Trabalho de Formatura Supervisionado

Seleção Não-Supervisionada de Métodos de Binarização para Documentos Históricos

Denis T. Ikeda

Orientador: Ronaldo Fumio Hashimoto

Departamento de Ciência da Computação
Instituto de Matemática e Estatística
Universidade de São Paulo

Dezembro de 2011

Sumário

Parte I - Parte Técnica....4
Capítulo 1 – Introdução....4
1.1 Apresentação....4
1.2 Aplicação....4
1.3 Definições para todo o documento....6
1.4 O Software de Avaliação....7
Capítulo 2 – Métodos de Binarização Utilizados....8
2.1 Otsu....9
2.1.1 Cálculo....9
2.1.2 Pseudocódigo....9
2.2 Niblack....10
2.2.1 Cálculo....10
2.2.2 Pseudocódigo....10
2.3 Sauvola....11
2.3.1 Cálculo....11
2.3.2 Pseudocódigo....11
2.4 White....12
2.4.1 Cálculo....12
2.4.2 Pseudocódigo....12
2.5 Su....13
2.5.1 Cálculo....13
2.5.2 Pseudocódigo....14
Capítulo 3 – Seleção Proposta....15
3.1 Descrição Inicial....16
3.2 Problemas....16
3.3 Soluções....17
3.4 Descrição Final....18
3.5 Cálculo....18
3.6 Considerações para Implementação....19
Capítulo 4 – Métricas de Avaliação....20
4.1 F-Measure (FM)....21
4.2 Pseudo F-Measure (pFM)....21
4.3 Peak Signal to Noise Ratio (PSNR)....22
4.4 Distance Reciprocal Distortion Metric (DRD)....23
4.5 Negative Rate Metric (NRM)....24
4.6 Misclassification Penalty Metric (MPM)....24
Capítulo 5 – Resultados....25
5.1 Formas de Avaliação....26
5.2. DIBCO 2009....27
5.2.1 Exemplos....28
5.3. H-DIBCO 2010....30
5.3.1 Exemplos....31
5.4. DIBCO 2011 – Métodos Implementados....32
5.4.1 Exemplos....33
5.5. DIBCO 2011 – Simulação....35
5.5.1 Exemplos....37
5.6. DIBCO 2011 – Validação da Avaliação....39
5.7 Implementação do Software....41

Capítulo 6 – Conclusão....42
6.1 Evolução da Seleção....43
6.2 Evolução do software....44
Bibliografia....45
Parte II – Parte Subjetiva....47
Capítulo 1 – Desafios e Frustrações....47
1.1 Em relação ao trabalho realizado....47
1.2 Em relação ao curso....48
Capítulo 2 – Disciplinas Relevantes....49
Agradecimentos....50

Parte I - Parte Técnica

Capítulo 1 – Introdução

1.1 Apresentação

Desde a invenção da escrita, a informação como texto conseguiu um tempo de vida próprio, independente de seu produtor, podendo alcançar leitores muitíssimo distantes no espaço e no tempo. E dentre os meios nos quais foi registrada, o mais importante até hoje continua sendo o papel, fácil de ser produzido, distribuído e alterado. Mesmo com a grande disseminação dos meios digitais nas últimas décadas, o papel está longe de se tornar obsoleto.

Todavia, os defeitos do papel como preservador de informação são vários: ele rasga com facilidade, não podendo ser plenamente reconstituído; mancha ao entrar em contato com uma infinidade de substâncias, podendo molhar e enfraquecer; e sofre desgaste natural com o passar do tempo. Estes defeitos, provenientes do manuseio necessário para ler o papel, tornam difícil preservá-lo, armazená-lo e, ao mesmo tempo, mantê-lo sempre acessível para quem quiser lê-lo.

Uma das soluções atuais para o dilema da preservação em detrimento da acessibilidade é a digitalização do papel. Arquivos de imagem, a princípio, não sofrem degradação temporal ou qualquer dano por manuseio e podem ser facilmente distribuídos através de uma simples cópia de arquivo. Apesar disso, a digitalização está suscetível a defeitos na mídia digital onde está o arquivo, e o processo de digitalização pode ser operado de forma inadequada, comprometendo a qualidade da imagem de forma irreversível até que o papel seja novamente digitalizado corretamente. A primeira vulnerabilidade pode desde acrescentar ruídos à imagem, chamados de artefatos, a inutilizar o arquivo, mas é facilmente resolvida aumentando sua redundância através do *backup* para outras mídias, restaurando arquivos danificados de acordo com a necessidade.

Por causa das vantagens de se ter documentos em arquivos, acervos inteiros são digitalizados e armazenados na melhor qualidade possível. Isso restringe a sua acessibilidade, pois imagens de alta resolução chegam a ocupar 10 Mb, sendo de difícil armazenamento e distribuição. Em geral, também não se pode fazer buscas por arquivos, levando a várias transferências de arquivos enormes em vão. Reduzir a qualidade das imagens pode comprometer o conteúdo, e indexar arquivos manualmente consome muito tempo. Por isso, a solução é binarizá-los.

A binarização transforma a imagem de um documento em texto e fundo, utilizando só as cores branca (para o fundo) e preta (para o texto). Dela, pode-se obter a informação de texto digital (como caracteres *ASCII*, por exemplo) através do *OCR* (*Optical Character Recognition*, ou reconhecimento ótico de caracteres), indexando imagens automaticamente. E sendo só texto e fundo, torna-se mais legível e ocupam menos espaço, chegando a poucas dezenas de Kbs.

1.2 Aplicação

Documentos históricos são particularmente problemáticos. Possuem valor inestimável por sua importância ao capturar a informação de uma época e têm caráter oficial, não podendo simplesmente transcrevê-los de tempos em tempos para evitar o desgaste. E, exatamente por serem importantes, precisam ser lidos sempre e preservados ao máximo. Por estes e outros motivos, como descaso do portador original do documento, eles costumam apresentar diversos defeitos graves ao mesmo tempo.

Alguns desses defeitos mais notáveis são:

1. envelhecimento do papel (Figura 1.2.1)
2. manchas (Figura 1.2.2)
3. iluminação desigual (Figura 1.2.3)
4. dobras (Figura 1.2.4)
5. escrita do verso aparecendo na frente (Figura 1.2.5)
6. pouco contraste entre texto fundo (o fundo escurece ao envelhecer, a tinta se desgasta) (Figura 1.2.6)

Esses defeitos complicam bastante a binarização. Tanto que, nos últimos 3 anos foram realizadas competições de métodos de binarização de documentos históricos, e mesmo apresentando diversos métodos de excelente desempenho, ainda há espaço para melhora.

Um aspecto deficitário na binarização de documentos históricos é a inconsistência dos métodos: para algumas imagens, produzem excelentes resultados; para outras, resultados inaceitáveis em todos as aspectos. Disso, conclui-se que a binarização não oferece a confiabilidade necessária para ser automatizada, pois precisa de supervisão humana para detectar estas eventuais falhas.

O objetivo deste trabalho foi criar uma seleção não-supervisionada de métodos de binarização para documentos históricos, de forma que estes resultados inadequados sejam descartados, podendo aumentar a confiabilidade da binarização ao ponto de tornar viável sua completa automatização. Por outro lado, seria ideal que, além de evitar os piores casos, também pudesse manter a qualidade próxima do melhor possível, sendo melhor que cada método individualmente.

Imagem 1.2.1 Exemplo de papel envelhecido

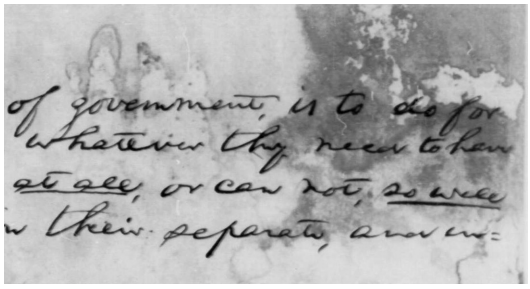
Imagem 1.2.2 Exemplo de manchas

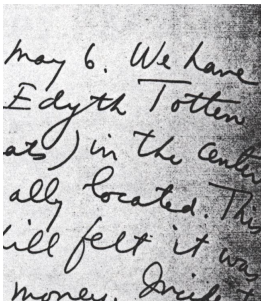
Imagem 1.2.3 Exemplo de iluminação desigual

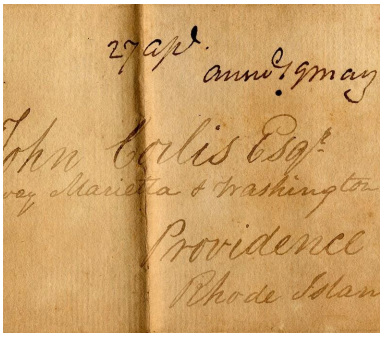
Imagem 1.2.4 Exemplo de dobra

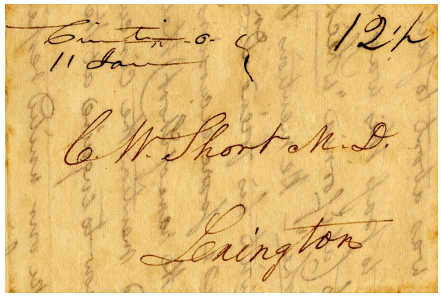
Imagem 1.2.5 Exemplo de verso aparecendo na frente

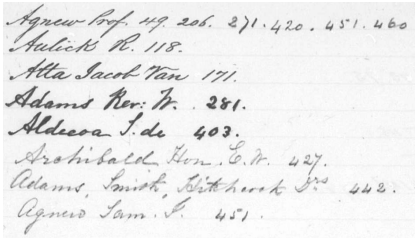
Imagem 1.2.6 Exemplo de pouco contraste

1.3 Definições para todo o documento

Neste documento, serão usados termos e expressões que necessitam de uma definição clara e precisa ou uma breve explicação para os não habituados com o domínio de imagens. As definições de natureza matemática sempre possuem uma versão mais descritiva em linguagem natural, então podem ser puladas sem qualquer risco. Elas são usadas para tornar as definições de cálculo mais precisas. **É recomendado que esta seção seja lida e usada como referência durante a leitura deste texto.**

Um **pixel** (*picture element*, ou **elemento de imagem**) é um quadrado preenchido por uma **intensidade** luminosa, uma cor. O **valor** de um pixel é o valor numérico da intensidade que o preenche. Para este estudo, serão consideradas apenas intensidades em níveis de cinza de 8 bits, i.e., existem 256 tons de cinza, variando de 0 (preto) a 255 (branco). Todas as imagens coloridas terão seus pixels convertidos para este intervalo antes de serem utilizados. Uma imagem costuma ter de dezenas de milhares a milhões de pixels.

