mesma distribuição *P*.

O erro verdadeiro da hipótese ψ agora deve ser definido como a probabilidade de que ψ classifique incorretamente um exemplo $(q_{p_i}^{\mathbf{x}}, Q^{\mathbf{y}}(p_i))$ escolhido aleatoriamente por *P*:

$$\text{t\_error}_P(\psi) = P\left\{(q_{p_i}^{\mathbf{x}}, Q^{\mathbf{y}}(p_i)) \in \{0,1\}^w \times \{0,1\} \middle| \psi(q_{p_i}^{\mathbf{x}}) \neq Q^{\mathbf{y}}(p_i)\right\}$$

Na situação ruidosa, não existe uma função alvo claramente definida. No seu lugar, existe uma função ψ^* com o menor erro verdadeiro. Vamos definir o erro empírico (e-erro) de uma hipótese ψ sobre uma seqüência $\vec{a}$ como a proporção de erros cometidos quando ψ classifica as instâncias de $\vec{a}$:

$$\text{e\_error}_{\vec{a}}(\psi) = \left(\frac{1}{m}\right) \left| \left\{(a_{p_i}^{\mathbf{x}}, A^{\mathbf{y}}(p_i)) \in \vec{a} \;\middle|\; \psi(a_{p_i}^{\mathbf{x}}) \neq A^{\mathbf{y}}(p_i)\right\} \right|,$$

onde *m* é o comprimento de $\vec{a}$.

Seja $\hat{\psi}$ a hipótese com o menor e-erro sobre $\vec{a}$ e seja ψ^* a hipótese com o menor erro verdadeiro. Então [Haussler, 1992]

$$\Pr\left[\text{t\_error}_P(\hat{\psi}) - \text{t\_error}_P(\psi^*) > \varepsilon\right] < \delta,$$

desde que *H* seja finito e o comprimento *m* de $\vec{a}$ satisfaça:

$$m \geq \frac{1}{2\varepsilon^2}\left[\ln\left(\frac{1}{\delta}\right) + \ln\left(2|H|\right)\right]. \qquad (2.2)$$

Infelizmente, a complexidade de amostra acima é uma superestimativa ainda maior que a da equação (2.1). Dada uma seqüência de amostras $\vec{a}$, a hipótese empiricamente ótima (e-ótima) $\hat{\psi}$ pode ser construída facilmente. Vamos definir que um aprendiz

A é e-ótimo se ele gerar sempre uma hipótese e-ótima sobre a seqüência de treinamento. Se **A** fosse e-ótimo, dado um padrão de busca $q_{p_i}^{\mathbf{x}}$, qual deveria ser a sua classificação $\hat{\psi}(q_{p_i}^{\mathbf{x}}) = \mathbf{A}(\vec{a})(q_{p_i}^{\mathbf{x}})$? Sejam $(a_{r_1}^{\mathbf{x}}, A^{\mathbf{y}}(r_1))$, ..., $(a_{r_N}^{\mathbf{x}}, A^{\mathbf{y}}(r_N))$ os N exemplos de treinamento de $q_{p_i}^{\mathbf{x}}$ em $\vec{a}$, isto é, $a_{r_j}^{\mathbf{x}} = q_{p_i}^{\mathbf{x}}$, $1 \le j \le N$ (não há outros exemplos de $q_{p_i}^{\mathbf{x}}$ em $\vec{a}$ além desses). Como há ruído, os N exemplos acima podem não concordar sobre a classificação de $q_{p_i}^{\mathbf{x}}$. Para minimizar e-erro, a classificação deve ser decidida pela maioria dos votos desses exemplos de treinamento:

$$\hat{\psi}(q_{p_i}^{\mathbf{x}}) \leftarrow \text{moda}(A^{\mathbf{y}}(r_1), \ldots, A^{\mathbf{y}}(r_N)).$$

Note que todo aprendiz e-ótimo é consistente num ambiente sem ruído. Apresentamos abaixo um exemplo.

Exemplo 2.4: As imagens de impressões digitais 2.2a e 2.2c foram corrompidas pelo ruído "sal e pimenta", resultando em imagens 2.3a e 2.3b. Em média, 1 em cada 40 pixels mudou de cor. Gostaríamos de projetar um W-operador 3×3 $\hat{\Psi}$ tal que uma imagem semelhante à saída ideal $A^{\mathbf{y}}$ (figura 2.2b) resulte, apesar do ruído, quando a imagem 2.3a é processada por $\hat{\Psi}$. Para atingir este objetivo, um W-operador $\hat{\Psi}$ foi projetado por um aprendiz e-ótimo usando as imagens 2.3a e 2.2b como amostras de treinamento. Como as imagens 2.3a e 2.2b têm 200×200 pixels, com probabilidade pelo menos 99%, a diferença entre os erros verdadeiros do operador ótimo Ψ^* e do operador $\hat{\Psi}$ será menor que 6,71%, i.e., $\text{t\_error}_P(\hat{\psi}) - \text{t\_error}_P(\psi^*) \le 0{,}0671$, pois:

$$\frac{1}{2 \times 0{,}0671^2}\left[\ln\left(\frac{1}{0{,}01}\right) + \ln\left(2 \times 2^{2^9}\right)\right] \cong 40000.$$

No exemplo 2.5, este problema será analisado novamente. ■

(2.3a) Amostra de entrada ruidosa.

(2.3b) Imagem ruidosa a-ser-processada.

(2.3c) Imagem processada.

Fig. 2.3: Aprendizagem de W-operador num ambiente ruidoso.

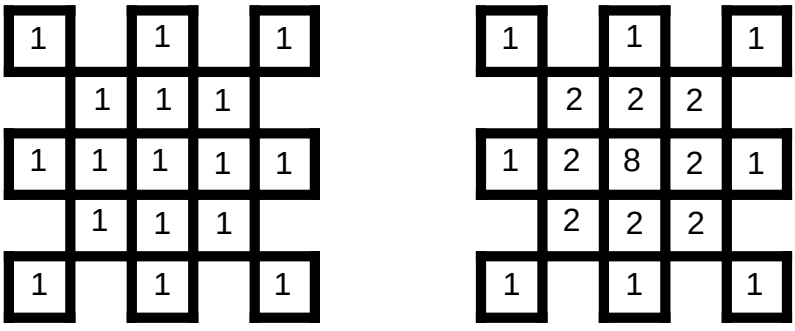


Fig. 2.4: Janelas sem e com pesos, com 17 furos-de-espiar.

Estimação estatística da taxa de erro

Esta subsubseção irá expor as técnicas para calcular um limite mais estreito para a taxa de erro. Estas técnicas serão muito úteis, pois as equações (2.1) e (2.2) normalmente superestimam a complexidade de amostra e a taxa de erro. Ao contrário das fórmulas anteriores, as técnicas desta subsubseção podem ser aplicadas somente após ter projetado W-operador, com a condição adicional de que a imagem de saída ideal $Q^{\mathbf{y}}$ esteja disponível. É lícito supor que a saída ideal estará disponível para se realizar testes, pois estamos supondo que um par de imagens entrada-saída de treinamento está disponível para projetar W-operador. E, se as imagens de treinamento estão disponíveis, elas podem ser quebradas em dois pedaços: imagens de treinamento ($A^{\mathbf{x}}$, $A^{\mathbf{y}}$) e imagens de teste ($Q^{\mathbf{x}}$, $Q^{\mathbf{y}}$).

Portanto, supondo que a saída ideal $Q^{\mathbf{y}}$ esteja disponível, uma simples contagem de pixels diferentes entre $Q^{\mathbf{y}}$ e $\hat{Q}^{\mathbf{y}}$ irá fornecer o e-erro. E, dada a acurácia observada de uma hipótese sobre uma amostra de dados limitada, é possível conhecer o quanto esta irá conseguir estimar a acurácia sobre exemplos adicionais. Para isso, vamos construir intervalos de confiança unilateral ou bilateral. Explicações adicionais sobre intervalos de confiança da média de variáveis aleatórias binomiais encontram-se em [Mitchell, 1997] ou em muitos livros elementares de Estatística. Com N% de confiança:

$$\text{t\_error}_P(\hat{\psi}) \in \text{e\_error}_{\vec{q}}(\hat{\psi}) \pm z_N \sqrt{\frac{\text{e\_error}_{\vec{q}}(\hat{\psi})(1-\text{e\_error}_{\vec{q}}(\hat{\psi}))}{n}}, \qquad (2.3)$$

$$\text{t\_error}_P(\hat{\psi}) \leq \text{e\_error}_{\vec{q}}(\hat{\psi}) + z'_N \sqrt{\frac{\text{e\_error}_{\vec{q}}(\hat{\psi})(1-\text{e\_error}_{\vec{q}}(\hat{\psi}))}{n}}, \qquad (2.4)$$

onde n é o comprimento de $\vec{q}$; z_N define a metade da largura do menor intervalo em torno da média que inclui N% da massa da probabilidade total sob distribuição normal com desvio-padrão 1; e $z'_N \equiv z_{2N-1}$. Por exemplo, $z'_{84\%} = z_{68\%} = 1{,}00$, $z'_{95\%} = z_{90\%} = 1{,}64$, $z'_{99\%} = z_{98\%} = 2{,}33$ e $z_{99\%} = 2{,}58$. As fórmulas (2.3) e (2.4) nor-

malmente produzem uma estimativa da taxa de erro muito mais acurada que as equações (2.1) e (2.2).

No caso sem ruído, basta conhecer um limite superior para a taxa de erro verdadeiro do operador projetado ($\text{t\_error}_P(\hat{\psi})$). Porém, para os casos ruidosos, o erro mínimo ($\text{t\_error}_P(\psi^*)$) também deve ser estimado pois, como o operador projetado nunca poderá atingir uma taxa de erro verdadeiro menor que o mínimo, um operador pode ser considerado uma boa solução se o seu erro verdadeiro estiver próximo do mínimo. Infelizmente, não há meios para se estimar $\text{t\_error}_P(\psi^*)$ diretamente, pois o operador ótimo é desconhecido. Descrevemos abaixo um artifício que tem conseguido estabelecer bons limites inferiores para $\text{t\_error}_P(\psi^*)$. Embora muito simples, nunca vimos esta técnica descrita na literatura.

Para isso, vamos construir a hipótese $\hat{\psi}^*$ e-ótima sobre $\vec{q}$. Se o aprendiz **A** for e-ótimo, $\hat{\psi}^* = \mathbf{A}(\vec{q})$. Note que estamos treinando o operador com as próprias imagens ($Q^{\mathbf{x}}$, $Q^{\mathbf{y}}$) que serão utilizadas no teste. Claramente, $\text{e\_error}_{\vec{q}}(\hat{\psi}^*) \leq \text{e\_error}_{\vec{q}}(\psi^*)$ e $\text{e\_error}_{\vec{q}}(\hat{\psi}^*)$ pode ser medido experimentalmente. Então, utilizamos a seguinte desigualdade para estabelecer um limite inferior para $\text{t\_error}_P(\psi^*)$:

$$\begin{aligned} \text{t\_error}_P(\psi^*) &\geq \text{e\_error}_{\vec{q}}(\psi^*) - z'_N \sqrt{\frac{\text{e\_error}_{\vec{q}}(\psi^*)(1-\text{e\_error}_{\vec{q}}(\psi^*))}{n}} \\ &\geq \text{e\_error}_{\vec{q}}(\hat{\psi}^*) - z'_N \sqrt{\frac{\text{e\_error}_{\vec{q}}(\hat{\psi}^*)(1-\text{e\_error}_{\vec{q}}(\hat{\psi}^*))}{n}} \end{aligned} \quad (2.5)$$

A desigualdade acima é verdadeira, com nível de confiança *N*%, toda vez que:

$$\frac{b+1-\sqrt{b(b+1)}}{2(b+1)} \leq \text{e\_error}_{\vec{q}}(\hat{\psi}^*) \leq \text{e\_error}_{\vec{q}}(\psi^*) \leq \frac{b+1+\sqrt{b(b+1)}}{2(b+1)} \quad (2.6)$$

onde $b = n/(z'_N)^2$. Note que a desigualdade (2.6) é verdadeira para praticamente todos os problemas práticos e conseqüentemente a desigualdade (2.5) também é sempre verdadeira na prática.

Exemplo 2.5: No exemplo 2.4, tínhamos concluído com 99% de confiança que o operador $\hat{\Psi}$ obtido comete no máximo 6,71% mais erros que o operador ótimo 3×3. A fim de estabelecer um limite de erro mais estreito, o e-erro de $\hat{\Psi}$ (a diferença entre as imagens 2.2d e 2.3c) foi medido e descobriu-se que valia 4,992%. Utilizando a equação (2.3), concluímos com 99% de confiança que o erro verdadeiro de $\hat{\Psi}$ pertence ao intervalo (4,992±0,281)%. O operador $\hat{\Psi}^*$ e-ótimo sobre as imagens de teste (figuras 2.3b e 2.2d) foi construído e cometeu e-erro 4,723% quando processou a imagem 2.3b. Utilizando a desigualdade (2.5), concluímos com 99% de confiança que o erro verdadeiro do operador ótimo 3×3 é maior que (4,723-0,247)%. Conseqüentemente, com confiança de pelo menos 99%, o erro verdadeiro de $\hat{\Psi}$ é no máximo 0,797% maior que o erro verdadeiro do operador ótimo, isto é:

$$\text{t\_error}_P(\hat{\psi}) - \text{t\_error}_P(\psi^*) \leq 0{,}00797 .$$

Este resultado confirma que a equação (2.2) superestima a taxa de erro, pois 0,797% é muito menor que 6,71%. ■

Viés indutivo ek-NN

Nas subsubseções anteriores, tínhamos suposto que o aprendiz era e-ótimo (ou consistente) para calcular a complexidade de amostra. Porém a e-otimalidade sozinha não especifica inteiramente um algoritmo de aprendizagem, pois existem muitos diferentes aprendizes e-ótimos. Para especificar completamente um aprendiz, um método de generalização (viés indutivo) também deve ser escolhido.

Para a aprendizagem de W-operador, sugerimos que se utilize a generalização *k*-NN [Mitchell, 1997], pois nos parece bastante natural que padrões semelhantes sejam classificados similarmente. Uma outra possibilidade seria utilizar a generalização dada pela árvore de decisão [Mitchell, 1997], pois se aproxima muito da generalização *k*-NN. Evidentemente, ao se escolher um viés indutivo, deve-se levar em conta a existência de algoritmos computacionalmente eficientes que consigam implementá-lo. Também se deve tomar cuidado para que a generalização mantenha a e-otimalidade pois, caso contrário, a teoria PAC se tornará inválida.

