Método Híbrido de Reconhecimento Ótico de Caracteres

William Ivanski, Luciano Silva, Olga R. P. Bellon
{wiva06, luciano, olga}@inf.ufpr.br
Grupo IMAGO de Pesquisa*† – UFPR, Cx.P. 19902 – 81531-990 – Curitiba-PR

Resumo

Com os recentes avanços nas áreas de Processamento de Imagens e Visão Computacional, reconhecer caracteres em imagens tem se tornado viável para uma variedade cada vez maior de aplicações, incluindo a integração entre OCR (Optical Character Recognition) e ferramentas de acessibilidade. Este trabalho apresenta um sistema de OCR híbrido, baseado em Análise Estrutural e Comparação de Modelos, e ainda conta com um método de segmentação de caracteres conexos. Para mostrar a efetividade da solução proposta, foi realizado um experimento comparativo utilizando outros dois sistemas de OCR para software livre.

1. Introdução

Reconhecimento Ótico de Caracteres, ou no inglês *Optical Character Recognition* (OCR) [2, 10, 13], é uma das mais importantes áreas de Visão Computacional. O OCR consiste em extrair texto de imagens e, através de um algoritmo de classificação, gerar um texto editável equivalente ao texto contido na imagem. É usado em *software* de *scanners* de mesa, que digitalizam um documento e reconhecem o texto nele contido, permitindo que o usuário altere o documento da forma que desejar [4]. Para isso, o sistema OCR deve ser capaz de processar imagens degradadas, com ruído, apresentando textos com fontes inclinadas ou distorcidas, em diferentes formatos e resoluções [4, 9].

Recentemente, OCR também tem sido usado como ferramenta de acessibilidade, integrado a outras tecnologias assistivas, tais como leitores e magnificadores de tela [5]. Para tal finalidade, um sistema OCR ideal deve conseguir reconhecer texto com fontes nos tamanhos de 10 a 12, mais usados em aplicações *Desktop*, e nos tamanhos de 12 a 14, mais usados na *Web* [7].

De acordo com o método de extração de características utilizado na fase de reconhecimento, os sistemas OCR podem ser divididos em duas categorias: os sistemas baseados em Análise Estrutural e os sistemas baseados em Comparação de Modelos [11]. A Análise Estrutural extrai de cada caractere um conjunto de características simples, também chamadas de descritores, que formam a representação estrutural do caractere [11]. Como exemplos de sistemas baseados somente em Análise Estrutural, temos o *gocr* [1], o *ocrad* [2] e o *tesseract* [3]. A Comparação de Modelos utiliza apenas uma matriz, o chamado modelo, para representar a forma do caractere, de acordo com o posicionamento de seus pontos [11]. Como exemplos de sistemas que utilizam somente Comparação de Modelos, temos o *calera* [2] e o sistema descrito em [12, 13].

Os sistemas que baseiam-se em somente uma dessas categorias são chamados puros, e os sistemas que baseiam-se nas duas são chamados híbridos [11]. O problema de utilizar métodos puros de reconhecimento é que o número de amostras de treinamento da base de dados deve ser muito grande para que o sistema atinja a robustez. O *tesseract*, por exemplo, utiliza 60.160 amostras em sua fase de treinamento [10], e um algoritmo de reconhecimento detalhado em [1] utiliza 1.175.000 amostras. Ainda, se o método de reconhecimento for baseado somente em Análise Estrutural, o número de características deve ser muito grande. O *tesseract* extrai entre 50 e 100 características para cada caractere [10]. Outro problema da Análise Estrutural pura é a intolerância a variações estruturais no caractere, causadas devido à degradação ou à baixa qualidade da imagem [2].

Neste trabalho é apresentado um método híbrido que explora as propriedades da Análise Estrutural e da Comparação de Modelos. Da união destes dois métodos, o método proposto possui as vantagens de utilizar poucos descritores e menos amostras na fase de treinamento, como será apresentado nas seções seguintes. Para a implementação do sistema OCR proposto, denominado *imagocr*, foram utilizados algoritmos clássicos encontrados na literatura, porém agregando novas e importantes contribuições: (1) algoritmo de corte para segmentação

---

∗ www.imago.ufpr.br

† Os autores gostariam de agradecer ao CNPq e a FINEP pelo suporte financeiro.

1 http://sourceforge.net/projects/jocr/

2 http://www.gnu.org/software/ocrad/

3 http://sourceforge.net/projects/tesseract-ocr/

de caracteres conexos; (2) algoritmo de segmentação de palavras; (3) conjunto de descritores; (4) novo algoritmo de reconhecimento, que baseia-se no conjunto de descritores. O objetivo da implementação deste sistema é a sua aplicação em ferramentas de acessibilidade para pessoas com necessidades especiais [5].