Uma **imagem** é uma associação das posições de uma grade de quadrados aos pixels. Mais formalmente, ela é uma função $I : AxB \rightarrow C$, onde $A , B \subset \mathbb{Z}_+$ são as posições válidas da imagem, representadas pelas suas coordenadas horizontal e vertical, e *C* é o conjunto de intensidades. Normalmente, ao invés de usar a notação *I (x, y)*, será dado um conjunto de pontos $W \subseteq AxB$ e a notação usada será $I(w), w \in W$.

Uma **imagem binária** é uma imagem que possui apenas pixels representando as cores preto e branco i.e., seu conjunto *C = {0, 255}*. Para o domínio de imagens de informação textual, geralmente preto é considerada a cor do texto, e branco, a cor do fundo.

Uma **imagem em níveis de cinza** é uma imagem que possui conjunto *C = {0, 1, 2,..., 255}*. Formaliza o que está descrito na definição de pixel.

Binarização ou **limiarização** é o ato de transformar uma imagem em níveis de cinza em uma imagem binária. Neste estudo, este ato preocupa-se em extrair grande parte da informação textual contida na imagem, eliminando os ruídos e o fundo. O nome limiarização vem do fato de muitos métodos de binarização utilizarem um valor de intensidade **limiar**. Os valores dos pixels na imagem abaixo do ou igual ao limiar são classificados como texto; os acima, como fundo.

Binarizar é efetuar uma binarização.

Um **método de binarização** é um algoritmo que realiza uma binarização. Ele é chamado de **paramétrico** caso necessite de algum parâmetro de entrada, além da imagem em níveis de cinza. Se uma pequena mudança em um dado parâmetro k modifica radicalmente o resultado de uma binarização, podendo ir de muito bom a inaceitável e vice versa, ele é dito **sensível ao parâmetro k**. O valor de uma pequena mudança não será definido por ser dependente do contexto, embora seja facilmente observável na prática.

Ground truth *(GT)* é uma imagem binária ideal feita a partir de uma imagem em níveis de cinza. Nela, toda a informação textual é preservada e todos os ruídos são descartados.

Uma **métrica de imagens binárias** avalia uma binarização, atribuindo um valor numérico que indica quão boa foi a binarização. Existem vários tipos de métricas, mas aqui somente métricas que comparam as binarizações com seus respectivos *ground truth* serão utilizadas.

1.4 O Software de Avaliação

Toda vez que um método de binarização é desenvolvido ou avaliado, um novo software precisa ser criado, mesmo que muito simples, para implementar vários métodos (geralmente para compará-los) e criar um sistema de avaliação, seja através de métricas ou uma análise visual do conjunto de dados. Além disso, cada avaliador ou desenvolvedor escolhe seu conjunto de dados e seus métodos de avaliação para obter suas conclusões. Muitas vezes, os resultados experimentais parecem ser surpreendentes, e é desejável saber se eles continuam valendo em um outro domínio. Este problema também está presente em outros domínios, como o de segmentação de vídeo [18].

Por exemplo, na avaliação de Sezgin *et al* [14], ele conclui que o melhor método para binarização de textos é o de Kittler e Illingworth [12], um método global. Mas o conjunto de dados utilizado não possui as características de documentos históricos, por ter sido criado artificialmente, como má qualidade de digitalização, iluminação desigual e manchas. Então para saber se este método também possui excelentes resultados no domínio de documentos históricos, caso o software utilizado não esteja publicamente disponível, será necessário reimplementar todos os métodos, o que por si só pode não garantir os mesmos resultados, como as métricas utilizadas terão de ser reimplementadas. E mesmo se o software estiver disponível, ao imaginar como seria a performance deste método com as métricas do DIBCO 2009, elas também teriam de ser reimplementadas, e se possível, integradas neste software que pode ter sido desenvolvido com as métricas escritas de forma que, ao inserir uma nova métrica, todo software tenha que ser revisado e atualizado.

Uma solução comum para este problema é criar cada módulo como um utilitário de linha de comando independente e integrá-los através de um script. Esta solução resolve parcialmente o problema, aumentando a flexibilidade, mas ainda fica dependente da implementação de cada módulo, podendo não ser portável ou possuir interfaces incompatíveis.

Foi com isso em mente que, ao pensar em criar um novo método, implementar métodos interessantes para comparação e escrever métricas para avaliá-los, não me detive em tentar produzir mais um software simples que só eu conseguiria reutilizar e possivelmente com muito custo. O objetivo é implementar uma métrica ou um método uma única vez, e variar qual conjunto de métricas, métodos e dados serão usados para uma certa avaliação, sem modificar o software. Desta forma, um desenvolvedor poderia utilizar um conjunto de métodos relevantes e métricas já implementados, implementar somente seu método e mostrar seus resultados para um certo conjunto de dados. Para reproduzir a pesquisa com outros dados de outro domínio, bastaria obter a implementação disponibilizada do novo método e modificar os dados da entrada.

Capítulo 2 – Métodos de Binarização Utilizados

Para iniciar o estudo sobre o domínio das técnicas de binarização de imagens de texto, alguns métodos foram estudados e implementados. Estes métodos servirão de base para comparar o novo método proposto, assim como compõe a implementação inicial do software desenvolvido. Nesta seção, eles serão explicados resumidamente para aqueles que não estão familiarizados com o domínio ou com algum método em particular.

Pode-se dividir as técnicas de binarização em duas categorias: global e local. As técnicas globais supõe que o fundo e o texto da imagem inteira estão em faixas diferentes de intensidade, bastando então escolher muito bem um limiar durante a binarização. No entanto, documentos históricos apresentam altos níveis de degradação que violam esta suposição, reduzindo bastante sua eficácia. Por isso, todos os métodos apresentados, exceto um, são locais, que obtêm a binarização da imagem através das informações contidas na vizinhança de cada pixel, e não através das contidas na imagem como um todo. Estas técnicas, chamadas de local adaptativas, são as que possuem melhor desempenho na binarização de documentos [14].

Para algoritmos que utilizam janelas, as janelas geralmente são quadradas de dimensões ímpares, para que o centro esteja bem definido. Além disso, para os pixels próximos à borda da imagem, onde uma janela do tamanho especificado tentaria acessar pixels fora da imagem, não existe abordagem única. Uma solução seria ignorá-los, o que pode não ser desejado caso algum texto esteja próximo à borda da imagem. Outra seria considerar os pixels ausentes como tendo intensidade fixa pré-determinada, ou igual à média dentro da imagem ou janela. Uma terceira opção seria utilizar somente aqueles que fazem parte da imagem na análise, possivelmente utilizando janelas retangulares e de dimensão par para eles, além de quebrar a regularidade da amostragem. Esta última foi a abordagem utilizada na implementação destes métodos, por poder tratar imagens e janelas de forma uniforme.

Considere uma janela de pontos W centrada no ponto $p \in W$:

- $m(p) = \frac{\sum_{w \in W} I(w)}{|W|}$ é a média da janela,
- $s(p) = \frac{\sum_{w \in W} [I(w) - m(p)]^2}{|W|}$ é o desvio padrão da janela,
- $T(p)$ é o limiar da janela

2.1 Otsu

O método de Otsu [5] é único método global a ser apresentado, por ser o mais conhecido dos métodos de binarização e por obter resultados muito bons, exceto quando há manchas e iluminação desigual, não havendo duas classes distintas de intensidade para toda a imagem.

Ele supõe que a imagem pode ser dividida em duas classes, *C0* e *C1*, uma do texto e outra fundo, ao escolher um limiar *k* entre as duas. Então, para que esta escolha seja não paramétrica, ele define três medidas equivalentes *(λ, κ e η)* que determinam quão boa é a escolha do limiar *k*, transformando o problema da escolha de *k* num problema de otimização da medida dentre as três que possui cálculo mais simples. Em outras palavras, a intensidade que maximiza esta medida será o limiar.

2.1.1 Cálculo

Um histograma de uma imagem é uma função que leva uma intensidade em níveis de cinza ao número de pixels na imagem que possuem aquela intensidade, a frequência da intensidade. Ao dividir cada frequência pelo número total de pixels, é obtido um histograma normalizado, que representa uma distribuição de probabilidades de cada nível de cinza.

Considere esta distribuição de probabilidades, i.e., os p_i, onde i é uma intensidade, cujo valor é o número de pixels de intensidade i dividido pelo número total de pixels. Além disso, para este método, as intensidades i vão de 1 a L, como no *paper* original. São definidas, então, as somas parciais acumuladas de ordem zero e um, respectivamente, de uma dada intensidade k:

$$\omega(k) = \sum_{i=1}^{k} p_i \ , \qquad \mu(k) = \sum_{i=1}^{k} i \cdot p_i$$

As somas totais são:

$$\omega(L) = \sum_{i=1}^{L} p_i = 1 \ , \qquad \mu_T = \mu(L) = \sum_{i=1}^{L} i \cdot p_i$$

A medida mais simples adotada é a η, a variância interclasse, cuja maximização depende só de:

$$\sigma_B^2(k) = \frac{[\mu_T \cdot \omega(k) - \mu(k)]^2}{\omega(k) \cdot [1 - \omega(k)]}$$

Portanto, o valor k^* usado como limiar é:

$$\sigma_B^2(k^*) = \max_{1 \le k < L} \sigma_B^2(k)$$

2.1.2 Pseudocódigo

1. Obtenha a frequência de cada intensidade $1 \le i \le L$ (histograma da imagem)
2. Divida cada frequência pelo número total de pixels, obtendo as probabilidades (normalização do histograma)
3. Obtenha as somas parciais acumuladas ω e μ de cada intensidade através das probabilidades
4. Para cada intensidade k, calcule $\sigma_B^2(k)$. Se este for o maior valor encontrado, atualize o limiar k^* com k.
5. Para cada pixel, se a intensidade do pixel for menor que ou igual a k^*, ele será considerado texto; caso contrário, fundo.