Para ilustrar o perigo da não e-otimalidade, considere a aprendizagem *k*-NN. O seu viés indutivo corresponde à suposição de que a classificação de uma instância será mais parecida à classificação de outras instâncias que estão próximas em distância. No algoritmo *k*-NN ingênuo, o treinamento consiste simplesmente em armazenar os dados de treinamento apresentados. De acordo com a regra *k*-NN, para cada padrão de busca $q_{p_i}^{\mathbf{x}}$, os *k* padrões exemplos de entrada "mais parecidos" devem ser procurados em $\vec{a}$. Como estamos lidando com imagens binárias, as distâncias entre $q_{p_i}^{\mathbf{x}}$ e os padrões de treinamento devem ser medidas utilizando a distância de Hamming (isto é, o número de bits discordantes) ou a distância de Hamming com pesos. No último caso, pode-se dar mais peso a alguns furos de espiar (por exemplo, os furos centrais) do que a outros (por exemplo, os furos periféricos). A figura 2.4 mostra duas janelas sem e com pesos. A saída é definida como a classificação mais comum entre os *k* exemplos de treinamento mais próximos. Claramente, esta regra *k*-NN original não é e-ótima. Porém, mudando-a ligeiramente como segue, ela torna-se e-ótima:

1) Se o padrão a-ser-processado $q_{p_i}^{\mathbf{x}}$ aparecer uma ou mais vezes em $\vec{a}$, a sua classificação será dada pela maioria dos votos somente dessas instâncias de treinamento. Isto é, sejam $a_{r_1}^{\mathbf{x}}$, ..., $a_{r_N}^{\mathbf{x}}$ as instâncias de treinamento tais que $a_{r_j}^{\mathbf{x}} = q_{p_i}^{\mathbf{x}}, 1 \le j \le N$. Então, faça $\hat{q}_{p_i}^{\mathbf{y}} \leftarrow \underset{1 \le j \le N}{\text{moda}}\left(A^{\mathbf{y}}(r_j)\right)$. Neste caso, *N* pode ser maior, igual ou menor que *k*.

2) Por outro lado, se o padrão a-ser-processado $q_{p_i}^{\mathbf{x}}$ nunca foi visto antes, procure pelas suas *k* instâncias mais semelhantes em $\vec{a}$ e escolha o voto majoritário delas. Isto é, sejam $a_{r_1}^{\mathbf{x}}$, ..., $a_{r_N}^{\mathbf{x}}$ as *N* instâncias mais semelhantes à $q_{p_i}^{\mathbf{x}}$, de acordo com alguma medida de distância. Então, novamente faça $\hat{q}_{p_i}^{\mathbf{y}} \leftarrow \underset{1 \le j \le N}{\text{moda}}\left(A^{\mathbf{y}}(r_j)\right)$. Neste caso, *N* pode ser igual ou maior que *k* (se houver empate), mas nunca pode ser menor que *k*.

Chamamos esta regra modificada de aprendizagem *k* vizinhos mais próximos empiricamente ótima (abreviado como ek-NN). A aprendizagem ek-NN parece ser muito apropriada para ser usada na aprendizagem de W-operadores. Porém, para ser realmente útil, deveriam existir estruturas de dados e algoritmos que permitam uma implementação eficiente. As implementações possíveis são as mesmas da aprendizagem *k*-NN, já vistas na introdução: a força-bruta, a LUT e a kd-árvore. A força-bruta é muito lenta. A LUT é extremamente rápida na aplicação, porém a sua velocidade de treinamento e a memória gasta crescem exponencialmente com o aumento da janela. A kd-árvore pode ser treinada rapidamente e a memória gasta é razoável, porém o seu tempo de busca torna-se proibitivo nas dimensões altas.

Aprendizagem por árvore de decisão

A aprendizagem ek-NN vista na subsubseção anterior não pode ser usada para projetar W-operadores definidos em janelas amplas, pois não existem algoritmos e estruturas de dados eficientes. Assim, somos forçados a buscar alternativas. Vamos examinar a aprendizagem por árvore de decisão (DT) [Mitchell, 1997]. Ela é uma das técnicas mais amplamente utilizadas para aproximar funções alvos discretos. A função aprendida é representada como uma árvore (no nosso problema, uma árvore binária). Na realidade, a árvore de decisão é muito similar à kd-árvore usada na aprendizagem *k*-NN. A diferença principal está no estágio de busca: não existe um processo de backtracking. Isto torna a busca muito rápida, na prática milhões de vezes mais rápida que a kd-árvore, superando a deficiência que torna impossível o uso da kd-árvore em aprendizagem de W-operador com janela grande. A eliminação de backtracking também elimina a necessidade de armazenar padrões de entrada nas folhas, diminuindo o uso de memória.

A aprendizagem DT é e-ótima. Esta propriedade fixa os valores de saída para todos padrões de busca que aparecem pelo menos uma vez na seqüência de treinamento. Por outro lado, se o aprendiz nunca viu o padrão de busca, o valor de saída é escolhido de acordo com o viés indutivo de aprendizagem DT: prefira as árvores que colocam atributos com alto ganho de informação mais próximos à raiz sobre aqueles que

não fazem isso. Este costume torna o comportamento da aprendizagem DT bastante similar ao da aprendizagem ek-NN. Também aproxima o viés indutivo conhecido como a "navalha de Occam": prefira a hipótese mais simples que explica os dados observados.

Para explicar a construção de uma árvore de decisão, sejam dados *n* padrões amostras de entrada com as correspondentes cores de saída:

$$\vec{a} = \left((a^{\mathbf{x}}_{p_1}, A^{\mathbf{y}}(p_1)),\ldots,(a^{\mathbf{x}}_{p_m}, A^{\mathbf{y}}(p_m))\right),\ a^{\mathbf{x}}_{p_i} \in \{0,1\}^w \text{ e } A^{\mathbf{y}}(p_i) \in \{0,1\}.$$

No processo de geração da árvore DT, um atributo de corte $s \in [1...w]$ é escolhido e o espaço de padrões $\{0,1\}^w$ é cortado em duas metades. Todas as amostras com atributo *s* preto irão pertencer a um semi-espaço e aquelas com branco ao outro. Em cada corte, um nó interno é criado e o atributo de corte *s* armazenado nele.

Para obter uma árvore otimizada, em cada estágio de corte, o atributo *s* deve ser escolhido de forma que o ganho de informação seja maximizado. Assim, em cada corte, os ganhos de informação de todos os atributos são calculados e o atributo com o maior ganho é escolhido como o atributo de corte. O ganho de informação é a redução de entropia esperada causada ao particionar os exemplos de acordo com o atributo *s*:

$$\text{Gain}(\vec{a}, s) = \text{Entropy}(\vec{a}) - \left(\frac{b}{m}\text{Entropy}(\vec{a}_{v_s=0}) + \frac{m-b}{m}\text{Entropy}(\vec{a}_{v_s=1})\right)$$

onde $\vec{a}_{v_s=0}$ ($\vec{a}_{v_s=1}$) é a subseqüência de $\vec{a}$ com todas as amostras cujo valor no atributo *s* é preto (branco). Utilizamos a notação v_s para denotar o valor do atributo *s*. A entropia de uma seqüência de amostra $\vec{a}$ com *b* saídas pretas (e conseqüentemente *m*-*b* saídas brancas) é:

$$\text{Entropy}(\vec{a}) \quad = \quad -\left(\frac{b}{m}\right)\log_2\left(\frac{b}{m}\right) - \left(\frac{m-b}{m}\right)\log_2\left(\frac{m-b}{m}\right).$$

Para cada um dos dois semi-espaços obtidos, o processo de corte continua recursivamente, gerando subespaços cada vez menores. Este processo pára quando cada subespaço contiver ou somente amostra com a mesma cor de saída ou somente amostras

com o mesmo padrão de entrada (mas com duas diferentes cores de saída). No primeiro caso, um nó terminal é criado e a cor de saída é armazenada nele. No segundo caso, um nó terminal também é criado e, para assegurar a e-otimalidade, a moda das cores de saída é avaliada e armazenada.

A árvore de decisão construída representa a função característica $\hat{\psi}$. Dado um padrão de busca $q_{p_i}^{\mathbf{x}}$, a sua cor de saída $\hat{Q}^{\mathbf{y}}(p_i) = \hat{\psi}(q_{p_i}^{\mathbf{x}})$ é calculada executando uma busca na árvore. A busca começa no nó raiz. Em cada nó interno, a direção a seguir (esquerda ou direita) é escolhida de acordo com o valor do padrão de busca no atributo de corte *s*. O processo é repetido até chegar a um nó terminal. O valor da função característica $\hat{\psi}$ é a cor de saída armazenada no nó terminal.

Dadas *m* amostras e *n* pontos de busca no espaço de padrões de dimensão *w*, pode ser mostrado que a árvore de decisão pode ser construída em tempo médio $O(wm\log m)$. A aplicação leva $O(n\log m)$ e a complexidade de uso de memória é $O(m)$. Esta análise mostra que tanto a construção quanto a busca são extremamente rápidas, enquanto a memória é utilizada economicamente mesmo em dimensões altas.

Comparação dos diferentes vieses indutivos

Freqüentemente, estamos interessados em comparar o desempenho de dois algoritmos de aprendizagem $\mathbf{A}_1$ e $\mathbf{A}_2$ em vez de duas hipóteses específicas. Por exemplo, podemos querer determinar se o viés indutivo de ek-NN é mais efetivo que os outros. Em outras palavras, gostaríamos estimar a diferença esperada entre as taxas de erros verdadeiros:

$$E\left[\text{t\_error}_P(\mathbf{A}_1(\vec{a})) - \text{t\_error}_P(\mathbf{A}_2(\vec{a}))\right] = \sum_{\vec{a}\in(\{0,1\}^w\times\{0,1\})^m}\left[\text{t\_error}_P(\mathbf{A}_1(\vec{a})) - \text{t\_error}_P(\mathbf{A}_2(\vec{a}))\right]P^m(\vec{a})\cdot$$

Para estabelecer um intervalo de confiança para a quantidade acima, os dois aprendizes $\mathbf{A}_1$ e $\mathbf{A}_2$ devem ser treinados utilizando *K* seqüências de treinamento independen-

tes $\vec{a}_i$, $1 \le i \le K$, e as hipóteses resultantes aplicadas a *K* diferentes seqüências de teste $\vec{q}_i$, $1 \le i \le K$. Este processo irá gerar *K* diferenças entre os e-erros de $\mathbf{A}_1$ e $\mathbf{A}_2$:

$$\delta_i = \text{e\_error}_{\vec{q}_i}(\mathbf{A}_1(\vec{a}_i)) - \text{e\_error}_{\vec{q}_i}(\mathbf{A}_2(\vec{a}_i)),\ 1 \le i \le K.$$

Intervalos de confiança unilateral ou bilateral podem ser construídos a partir de δ_1, ..., δ_K utilizando a distribuição *t* de Student. Com confiança *N*%:

$$\underset{\vec{a} \in P^m}{E}\left[\text{t\_error}_P(\mathbf{A}_1(\vec{a})) - \text{t\_error}_P(\mathbf{A}_2(\vec{a}))\right] \in \overline{\delta} \pm t_{N,k-1} s_{\overline{\delta}} \tag{2.7}$$

$$\underset{\vec{a} \in P^m}{E}\left[\text{t\_error}_P(\mathbf{A}_1(\vec{a})) - \text{t\_error}_P(\mathbf{A}_2(\vec{a}))\right] > \overline{\delta} - t'_{N,k-1} s_{\overline{\delta}} \tag{2.8}$$

onde:

- $s_{\overline{\delta}} \equiv \sqrt{\dfrac{1}{K(K-1)} \sum_{i=1}^{K} (\delta_i - \overline{\delta})^2}$;
- $\overline{\delta} \equiv \left(\delta_1 + \cdots + \delta_K\right) / K$;
- $t_{N,K-1}$ define a meia largura do menor intervalo em torno da média que inclui *N*% da massa de probabilidade total sob a distribuição *t* normalizada com (*K*-1) graus de liberdade; e $t'_{N,K-1} \equiv t_{2N-1,K-1}$.

Por exemplo, $t'_{95\%,2} = t_{90\%,2} = 2{,}92$, $t'_{97.5\%,2} = t_{95\%,2} = 4{,}30$ e $t'_{99\%,2} = t_{98\%,2} = 6{,}96$.

2.3 Aumento de Resolução de Imagens Binárias

Introdução

Esta seção descreve uma contribuição científica original minha. Os resultados descritos nesta seção estão documentados em artigos [Ri03; Ci02; Cn10].

Nesta seção, usaremos a teoria desenvolvida na seção anterior para aumentar a resolução de imagens binárias de documentos impressos ou manuscritos. Num ambiente de escritório típico, as imagens digitais e os documentos são manipulados por um conjunto de equipamentos e softwares não-homogêneos que formam um sistema capaz de escanear, editar, mostrar, imprimir, transmitir, efetuar OCR, e executar várias outras tarefas de Processamento e Análise de Imagens. Como cada componente do sistema pode operar numa resolução espacial diferente, freqüentemente aparece a necessidade da conversão de resolução, para permitir que as imagens e os documentos digitais migrem de um componente do sistema a outro.

A diminuição da resolução espacial é uma tarefa relativamente fácil. Em contraste, o aumento da resolução espacial (ou ampliação ou zoom) é difícil, pois a imagem de entrada normalmente não contém toda a informação necessária para gerar uma imagem de saída perfeitamente ampliada. Além disso, a ampliação ideal depende do "contexto" da aplicação. Por exemplo, o operador ótimo, projetado para ampliar duas vezes os caracteres "Times, 12 pt., 300 dpi", pode não ser ótimo para uma outra fonte ou um documento manuscrito.

Muitos algoritmos de ampliação de imagens foram desenvolvidos para imagens em níveis de cinza e coloridas. Porém, parece que a ampliação de imagens binárias tem recebido muito menos atenção até agora. Loce et al. [Loce and Dougherty, 1997; Loce et al., 1997] apresentam algumas técnicas, entre o pequeno número publicadas na literatura, para a ampliação de imagens binárias. Isto causa certa surpresa, pois a ampliação de imagens binárias é muitas vezes necessária na prática. Por exemplo, con-

sidere o número de vezes em que uma imagem em 300 dpi teve que ser impressa numa impressora 600 dpi.

Muitas tarefas de processamento de imagens estão baseadas em operadores restritos à janela (W-operadores). Usaremos a aprendizagem *k*-NN (*k* vizinhos mais próximos) para projetar os operadores de ampliação restritos à janela (WZ-operadores, Z de zoom).

Loce et al. [Loce and Dougherty, 1997; Loce et al., 1997] em essência expõem duas técnicas para ampliar as imagens binárias. Eles usam filtros não-crescentes e crescentes (utilizamos as palavras “filtro” e “operador” como sinônimos). Nesta seção, propomos algumas melhorias sobre essas técnicas anteriores.