2. Visão Geral do Sistema

O sistema OCR possui quatro etapas principais: (1) Aquisição da Imagem, (2) Pré-Processamento, (3) Segmentação e (4) Reconhecimento.

2.1. Aquisição da Imagem e Pré-Processamento

As imagens de entrada do sistema são imagens de ambientes Desktop e Web, capturadas com o auxílio da ferramenta *MouseLupa* [5]. A princípio, consideramos apenas imagens com fundo bem comportado, sem a presença de figuras e sem ruído ou degradação. Foi desenvolvida uma pequena biblioteca para processar essa imagens.

Uma vez carregada na memória, a imagem passa pela etapa de pré-processamento, onde é convertida para escala de cinza e em seguida é binarizada utilizando o algoritmo de Otsu [8], que encontra um limiar de binarização automaticamente. Não foi usado um algoritmo de binarização com limiar fixo porque diversas fontes representam caracteres pequenos utilizando fatores de cinza muito menores do que os utilizados em caracteres maiores, representados pela mesma fonte.

2.2. Segmentação

Através da projeção horizontal da imagem binária, são detectadas linhas de texto em um documento [3], isolando linha por linha. Para isolar os caracteres em cada linha, poderia ser utilizada projeção vertical [3] ou rotulação de componentes conexos [4]. A segunda facilita a extração de caracteres em fundos com uma certa variação, e generaliza o processo de reconhecimento. De agora em diante, cada caractere extraído de uma linha, através de rotulação, será chamado de objeto.

Quando a fonte é muito pequena, usualmente no texto se encontram caracteres que se tocam, *i.e.*, estão "grudados", como apresentado na Fig. 1. Nesses casos, o objeto representa mais de um caractere, de forma que não se pode reconhecer tal objeto utilizando o processo de reconhecimento a nível de caractere. Na literatura são encontrados muitos métodos que tratam desse problema com muita eficiência quando aplicados a fontes de largura fixa. Entretanto, o problema ainda está em aberto quando são tratadas fontes de largura proporcional ou variável [3].

(a) (b)

Figura 1. Exemplos de palavras e suas projeções verticais: (a) caracteres que não se tocam, e que são facilmente segmentados através de rotulação; (b) caracteres que se tocam, e que são segmentados com o auxílio de pontos de corte (linhas verticais delimitando o caractere "A").

Para segmentar caracteres grudados, foi utilizado um método de corte que utiliza projeção vertical (veja Fig. 1), e é baseado em reconhecimento [3, 9, 12]. Tal método encontra um ou mais pontos de corte em um dado objeto analisando a projeção vertical desse objeto, e em seguida decide se os pontos de corte estão corretos ou não, através do reconhecimento das partes isoladamente. Esse processo, chamado janelas deslizantes, é feito com o auxílio de uma estrutura de dados (árvore), e cada parte reconhecida isoladamente é chamada de janela. O caminho com a maior probabilidade, da raíz da árvore até suas folhas, é escolhido como o correto [9, 12].

Como os pontos de corte sempre são os pontos mínimos na projeção vertical do objeto [3, 9], teríamos pontos de corte no meio de caracteres tais como 0, o, e, B, e a, dentre outros. Por essa razão, foi acrescentada ao método de corte uma condição a mais na escolha de pontos de corte. Junto com a projeção vertical, também é obtida uma projeção vertical de transformações, que contabiliza, para cada coluna, a soma das transformações de 0 para 1. Então, pontos mínimos na projeção vertical do objeto não são considerados pontos de corte se, naquele ponto, o valor da projeção vertical de transformações for maior do que 1. Isso elimina os cortes desnecessários no meio de caracteres.

As distâncias entre os caracteres atual e anterior são usadas para segmentar palavras, o que significa saber onde inserir espaços em branco em uma seqüência de caracteres reconhecidos. A solução adotada reside no fato de que, dentro de um conjunto de distâncias entre caracteres, os espaçamentos entre palavras são sempre maiores do que espaçamentos entre caracteres dentro de uma mesma palavra. Isso pode ser feito usando o algoritmo de Otsu [8], que encontra um limiar divisor entre dois grupos de valores. Se a distância entre o caractere atual e o anterior for maior do que o limiar, então temos um espaçamento entre palavras; caso contrário, temos um espaçamento simples entre caracteres numa mesma palavra.

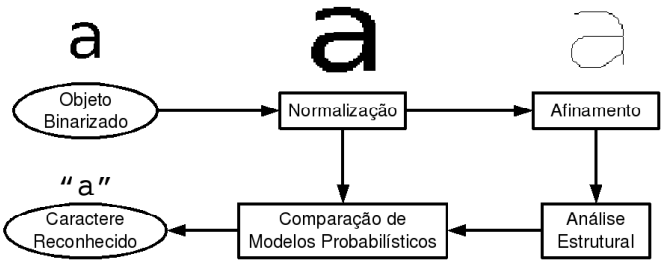

Figura 2. Diagrama do sistema proposto para o reconhecimento de caracteres.