2.2 Niblack

O método de Niblack [6] foi um dos primeiros a se destacar entre os locais adaptativos, sendo utilizado como base para vários outros métodos da mesma espécie. Ele consegue extrair muito bem a informação textual de sua vizinhança, mas necessita de um pós-processamento, pois boa parte do fundo também é reconhecido erroneamente como texto [13]. Além disso, ele é paramétrico, sendo muito sensível ao parâmetro de peso.

Este método é bem simples: para cada pixel da imagem, é definida uma janela de comprimento e largura n centrada neste pixel, e são calculados a média e o desvio padrão desta janela. Se o valor do pixel for menor que o limiar, obtido da média, do desvio padrão e do parâmetro peso, ele é considerado texto; se for maior, fundo.

2.2.1 Cálculo

Para este método, a janela W é quadrada n por n e seja k o parâmetro peso do desvio padrão. O limiar desta janela é:

$$T(p) = m(p) + k \cdot s(p)$$

2.2.2 Pseudocódigo

1. Para cada pixel:
    1. Defina uma janela centrada neste pixel
    2. Calcule a média e o desvio padrão da janela
    3. Calcule o limiar a partir da média, do desvio padrão e do parâmetro peso
    4. Se a intensidade do pixel for menor que o limiar, ele será considerado texto; caso contrário, fundo

2.3 Sauvola

O método de Sauvola [7] é considerado um dos melhores métodos adaptativos clássicos para documentos [14]. Ele é uma adaptação do método de Niblack. O paper de Sauvola *et al.* também define a binarização de elementos não-textuais de uma imagem, mas esta parte não será considerada neste estudo.

A diferença deste método para o de Niblack está na forma do cálculo do limiar, que utiliza um segundo parâmetro: a faixa dinâmica do desvio padrão. Ao reorganizar a fórmula e adicionar um novo parâmetro, ele diminui a sensibilidade em relação ao parâmetro peso, mas se torna sensível ao parâmetro faixa.

2.3.1 Cálculo

Para este método, a janela W é quadrada n por n e sejam k e R os parâmetros peso e faixa dinâmica do desvio padrão, respectivamente. O limiar desta janela é:

$$T(p) = m(p) \cdot [1 + k \cdot (\frac{s(p)}{R} - 1)]$$

2.3.2 Pseudocódigo

1. Para cada pixel:
    1. Defina uma janela centrada neste pixel
    2. Calcule a média e o desvio padrão da janela
    3. Calcule o limiar a partir da média, do desvio padrão e dos parâmetros
    4. Se a intensidade do pixel for menor que o limiar, ele será considerado texto; caso contrário, fundo

2.4 White

O método de White [8] é mais simples que os dois anteriores, não possuindo igual performance ou destaque. Mas, por ser bastante sensível ao seu parâmetro de viés, pode prover excelentes resultados para um dado valor de viés para uma certa imagem.

Semelhante ao Niblack, com um parâmetro viés no lugar do peso, calcula um limiar através da média da janela, mas não utiliza o desvio padrão. Esta descrição está mais de acordo com a de Sezgin *et al* [14], por ser mais simples do que a original.

2.4.1 Cálculo

Para este método, a janela W é quadrada n por n e seja $bias$ o viés da média. O limiar desta janela é:

$$T(p) = \frac{m(p)}{bias}$$

(a descrição em [14] não define um valor para o limiar, define uma função binária que vale 1 se $m(p) < I(p) \cdot bias$):

2.4.2 Pseudocódigo

1. Para cada pixel:
    1. Defina uma janela centrada neste pixel
    2. Calcule a média desta janela
    3. Calcule o limiar a partir da média e do parâmetro viés
    4. Se a intensidade do pixel for menor que o limiar, ele será considerado texto; caso contrário, fundo

2.5 Su

O método de Su [9] foi um dos primeiro colocados do H-DIBCO 2010 (houve um empate), dos 17 algoritmos participantes vindos de 16 instituições.

Ele é dividido em duas partes:

1. Antes de começar a binarização propriamente dita, ele detecta os pixels de alto de contraste, através da binarização da imagem de contraste baseada no máximo e mínimo locais. O método de Otsu é utilizado porque a imagem de contraste possui duas classes distintas bem definidas, mesmo que a imagem original não tenha.
2. Com os pixels de alto contraste, binariza a imagem original utilizando um limiar baseado na média e no desvio padrão de uma janela, como no Niblack, mas somente utiliza para o cálculo destes valores os pixels de alto contraste, i.e., os pixels de texto da imagem de alto contraste construída no passo anterior. Além disso, se o número de pixels de alto contraste da janela centrado num certo pixel for menor que o parâmetro *Nmin* dado, este pixel é imediatamente considerado fundo.

Este algoritmo utiliza dois parâmetros importantes: o tamanho da janela e o número de pixels de alto contraste mínimo numa janela *Nmin*. Tais parâmetros são derivados da estimativa da grossura do traço do texto, utilizando a distância mais frequente entre picos de máximo contraste ao percorrer a imagem linha a linha. Esta estimativa não foi implementada.

2.5.1 Cálculo

No primeiro passo, a imagem de contraste é construída achando o máximo e o mínimo numa janela pequena (3x3 é o utilizado originalmente) e calculando o contraste da seguinte forma:

$$D(p) = \frac{f_{max}(p) - f_{min}(p)}{f_{max}(p) + f_{min}(p) + \epsilon}$$

$f_{max}(p)$ e $f_{min}(p)$ são as intensidades máxima e mínima, respectivamente, da janela da imagem original em torno do ponto p, $D(p)$ é a intensidade do pixel no ponto p da imagem de contraste e ε é um valor positivo infinitamente pequeno adicionado para o caso de o máximo ser zero.

Seja W o conjunto de pontos da janela n por n centrado no ponto $p \in W$, e $W' \subseteq W$ o conjunto dos pontos de W tais que exista um pixel de alto contraste correspondente, i.e., seja I_c a imagem binária dos pixels de alto contraste, então $W' = \{w \in W : I_c(w) = 0\}$. Logo:

$$m(p) = \frac{\sum_{w \in W'} I(w)}{|W'|} \quad , \quad s(p) = \frac{\sum_{w \in W'} [I(w) - m(p)]^2}{|W'|}$$

$$T(p) = m(p) + \frac{s(p)}{2}$$

Lembrando que, se $|W'| < Nmin$, o pixel no ponto p é considerado fundo sem avaliar o limiar.

No texto original, $|W'| = Ne,\ m(p) = E_{mean},\ s(p) = E_{std}$, e, ao invés de definir $T(p)$, é definida uma função binária $R(x, y)$ que engloba a condição do *Nmin* e a do limiar.

A estimativa dos parâmetros não foi inteiramente compreendida para poder ser explicada aqui, tampouco foi implementada pelo mesmo motivo.

2.5.2 Pseudocódigo

1. Para cada pixel:
    1. Defina uma janela 3x3 em torno dele
    2. Calcule as intensidades máxima e mínima da janela
    3. Atribua o valor do contraste ao pixel correspondente na imagem de contraste
2. Após a construção da imagem de contraste, estime *Nmin* e o *n*, o tamanho da janela
3. Binarize a imagem de contraste através do método de Otsu
4. Para cada pixel:
    1. Defina uma janela *n* por *n* em torno dele
    2. Calcule $|W'|$
    3. Se $|W'| < Nmin$, o pixel é considerado fundo
    4. Calcule a média e o desvio padrão como citado acima
    5. Se a intensidade do pixel for menor que o limiar, ele será considerado texto; caso contrário, fundo

Capítulo 3 – Seleção Proposta

As competições DIBCO [2][3][4] dos últimos anos mostraram que ainda há interesse na pesquisa da elaboração de novos métodos de binarização. Além disso, os métodos apresentados de alto desempenho costumam ser bastante intrincados, possuindo várias etapas, e cada etapa possui sua própria complexidade. Por isso, embora durante este estudo fora tentado criar um novo método, nenhum com resultado equiparável aos melhores lugares pôde ser obtido.

Além disso, foi percebido no DIBCO 2011, o primeiro a fornecer resultados por imagem e disponibilizar as imagens dos métodos, que mesmo os primeiros lugares são inconsistentes, com resultados indo de excelente para mediano e às vezes até inaceitável. Isto faz da binarização uma tarefa supervisionada, pois tais resultados inaceitáveis não teriam como ser detectados automaticamente sem que haja uma binarização ideal criada manualmente (o *ground truth*, explorado no próximo capítulo).

Então, como existem métodos excelentes e muito interesse em melhorá-los, mas nenhum deles individualmente é confiável, é proposta uma seleção não-supervisionada de métodos de binarização, visando aumentar a confiabilidade e consistência da binarização ao utilizar sempre o melhor método para cada imagem. Como cada aplicação tem sua definição de melhor (mais legível, mais palavras detectadas, menos ruído...), muitas vezes conflitantes ou subjetivas, ao invés de adotar uma ou outra definição de melhor, a seleção tenta sempre obter resultados aceitáveis, mesmo que outro resultado possa ser considerado melhor para uma dada imagem.

A aceitabilidade de um resultado não será definida formalmente, mas, diferentemente da superioridade, um resultado geralmente é aceitável na maioria dos contextos ao mesmo tempo. De outra forma, a aceitabilidade é facilmente avaliada visualmente ao comparar o resultado somente com a imagem original, sem a necessidade de consultar outras imagens.

Vale ressaltar que está sendo proposta uma **seleção**, não uma **combinação** de métodos. Existem várias combinações na literatura, como a de Su [10]. **Não foram encontradas outras seleções.** Esta seleção pode ser usada, inclusive, para escolher uma dentre várias combinações de métodos diferentes, o que poderia aumentar ainda mais a confiabilidade da binarização, uma vez que combinações tendem a suavizar os problemas de cada método, podendo ser melhor que cada um isoladamente. Entretanto, interações como esta não foram testadas para este estudo.

Neste capítulo, a seleção será mostrada como inicialmente concebida, quais problemas há nesta concepção, como foram resolvidos, e a versão final dela. Em seguida, os cálculos necessários serão explorados em detalhes, e alguns detalhes da implementação serão discutidos.