Primeiro, como a distribuição de probabilidade verdadeira que governa o processo de ampliação é normalmente desconhecida, na prática as estatísticas derivadas das imagens amostras de entrada-saída devem ser utilizadas no seu lugar. Conseqüentemente, o melhor operador que alguém pode obter na prática é o operador que é ótimo sobre as imagens de treinamento (não levando em conta o viés indutivo). Chamamos isto de operador empiricamente ótimo (e-ótimo). A técnica de filtro crescente [Loce and Dougherty, 1997, chap. 9; Loce et al., 1997] pode gerar uma solução sub-ótima, enquanto que a abordagem [Loce and Dougherty, 1997, chap. 6] e a nossa sempre projetam um operador e-ótimo. Além disso, os trabalhos anteriores não analisam a diferença entre os operadores empiricamente ótimo e verdadeiramente ótimo, assumindo implicitamente que as estatísticas derivadas das imagens amostras são uma aproximação próxima da verdadeira distribuição de probabilidade. Propomos usar as técnicas estatísticas para estimar a diferença entre as duas taxas de erro. Além disso, as técnicas prévias não adotam qualquer viés indutivo explícito. O viés indutivo é o conjunto de suposições *a priori* pelo qual o aprendiz generaliza além dos dados de treinamento observados, para inferir as classificações das novas instâncias. Um aprendiz que não assume nenhuma suposição *a priori* quanto à identidade do conceito alvo não possui nenhuma base racional para classificar qualquer instância ainda não vista. Adotamos a aprendizagem *k*-NN porque ela possui um viés indutivo sólido,

exaustivamente testado em muitas aplicações diferentes. Mostramos experimentalmente a sua eficácia para o problema em questão.

Em segundo lugar, os trabalhos anteriores parecem necessitar de certa intervenção humana no projeto de operador. A nossa técnica é, ao contrário, totalmente automática.

Em terceiro lugar, a técnica de filtro crescente é uma tentativa de melhorar a técnica não-crescente, visando a implementação em hardware: ela está focalizada em projetar um operador "logicamente eficiente", isto é, um operador representado utilizando um número pequeno de portas lógicas. A nossa abordagem está focalizada em implementação por software, onde a redução de lógica perde a sua atratividade porque uma lógica mais simples não necessariamente significa uma técnica mais rápida. Em seu lugar, o uso de algoritmos e estruturas de dados apropriados pode levar a métodos mais rápidos, reduzindo a complexidade computacional. Para acelerar a aplicação de W-operadores, Jones e Svalbe [Jones and Svalbe, 1994] usam look-up-table (LUT), Kim et al. [Ri01; Ci01; Cn06] usam uma estrutura de dados em forma de árvore, e Robert e Malandain [Robert and Malandain, 1998] usam diagrama de decisão binária. A LUT é extremamente rápida no estágio de aplicação e permite a implementação da aprendizagem *k*-NN exata, mas a sua demanda pela memória e tempo de treinamento cresce exponencialmente à medida que a janela cresce. A estrutura de árvore requer somente uma quantidade moderada de memória e tempo de treinamento, mas o seu tempo de aplicação é bem maior que LUT (quando ela implementa a aprendizagem *k*-NN exata usando uma estrutura de dado conhecida como kd-árvore, que requer um processo de back-tracking) ou ligeiramente maior que LUT (quando ela implementa uma árvore de decisão, uma estratégia de aprendizagem de máquina bastante semelhante à aprendizagem *k*-NN). O diagrama de decisão binária é tão rápido quanto a estrutura de árvore em aplicação e usa menos memória, mas o seu processo de treinamento é muito lento. Adotamos a solução LUT, pois os resultados experimentais mostraram que a ampliação de documentos impressos e manuscritos não necessita de janelas grandes.

Projeto de WZ-operador pela aprendizagem k-NN

Vamos definir o operador de aumento de resolução restrito à janela (WZ-operador, Z de zoom). Um WZ-operador Ψ é definido através da janela W e f^2 funções características ψ_0, ..., ψ_{f^2-1}, onde f é o fator de zoom. Trabalharemos somente com aumentos de resolução por fatores inteiros f. Além disso, para simplificar a notação, assumiremos que os fatores de aumento de linha e coluna são iguais. Por exemplo, $f=2$ aumenta a resolução espacial duas vezes em cada coordenada. Cada função característica é uma função booleana $\psi_i : \{0,1\}^w \to \{0,1\}$ e referiremos ao conjunto de f^2 funções ψ_i como ψ ($\psi : \{0,1\}^w \to \{0,1\}^{(f^2)}$). As funções ψ_i convertem um pixel de entrada p em f^2 pixels de saída y_i baseado no conteúdo da janela W deslocada para p, isto é, para $0 \le i < f^2$ (figura 2.5):

$$y_i = \Psi(Q)(f\ p + d_i) = \psi_i(Q(W_1 + p), \ldots, Q(W_w + p)),$$

onde $p \in \mathbb{Z}^2$ e d_i é o vetor de deslocamento associado à i-ésima função característica. Na figura 2.5, as funções características ψ_0, ..., ψ_3 convertem o pixel p em pixels y_0, ..., y_3 baseado no conteúdo da janela 3×3.

Para poder aplicar um WZ-operador a uma imagem $Q^{\mathbf{x}}$, a regra ek-NN deve ser aplicada a cada padrão a-ser-ampliado $q^{\mathbf{x}}_{p_i}$ de $Q^{\mathbf{x}}$. Infelizmente, este processo é excessivamente lento: para aumentar a resolução de cada pixel, a imagem amostra $A^{\mathbf{x}}$ inteira deve ser analisada. As nossas experiências mostram que este algoritmo ingênuo leva meses ou mesmo anos para aumentar a resolução de uma única imagem, utilizando um computador convencional.

Utilizamos look-up-table (LUT) para acelerar este processo. A LUT permite implementar a aprendizagem ek-NN e é extremamente rápida no tempo de avaliação, o que a torna adequada para aplicações de tempo real. Porém, a sua demanda pela memória e tempo de treinamento aumentam exponencialmente à medida que o tamanho da janela cresce. Isto torna impossível o seu uso para janelas grandes. Felizmente, as experiências mostraram que as janelas pequenas com 3×3, 4×4 ou 17

furos-de-espiar (figura 2.4) podem gerar bons WZ-operadores para aumentar a resolução de documentos impressos ou manuscritos.

Evidentemente, una função booleana $\psi:\{0,1\}^w \rightarrow \{0,1\}$ pode ser representada como uma LUT com 2^w linhas, numeradas de 0 a 2^w-1, onde cada célula é ou 0 ou 1. Portanto, uma tabela para representar f^2 funções deve ter 2^w linhas e f^2 colunas, ocupando $2^w f^2$ bits. Por exemplo, usando uma janela 4×4 e fator de zoom f=3, a tabela irá ocupar 589824 bits ou 73728 bytes. Cada coluna irá representar uma função característica ou hipótese $\hat{\psi}_i$.

O processo de aprendizagem *k*-NN deve preencher a LUT. O índice *l* de cada linha representa um padrão binário $q_l^{\mathbf{x}}$, de comprimento *w*. Para cada $q_l^{\mathbf{x}}$, a regra ek-NN deve ser aplicada. Este processo pode ser acelerado substancialmente criando um vetor onde cada padrão de entrada de $A^{\mathbf{x}}$ aparece uma única vez, junto com o número de votos para as saídas branca e preta. Depois, a busca é executada neste vetor, em vez de em $A^{\mathbf{x}}$. Note que o vetor de padrões não repetidos pode ser criado rapidamente utilizando qualquer algoritmo de ordenação $O(m \log m)$, como quicksort ou heapsort [Cormen et al., 1990], seguido por um algoritmo $O(m)$ para eliminar os padrões repetidos.

Depois que a LUT esteja completamente preenchida, dado um padrão a-ser-ampliado, os pixels de saída y_i podem ser calculados sem esforço simplesmente indexando a linha correspondente da LUT.

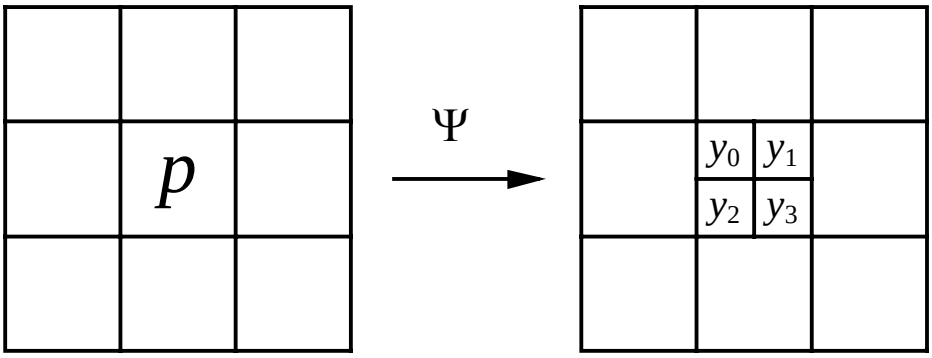


Fig. 2.5: Operador de aumento de resolução restrito à janela 3×3 (WZ-operador) com fator de zoom f =2.

	janela	ψ_1	ψ_2	ψ_3	ψ_4	média
$e\_error_{\vec{q}}(\hat{\psi}_i)$	3×3	1,10%	1,03%	1,05%	1,05%	1,058%
$e\_error_{\vec{q}}(\hat{\psi}_i^*)$, e-ótima sobre $\vec{q}$	3×3	1,09%	1,02%	1,04%	1,03%	1,045%
$e\_error_{\vec{q}}(\hat{\psi}_i)$	17	1,01%	0,95%	1,00%	1,02%	0,995%
$e\_error_{\vec{q}}(\hat{\psi}_i^*)$, e-ótima sobre $\vec{q}$	17	0,95%	0,89%	0,92%	0,92%	0,920%

Tab. 2.1: Erros empíricos obtidos usando a regra e1-NN ao aumentar a resolução de documento impresso.

Aumento de resolução de caracteres impressos

Para testar as idéias expostas acima, projetamos um WZ-operador 3×3 para aumentar a resolução de documentos contendo caracteres “Times 12 pt.” (tanto normal como itálico) de 300 dpi para 600 dpi. As imagens de treinamento foram obtidas imprimindo os documentos eletrônicos para arquivos “.PS” através de um *driver* de uma impressora *PostScript*, e então convertendo esses arquivos para as imagens binárias. Embora as imagens estejam sem ruído, o problema deve ser considerado ruidoso, pois um único padrão 3×3 em 300 dpi pode corresponder a dois ou mais padrões diferentes em 600 dpi.

Vamos utilizar a equação 2.2 para estimar o tamanho necessário das imagens de treinamento para, usando a janela 3×3, obter um WZ-operador $\hat{\Psi}$ com uma taxa de erro no máximo 2% maior que o operador ótimo. Usando nível de confiança 99%:

$$m \geq \frac{1}{2\varepsilon^2}\left[\ln\left(\frac{1}{\delta}\right)+\ln\left(2|H|\right)\right]=\frac{1}{2\times 0{,}02^2}\left[\ln\left(\frac{1}{0{,}01}\right)+\ln(2)+2^9\times\ln(2)\right]\cong 450238\,.$$

Temos dois pares de imagens de amostra independentes ($A^{\mathbf{x}}$, $A^{\mathbf{y}}$) e ($Q^{\mathbf{x}}$, $Q^{\mathbf{y}}$) com caracteres Times 12 pt. (figura 2.6) cujos tamanhos são (554×813, 1108×1626) e (558×740, 1116×1480), respectivamente. Note que a imagem $A^{\mathbf{x}}$ é grande o suficiente para obter a acurácia desejada, pois 554×813=450402. Um WZ-operador foi construído utilizando a aprendizagem 1-NN. O treinamento levou 5s e a aplicação menos que 1s num Pentium 300MHz.

A imagem processada $\hat{Q}^{\mathbf{y}}$ (figura 2.6c) e a imagem ideal $Q^{\mathbf{y}}$ (figura 2.6b) diferiam em 1,058% dos pixels e eles são visualmente bastante semelhantes. Note que na realidade 4 funções características independentes foram projetadas e os seus e-erros individuais estão descritos na primeira linha da tabela 2.1. Uma vez que o e-erro foi medido, pode surgir a seguinte pergunta: “É possível aumentar substancialmente a acurácia do operador projetado?” Utilizaremos as desigualdades (2.5) e (2.6) para

mostrar que é impossível obter qualquer melhora substancial na qualidade do WZ-operador, enquanto janela 3×3 estiver sendo utilizada. Mostraremos que:

1) O e-erro obtido é uma boa estimativa do erro verdadeiro de $\hat{\Psi}$.

2) O erro verdadeiro do operador 3×3 ótimo está muito próximo ao do $\hat{\Psi}$.

Usando equação 2.5, com confiança 99%:

$$\text{t\_error}_P(\hat{\psi}) \leq 0{,}01058 + 2{,}33\sqrt{\frac{0{,}01058(1-0{,}01058)}{1116\times1480}} = (1{,}058 + 0{,}019)\%,$$

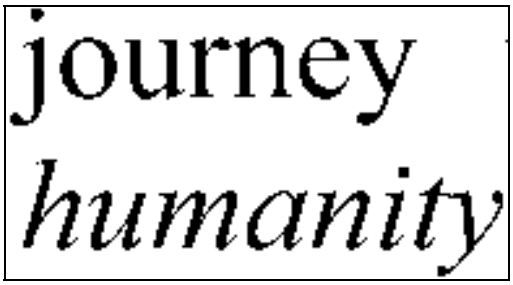
o que demonstra a primeira afirmação.

Para demonstrar a segunda afirmação, projetamos WZ-operador $\hat{\Psi}$* e-ótimo sobre ($Q^{\mathbf{x}}$, $Q^{\mathbf{y}}$) e o aplicamos na imagem $Q^{\mathbf{x}}$. Os e-erros obtidos estão mostrados na segunda linha da tabela 2.1. Usando os dados obtidos e a equação 2.6, concluímos com confiança 99% que:

$$\text{t\_error}_P(\psi^*) \geq 0{,}01045 - 2{,}33\sqrt{\frac{0{,}01045(1-0{,}01045)}{1116\times1480}} = (1{,}045 - 0{,}018)\%.$$

Isto mostra claramente que não pode existir qualquer WZ-operador 3×3 substancialmente melhor que $\hat{\Psi}$ pois, com probabilidade 99%, o erro verdadeiro de $\hat{\Psi}$ é no máximo 1,077% enquanto que com a mesma probabilidade o erro verdadeiro do WZ-operador 3×3 ótimo é pelo menos 1,027%.

Uma vez que demonstramos que o WZ-operador obtido é virtualmente o melhor WZ-operador 3×3, uma outra questão pode surgir: “Poderia melhorar a qualidade do operador escolhendo uma janela maior”? Repetimos os testes utilizando a janela com 17 furos sem peso (figura 2.4). A terceira linha da tabela 2.1 mostra os e-erros obtidos. A qualidade de WZ-operador melhorou somente ligeiramente. Além disso, a linha 4 mostra que, mesmo usando uma janela com 17 furos, o erro mínimo não pode ser substancialmente menor que 0,92%. Desta vez, o treino levou 148s mas a aplicação ainda levou menos de 1s.