2.3. Reconhecimento

Quanto menos características forem usadas na Análise Estrutural, menos eficiente é o método de reconhecimento, pois faltam elementos discriminantes para diferenciar caracteres parecidos. Além disso, quanto menos amostras forem utilizadas para treinar um método de Comparação de Modelos, menos robusto será o reconhecimento, pois os modelos da base não estarão aptos a se adaptar a determinadas variações na forma do caractere não previstas pela fase de treinamento. Em ambos os casos, o algoritmo de reconhecimento não saberá discernir corretamente determinados caracteres parecidos.

A estratégia adotada no sistema proposto explora os pontos fracos e fortes da Análise Estrutural e da Comparação de Modelos, através de um método híbrido, que combina os dois métodos. O reconhecimento foi dividido em duas fases: uma fase de Análise Estrutural, que utiliza menos descritores do que um método puro, e outra fase de Comparação de Modelos Probabilísticos. Outro ponto importante do método proposto é usar um número muito menor de amostras na fase de treinamento: foram utilizadas 35 amostras de 86 caracteres, totalizando 3010 amostras.

O algoritmo de reconhecimento de caracteres é ilustrado na Fig. 2. Primeiro, a matriz binarizada do objeto é normalizada, o que significa que ela é redimensionada para o tamanho dos caracteres contidos na base de dados [4]. Em seguida, ela é afinada, utilizando o algoritmo de esqueletização descrito em [9]. A partir do esqueleto, são extraídas as seguintes características: (1) acentuação: diz se o caractere é acentuado ou não, a partir da análise do objeto imediatamente superior ao caractere em questão; (2) medianas: número de cortes nos eixos vertical e horizontal [11]; (3) lagos: número de lagos e posicionamento dos seus centros [6]; (4) finais de linha: número e posicionamento dos finais de linha de um caractere [6, 11]; (5) junções: número e posicionamento das junções do caractere [6, 11].

Durante a análise dessas características, vão sendo eliminados os caracteres da base que não são semelhantes ao objeto que está sendo comparado. Ao final, podem restar zero, um ou mais de um caractere da base. Se restar zero ou um caractere, então o reconhecimento atingiu o seu fim. Se restar mais de um caractere, então entra em ação a fase de Comparação de Modelos Probabilísticos. Esta fase compara os modelos dos caracteres da base que restaram com o objeto que está sendo comparado. Os modelos da base são ponderados, durante a fase de treinamento, de acordo com a freqüência dos valores que cada posição do modelo pode assumir. Ao final da comparação, para cada caractere restante da base, temos uma probabilidade de o objeto que está sendo comparado ser aquele caractere. O caractere da base com a maior probabilidade é escolhido como o correto.

3. Resultados Experimentais

O experimento realizado foi baseado nos testes feitos em [1], que utiliza imagens em escala de cinza contendo todos os caracteres da tabela ASCII, formatados em 100 fontes diferentes e 10 tamanhos entre 5 e 14. Foram feitas algumas modificações no experimento apresentado em [1]. Foram utilizadas 168 imagens em escala de cinza, cada imagem continha todas as letras do alfabeto inglês, maiúsculas e minúsculas, e também dígitos de 0 a 9. Os textos nas imagens foram formatados em 24 tamanhos de 6 a 60, e em 7 tipos de fonte: *Arial*, *Avant Garde*, *Bitstream Vera Sans*, *Courier 10 Pitch*, *Helvetica*, *Times*, *Times New Roman*, que são as fontes mais utilizadas em ambientes Desktop e Web [7].

A comparação foi feita entre o *imagocr*, o *gocr* e o *ocrad*. Não foram inseridos no experimento os sistemas anteriormente mencionados: (1) *calera*, por ser proprietário; (2) o sistema descrito em [12, 13], por não ser disponível para download gratuito; e (3) *tesseract*, por somente aceitar como entrada imagens bitonais.

A Fig. 3 mostra um exemplo de imagem utilizada no experimento, bem como os resultados correspondentes para cada um dos sistemas comparados. A Fig. 4 apresenta, para os três sistemas testados, a taxa de reconhecimento média em cada tamanho de fonte. A avaliação completa para cada sistema está disponível na página do grupo [4].

4 www.imago.ufpr.br/imagocr

```
                THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG
imagocr: THE QU7cK BROWN FOX JuMPS oVER THE _ZY DoG
gocr:    THE au cH BRowN Fox Jums ovER THE  DoG
ocrad:   THE OUICK BROWN FOX_UMP_ OV_R THE _ DOC
```

Figura 3. Exemplo de imagem usada no experimento e saídas correspondentes dos três sistemas. De cima para baixo: imagem original, fonte Arial tamanho 14; e saídas do imagocr, gocr e ocrad, respectivamente.