3.1 Descrição Inicial

A seleção foi elaborada inicialmente da seguinte forma:

1. Os métodos do conjunto para escolha são executados, e seus resultados guardados
2. Os resultados são avaliados através de um critério que os ordena de forma crescente
3. O resultado na primeira posição é escolhido

Como critério de ordenação, foram utilizadas duas métricas muito comuns nesta área: *Precision* e *Recall*. Porém, estas métricas são geralmente calculadas em conjunto com *ground truth*, e como sua criação é manual, não pode ser usado se é almejada uma completa automatização da binarização. Por isso, elas são estimadas da forma como apresentado por Lamiroy e Sun [11].

Para simplificar as definições, um pixel está correto se ele for de texto no resultado correto, e um pixel está num resultado se ele é de texto no resultado. Logo, os casos se dividem em um pixel estar em um resultado ou estar correto. Para tornar o texto ainda mais conciso, será dito simplesmente pixel correto ou pixel de um resultado.

Adaptando o contexto de obtenção de informação para imagens, a estimativa interpreta *Precision* como a probabilidade de um pixel de um dado resultado estar correto; e *Recall*, de um pixel correto estar em um dado resultado. Cada resultado de um método de binarização seria um possível resultado correto. Além deles, são introduzidos dois resultados artificiais, que correspondem a uma imagem completamente branca e outra completamente preta.

As duas medidas dependem do cálculo da probabilidade de um pixel estar correto. Ao assumir que todos os métodos são igualmente confiáveis, e que todos os pixels têm mesma chance de estarem corretos, esta probabilidade corresponde ao número de métodos nos quais um pixel está dividido pelo número total de métodos (incluindo os artificiais). Desta forma, tanto *Precision* e quanto *Recall* podem ser calculados diretamente ou através de probabilidade condicional.

Para ordenar os resultados, as medidas são combinadas através de sua média harmônica, formando o chamado *F-Measure*. Portanto, o resultado de maior *F-Measure* é considerado ser o mais correto de acordo com a estimativa, ou simplesmente o mais aceitável.

3.2 Problemas

A abordagem descrita foi implementada, mas seu desempenho foi terrível, pois geralmente as imagens com muito ruído ou com muito texto faltando eram escolhidas pela seleção. Ao avaliar os números com cuidado, foram feitas algumas observações:

1. *Precision* atinge probabilidade 1 quando todos os pixels do resultado forem corretos, logo seu valor é limitado pelo número de pixels de cada resultado; já *Recall*, quando todos os pixels corretos estiverem no resultado, sendo limitado pelo número de pixels corretos, cuja estimativa depende de todos os resultados ao mesmo tempo.
2. Por *F-Measure* ser a média harmônica das métricas, valores pouco abaixo da média de *Precision* são compensados por altos valores de *Recall*, e vice versa.
3. A hipótese de igual frequência dos pixels de texto não serve para documentos, pois geralmente boa parte da imagem é fundo.

Observações 1 e 2 implicam em um grande problema: a presença de uma pequena parcela de resultados com muito ruído aumenta significativamente os valores estimados de *Recall*, mas *Precision* diminui relativamente pouco, o que torna o *F-Measure* destes resultados maior que os dos demais. O contrário vale para imagens com muito texto faltando: altíssimo *Precision*, baixo *Recall*, mas *F-Measure* alto. Dos dois, o primeiro caso é bem mais frequente.

A observação 3 também contribui para imagens com ruído, pois os pixels do fundo, que são a maioria, acabam tendo uma probabilidade significativa de serem considerados como texto, principalmente por causa da imagem preta introduzida.

3.3 Soluções

Para o primeiro problema encontrado, a solução foi filtrar as imagens que possuem *Recall* muito diferente da maioria. Para isto, é definido um intervalo aceitável dos valores, centrado na média μ dos valores e tamanho igual a duas vezes o desvio padrão σ^2, i.e., vai de $\mu - \sigma^2$ a $\mu + \sigma^2$. A imagem mais longe do intervalo aceitável, acima ou abaixo, é retirada do conjunto, e as métricas são recalculadas, definindo um novo intervalo aceitável. Este procedimento é feito até que todas as imagens estejam dentro do intervalo aceitável atual, prosseguindo para a etapa de ordenação.

Tentou-se ampliar e reduzir o intervalo, assim como alterar o cálculo do *F-Measure*, dando pesos diferentes para uma ou outra métrica, e até mesmo mudando para média aritmética, mas nenhuma variação obteve melhor resultados do que o intervalo inicial e o *F-Measure* padrão.

Já o segundo problema foi proveniente de um erro de modelagem, então foi necessário utilizar um modelo mais parecido com o de imagens de documentos: os pixels de texto são bem menos frequentes que os de fundo. Pensou-se em duas maneiras de corrigir a modelagem: utilizar outro modelo estatístico, não-uniforme, ou alterar a estimativa inicial, que são as imagens artificiais. A primeira opção, além de complicar o processo, geralmente depende de hipóteses restritivas que não se aplicam a este domínio. Portanto, foi adotada a segunda opção.

Por os pixels de fundo serem mais frequentes que os de texto, a imagem branca funciona bem, gerando uma pequena incerteza para todos a imagem. Resta alterar a imagem preta. Várias estratégias foram adotadas para substituí-la: simplesmente diminuir o peso dela, transformando-a numa imagem cinza uniforme; utilizar algum tipo de mapa de probabilidades, que dê alta probabilidade para o texto e pouca para o fundo; e uma mistura das dois, que seria uma máscara de probabilidade uniforme para os prováveis pixels de texto.

Como o contraste identifica bem os pixels de texto, este valor foi usado como probabilidade do mapa, formando mapas de contraste. As funções de contraste foram a de máximo e mínimo, do método de Su [9], e a de máximo [10]. As máscaras usadas são a binarização destes mapas.

3.4 Descrição Final

A seleção como é proposta atualmente funciona desta forma:

1. Calcula-se a estratégia de estimativa inicial de texto (homogêneo, mapa ou máscara de contraste)
2. Os métodos do conjunto para escolha são executados, e seus resultados guardados
3. Os resultados são avaliados através de um critério que os ordena de forma crescente
4. Caso exista alguma imagem fora do intervalo aceitável,
    1. A mais distante é retirada do conjunto e volta para o passo 3
5. O resultado na primeira posição é escolhido

3.5 Cálculo

Para a seleção, é útil considerar o conjunto de valores dos pixels $C = \{x \in \mathbb{R} \mid 0 \leq x \leq 1\}$, com 0 para branco (fundo) e 1 para preto (texto). Logo, imagens binárias só possuem uns e zeros.

Seja I_0 a estimativa inicial das probabilidades por alguma estratégia citada, $I_k, 1 \leq k \leq n,$ os resultados dos métodos de binarização do conjunto para escolha, e $I_{n+1}(p) = 0, \forall\, p \in AxB$ a imagem branca. Então, a probabilidade de um pixel no ponto *p* estar correto é:

$$P(p) = \frac{1}{n+2} \sum_{k=0}^{n+1} I_k(p)$$

Lembrando que foi suposto que todos os métodos são igualmente confiáveis.

Precision é a probabilidade de um pixel de um resultado dado estar correto. Para cada documento, então, é calculado:

$$Precision(I_k) = \frac{\sum_{p \in AxB} P(p) I_k(p)}{\sum_{p \in AxB} I_k(p)}, k = 1, \ldots, n$$

Recall é a probabilidade de um pixel correto estar num resultado dado. Logo:

$$Recall(I_k) = \frac{\sum_{p \in AxB} P(p) I_k(p)}{\sum_{p \in AxB} P(p)}, k = 1, \ldots, n$$

F-Measure é simplesmente é a média harmônica das duas:

$$F - Measure(I_k) = \frac{2\, Precision(I_k) Recall(I_k)}{Precision(I_k) + Recall(I_k)}, k = 1, \ldots, n$$

Se a definição de *Precision* for alterada para a probabilidade de um pixel estar correto dado que está num certo resultado, e *Recall* como a probabilidade de um pixel estar num certo resultado dado que está correto, então um pode ser escrito em função do outro através de probabilidade condicional juntamente com o teorema de Bayes.

A estimativa inicial homogênea simplesmente define $I_0(p) = k$, $k \in \mathbb{R}$, $0 \leq k \leq 1$. Já os mapas de contraste utilizam as funções de contraste:

- do máximo:

$$I_0(p) = \frac{f_{max}(p) - I(p)}{f_{max}(p) + \varepsilon}$$

- dos máximo e mínimo:

$$I_0(p) = \frac{f_{max}(p) - f_{min}(p)}{f_{max}(p) + f_{min}(p) + \varepsilon}$$

Para as funções, $f_{max}(p)$ e $f_{min}(p)$ são as intensidades máxima e mínima, respectivamente, da janela da imagem original em torno do ponto p, $I(p)$ é a intensidade da imagem original no ponto p e ε é um valor positivo infinitamente pequeno adicionado para o caso de o máximo ser zero.

A função de máximo e mínimo é a mesma do método de Su, do capítulo 2, mas lá o valor do contraste é mapeado para uma intensidade válida de pixel.

No caso das máscaras, como a imagem derivada do mapa de contraste possui duas faixas de intensidade bem distintas, ela pode ser binarizada pelo método de Otsu. As imagens binarizadas precisam ser invertidas, no entanto, pois os maiores contrastes são da parte do texto, que é mapeada para a menor intensidade, a cor preta. Os pontos mascarados têm todos uma probabilidade fixa dada, como na estratégia homogênea.

3.6 Considerações para Implementação

A ordenação descrita não precisa ser feita de forma explícita, bastando escolher a imagem de maior *F-Measure*, o que representa uma redução de um fator logarítmico no tamanho do conjunto de imagens. Para conjuntos pequenos, esta alteração não surtirá qualquer efeito observável.

Esta seleção é paramétrica, pois precisa receber uma estratégia que define a estimativa inicial dos resultados I_0, junto com seus possíveis parâmetros. No caso da estratégia homogênea, o parâmetro é um valor de 0 a 1; para os mapas, o tamanho da janela para calcular o contraste; e no caso híbrido, os dois.

Implementada de forma trivial, esta seleção leva tempo proporcional à soma dos tempos de execução de cada método, que pode ser demasiadamente longo para alguns métodos demorados ou uma grande quantidade de métodos. Este problema pode ser aliviado através da paralelização da execução dos métodos, uma vez que cada método é independente de cada outro, pois só leem uma imagem de entrada e devolvem uma outra nova, binarizada. Esta independência pode ser garantida caso a imagem de entrada seja imutável. Desta forma, a seleção passa a levar tempo proporcional ao tempo do método mais demorado, o que é escalável.