(2.6a) Imagem original $Q^{\mathbf{x}}$ em 300 dpi.

(2.6b) Saída ideal $Q^{\mathbf{y}}$ em 600 dpi.

(2.6c) Imagem 600 dpi $\hat{Q}^{\mathbf{y}}$ gerada pela aprendizagem.

Fig. 2.6: Aumento de resolução de caracteres impressos (Times, 12 pt.) usando WZ-operadores projetados pela aprendizagem 1-NN.

	Janela	teste 1	teste 2	teste 3	média
1. Viés aleatório	17	1,638%	1,680%	1,622%	1,647%
2. e1-NN	17 sem peso	1,218%	1,238%	1,174%	1,210%
3. e5-NN	17 sem peso	1,208%	1,234%	1,166%	1,203%
4. e10-NN	17 sem peso	1,206%	1,236%	1,168%	1,203%
5. e20-NN	17 sem peso	1,212%	1,241%	1,166%	1,206%
6. e40-NN	17 sem peso	1,218%	1,244%	1,168%	1,210%
7. e1-NN	17 com peso	1,191%	1,206%	1,143%	1,180%
8. e5-NN	17 com peso	1,180%	1,202%	1,130%	1,171%
9. e10-NN	17 com peso	1,178%	1,199%	1,129%	1,169%
10. e20-NN	17 com peso	1,180%	1,200%	1,124%	1,168%
11. e40-NN	17 com peso	1,184%	1,201%	1,130%	1,172%
12. $\text{e\_error}_{\bar{q}}(\hat{\psi}_i^*)$	17	0,920%	1,012%	0,922%	0,952%
13. Replicação de pixels	-	1,540%	1,670%	1,580%	1,597%

Tab. 2.2: Erros empíricos usados para comparar os diferentes vieses indutivos.

Avaliação do viés indutivo ek-NN

Nesta subsubseção, iremos testar se o viés indutivo da aprendizagem ek-NN é efetivo no aumento da resolução. Para esta finalidade, as diferentes aprendizagens ek-NN foram comparadas com o aprendiz e-ótimo com viés indutivo aleatório. Um aprendiz com viés indutivo aleatório classifica aleatoriamente qualquer padrão não visto. Para tornar evidente as diferenças dos vieses indutivos, pequenas imagens de treinamento (116×516, 232×1032) foram usadas. Por outro lado, as imagens de teste ($Q^{\mathbf{x}}$, $Q^{\mathbf{y}}$) foram razoavelmente grandes (740×558, 1480×1116) para obter estimativas acuradas dos erros verdadeiros. Os testes foram repetidos 3 vezes, cada vez utilizando um conjunto de imagens completamente independente.

Os resultados estão listados na tabela 2.2. A primeira linha apresenta taxas de erro do aprendiz e-ótimo com o viés aleatório. As linhas 2-6 apresentam e-erros das aprendizagens ek-NN para diferentes valores de *k* usando uma janela com 17 furos sem peso, e linhas 7-11 usando janela com 17 furos com peso. A linha 12 é o e-erro do operador e-ótimo sobre as imagens de teste. Finalmente, como mera curiosidade, a linha 13 mostra os e-erros obtidos pela simples replicação de cada pixel quatro vezes.

Para mostrar a eficácia do viés indutivo ek-NN, vamos comparar o seu viés aleatório (linha 1) com o de e1-NN sem peso (linha 2). Note que e1-NN sem peso apresenta o maior taxa de e-erro entre os ek-NN's. A diferença média entre os dois aprendizes foi $\overline{\delta} = 0{,}437\%$. Esta diferença é significativa estatisticamente? Para responder a esta questão, vamos construir um intervalo de confiança. Usando a equação 2.8, com confiança 95%:

$$\underset{\vec{a}\in P^m}{E}\left[\text{t\_error}_P(\mathbf{A}_1(\vec{a})) - \text{t\_error}_P(\mathbf{A}_2(\vec{a}))\right] > (0{,}437 - 0{,}025)\%\,.$$

Isto mostra claramente que o viés indutivo ek-NN ajuda a diminuir a taxa de erro.

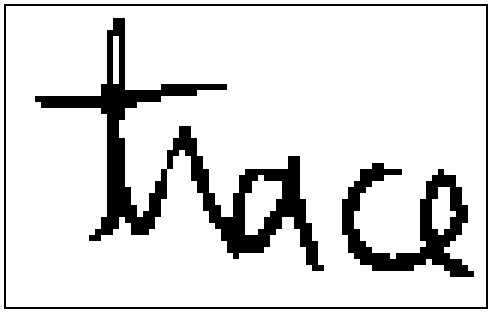
De acordo com a tabela 2.2, parece que as janelas com peso geram menos erros que as janelas sem peso e que o erro torna-se mínimo para $k \cong 10$. Porém, como essas

diferenças são muito pequenas, mais testes são necessários para validar essas suposições.

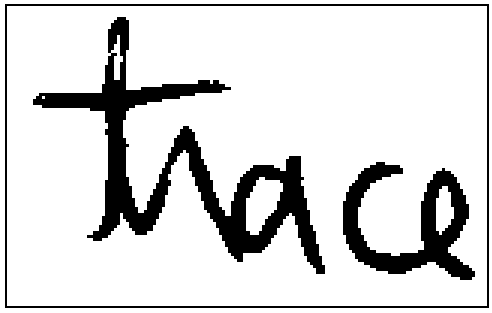
Documentos manuscritos

A técnica acima também foi aplicada para documentos manuscritos (figura 2.7). Os tamanhos das imagens treinamentos ($A^{\mathbf{x}}$, $A^{\mathbf{y}}$) e imagens de teste ($Q^{\mathbf{x}}$, $Q^{\mathbf{y}}$) foram (672×848, 1344×1696). O treino levou 9s usando a janela 3×3 sem peso, enquanto a aplicação levou menos de 1s. A imagem 2.7a é o documento original $Q^{\mathbf{x}}$, 2.7b é a saída ideal $Q^{\mathbf{y}}$ e 2.7c é a imagem processada $\hat{Q}^{\mathbf{y}}$. A diferença entre as imagens $Q^{\mathbf{y}}$ e $\hat{Q}^{\mathbf{y}}$ é 1,14%. O operador 3×3 $\hat{\Psi}^{*}$, e-ótimo sobre imagens de teste, apresentou e-erro de 1,13%. Isto mostra claramente que o WZ-operador projetado é virtualmente o melhor.

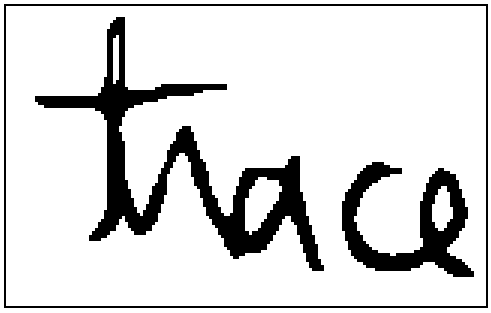
Como uma curiosidade, o WZ-operador 3×3 projetado para aumentar a resolução de documentos impressos foi aplicado em manuscrito 2.7a, gerando a figura 2.7d. O erro foi 1,56%. Isto mostra que o WZ-operador projetado para aumentar a resolução de caracteres impressos não é adequado para aumentar a resolução de documentos manuscritos, pois o erro de 1,56% é consideravelmente maior que 1,14%, obtido com o WZ-operador projetado para ampliar as imagens manuscritas. De um modo geral, a aptidão de um WZ-operador depende do contexto da aplicação.



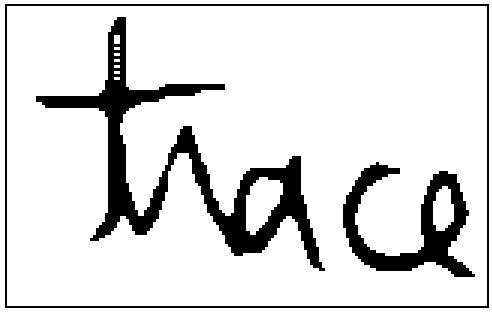
(2.7a) Imagem original $Q^{\mathbf{x}}$.



(2.7b) Imagem de saída ideal $Q^{\mathbf{y}}$ (supostamente desconhecida).



(2.7c) Imagem $\hat{Q}^{\mathbf{y}}$ com resolução aumentada.



(2.7d) Imagem obtida usando operador projetado para aumentar resolução de caracteres impressos.

Fig. 2.7: Aumento de resolução de um documento manuscrito, usando o WZ-operador projetado pela aprendizagem 1-NN.

2.4 Aumento de Resolução de Imagens Meio-Tom

Introdução

Esta seção descreve uma contribuição científica original minha. Os resultados descritos nesta seção estão documentados em artigos [Ri05; Ci05].

A maioria das impressoras jato-de-tinta ou laser atuais na verdade não consegue imprimir as tonalidades de cinza. Elas conseguem imprimir somente pontos minúsculos no papel (dispositivos coloridos não serão considerados aqui). Portanto, qualquer imagem em níveis de cinza deve primeiro ser convertida numa imagem binária por um processo de meio-tom digital antes que a impressão realmente seja efetuada. As técnicas de meio-tom simulam as tonalidades de cinza espalhando quantidades apropriadas de pontos pretos e brancos. Isto é, dada uma imagem em níveis de cinza $G: \mathbb{Z}^2 \to [0,1]$, o meio-tom gera uma imagem binária $B: \mathbb{Z}^2 \to \{0,1\}$ de tal forma que para qualquer pixel *p*:

$$\overline{B}(p) \cong G(p),$$

onde $\overline{B}(p)$ é o valor médio da imagem *B* numa vizinhança em torno do pixel *p*.

Existe uma variedade enorme de técnicas de meio-tom. Os dois mais amplamente conhecidos são a difusão de erro e a excitação ordenada (ordered dithering, abreviada como OD) [Knuth, 1987; Ulichney, 1987]. Existem muitas outras técnicas de meio-tom, por exemplo, a difusão de ponto e as máscaras de ruído azul [Knuth, 1987; Ulichney, 1987]. Algumas delas são projetadas para tecnologias de impressão específicas, para superar as limitações que certas impressoras têm em imprimir pequenos pontos isolados ou os pontos pretos e brancos finamente intercalados.

Muitas tarefas de Processamento e Análise de Imagens são realizadas com operadores restritos à janela (W-operadores). Alguns trabalhos utilizam a abordagem de aprendizagem de máquina para projetar automaticamente um W-operador a partir de imagens amostras de treinamento entrada-saída [Dougherty, 1992a; Dougherty,

1992b; Ri01; Ci01; Cn06; Ri03; Ci02]. Especificamente, propusemos armazenar um W-operador criado pelo processo de aprendizagem de máquina numa estrutura de dados em forma de árvore [Ri01; Cn06]. Aqui, usamos uma idéia similar para projetar o operador de ampliação de imagem (WZ-operador) para aumentar a resolução de imagens binárias meio-tom.

Na literatura, existem muitos artigos sobre o aumento de resolução de imagens em níveis de cinza. Surpreendentemente, somente uns poucos artigos foram escritos sobre a ampliação de imagens binárias [Ri03; Loce and Dougherty, 1997; Loce et al., 1997]. Todas essas técnicas estão baseadas em alguma forma de aprendizagem de máquina e podem ampliar de forma acurada os caracteres impressos ou manuscritos. Além disso, essas técnicas podem ser treinadas para executar algum processamento de imagem simples ao mesmo tempo em que aumenta a resolução. Por exemplo, elas podem atenuar o ruído enquanto aumenta a resolução. Infelizmente, essas técnicas não conseguem levar em conta uma vizinhança ampla para decidir as cores dos pixels com a resolução aumentada, pois os seus tempos de processamento explodem com o aumento dos tamanhos da janela e das imagens exemplos. Uma janela pequena (por exemplo, 3×3 ou 4×4) pode ser bom para ampliar os caracteres impressos ou manuscritos, porém ela não pode ampliar com acurácia as imagens meio-tom. As nossas experiências mostram que janelas do tamanho 8×8 ou 9×9 são necessárias para ampliar com acurácia uma imagem meio-tom.

Esta seção apresenta um algoritmo melhorado para ampliar as imagens binárias baseado em aprendizagem de máquina que permite ampliar de forma acurada até mesmo as imagens meio-tom. A nova técnica está baseada em aprendizagem por árvore de decisão (DT). No conhecimento do autor, esta é a primeira técnica que consegue ampliar direta e com acurácia as imagens meio-tom. A estrutura de dados em forma de árvore permite-nos escrever algoritmos eficientes. A complexidade do tempo de treinamento da nova técnica é somente $O(wm\log m)$, onde w é o tamanho da janela e m é o tamanho da imagem amostra de entrada. A complexidade de aplicação é somente $O(n\log m)$, onde n é o tamanho da imagem a-ser-ampliada. Isto significa que o de-

sempenho deteriora só muito lentamente à medida que os tamanhos da janela e amostras crescem. Esta propriedade torna possível usar as janelas e as imagens amostras grandes. A nova técnica também pode ser usada para ampliar os caracteres impressos ou manuscritos. A nova técnica é incapaz de ampliar com acurácia as imagens geradas pela difusão de erro [Knuth, 1987; Ulichney, 1987], ou por qualquer outro algoritmo de meio-tom onde as cores de saída não são escolhidas como uma função das cores numa vizinhança local. Note que a saída da difusão de erro num pixel particular na realidade depende de todos os pixels previamente processados. Porém, surpreendentemente, a aprendizagem DT pode efetuar o meio-tom inverso acurado das imagens obtidas por difusão de erro, conforme mostramos num artigo recente [Ci11].

O meio-tom inverso é a técnica usada para recuperar a imagem em níveis de cinza a partir de uma imagem binária meio-tom [Wong, 1995; Luo et al., 1998]. O meio-tom inverso simples consiste simplesmente num filtro passa-baixas, por exemplo, um filtro gaussiano. É possível ampliar as imagens meio-tom usando um algoritmo de meio-tom inverso. Porém, a nossa abordagem apresenta uma série de diferenças:

1. Na nossa abordagem, não é necessário ter acesso ao processo de meio-tom em si. É suficiente ter um conjunto de imagens de treinamento entrada-saída. O último é um requerimento mais suave que o primeiro, pois se alguém tiver acesso ao processo de meio-tom, qualquer quantidade de imagens amostras pode ser obtida. O contrário não é verdadeiro.
2. Apesar deste requerimento mais suave, as imagens obtidas pela nova técnica são mais acuradas que aquelas obtidas usando as técnicas de ampliação baseadas em processos de meio-tom inverso simples. Utilizamos o filtro passa-baixas gaussiano e a média local como os processos de meio-tom inverso.
3. Não comparamos o nosso método contra as outras técnicas mais sofisticadas de meio-tom inverso. Porém, demonstramos que a qualidade do nosso processo está bem próxima da melhor qualidade possível de se obter para um processo de ampliação baseada em vizinhança, utilizando a distância de Hamming para quantificar o erro.