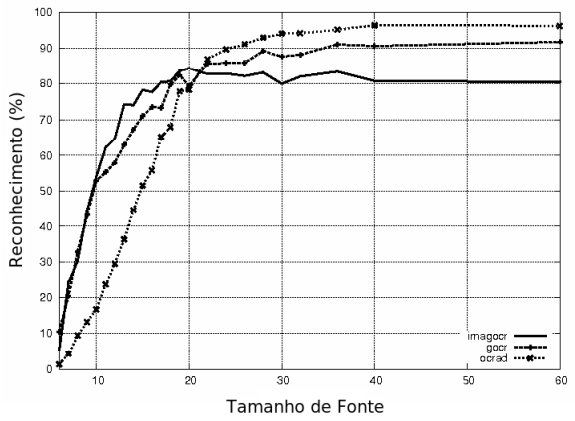

Figura 4. Comparação dos sistemas de OCR com a taxa de reconhecimento média em cada tamanho de fonte.

O sistema proposto atinge um desempenho equivalente aos outros sistemas utilizando uma base de dados menor. O tempo médio de execução do sistema, para cada imagem, foi abaixo de 0.3 segundos. Também é possível perceber que o *imagocr* é mais preciso quando o tamanho varia de 9 a 22. Segundo [7], os tamanhos de fonte mais utilizados em ambientes *Desktop* variam de 10 a 12, e na *Web* de 12 a 14. Por isso, o treinamento da base de dados foi realizado somente para estes tamanhos de fonte. As etapas de segmentação de caracteres, Análise Estrutural e Comparação de Modelos Probabilísticos também foram construídas utilizando caracteres pequenos. A consequência disso é que, para tamanhos maiores, a taxa de reconhecimento se estabiliza, pois, por mais que fontes maiores permitam a extração de mais características, o método limita-se apenas a características inerentes a fontes menores.

4. Conclusão

Através de um conjunto pequeno de características e um mecanismo simples de comparação de modelos probabilísticos, é possível implementar a base, ou núcleo, de um sistema OCR. Isso se deve ao uso de métodos híbridos de reconhecimento, que exploram as principais propriedades de métodos distintos. Utilizando um método híbrido, também é possível diminuir consideravelmente o número de amostras da fase de treinamento.

Como apresentado na seção anterior, o sistema proposto destaca-se no reconhecimento de fontes de tamanhos de 9 a 22, que são os tamanhos mais usados em computadores pessoais hoje em dia, seja para aplicativos *Desktop* ou *Web*. Devido às suas etapas de pré-processamento e segmentação, o sistema proposto limita-se apenas a imagens com fundo bem comportado. Entretanto, o sistema pode ser utilizado com eficiência no reconhecimento de textos em aplicativos de maneira a auxiliar pessoas com necessidades especiais no uso do computador.

Referências

[1] H. S. Baird and R. Fossey. A 100-font classifier. *Proc. ICDAR*, pages 332–340, 1991.

[2] M. Bokser. Omnidocument technologies. *Proceedings of the IEEE*, 80(7):1066–1078, 1992.

[3] R. G. Casey and E. Lecolinet. A survey of methods and strategies in character segmentation. *IEEE Trans. PAMI*, 18(7):690–706, 1996.

[4] R. C. Gonzalez and R. E. Woods. *Processamento de Imagens Digitais*. Edgard Blücher, 1 edition, 2000.

[5] IMAGO. Projeto linux acessível. Disponível em: http://www.imago.ufpr.br/pt_linuxacessivel.html. Acesso em: 25 ago 08.

[6] O. Martinsky. Algorithmic and mathematical principles of automatic number plate recognition systems. Master's thesis, Brno University of Technology, 2007.

[7] E. R. Oliveira. Avaliação ergonômica de interfaces da scielo - scientific electronic library online. Master's thesis, Universidade Federal de Santa Catarina, 2001.

[8] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. Systems, Man and Cybernetics*, 9(1):62–66, 1979.

[9] J. R. Parker. *Algorithms for Image Processing and Computer Vision*. John Wiley and Sons, 1997.

[10] R. Smith. An overview of the tesseract ocr engine. *Proc. ICDAR*, pages 629–633, 2007.

[11] O. D. Trier, A. K. Jain, and T. Taxt. Feature extraction methods for character recognition - a survey. *Pattern Recognition*, 29(4):641–662, 1996.

[12] S. Wachenfeld, H. U. Klein, S. Fleischer, and X. Jiang. Segmentation of very low resolution screen-rendered text. *Proc. of Int'l. Conf. Document Analysis and Recognition*, pages 1153–1157, 2007.

[13] S. Wachenfeld, H. U. Klein, and X. Jiang. Recognition of screen-rendered text. *Pattern Recognition*, pages 1086–1089, 2006.