Capítulo 4 – Métricas de Avaliação

As métricas a ser apresentadas nesta seção comparam os resultados dos métodos de binarização com seus respectivos *ground truth*. O GT não pode ser obtido por um processo completamente automatizado, pois se tal processo existisse, então ele mesmo seria o método de binarização perfeito. Por outro lado, a criação manual desta imagem é, além de bastante demorada, suscetível a falhas humanas. Portanto, um processo semiautomático foi desenvolvido para as competições DIBCO, em que a intervenção humana ocorre de forma bastante controlada. Este processo é parcialmente explicado em [1].

Existem outras formas de avaliação, como a manual, em que uma pontuação é dada por um avaliador treinado que verifica a imagem original e a imagem binarizada, mas esta é considerada pouco consistente (uma mesma pessoa pode dar pontuações diferentes para o mesmo par original-binarizado em momentos diferentes), além de ser lenta. Existe também a avaliação através de OCR, que verifica a quantidade de palavras ou caracteres corretos. Como o OCR não funciona bem com texto manuscrito, estas métricas só podem ser usadas em documentos históricos impressos [1].

Algumas das métricas fazem uso dos seguintes conceitos:

True Positive (TP): é um pixel de texto correto

True Negative (TN): é um pixel de fundo correto

False Positive (FP): é um pixel de texto errado

False Negative (FN): é um pixel de fundo errado

Um pixel da imagem binarizada é considerado **correto** se ele for igual ao pixel correspondente do *ground truth*; **errado**, se for diferente.

Estas siglas nas formúlas denotam o número de pixels em cada categoria. Note que elas formam uma partição do conjunto de pixels da imagem, portanto a soma do números de pixels de cada categoria é igual ao número de pixels da imagem inteira.

4.1 F-Measure (FM)

Presente em: DIBCO 2009, H-DIBCO 2010, DIBCO 2011

Esta métrica [15] é a média harmônica de duas medidas bastante comuns: *Precision* e *Recall*. *Precision* representa quanto do texto presente na imagem binarizada é correto, medindo a qualidade do resultado; e *Recall* representa quanto do texto do *ground truth* está na imagem binarizada, medindo a completude do resultado. *Recall* também é chamado de *Sensitivity*; e *F-Measure*, de *F-Score*.

Esta métrica difere da utilizada pela seleção ao utilizar o *ground truth*.

O cálculo é feito da seguinte forma:

$$Precision = \frac{TP}{TP + FP} \quad , \qquad Recall = \frac{TP}{TP + FN}$$

$$F\text{-}Measure = \frac{2 * Recall * Precision}{Recall + Precision}$$

Ela representa uma porcentagem, *i.e.*, está no intervalo real de 0 a 1, e quanto maior for seu valor, mais próximo do GT é a imagem binarizada.

4.2 Pseudo F-Measure (pFM)

Presente em: H-DIBCO 2010

Semelhante ao *F-Measure*, mas, ao invés de utilizar o *Recall*, utiliza o chamado *Pseudo Recall*, que nada mais é do que o *Recall* considerando o GT esqueletizada (no original, *skeletonized ground truth*). Esta imagem, definida em [1], é a silhueta de um pixel de largura que define o conteúdo textual, e é obtida a partir da imagem de teste através da binarização, esqueletização e correção manual das pequenas falhas restantes.

Esta métrica [3] diferencia-se das demais por necessitar de 2 imagens padrão para avaliar o resultado de uma binarização: o GT e o GT esqueletizado. Como esta métrica não foi utilizada no DIBCO 2009 e 2011, não é possível utilizá-la em seus conjuntos de dados a não ser que o GT esqueletizado seja obtida do GT, o que produz resultados bem diferentes de uma silhueta corrigida manualmente.

Seu cálculo é igual ao *F-Measure* em todos os outros aspectos: calcula-se o *Precision* com o GT, o *Recall* com o GT esqueletizado, que é o *Pseudo Recall*, e o resultado é a média harmônica das duas como no *F-Measure*, possuindo também mesmo significado.

4.3 Peak Signal to Noise Ratio (PSNR)

Presente em: DIBCO 2009, H-DIBCO 2010, DIBCO 2011

Esta é a medida de proximidade entre o resultado e o GT [16]. Significa **razão entre sinal de pico e ruído**, e utiliza o cálculo do *MSE* (*mean square error*, ou **erro quadrático médio**). Seu cálculo, como originalmente definido, é da seguinte forma, com I sendo a imagem binarizada, I' o GT, *M* e *N* suas dimensões e *C* a diferença entre texto e o fundo:

$$MSE = \frac{\sum_{x=1}^{M}\sum_{y=1}^{N}(I(x,y)-I'(x,y))^2}{M \cdot N}$$

$$PSNR = 10 \cdot \log\left(\frac{C^2}{MSE}\right)$$

Entretanto, como as imagens em questão são binárias, as seguintes simplificações podem ser feitas:

- Como a diferença ao quadrado entre o GT e a imagem binarizada é zero se os pixels forem iguais (TP e TN) ou C^2 se forem diferentes (FP e FN), pode-se substituir o somatório das diferenças entre as imagens por $(FP + FN) \cdot C^2$.
- *M* vezes *N* é o número de pixels, que pode ser expresso como *TP* + *FP* + *TN* + *FN*, pois cada categoria é uma partição da imagem.
- Como C^2 aparece no numerador e no denominador do logaritmo, ele pode ser eliminado

Portanto:

$$PSNR = 10 \cdot \log\left(\frac{TP + TN + FP + FN}{FP + FN}\right)$$

O *PSNR* tenta mensurar a distorção da imagem ponto a ponto, não levando em conta a relações mútuas entre pixels, como a medida seguinte, *DRD*. De forma semelhante ao *F-Measure*, quanto maior o valor do *PSNR*, mais parecidas são as imagens. Note que, caso as imagens sejam idênticas, seu valor é ilimitado.

4.4 Distance Reciprocal Distortion Metric (DRD)

Presente em: DIBCO 2011

Esta métrica [17] tenta medir a distorção visual como é percebida pela visão humana. Para isso, utiliza como peso a inversa da distância dos pixels errados (*FP* ou *FN*) em torno de um erro, pois a percepção de distorção aumenta quanto mais perto do foco for um erro do outro. Estes pesos são representados em uma matriz quadrada de dimensão ímpar m, W_m, definida da seguinte forma, com $1 \le i, j \le m$ e $i_C = j_C = \frac{(m+1)}{2}$ os índices da posição central da matriz:

$$W_m(i,j) = \begin{cases} 0, & se\ i = i_C\ e\ j = j_C \\ \frac{1}{\sqrt{(i-i_C)^2 + (j-j_C)^2}}, & caso\ contrário \end{cases}$$

Esta matriz é então normalizada dividindo cada elemento seu pela soma de todos os elementos:

$$W_{Nm}(i,j) = \frac{W_m(i,j)}{\sum_{i=1}^{m}\sum_{j=1}^{m} W_m(i,j)}$$

Seja *S* o conjunto dos pixels errados (*FP* ou *FN*), e DRD_k a distorção do k-ésimo pixel em $S, 1 \le k \le |S|$, calculada somando as diferenças entre o *GT* e a imagem binarizada de cada pixel na janela W_k de tamanho *m* por *m* centrada neste pixel, multiplicando-se pelo peso correspondente da matriz W_{Nm}. Como as imagens são binárias, somente serão somadas as diferenças dos *FP* e *FN*, e como elas são sempre iguais a *C* (como no *PSNR*), podem ser somadas somente no fim, reduzindo DRD_k à soma dos pesos da matriz W_{Nm} que corresponde aos pixels em $W_k \cap S$ multiplicada por *C*.

Portanto, o valor final é, com *NUBN* o número de blocos 8 x 8 não-uniformes da imagem (que não são ou só brancos ou só pretos):

$$DRD = \frac{\sum_{k=1}^{|S|} DRD_k}{NUBN}$$

Para o DIBCO 2011, $m = 5$. Ao contrário das métricas anteriores, quanto menor for o valor desta métrica, melhor será a qualidade da binarização.

A implementação desta métrica obtém valores diferentes dos publicados em [4]. A causa desta diferença não foi encontrada, e infelizmente os valores obtidos em [17], onde ela é definida, não puderam ser reproduzidos por não haver acesso às mesmas imagens.

4.5 Negative Rate Metric (NRM)

Presente em: DIBCO 2009, H-DIBCO 2010

Esta métrica [18] baseia-se nas diferenças pixel a pixel entre o resultado e o GT. Ela é a média aritmética da taxa de falso negativos NR_{FN} com a taxa de falso positivos NR_{FP}. Calcula-se:

$$NR_{FN} = \frac{FN}{FN + TP} \quad , \quad NR_{FP} = \frac{FP}{FP + TN}$$

$$NRM = \frac{NR_{FN} + NR_{FP}}{2}$$

Assim como a DRD, quanto menor seu valor, melhor o resultado. Esta e a métrica seguinte são encontradas na área de segmentação de vídeo.

4.6 Misclassification Penalty Metric (MPM)

Presente em: DIBCO 2009, H-DIBCO 2010, DIBCO 2011

Diferente das demais métricas, esta [19] utiliza as distâncias dos pixels até a borda do texto do *ground truth*. Primeiro, é extraída uma imagem com a borda do texto do GT e dela é obtida sua transformada distância. Então, calcula-se os três somatórios:

- $\sum_{i=1}^{FN} d_{FN}^{i}$ é soma das distância dos pixels falso negativos até a borda mais próxima,
- $\sum_{j=1}^{FP} d_{FP}^{j}$ é soma das distância dos pixels falso positivos até a borda mais próxima,
- D é a soma das distâncias de todos os pixels da imagem, chamado de fator de normalização.

Por fim, calcula-se:

$$MP_{FN} = \frac{\sum_{i=1}^{FN} d_{FN}^{i}}{D} \quad , \quad MP_{FP} = \frac{\sum_{j=1}^{FP} d_{FP}^{j}}{D}$$

$$MPM = \frac{MP_{FN} + MP_{FP}}{2}$$

Um baixo valor desta métrica indica que um método identifica bem a borda do texto. Ela é baseada no *Misclassification Penalty* definido em [20], que utiliza a transformada distância Chamfer e une as categorias FP e FN, juntando os dois somatórios sem tirar a média.