Os programas e as imagens usados aqui estão disponíveis em:

http://www.lps.usp.br/~hae/software/halfzoom.

Projeto de WZ-operador pela aprendizagem por árvore de decisão

O operador de ampliação restrito à janela (WZ-operador) foi definido na seção 2.3. Conforme vimos, um WZ-operador pode ser imaginado como um conjunto de f^2 W-operadores (onde f é um fator de ampliação inteiro). O projeto de um WZ-operador é comparável ao projeto de f^2 W-operadores. Assim, um programa computacional que projeta W-operadores pode ser aplicado f^2 vezes para projetar um WZ-operador com fator de ampliação f. Porém, no projeto de WZ-operador, todas as f^2 seqüências de treinamento têm os mesmos padrões de entrada, embora elas normalmente têm diferentes cores de saída. Este fato pode ser explorado para escrever programas mais rápidos e que gastam menos memória, especialmente construídos para o projeto de WZ-operadores.

Para ampliar uma imagem meio-tom usando a aprendizagem DT original, f^2 árvores de decisões independentes devem ser construídas e aplicadas. Isto é uma perda de tempo e de memória computacional. Propomos usar, no projeto de WZ-operadores, uma aprendizagem DT ligeiramente alterada para economizar tempo e espaço, que denominamos de aprendizagem WZDT. A alteração consiste em escolher o atributo de corte $s \in [1...w]$ que torna os dois semi-espaços resultantes a conterem um número tão semelhante quanto possível de pontos de treinamentos (em vez de escolher o atributo que maximiza o ganho de entropia). O novo critério é computacionalmente mais simples que o original. Certamente, o novo teste não é tão bom quanto a maximização do ganho de entropia. Porém, à medida que o tamanho das amostras cresce, os comportamentos das aprendizagens WZDT e DT tornam-se cada vez mais semelhantes. Para amostras grandes, os dois métodos tornam-se inteiramente idênticos (veja os resultados experimentais adiante). Além disso, o novo critério não depende dos valores de saída, enquanto que o critério original depende. Conseqüentemente, usando o

novo critério, todas as f^2 árvores de decisão serão exatamente iguais, exceto pelos seus valores de saída. Assim, uma única árvore de decisão, onde f^2 valores de saída são armazenadas em cada folha, pode representar um WZ-operador. Isto diminui o uso de memória aproximadamente por um favor de f^2. A velocidade também seria melhorada por um fator de f^2. Porém, como o novo critério é computacionalmente mais simples que o original, a aceleração na prática é muito maior que f^2.

Complexidade de amostra e estimação estatística da taxa de erro

Nesta subsubseção, usaremos a teoria de aprendizagem PAC explicada na seção 2.2 para calcular a complexidade de amostra. A acurácia desta complexidade de amostra é então medida utilizando a estimação estatística também explicada na seção 2.2.

Vamos adotar o fator de ampliação $f = 2$ e a janela 4×4. Vamos usar a equação 2.2 para estimar o tamanho necessário *m* da seqüência de treinamento para obter, com nível de confiança 99%, um WZ-operador $\hat{\Psi}$ com o t-erro no máximo 14,5% mais alto que o t-erro do WZ-operador ótimo 4×4 :

$$m \geq \frac{1}{2\varepsilon^2}\left[\ln\left(\frac{1}{\delta}\right) + \ln\left(2|H|\right)\right] = \frac{1}{2\times 0{,}145^2}\left[\ln\left(\frac{1}{0{,}01}\right) + \ln(2) + 2^{16}\times\ln(2)\right] \cong 1{,}1\times 10^6 .$$

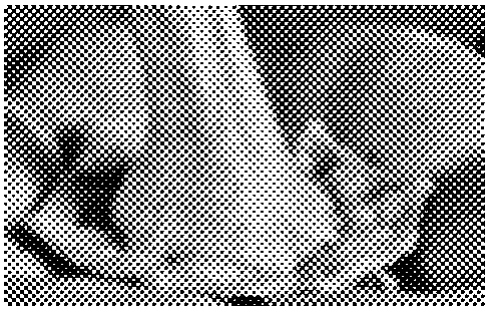
Usamos dois pares de imagens independentes entrada-saída (figuras 2.8a, 2.8b, 2.8c e 2.8d), respectivamente imagens *Peppers* ($A^{\mathbf{x}}$, $A^{\mathbf{y}}$) e *Lena* ($Q^{\mathbf{x}}$, $Q^{\mathbf{y}}$). Elas foram convertidas em imagens meio-tom em 150 e 300 dpi usando o HP LaserJet driver para Microsoft Windows, com a opção “pontos grandes”. $A^{\mathbf{x}}$ e $Q^{\mathbf{x}}$ são 1050×1050, e $A^{\mathbf{y}}$ e $Q^{\mathbf{y}}$ são 2100×2100. Portanto, a imagem $A^{\mathbf{x}}$ é suficientemente grande para produzir a acurácia requerida, pois 1050×1050 ≅ 1,1×10^6. Um WZ-operador $\hat{\Psi}$ foi construído pela aprendizagem WZDT. O treino levou 4s e a aplicação somente 1,2s num Pentium III 1GHz.

Por outro lado, para estabelecer um intervalo estreito para erro, o e-erro de $\hat{\Psi}$ (isto é, a proporção de pixels diferentes entre as imagens 2.8d e 2.8e) foi medido, resultando 7,540%. Usando a equação 2.3, concluímos com 99% de confiança que o t-erro de $\hat{\Psi}$ está contido no intervalo (7,540±0,032)%. Como explicamos anteriormente, o WZ-operador ótimo sobre as imagens de teste pode ser gerado por qualquer aprendiz e-ótimo usando as imagens de teste (Q^x, Q^y) como amostras de treinamento. Assim, usando as figuras 2.8c e 2.8d como amostras de treinamento, o WZ-operador e-ótimo $\hat{\Psi}^*$ foi projetado pela aprendizagem WZDT. Processando a imagem a-ser-ampliada (figura 2.8c) com $\hat{\Psi}^*$, obtivemos a imagem 2.8f. O e-erro desta imagem (a proporção de pixels diferentes entre 2.8d e 2.8f) foi 6,830%. O e-erro do operador e-ótimo $\hat{\Psi}^*$ é uma estimativa do t-erro do WZ-operador verdadeiramente ótimo Ψ^*. Usando a e-quação 2.5, concluímos com 99% de confiança que o t-erro de Ψ^* é pelo menos (6,830-0,028)%. Conseqüentemente, com confiança maior que 99%, o t-erro de $\hat{\Psi}$ é no máximo 0,77% maior que o t-erro do operador verdadeiramente ótimo Ψ^*, isto é:

$$\text{t\_error}_P(\hat{\psi}) - \text{t\_error}_P(\psi^*) \leq 0{,}0077\,.$$

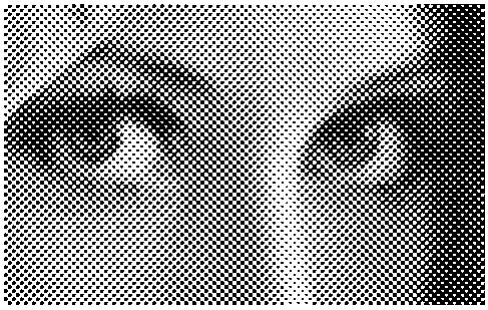
Este resultado confirma que a equação 2.2 produz uma taxa de erro superestimada, pois 0,77% é muito menor que 14,5%. Quanto maior for a janela, a estimativa da complexidade de amostra produzida pela equação 2.2 estará mais inflacionada.

(2.8a) Imagem amostra de entrada $A^{\mathbf{x}}$ em 150 dpi.

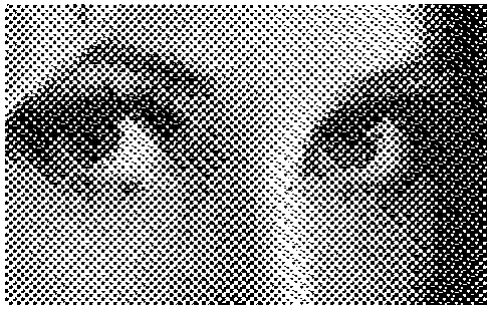


(2.8b) Imagem amostra de saída $A^{\mathbf{y}}$ em 300 dpi.

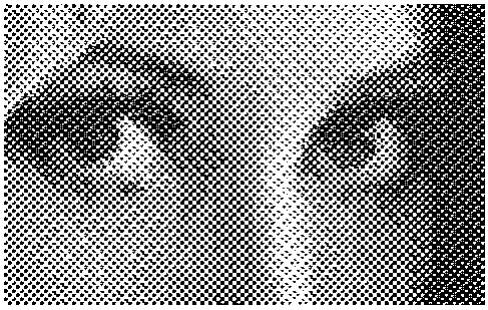
(2.8c) Imagem a-ser-ampliada $Q^{\mathbf{x}}$ em 150 dpi.



(2.8d) Imagem de saída ideal $Q^{\mathbf{y}}$ em 300 dpi.

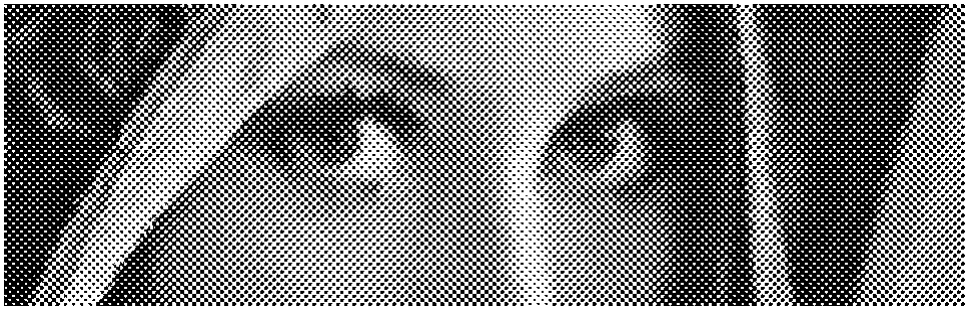


(2.8e) Imagem ampliada $\hat{Q}^{\mathbf{y}}$, usando a janela 4×4. Tamanho das imagens amostras: 1050×1050 e 2100×2100 pixels.



(2.8f) Imagem empiricamente ótima $\hat{\Psi}^{*}(Q^{\mathbf{x}})$, obtido usando a janela 4×4. As imagens de teste (figuras 2.8c e 2.8d) foram usadas como imagens amostras.

Fig. 2.8: Continua na próxima página.



(2.8g) Imagem ampliada $\hat{Q}^{\mathbf{y}}$, usando a janela 8×8. Tamanho das imagens amostras: 9610×1050 e 19220×2100 pixels. A diferença com a saída ideal foi em 1,466% dos pixels.

Fig. 2.8: Aumento de resolução das imagens meio-tom obtidas usando HP LaserJet driver (opção “pontos grandes”) e aprendizagem WZDT.

Escolha de uma janela adequada

Nesta subseção, selecionaremos uma janela apropriada para ampliar as imagens meio-tom na resolução 150 dpi geradas pelo HP LaserJet driver, opção “pontos grandes” (figura 2.8). A janela 4×4 que utilizamos na última subsubseção gerou uma taxa de e-erro excessivamente alta (7,540%). Testamos a aprendizagem WZDT com três conjuntos de imagens completamente independentes usando janelas quadradas de diferentes tamanhos. Os e-erros obtidos podem ser vistos na tabela 2.3. A janela 8×8 gerou o menor e-erro em todos os 3 testes. Isto não causa surpresa, pois o driver da HP provavelmente utiliza o algoritmo de difusão de ponto [Knuth, 1987] definido numa janela 8×8. Assim, parece que a janela 8×8 é a melhor escolha.

Entretanto, alguém poderia perguntar se temos evidências estatísticas para afirmar que a janela 8×8 é a melhor escolha. Usando a equação 2.8, podemos mostrar que, por exemplo, a janela 8×8 é melhor que 10×10. Podemos concluir, com 95% de confiança, que a diferença esperada entre os dois t-erros é pelo menos 0,096%, quando as amostras de tamanho m = 1050×1050 são usadas, isto é:

$$\mathop{\mathrm{E}}_{\vec{a}\in P^m}\left[\mathrm{t\_error}_P(\hat{\psi}_{8\times 8}) - \mathrm{t\_error}_P(\hat{\psi}_{10\times 10})\right] > 0{,}00096\,.$$

Porém, não podemos afirmar que a janela 8×8 seja melhor que 9×9 com 95% de confiança. Mais dados devem ser coletados para obter informação suficiente para formar uma evidência estatística.

Comparação de diferentes vieses indutivos

Nesta subsubseção, comparamos os diferentes vieses indutivos. Executamos 11 testes com as aprendizagens WZDT, DT original, e5-NN, e1-NN e o viés indutivo aleatório, sempre usando a janela 4×4. Não foi possível executar os testes usando uma janela maior (por exemplo, 8×8) pois a aprendizagem ek-NN é excessivamente lento: de acordo com as nossas estimativas, levaria 6 dias para executar um teste usando o algoritmo força bruta e 100 milhões de anos usando a implementação por look-up-

table. Para tornar as diferenças evidentes, pequenas imagens de treinamento ($A^{\mathbf{x}}$, $A^{\mathbf{y}}$) foram usadas (100×100, 200×200). Por outro lado, as imagens de teste ($Q^{\mathbf{x}}$, $Q^{\mathbf{y}}$) foram grandes (1050×1050, 2100×2100) para obter uma estimativa de t-erro acurada.

Os resultados estão ilustrados na tabela 2.4. O erro médio da aprendizagem WZDT é mais alto que os erros dos outros 3 algoritmos (DT original, e5-NN e e1-NN). Este resultado era esperado, pois escolhemos a aprendizagem WZDT devido ao seu desempenho computacional, com o sacrifício resultante da acurácia do WZ-operador obtido. Podemos executar testes para decidir se as diferenças observadas nas taxas de erro são estatisticamente significativas. Por exemplo, usando a equação 2.8, pode-se mostrar com 95% de confiança que a diferença esperada entre as taxas de t-erro dos algoritmos de WZDT e e1-NN é pelo menos 0,330%, usando m = 10000 exemplos de treinamento. Porém, um resultado similar não pode ser derivado para a diferença esperada entre os t-erros dos métodos de aprendizagem WZDT e DT. As diferenças entre os erros tende a desaparecer à medida que o tamanho das amostras cresce, conforme mostraremos na próxima subsubseção.