Capítulo 5 – Resultados

A seleção proposta foi testada com os conjuntos de dados das 3 competições DIBCO realizadas [2][3][4], com suas respectivas métricas. Como concorrentes, foram utilizados os métodos descritos no Capítulo 2 (Otsu, Niblack, Sauvola, White e Su) e os 18 métodos participantes da competição do DIBCO 2011. Estes últimos puderam ser simulados pois seus resultados nesta competição foram disponibilizados publicamente, o que não ocorreu nos anos anteriores, portanto não há como simulá-los da mesma forma. A seleção escolhe um resultado dentre os de seus concorrentes, para avaliar melhor os resultados. Na prática, o conjunto de métodos seria selecionado minuciosamente e suas interações testadas para cada parâmetro até obter bons resultados.

Como a seleção é paramétrica, foram escolhidos 5 parâmetros para testá-la, cada um representando uma estratégia. Geralmente, a introdução de um método altera as posições dos demais métodos, como poderá ser visto a seguir, mas como a seleção sempre escolhe uma imagem existente, ela empata com alguma outra imagem, não alterando as pontuações dos métodos restantes ao considerar a pontuação por imagem. Por este motivo, os resultados podem ser expressos de maneira compacta comparando os 5 representantes da seleção ao mesmo tempo sem perda de informação (exceto por trocas de lugar de métodos com pontuações próximas na avaliação pela média, mas estas serão desconsideradas). Cada representante será da seguinte forma:

- P: hom, para a homogênea com parâmetro 0,5
- P: map max, para o mapa de contraste do máximo de janela 15
- P: map mmin, para o mapa de contraste dos máximo e mínimo de janela 15
- P: bin max, para a máscara de contraste do máximo de janela 15 e parâmetro 1
- P: bin mmin, para a máscara de contraste dos máximo e mínimo de janela 15 e parâmetro 1

Além das avaliações, foi feita uma simulação somente com os métodos do competição de 2011, com o intuito de validar as métricas e a avaliação feita pelo software. Esta validação foi útil, pois mostra que o software está quase de acordo com o padrão estabelecido pela competição.

Os métodos tiveram parâmetros:

- Otsu (não tem)
- Niblack: janela 15; k = -0,2 (recomendados)
- Sauvola: janela 15; k = 0,5 ; R = 128 (recomendados)
- White: janela 15; bias = 2 (alterado para melhor desempenho)
- Su: janela 15; N_{min} = 8; janela de contraste 3 (deveria ser estimado, contraste recomendado)

Este capítulo começa com as formas de avaliação, mostrando em seguida os resultados para cada conjunto de dados e competidores. As conclusões serão feitas somente no próximo capítulo.

5.1 Formas de Avaliação

Cada método é associado a um valor por cada métrica em cada imagem. As competições avaliam os métodos de duas formas, uma pela média das métricas e outra levando em conta a pontuação de cada imagem:

- Na primeira forma, tira-se a média desses valores por imagem para cada método e métrica, ordena-se por métrica os métodos de acordo com as médias obtidas, e a soma das posições é utilizada para ordenar os métodos de forma crescente, obtendo as posições finais.
- Na segunda forma, para cada imagem, é feita a ordenação por métrica, e a soma das posições nessas ordenações é utilizada para ordenar os métodos e obter as posições finais.

Mais precisamente, sejam M, N e P o número de métricas, métodos, e imagens, respectivamente, e $m_i, f_j e\, I_k, 1 \le i \le M, 1 \le j \le N\, e\, 1 \le k \le P$ os métricas, métodos, e imagens. Pode-se ver os f_j como funções que levam a imagem original na imagem binarizada, e os m_i como funções que levam a imagem binarizada num valor numérico (o GT fica implícito para simplificar a notação). Logo, existirão MNP valores possíveis da combinação dos métricas, métodos, e imagens, aplicando $m_i(f_j(I_k)), \forall i, j, k$.

Pela avaliação da média, serão tiradas as médias $\mu_{i,j}$ das métricas *i* para o método *j*. Logo:

$$\mu_{i,j} = \frac{\sum_{k=1}^{P} m_i(f_j(I_k))}{P}$$

Os *j* métodos são ordenados pelos seus valores para cada métrica *i*, de forma crescente em qualidade (depende da natureza da métrica). A posição do método *j* para a métrica *i* será denotado $r_i(j)$. A pontuação final *R* do método *j* é:

$$R(j) = \sum_{i=1}^{M} r_i(j)$$

As posições finais são obtidos do ordenamento crescente de *R*.

Para a avaliação por imagem, define-se as posições $r_{i,k}(j)$ do método *j* em relação à métrica *i* para imagem *k*, como obtidas em $r_i(j)$, mas levando em conta os valores somente para a imagem *k*, não em relação à média. A pontuação final *R′* do método *j* é:

$$R'(j) = \sum_{i=1}^{M} \sum_{k=1}^{P} r_{i,k}(j)$$

As posições finais são obtidos do ordenamento crescente de *R′*.

Em todas os ordenamentos, em caso de empate, os métodos participantes do empate recebem a mesma posição *r* e o método de valor imediatamente posterior recebe a posição seguinte *r + 1*.

5.2. DIBCO 2009

O DIBCO 2009 utilizou um conjunto de 10 imagens, sendo 5 de documentos impressos e 5 manuscritos. Suas métricas foram F-Measure, PSNR, NRM e MPM, e sua forma de avaliação foi pela média.

Método	Posição	Pontuação
P: map max	1	8
P: bin mmin	1	8
P: hom	2	11
Su	3	15
P: map mmin	4	23
P: bin max	5	24
Otsu	6	29
White	7	30
Sauvola	8	33
Niblack	9	39

Tabela 5.2. 1– posições pela média

Método	Posição	Pontuação
P: map max	1	60
P: hom	1	60
P: bin mmin	2	65
Su	3	69
P: map mmin	4	74
P: bin max	5	75
Otsu	6	93
White	7	111
Sauvola	8	137
Niblack	9	190

Tabela 5.2.2 – posições por imagem

Método	Posição	FM (%)	PSNR	NRM ($x10^{-2}$)	MPM ($x10^{-3}$)
P: map max	1	91,43	18,68	5,33	0,84
P: bin mmin	1	91,48	18,64	4,95	1,05
P: hom	2	91,11	18,59	5,02	1,02
Su	3	89,97	18,06	6,93	0,75
P: map mmin	4	89,27	17,52	6,10	1,52
P: bin max	5	88,37	17,63	8,17	1,41
Otsu	6	78,53	15,26	5,54	13,86
White	7	80,49	15,40	13,41	2,85
Sauvola	8	61,66	13,84	25,46	1,35
Niblack	9	38,85	5,76	19,75	183,08

Tabela 5.2.3 – posições pela média e média das métricas

5.2.1 Exemplos

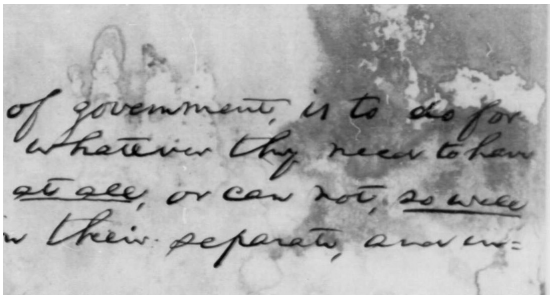

Imagem 5.2.1 H04 original

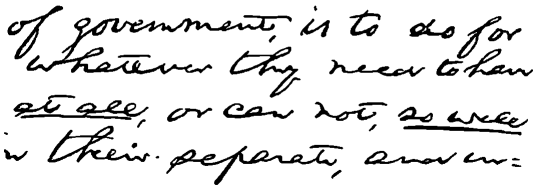

Imagem 5.2.2 H04 *ground truth*

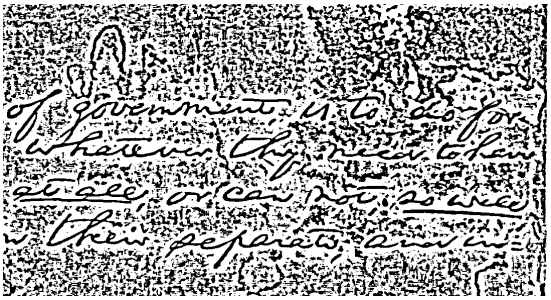

Imagem 5.2.3 H04 Niblack

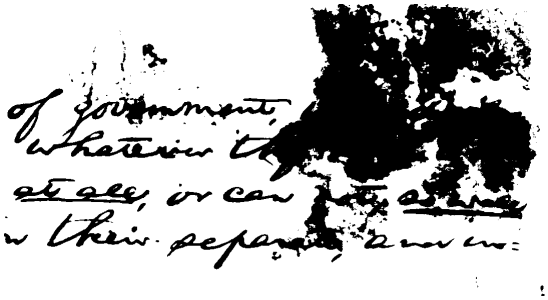

Imagem 5.2.4 H04 Otsu

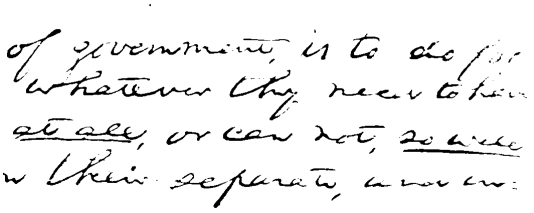

Imagem 5.2.5 H04 Sauvola

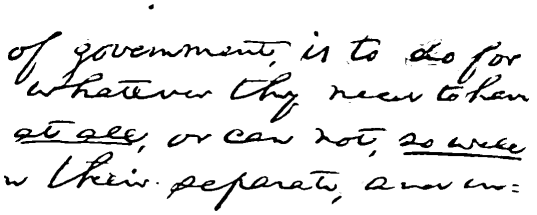

Imagem 5.2.6 H04 Su

Método	Escolhido
P: hom	Su
P: map max	Su
P: map mmin	Su
P: bin max	White
P: bin mmin	Su

Tabela 5.2.4 – escolhas para H04

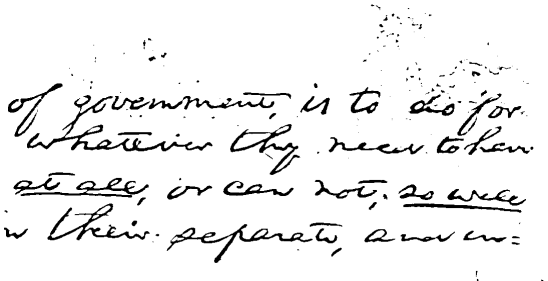

Imagem 5.2.7 H04 White

Imagem 5.2.8 P03 original

Imagem 5.2.9 P03 *ground truth*

Imagem 5.2.10 P03 Niblack

Imagem 5.2.11 P03 Otsu

Imagem 5.2.12 P03 Sauvola

Imagem 5.2.13 P03 Su

Método	Escolhido
P: hom	Otsu
P: map max	Otsu
P: map mmin	Su
P: bin max	Su
P: bin mmin	Otsu

Tabela 5.2.5 – escolhas para P03

Imagem 5.2.14 P03 White

5.3. H-DIBCO 2010

O H-DIBCO 2010 utilizou um conjunto de 10 imagens, todas documentos manuscritos. Suas métricas foram F-Measure, Pseudo F-Measure, PSNR, NRM e MPM, e sua forma de avaliação foi pela média.

Método	Posição	Pontuação
Su	1	13
P: bin mmin	2	14
P: map max	3	16
P: map mmin	4	20
P: hom	5	26
Otsu	6	27
White	7	36
P: bin max	8	37
Sauvola	9	38
Niblack	10	48

Tabela 5.3.1 – posições pela média

Método	Posição	Pontuação
Su	1	85
P: bin mmin	2	93
P: map max	3	98
P: hom	4	100
P: map mmin	5	105
Otsu	5	105
P: bin max	6	113
White	7	144
Sauvola	8	187
Niblack	9	229

Tabela 5.3.2 – posições por imagem

Método	Posição	FM (%)	pFM (%)	PSNR	NRM ($x10^{-2}$)	MPM ($x10^{-3}$)
Su	1	87,59	92,27	18,18	8,25	2,36
P: bin mmin	2	87,06	91,25	17,99	8,04	1,87
P: map max	3	86,12	91,58	17,83	9,28	1,23
P: map mmin	4	86,23	90,73	17,66	8,50	1,72
P: hom	5	86,04	90,00	17,52	8,15	3,09
Otsu	6	85,41	90,69	17,48	9,33	1,63
White	7	67,20	79,55	15,25	22,17	1,37
P: bin max	8	84,14	89,75	17,05	9,85	3,17
Sauvola	9	38,22	46,03	13,71	35,76	0,49
Niblack	10	29,05	29,54	5,35	21,19	186,63

Tabela 5.3.3 – posições pela média e média das métricas

5.3.1 Exemplos

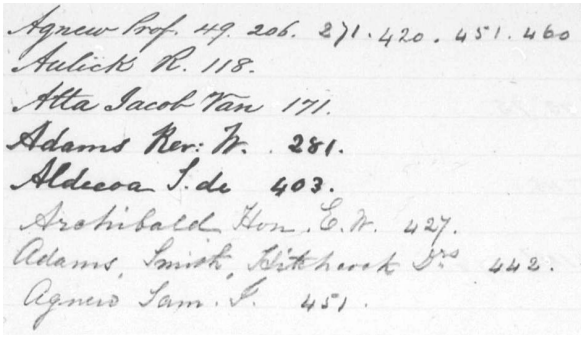

Imagem 5.3.1 H04 original

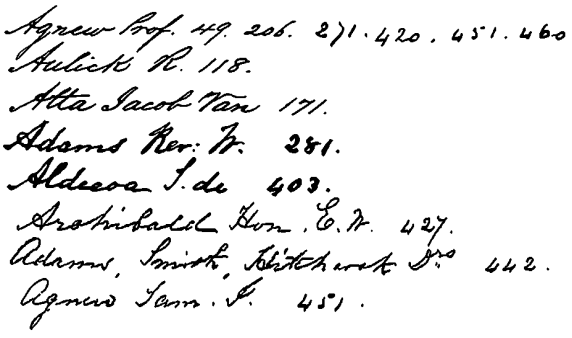

Imagem 5.3.2 H04 *ground truth*

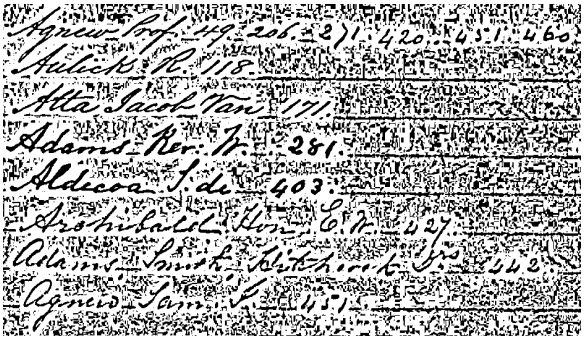

Imagem 5.3.3 H04 Niblack

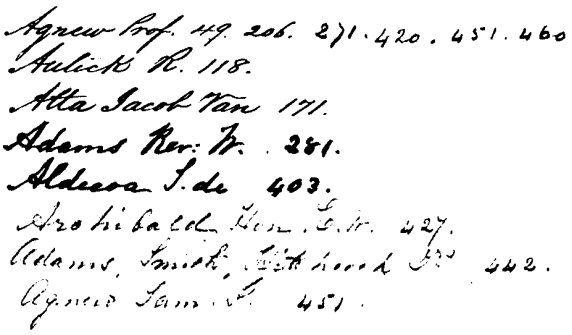

Imagem 5.3.4 H04 Otsu

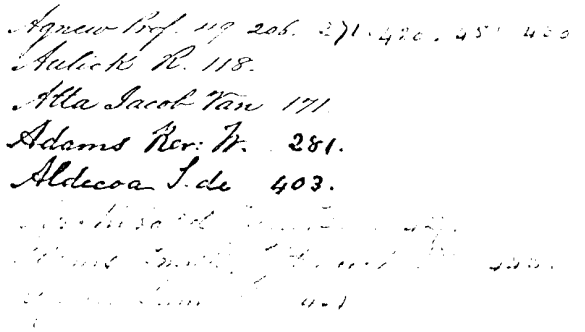

Imagem 5.3.5 H04 Sauvola

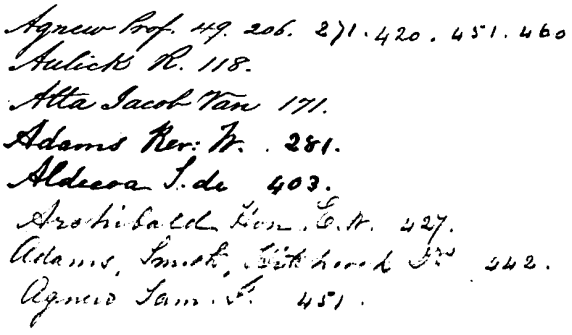

Imagem 5.3.6 H04 Su

Método	Escolhido
P: hom	Su
P: map max	Otsu
P: map mmin	Otsu
P: bin max	Otsu
P: bin mmin	Su

Tabela 5.3.4 – escolhas para H04

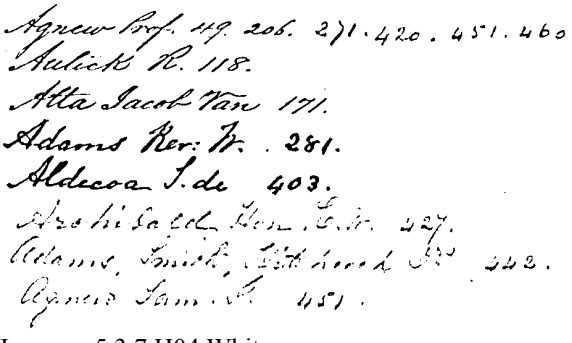

Imagem 5.3.7 H04 White

5.4. DIBCO 2011 – Métodos Implementados

O DIBCO 2011 utilizou um conjunto de 16 imagens, sendo 8 documentos impressos e 8 manuscritos. Suas métricas foram F-Measure, PSNR, DRD e MPM, e sua forma de avaliação foi por imagem.

Método	Posição	Pontuação
P: hom	1	6
P: map max	2	12
P: map mmin	3	13
P: bin max	4	17
P: bin mmin	5	19
Sauvola	6	26
White	6	26
Su	7	27
Otsu	8	34
Niblack	9	40

Tabela 5.4.1 – posições pela média

Método	Posição	Pontuação
Su	1	115
P: hom	2	117
P: map max	3	125
P: bin mmin	4	126
P: map mmin	4	126
P: bin max	5	132
Otsu	6	173
White	7	174
Sauvola	8	179
Niblack	9	319

Tabela 5.4.2 – posições por imagem

Método	Posição	FM (%)	PSNR	DRD	MPM ($x10^{-3}$)
P: hom	1	85,00	16,47	23,04	7,09
P: map max	2	84,54	16,34	23,33	7,04
P: map mmin	3	84,89	16,46	23,63	7,92
P: bin max	4	84,38	16,31	23,42	7,58
P: bin mmin	5	84,64	16,42	25,78	9,53
Sauvola	6	62,23	14,01	37,39	1,00
White	6	78,71	14,94	34,58	8,03
Su	7	83,17	16,08	37,94	8,17
Otsu	8	77,36	14,63	99,27	24,74
Niblack	9	36,10	5,51	438,78	186,96