Por outro lado, os e-erros da aprendizagem WZDT é notavelmente menor que os e-erros do viés indutivo aleatório. Usando novamente a equação 2.8, pode-se mostrar com 95% de confiança que é esperado que o viés indutivo aleatório cometa pelo menos 1,442% mais erros que a aprendizagem WZDT, usando m = 10000 exemplos de treinamento. Isto mostra claramente que o viés indutivo da aprendizagem WZDT ajuda a diminuir a taxa de erro, mesmo que ele não seja tão efetivo quanto os outros vieses indutivos computacionalmente mais caros.

	4×4	5×5	6×6	7×7	8×8	9×9	10×10	11×11
teste 1 (%)	7,540	2,644	2,025	1,806	1,710	1,772	1,870	2,013
teste 2 (%)	6,105	2,431	2,330	2,374	2,249	2,305	2,486	2,616
teste 3 (%)	7,546	3,676	2,589	2,359	2,354	2,525	2,688	2,740
média (%)	7,064	2,917	2,315	2,180	2,104	2,201	2,348	2,456

Tab. 2.3: Erros empíricos obtidos usando a aprendizagem WZDT com janelas de diferentes tamanhos. Os tamanhos das imagens amostras foram 1050×1050 e 2100×2100 pixels.

	aprendizagem WZDT	aprendizagem original DT	aprendizagem e5-NN	aprendizagem e1-NN	viés indutivo aleatório
teste 1 (%)	8,699	8,664	8,592	8,602	9,262
teste 2 (%)	14,647	14,897	13,791	13,604	15,924
teste 3 (%)	11,541	11,796	11,563	11,471	13,150
teste 4 (%)	11,861	11,263	11,707	11,771	13,961
teste 5 (%)	14,700	13,434	15,922	13,925	18,349
teste 6 (%)	10,699	10,295	10,403	10,386	11,200
teste 7 (%)	14,439	14,436	13,384	13,898	17,369
teste 8 (%)	12,644	12,677	12,109	12,004	13,891
teste 9 (%)	14,483	13,965	13,675	13,995	17,660
teste 10 (%)	17,521	17,951	15,996	16,425	22,370
teste 11 (%)	22,523	20,982	20,267	20,362	24,607
média (%)	13,978	13,669	13,401	13,313	16,158

Tab. 2.4: Erros empíricos dos diferentes algoritmos de aprendizagem. Uma janela 4×4 e imagens amostras com 100×100 e 200×200 pixels foram usadas.

		1050×1050 (1 par)	3190×1050 (3 pares)	6400×1050 (6 pares)	9610×1050 (9 pares)	1050×1050 (e-ótimo)
aprendizagem WZTD	e-erro (%)	1,710	1,561	1,494	1,466	1,111
	treino (s)	11,04	37,35	84,36	146,11	10,49
	aplicação (s)	1,64	2,47	3,35	4,01	1,65
aprendizagem DT original	e-erro (%)	1,617	1,547	1,489	1,464	1,111
	treino (s)	538	2320	7×10^3	128×10^3	540
	aplicação (s)	2,9	3,9	4,0	4,8	3,1

Tab. 2.5: Os erros empíricos diminuem à medida que os tamanhos das amostras crescem. A última coluna mostra o erro do WZ-operador empiricamente ótimo, obtido usando as imagens de teste como as amostras de treinamento. A janela 8×8 foi usada.

Algoritmos de aprendizagem DT e WZDT

Nesta subsubseção, examinaremos cuidadosamente a variação do e-erro à medida que o número de exemplos de treinamento cresce, para obter o melhor WZ-operador possível. Executaremos todos os testes usando a janela 8×8, pois ela parece ser a melhor para a aplicação que estamos estudando. Testaremos somente as aprendizagens WZDT e DT, pois a aprendizagem ek-NN é excessivamente lenta para poder testar.

A tabela 2.5 mostra os resultados experimentais. Usamos, como as imagens amostras, 1, 3, 6 e 9 pares de imagens com (1050×1050, 2100×2100) pixels, grudadas horizontalmente mas separadas por algumas colunas brancas. O par de imagens de teste foi "Lena", com (1050×1050, 2100×2100) pixels (obviamente, o conjunto de imagens de treinamento não incluiu "Lena"). Como era esperado, os e-erros diminuíram à medida que o tamanho das amostras cresceu. Porém, os e-erros diminuíram muito pouco de 6 para 9 pares de imagens amostras, sugerindo que provavelmente já há uma quantidade suficiente de amostras de treinamento e o erro deve estar convergindo a algum limite inferior.

À medida que o tamanho das imagens amostras cresce, as diferenças entre as aprendizagens WZDT e DT diminuem. Para imagens amostras grandes (9 pares de imagens 1050×1050 e 2100×2100), as duas taxas de erro são praticamente idênticas: 1,466% e 1,464%. Porém, o treino da aprendizagem DT original leva 870 vezes mais tempo do que a aprendizagem WZDT. Portanto, na prática, a aprendizagem WZDT é o melhor algoritmo para ser usado para o projeto de WZ-operadores.

O melhor e-erro obtido pela aprendizagem WZDT é 1,466% (a penúltima coluna da tabela 2.5) e o menor e-erro possível é 1,111% (a última coluna da tabela 2.5). Usando as equações 2.4 e 2.5, concluímos com 95% de confiança que o t-erro do operador obtido é no máximo (1,466+0,009)% e o t-erro do operador verdadeiramente ótimo é pelo menos (1,111–0,008)%. Muito provavelmente, este limite inferior está subestimado. Para obter o menor e-erro, supusemos que a imagem de saída ideal estava disponível durante o estágio de treinamento. Isto não acontece numa situação real. As-

sim, o operador obtido pode ser considerado muito próximo do operador ótimo com respeito à distância de Hamming.

Ampliação baseada em meio-tom inverso

Nesta subsubseção, compararemos a aprendizagem WZDT com as ampliações baseadas em meio-tom inverso. As nossas experiências mostram que a aprendizagem WZDT é consideravelmente mais acurada que as ampliações baseadas em meio-tom inverso simples. Uma ampliação baseada em meio-tom inverso pode ser descrita como:

1. Dada uma imagem meio-tom *B*, use algum algoritmo de meio-tom inverso para obter a imagem em níveis de cinza *G* correspondente. Testamos dois filtros passa-baixas como os algoritmos de meio-tom inverso: o filtro gaussiano e a média móvel.
2. Aumente a resolução da imagem *G* usando alguma técnica de ampliação de imagem em níveis de cinza, obtendo a imagem ampliada *G*'. Utilizamos a interpolação linear como a técnica de ampliação em níveis de cinza.
3. Aplique o algoritmo de meio-tom à imagem *G*', para obter a imagem meio-tom ampliada *B*'.

Os e-erros obtidos estão listados na tabela 2.6. O menor e-erro foi 1,929% usando o filtro gaussiano e 1,947% usando o filtro média móvel. Ambas taxas de erro são consideravelmente mais altas que 1,466%, que é a menor taxa de erro obtida usando a aprendizagem WZDT. Os testes foram repetidos mais duas vezes usando imagens diferentes e resultados similares foram obtidos.

	Desvio-padrão da gaussiana (pixels) / tamanho da janela da média móvel (pixels)	erro empírico (%)
Meio-tom inverso usando a filtragem por gaussiana	2,0	3,192
	2,3	2,144
	2,5	1,962
	2,8	1,929
	3,0	2,012
	3,5	2,107
	4,0	2,286
Meio-tom inverso usando a filtragem média móvel	7×7	2,875
	8×8	1,947
	9×9	2,470

Tab. 2.6: Erros empíricos observados utilizando a ampliação baseada em meio-tom inverso.

Mais dados experimentais

Nesta subsubseção, aplicaremos a aprendizagem WZDT para ampliar as imagens meio-tom geradas por diferentes técnicas de meio-tom.

A figura 2.9 mostra a ampliação de imagens meio-tom geradas pela HP LaserJet driver, opção “pontos pequenos”. O melhor operador foi obtido usando a janela 8×8 e um par de imagens amostras com (9610×1050, 19220×2100) pixels. Aplicando-o à imagem “Lena”, a imagem processada apresentou uma taxa de e-erro 1,429%.

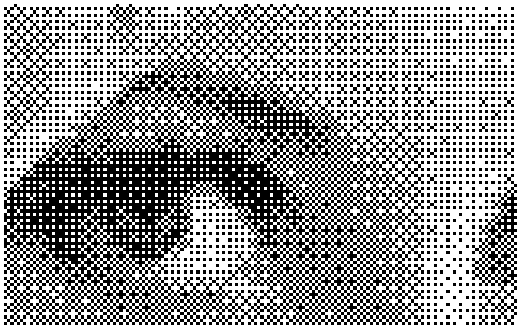
A figura 2.10 mostra a ampliação de imagens meio-tom geradas pelo algoritmo de excitação ordenada pontos aglutinados (clustered-dot ordered dithering) incluído no programa “Image Alchemy” de “Handmade Software, Inc”. A imagem processada tinha uma taxa de e-erro de 1,387%.

As imagens de entrada e saída não necessariamente devem usar a mesma técnica de meio-tom. Por exemplo, podemos usar imagens meio-tom 150 dpi geradas pela HP driver “pontos grandes” como entrada e imagens 300 dpi geradas pelo algoritmo de excitação ordenada pontos aglutinados como saída. Neste caso, a aprendizagem WZDT converte uma técnica de meio-tom numa outra ao mesmo tempo em que se aumenta a resolução. Testamos esta idéia e a imagem processada tinha uma taxa de e-erro de 1,494%. Também testamos o inverso: a conversão de uma imagem meio-tom 150 dpi gerada pela excitação ordenada pontos aglutinados na imagem meio-tom 300 dpi tipo HP pontos grandes. O e-erro resultante foi 1,687%.

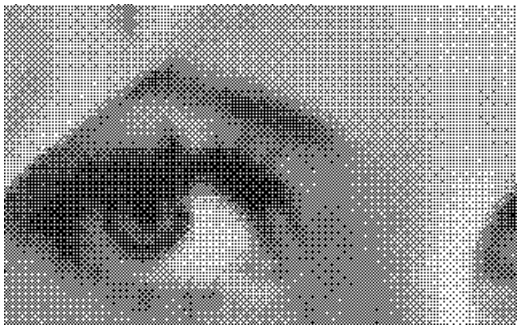
Finalmente, a aprendizagem WZDT foi aplicada para aumentar a resolução de imagens obtidas usando o algoritmo de difusão de erro. Infelizmente, resultados muito ruins foram obtidos. Usando driver HP opção “difusão de erro”, obtivemos uma taxa de e-erro de 12,90%. Usando o algoritmo de Floyd-Steinberg do programa Image Alchemy, obtivemos o e-erro de 42,77% (“algoritmo de difusão de erro” e “algoritmo de Floyd Steinberg” são sinônimos). Estes altos erros eram esperados, pois o algoritmo de difusão de erro não escolhe uma cor de saída em função das cores de uma vi-

zinhança local. Porém, surpreendentemente, a aprendizagem DT pode efetuar o meio-tom inverso com acurácia [Ci11].

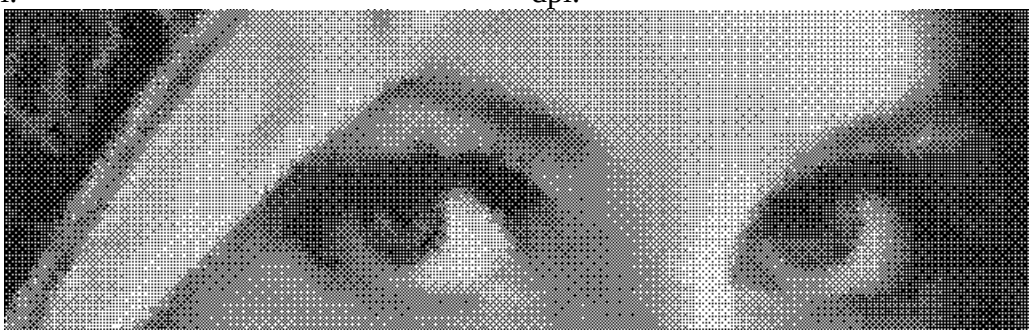
Uma imagem gerada pela difusão de erro pode ser ampliada por um processo de ampliação baseada em meio-tom inverso, resultando numa imagem com uma qualidade visual razoável. Porém, o e-erro resultante é muito alto. Uma imagem meio-tom gerada pela difusão de erro foi convertida numa imagem em níveis de cinza usando um filtro gaussiano com desvio padrão de 2,8 pixels. A imagem em níveis de cinza resultante foi ampliada e convertida novamente numa imagem meio-tom pela difusão de erro. O e-erro obtido foi 43,25%, embora a qualidade visual seja razoável (compare com 42,77% obtido com a aprendizagem WZDT). A distância de Hamming parece não ser uma medida apropriada para quantificar a qualidade das imagens produzidas por processos onde podem ocorrer “deslocamentos de fase”.



(2.9a) Imagem a-ser-ampliada $Q^{\mathbf{x}}$ em 150 dpi.



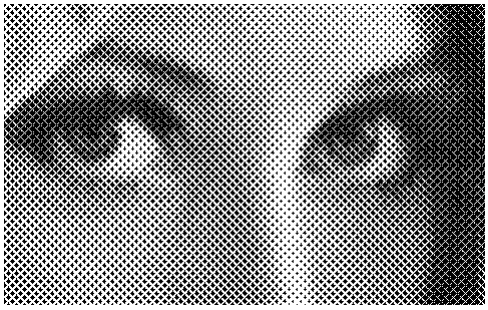
(2.9b) Imagem de saída ideal $Q^{\mathbf{y}}$ em 300 dpi.



(2.9c) Imagem ampliada $\hat{Q}^{\mathbf{y}}$, usando a janela 8×8. Os tamanhos das amostras de treinamento foram 9610×1050 e 19220×2100 pixels. A diferença para a saída ideal foi em 1,429% dos pixels.

Fig. 2.9: Aumento de resolução de imagens meio-tom geradas pela HP LaserJet driver, opção “pontos pequenos”, usando a aprendizagem WZDT.

(2.10a) Imagem a-ser-ampliada $Q^{\mathbf{x}}$ em 150 dpi.



(2.10b) Imagem de saída ideal $Q^{\mathbf{y}}$ em 300 dpi.

(2.10c) Imagem ampliada $\hat{Q}^{\mathbf{y}}$, usando a janela 8×8. Os tamanhos das imagens de treinamento foram 9610×1050 e 19220×2100 pixels. A diferença para a saída ideal foi em 1,387% dos pixels.

Fig. 2.10: Aumento de resolução das imagens meio-tom geradas pelo algoritmo de excitação ordenada, pontos aglutinados, usando a aprendizagem WZDT.

2.5 Meio-Tom Inverso pela Aprendizagem

Introdução

Esta seção descreve uma contribuição científica original minha. Os resultados descritos nesta seção estão documentados no artigo [Ci11].

A maioria de impressoras jato-de-tinta ou laser pode imprimir somente minúsculos pontos pretos sobre o papel. Assim, qualquer imagem em níveis de cinzas deve ser primeiro convertida numa imagem binária antes que a impressão seja efetuada. O processo de meio-tom simula os níveis de cinza distribuindo apropriadamente pixels pretos e brancos. As técnicas de meio-tom populares são a difusão de erro, a excitação ordenada (ordered dithering) e as máscaras de ruído azul [Roetling and Loce, 1994; Ulichney, 1998]. As imagens meio-tom podem ser ou ortográficas (dados digitais perfeitos antes da impressão) ou escaneadas.

O meio-tom inverso (em inglês, inverse halftoning ou descreening, abreviado aqui como IH) é o processo para recuperar a imagem em níveis de cinzas a partir da imagem meio-tom. Como meio-tom é um processo “muitos para um”, não existe uma única imagem em níveis de cinza para uma dada imagem meio-tom. Assim, outras propriedades das imagens devem ser utilizadas em IH. O método IH mais simples é um filtro passa-baixas. Embora este processo produza as imagens em níveis de cinza, borra as arestas e destrói os detalhes finos. Muitos métodos diferentes foram desenvolvidos para melhorar IH por filtragem passa baixas. Veja Luo et al. [Luo et al., 1998] para mais detalhes.

Mese e Vaidyanathan propuseram recentemente uma abordagem baseada na aprendizagem de máquina para fazer o meio-tom inverso das imagens ortográficas. Eles primeiro propuseram um algoritmo que usa look-up-table (LUT) [Mese and Vaidyanathan, 2001] e depois um outro que usa uma estrutura de árvore [Mese and Vaidya-

nathan, 2002]. Em ambos os métodos, existe uma fase de treinamento onde as imagens amostras são utilizadas para construir a estrutura de dados.

A aprendizagem de máquina tem sido usada em muitas aplicações de processamento de imagens. Em particular, pode-se usá-la para aumentar a resolução de imagens meio-tom usando uma LUT [Ri03; Ci02] ou uma árvore de decisão (DT) [Ci05; Ri05]. Estas técnicas podem ser adaptadas de forma direta para o problema IH.

Em [Mese and Vaidyanathan, 2002], um algoritmo de aprendizagem de máquina com uma estrutura de árvore "ad hoc" é empregado. Nós, ao contrário, propomos usar uma abordagem teórica e algorítmica baseada em aprendizagem DT para resolver IH. Mais especificamente, usamos uma versão da aprendizagem DT chamada ID3. Esta abordagem apresenta algumas vantagens sobre a proposta de [Mese and Vaidyanathan, 2002]. Há duas razões para isso. Primeiro, em [Mese and Vaidyanathan, 2002], o usuário deve selecionar cuidadosamente a máscara (isto é, a janela). Eles até apresentam um algoritmo para selecionar uma "boa" janela [Mese and Vaidyanathan, 2001]. A aprendizagem DT isenta o usuário de ter de escolher explicitamente uma janela, porque irá usar automaticamente somente aqueles atributos (furos de espiar, peepholes ou pixels) que são mais necessários para decidir a cor de saída em níveis de cinzas. Enquanto eles usaram janelas com no máximo 19 furos (que pode ser pequena demais para muitos problemas de IH), usando a aprendizagem DT, optamos por uma janela inicial muito maior (por exemplo, 8×8=64 pixels). O algoritmo de aprendizagem por si irá usar automaticamente somente os furos apropriados na ordem apropriada. A escolha dos furos está baseada na maximização de redução esperada da entropia, e esta política tem produzido árvores de alturas mais baixas e com boa capacidade de generalização [Mitchell, 1997; Quinlan, 1986]. Além disso, Mese e Vaidyanathan usam uma estrutura de dados não-ortodoxa (apesar de interessante) que combina a LUT e a árvore binária. Embora não há nada de errado com esta abordagem, a árvore de decisão original é claramente mais elegante, e apresenta um desempenho computacional equivalente.

Acreditamos que [Mese and Vaidyanathan, 2002] possua um desempenho estado de arte e que nosso algoritmo seja o próximo melhoramento natural, pois obtivemos imagens com PSNR que parecem estar 4 dB acima daquelas relatadas em [Mese and Vaidyanathan, 2002]. Os programas e as imagens usados aqui estão disponíveis em:

http://www.lps.usp.br/~hae/software/invhalf.

O problema

As imagens binária e em níveis de cinzas são definidas respectivamente como funções $Q^{\mathbf{x}}: \mathbb{Z}^2 \to \{0,1\}$ e $Q^{\mathbf{y}}: \mathbb{Z}^2 \to [0...255]$. O suporte de uma imagem é um subconjunto finito de $\mathbb{Z}^2$, onde a imagem está realmente definida.

Um operador restrito à janela Ψ "binário para níveis de cinza" é uma função que mapeia uma imagem binária $Q^{\mathbf{x}}$ numa imagem em níveis de cinza $Q^{\mathbf{y}}$. Ele é definido através de um conjunto de w pontos chamada janela $W = \{W_1, \ldots, W_w\}$, $W_i \in \mathbb{Z}^2$, e uma função característica $\psi : \{0,1\}^w \to [0...255]$ como segue:

$$Q^{\mathbf{y}}(p) = \Psi(Q^{\mathbf{x}})(p) = \psi(Q^{\mathbf{x}}(W_1 + p), \ldots, Q^{\mathbf{x}}(W_w + p)),$$

onde $p \in \mathbb{Z}^2$. Cada elemento W_i da janela é chamado furo de espiar (peephole) ou atributo (feature).

Sejam $A^{\mathbf{x}}$, $A^{\mathbf{y}}$, $Q^{\mathbf{x}}$ e $Q^{\mathbf{y}}$ respectivamente as imagens amostra de entrada, amostra de saída, a-ser-processada e ideal (esta supostamente desconhecida). Podemos supor que existe um único par de imagens de treinamento ($A^{\mathbf{x}}$ e $A^{\mathbf{y}}$). Se existirem mais pares, eles podem ser colados para formar um único par.

Vamos denotar o conteúdo em $A^{\mathbf{x}}$ da janela W deslocada para $p \in \mathbb{Z}^2$ como $a_p^{\mathbf{x}}$ e chamá-lo de instância de treinamento ou padrão de entrada de treinamento no pixel p:

$$a_p^{\mathbf{x}} = \left(A^{\mathbf{x}}(W_1 + p), A^{\mathbf{x}}(W_2 + p), \ldots, A^{\mathbf{x}}(W_w + p)\right) \in \{0,1\}^w.$$

A cada padrão $a_p^{\mathbf{x}}$, está associada uma cor de saída $A^{\mathbf{y}}(p) \in [0...255]$. Vamos denotar o conjunto obtido quando todos os pixels de $A^{\mathbf{x}}$ e $A^{\mathbf{y}}$ são varridos por

$$a = \left\{(a^{\mathbf{x}}_{p_1}, A^{\mathbf{y}}(p_1)), \ldots, (a^{\mathbf{x}}_{p_m}, A^{\mathbf{y}}(p_m))\right\}$$

e chamá-lo de conjunto amostra ou conjunto de treinamento (*m* é a quantidade de pixels das imagens $A^{\mathbf{x}}$ e $A^{\mathbf{y}}$). Vamos construir de forma similar o conjunto

$$q = \left\{(q^{\mathbf{x}}_{p_1}, Q^{\mathbf{y}}(p_1)), \ldots, (q^{\mathbf{x}}_{p_n}, Q^{\mathbf{y}}(p_n))\right\}$$

a partir das imagens $Q^{\mathbf{x}}$ e $Q^{\mathbf{y}}$ (*n* é a quantidade de pixels de $Q^{\mathbf{x}}$ e $Q^{\mathbf{y}}$). Cada padrão $q^{\mathbf{x}}_{p_i}$ é chamado de um padrão de busca ou uma instância a ser processada e a cor $Q^{\mathbf{y}}(p_i) \in [0...255]$ é chamada de cor de saída ideal.

No problema de IH, um algoritmo de aprendizagem **A** constrói um operador $\hat{\Psi}$ baseado em $A^{\mathbf{x}}$ e $A^{\mathbf{y}}$ tal que, quando $\hat{\Psi}$ é aplicado a $Q^{\mathbf{x}}$, espera-se que a imagem processada resultante $\hat{Q}^{\mathbf{y}} = \hat{\Psi}(Q^{\mathbf{x}})$ seja similar à imagem de saída ideal $Q^{\mathbf{y}}$. Para descrever este processo de forma mais precisa, vamos definir uma função de perda (ou erro) *l* que será usada para medir a diferença entre as saídas ideal e processada. Exemplos de possíveis funções de perda são:

Perda quadrática: $l(Q^{\mathbf{y}}(p), \hat{Q}^{\mathbf{y}}(p)) = (Q^{\mathbf{y}}(p) - \hat{Q}^{\mathbf{y}}(p))^2$

Perda absoluta: $l(Q^{\mathbf{y}}(p), \hat{Q}^{\mathbf{y}}(p)) = \left|Q^{\mathbf{y}}(p) - \hat{Q}^{\mathbf{y}}(p)\right|$.

Re-enunciando o problema IH, o aprendiz **A** deve construir uma função característica ou hipótese $\hat{\psi}$ baseada em conjunto amostra *a* tal que, quando $\hat{\psi}$ é aplicado a um padrão de busca $q^{\mathbf{x}}_{p_i}$ gerando a cor de saída $\hat{Q}^{\mathbf{y}}(p_i) = \hat{\psi}(q^{\mathbf{x}}_{p_i})$, a perda $l\left(Q^{\mathbf{y}}(p_i), \hat{Q}^{\mathbf{y}}(p_i)\right)$ deve ser baixa com alta probabilidade.

Algoritmos

Existem muitas técnicas de aprendizagem que poderiam ser usadas para conseguir o objetivo acima. Porém, para ser realmente útil, a técnica deve apresentar as seguintes características:

1. Ela deve generalizar para além do conjunto de treinamento. Isto é, a técnica deve gerar saídas com pequena perda não somente para os padrões do conjunto de treinamento mas também para outros padrões nunca vistos.
2. Ela deve ser rápida no estágio de aplicação do operador.
3. O espaço de memória necessário deve ser moderado.
4. A fase de treinamento não deve ser muito lenta, embora alguma lentidão possa ser tolerável.

Felizmente, a aprendizagem DT satisfaz todos esses requerimentos. Descrevemos brevemente abaixo o algoritmo de construção DT. Existem muitas versões de aprendizagem DT. Nesta seção, usamos o algoritmo ID3 [Mitchell, 1997; Quinlan, 1986]. No processo de geração de DT, o espaço de entrada $\{0,1\}^w$ é particionado em duas metades, e todos os padrões de treinamento com cor preta no atributo W_s irão pertencer a um semi-espaço e aqueles com cor branca a outro. A dimensão dos semi-espaços assim obtidos é um a menos que o espaço original, isto é, $\{0,1\}^{w-1}$. Para cada um dos dois semi-espaços obtidos, o processo que partição continua recursivamente, gerando espaços cada vez menores. Em cada partição, um nó interno é criado e o atributo *s* da partição é armazenado. Este processo pára quando cada espaço contiver ou somente amostras com a mesma cor de saída ou somente amostras com o mesmo padrão de entrada (mas com dois ou mais diferentes cores de saída). No primeiro caso, um nó terminal é criado e a cor de saída é armazenada nele. O segundo caso é chamado de conflito. Neste caso, um nó terminal é criado, mas a média dos valores de saída é calculada e armazenada (se a perda quadrática ou PSNR deve ser minimizada). A média deve ser substituída pela mediana se a perda absoluta deve ser minimizada.

Se não existem conflitos, o algoritmo acima deve classificar perfeitamente o conjunto de treinamento. Porém, os pesquisadores de aprendizagem de máquina observaram que esta estratégia pode levar a “superencaixamento” (overfitting). Dizemos que uma hipótese está superencaixada nos exemplos de treinamento se alguma outra hipótese que se encaixa mais pobremente nos exemplos de treinamento possui um desempe-

nho melhor sobre a distribuição global das instâncias. Na literatura, existem algumas estratégias sofisticadas para evitar o superencaixamento. Porém, usando-as, a fase de treinamento torna-se excessivamente demorada. Assim, usamos a seguinte estratégia simples: a média (ou mediana) é calculada toda vez que existirem *k* ou menos amostras num subespaço dos padrões. Denotaremos esta estratégia como *k*-ID3, por exemplo, 1-ID3 ou 10-ID3. Quando $k = 1$, temos o algoritmo ID3 original.

A questão central no algoritmo ID3 é como selecionar o atributo a ser usado para particionar o espaço dos padrões em cada nó interno da árvore. Diferentes escolhas gerarão diferentes árvores de decisão. Dadas duas ou mais DTs, é amplamente aceito que a mais simples (ou seja, a árvore de altura mais baixa) deve ser a preferida. Esta escolha é conhecida como "navalha de Occam" e muitos estudos apontam a sua superioridade, incluindo a nossa própria experiência com o problema IH. Uma maneira de implementar a "navalha de Occam" seria gerar todas as possíveis DTs e selecionar a mais baixa entre elas. Claramente, esta abordagem é impraticável pois levaria um tempo excessivamente longo.

O algoritmo ID3 utiliza o seguinte critério que segue de perto a "navalha de Occam". Consiste em colocar os atributos com alto ganho de informação mais perto da raiz. Vamos definir a entropia de um conjunto de amostras *a* onde cada amostra pode assumir um entre *c* diferentes valores de saída:

$$\text{Entropy}(a) \equiv \sum_{i=1}^{c} - p_i \log_2 p_i$$

onde p_i é a proporção de exemplos de *a* que pertence à classe *i*. Para o problema de IH, discretizamos (isto é, quantizamos) 256 possíveis valores em $c = 16$ categorias: [0...15], [16...31], etc. Esta discretização é usada somente para calcular a entropia. O valor preciso da saída é ainda armazenado nas folhas da DT. O ganho de informação, definido abaixo, é a redução esperada da entropia causada pela partição dos exemplos de acordo com o atributo *s*:

$$\text{Gain}(a, s) \equiv \text{Entropy}(a) - \sum_{v \in \{0,1\}} \frac{|a_v|}{|a|} \text{Entropy}(a_v)$$

onde a_v é o subconjunto de a na qual o atributo s tem valor v. Em cada nó interno, o algoritmo de aprendizagem escolhe o atributo que maximiza o ganho de informação.

Depois que a DT tiver sido construída, a sua aplicação é direta: dado um padrão de busca $q_p^{\mathbf{x}}$, a DT é percorrida de cima para baixo, até chegar numa folha. A informação contida nesta folha é então escolhida como o valor de saída $\hat{Q}^{\mathbf{y}}(p_i)$.

Denominaremos de "algoritmo seqüencial" o algoritmo de aprendizagem DT que escolhe os furos seqüencialmente numa ordem pré-determinada, sem usar o critério de maximização de ganho de entropia. Para verificar a eficácia do viés indutivo de maximização de ganho de entropia, compararemos os algoritmos seqüencial e ID3.

Resultados e Dados Experimentais

A técnica proposta foi implementada e testada. Usamos três pares de imagens com 1050×1050 pixels para o treinamento. Aplicamos o sistema IH resultante em três imagens 1050×1050 completamente independentes. Usamos a janela 8×8 em todos os casos. Os testes foram repetidos para 4 diferentes tipos de meio-tom:

1. Difusão de erro (algoritmo de Floyd-Steinberg).
2. Excitação ordenada, pontos dispersos (algoritmo de Bayer).
3. Excitação ordenada, pontos aglutinados.
4. O meio-tom do HP LaserJet driver para Windows, opção "pontos grandes".

A figura 2.11 mostra a aplicação da técnica proposta a imagens meio-tom obtidas pela difusão de erro. Parte de uma das imagens de treinamento é mostrada nas figuras 2.11a e 2.11b. Parte de uma das imagens a serem processadas é mostrada na figura 2.11c e a correspondente saída ideal (supostamente desconhecida) na figura 2.11d. As figuras 2.11e e 2.11f são as imagens obtidas usando algoritmos seqüencial e 10-ID3, respectivamente. As suas PSNRs são 26,75 e 34,75 dB para a imagem Lena. Isto demonstra que a maximização de ganho de informação desempenha um papel importante em melhorar a qualidade do sistema IH obtido. Aplicando o sistema IH a 3 imagens de teste diferentes (uma das quais é Lena), podemos ver que a PSNR varia

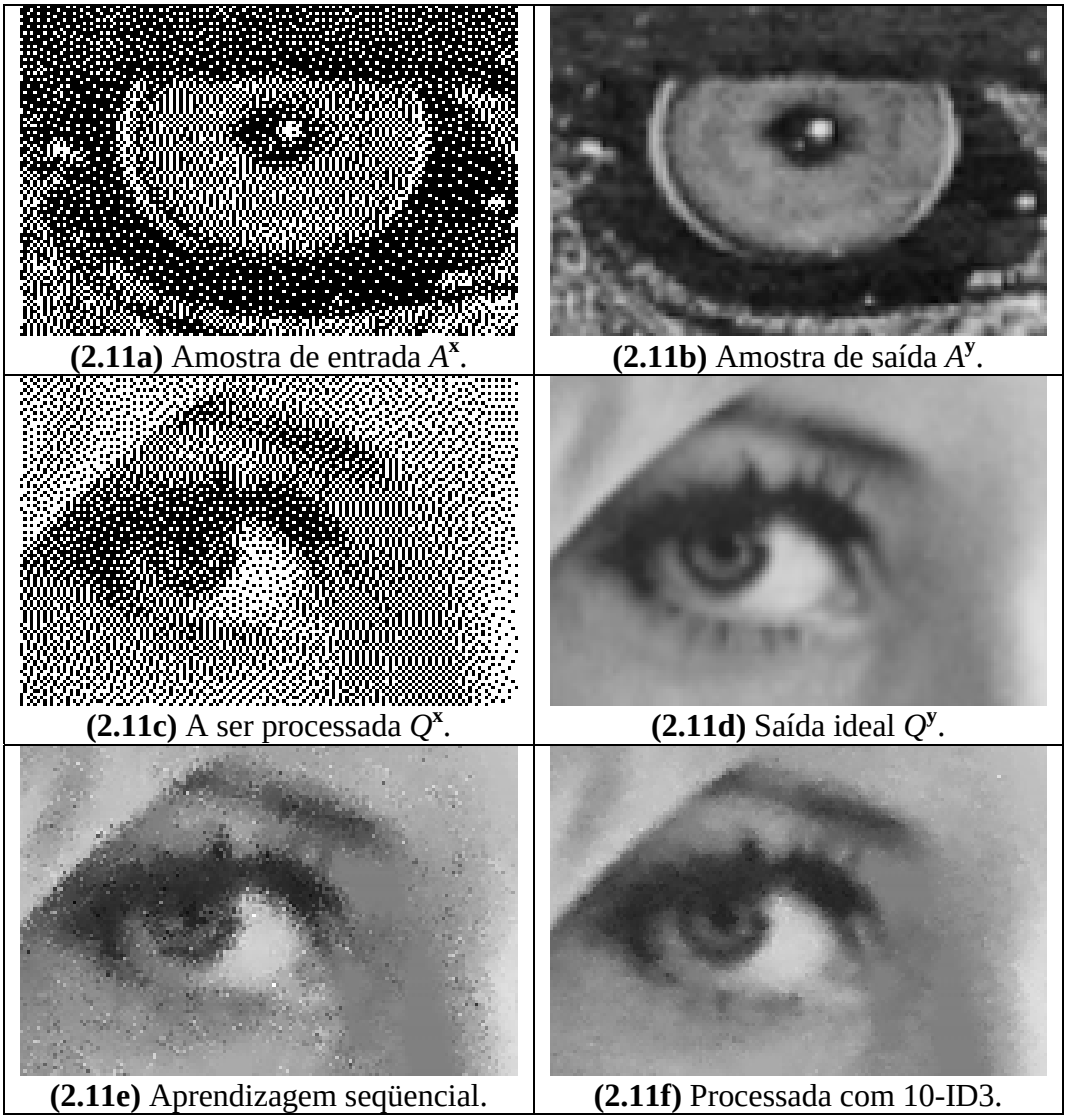
consideravelmente, conforme mostrada na tabela 2.7. A imagem “Lena” parece ser “boa” para fazer o meio-tom inverso, provavelmente por causa das suas amplas áreas suaves.

A figura 2.12 mostra partes das imagens meio-tom obtidas usando diferentes algoritmos e as suas respectivas imagens obtidas através do meio-tom inverso por 10-ID3. Suas PSNRs são mostradas na tabela 2.7. Em [Mese and Vaidyanathan, 2002], a PSNR relatada foi 27,08 dB para a difusão de erro. A nossa técnica produziu PSNR de 31,80 dB para um conjunto de imagens, que, embora diferente das usadas por [Mese and Vaidyanathan, 2002], contém imagens com características semelhantes. Testando somente a imagem Lena, [Mese and Vaidyanathan, 2002] relata 30,95 dB enquanto a nossa técnica produz 34,75 dB. Para a excitação ordenada, o nosso método produz 30,14 dB e 28,91 dB respectivamente para as opções “pontos dispersos” e “pontos aglutinados”. Os números correspondentes em [Mese and Vaidyanathan, 2002] são 25,82 dB e 24,26 dB. Isto parece apontar para um ganho de 4 dB em comparação com os resultados anteriores.

A fase de aplicação de IH leva somente 5 segundos num Pentium-1GHz, para uma imagem teste com 1050×1050 pixels. A fase de treinamento leva aproximadamente 20 minutos para ID3 e 1 minuto para a aprendizagem seqüencial (usando imagens com 1050×3150 pixels). A estrutura DT ocupa aproximadamente 5 MBytes.

		10-ID3 (dB)	1-ID3 (dB)	Seqüencial(dB)
Difusão de erro	3 imagens	31,80	31,02	25,39
	Somente Lena	34,75	33,20	26,75
OD disperso	3 imagens	30,14	29,73	28,11
	Somente Lena	33,69	33,17	32,97
OD aglutinado	3 imagens	28,91	28,56	27,78
	Somente Lena	32,59	32,01	31,83
HP laserjet, pontos grandes	3 imagens	27,66	27,32	27,35
	Somente Lena	31,72	31,16	31,07

Tab. 2.7: PSNRs obtidas usando diferentes algoritmos de aprendizagem DT e diferentes algoritmos de meio-tom. “3 imagens” refere às PSNRs obtidas testando o sistema IH em 3 imagens de teste diferentes, uma das quais era Lena. “Somente Lena” refere à PSNR obtida fazendo o meio-tom inverso da imagem Lena.



(2.11a) Amostra de entrada $A^{\mathbf{x}}$.

(2.11b) Amostra de saída $A^{\mathbf{y}}$.

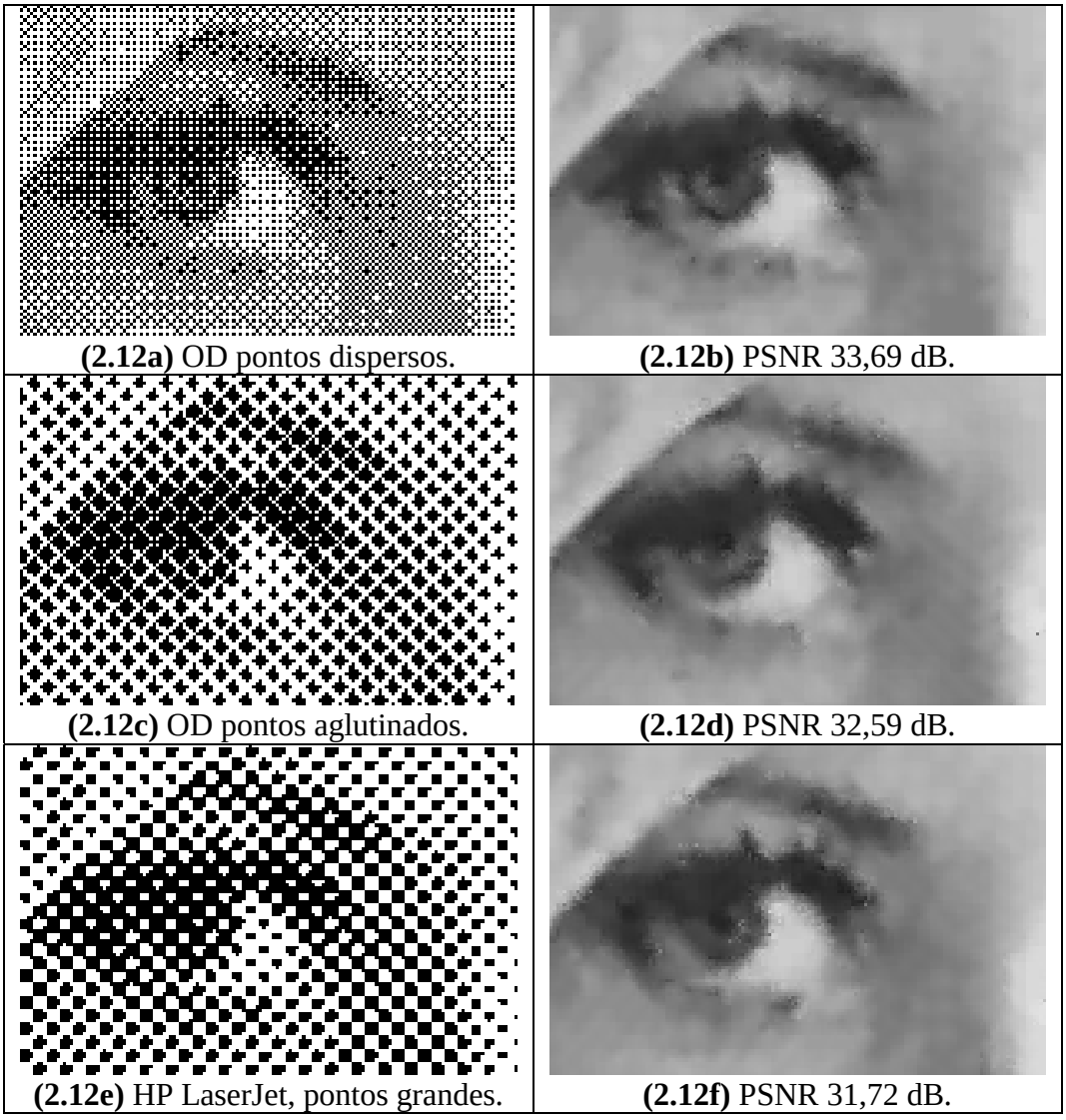
(2.11c) A ser processada $Q^{\mathbf{x}}$.

(2.11d) Saída ideal $Q^{\mathbf{y}}$.

(2.11e) Aprendizagem seqüencial.

(2.11f) Processada com 10-ID3.

Fig. 2.11: Meio-tom inverso das imagens obtidas pela difusão de erro, pela aprendizagem DT. (a, b) Amostra de entrada $A^{\mathbf{x}}$ e amostra de saída $A^{\mathbf{y}}$ (Mandrill). (c, d) Imagem a ser processada $Q^{\mathbf{x}}$ e a saída ideal $Q^{\mathbf{y}}$, supostamente desconhecida (Lena). (e) Imagem obtida usando aprendizagem seqüencial (PSNR 26,75 dB). (f) Imagem obtida usando algoritmo 10-ID3 (PSNR 34,75 dB).



(2.12a) OD pontos dispersos. **(2.12b)** PSNR 33,69 dB.

(2.12c) OD pontos aglutinados. **(2.12d)** PSNR 32,59 dB.

(2.12e) HP LaserJet, pontos grandes. **(2.12f)** PSNR 31,72 dB.

Fig. 2.12: Meio-tom inverso pelo algoritmo 10-ID3 aplicado em diferentes tipos de imagens meio-tom. A coluna da esquerda é a imagem meio-tom e a coluna da direita é a correspondente imagem em níveis de cinza obtida pelo algoritmo de meio-tom inverso 10-ID3. (a, b) Excitação ordenada, pontos dispersos (algoritmo de Bayer). (c, d) Excitação ordenada pontos aglutinados. (e, f) Imagem meio-tom gerada pelo driver para HP LaserJet para MS-Windows, usando a opção “pontos grandes”.

2.6 Conclusões

O objetivo deste capítulo foi apresentar as nossas contribuições científicas no uso das técnicas de aprendizagem de máquina no projeto automático de operadores restritos à janela (W-operadores).

Para isso, formalizamos o problema de aprendizagem de W-operadores usando a teoria PAC. Descrevemos como a estimação estatística pode ser utilizada para verificar se um operador obtido pela aprendizagem está próximo (ou não) do operador ótimo. Também descrevemos como usar a estimação estatística para comparar a eficácia de dois diferentes algoritmos de aprendizagem quanto à acurácia esperada do operador obtido. Descrevemos diversos algoritmos de aprendizagem, juntamente com as suas complexidades computacionais do treinamento, da aplicação e da memória necessária. Adaptamos alguns algoritmos de aprendizagem para que sejam mais eficientes no problema que estamos tratando: a aprendizagem k-NN tornou-se ek-NN para que a teoria PAC pudesse ser aplicada, e a aprendizagem DT tornou-se WZDT para melhorar o seu desempenho computacional. Aplicamos as teorias e os algoritmos desenvolvidos em três problemas: a ampliação de imagens binárias, a ampliação de imagens meio-tom e o meio-tom inverso, obtendo em todos eles bons resultados quanto à acurácia da solução e ao desempenho computacional.