Tabela 5.4.3 – posições pela média e média das métricas

5.4.1 Exemplos

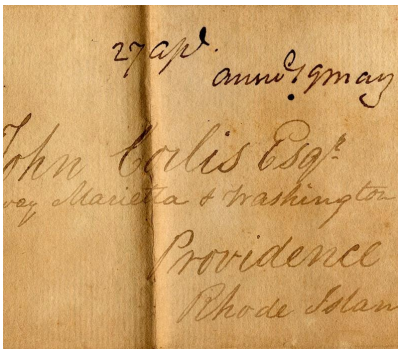

Imagem 5.4.1 HW6 original

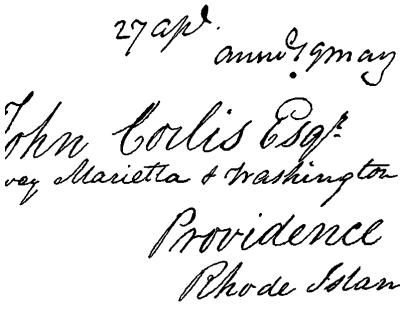

Imagem 5.4.2 HW6 *ground truth*

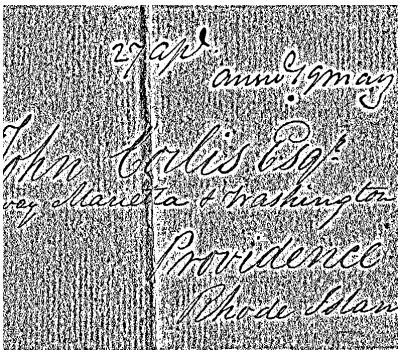

Imagem 5.4.3 HW6 Niblack

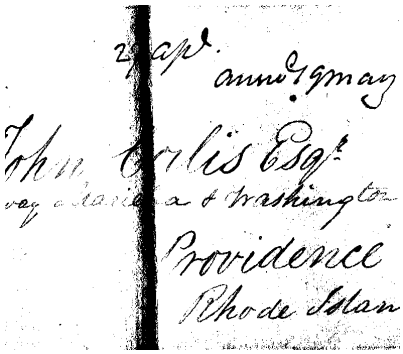

Imagem 5.4.4 HW6 Otsu

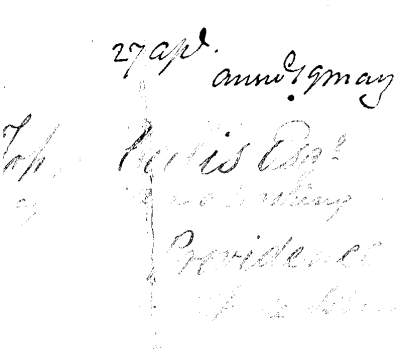

Imagem 5.4.5 HW6 Sauvola

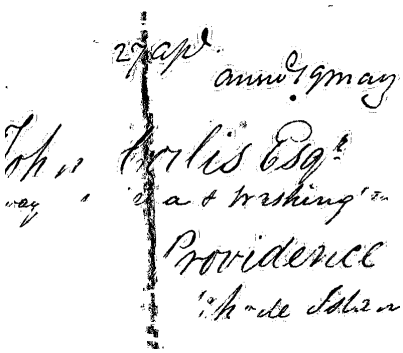

Imagem 5.4.6 HW6 Su

Método	Escolhido
P: hom	Su
P: map max	Su
P: map mmin	Su
P: bin max	Su
P: bin mmin	Su

Tabela 5.4.4 – escolhas para HW6

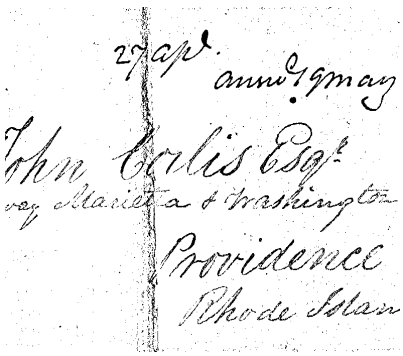

Imagem 5.4.7 HW6 White

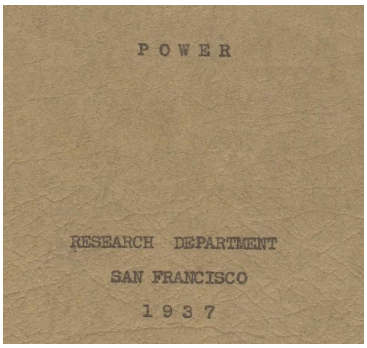

Imagem 5.4.8 PR7 original

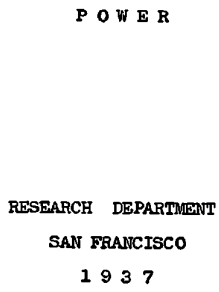

Imagem 5.4.9 PR7 *ground truth*

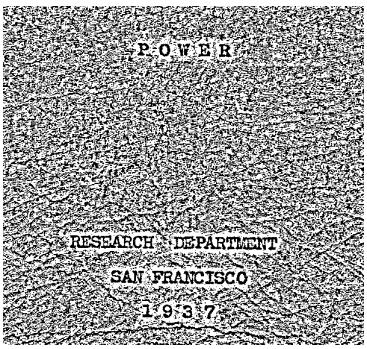

Imagem 5.4.10 PR7 Niblack

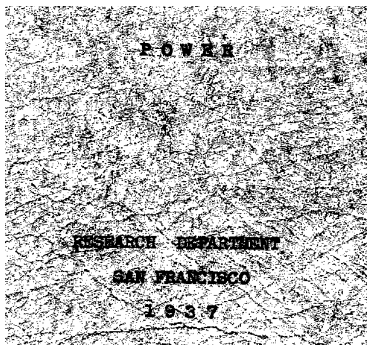

Imagem 5.4.11 PR7 Otsu


Imagem 5.4.12 PR7 Sauvola

Imagem 5.4.13 PR7 Su

Método	Escolhido
P: hom	White
P: map max	White
P: map mmin	White
P: bin max	White
P: bin mmin	White

Tabela 5.4.5 – escolhas para PR7

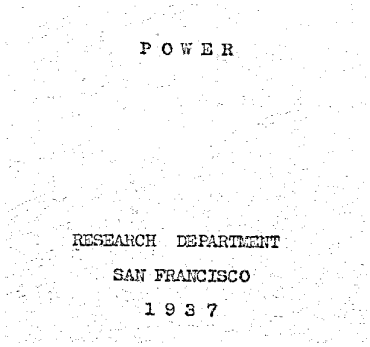

Imagem 5.4.14 PR7 White

5.5. DIBCO 2011 – Simulação

A simulação utiliza a mesma avaliação da seção anterior, mas os métodos são os 18 participantes originais da competição, simulando sua execução carregando as imagens de cada resultado dos arquivos disponibilizados. Como nos anteriores, a seleção escolhe um resultado dentre os seus concorrentes, só que agora são os 18 originais. Os métodos serão referenciados pelas suas numerações em [4].

Método	Posição	Pontuação
P: bin mmin	1	8
P: map mmin	2	12
11	3	15
P: bin max	4	20
P: hom	5	23
4	6	27
P: map max	7	28
6	8	36
3	9	37
2	10	40
8	11	43
1a	12	44
13	13	50
14	14	61
12	15	62
9	16	65
17	17	67
16	18	70
5	19	71
10	20	75
1b	21	78
7	22	80
15	23	92

Tabela 5.5.1 – posições pela média

Método	Posição	Pontuação
10	1	316
8	2	351
11	3	428
P: bin mmin	4	465
P: map mmin	5	466
6	6	467
4	7	486
P: hom	8	503
5	9	515
P: bin max	10	521
7	11	535
P: map max	12	585
9	13	596
1a	14	614
2	15	622
16	16	639
3	17	645
12	18	672
17	19	679
13	20	711
1b	21	792
14	22	819
15	23	1044

Tabela 5.5.2 – posições pela média

Método	Posição	FM (%)	PSNR	DRD	MPM ($x10^{-3}$)
P: bin mmin	1	87,39	17,34	20,31	6,41
P: map mmin	2	87,32	17,17	20,51	7,02
11	3	88,73	17,85	21,20	8,51
P: bin max	4	86,92	16,99	21,41	7,60
P: hom	5	86,79	17,05	22,11	8,00
4	6	85,22	16,61	24,59	4,38
P: map max	7	86,47	16,86	22,84	8,29
6	8	83,59	16,72	31,22	4,58
3	9	85,05	16,44	23,39	8,00
2	10	84,77	16,47	23,26	8,74
8	11	85,20	17,17	59,49	9,04
1a	12	85,78	16,52	26,25	11,94
13	13	83,60	15,97	25,74	10,27
14	14	78,00	14,88	30,32	7,16
12	15	82,29	15,74	34,70	13,76
9	16	81,87	15,83	61,94	11,49
17	17	80,49	15,46	38,52	10,99
16	18	81,22	15,56	57,55	17,39
5	19	81,40	15,97	118,20	23,04
10	20	80,86	16,14	398,35	64,29
1b	21	79,13	14,89	55,27	22,93
7	22	79,45	15,30	130,69	19,38
15	23	58,74	9,86	526,28	121,68

Tabela 5.5.3 – posições pela média e média das métricas

5.5.1 Exemplos

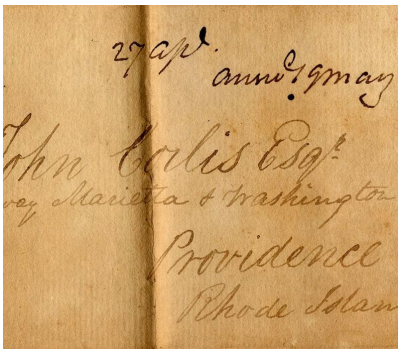

Imagem 5.5.1 HW6 original

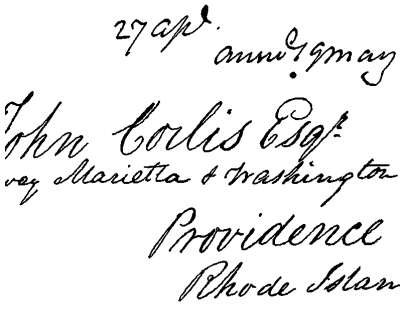

Imagem 5.5.2 HW6 *ground truth*

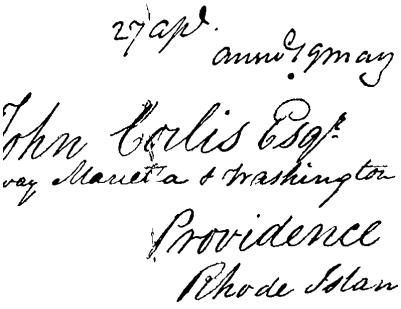

Imagem 5.5.3 HW6 método 10

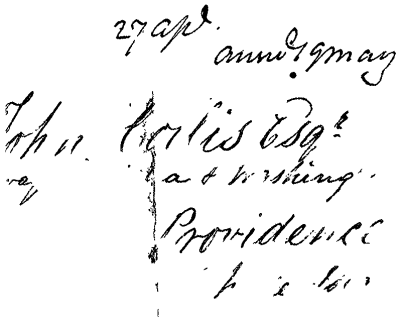

Imagem 5.5.4 HW6 método 8

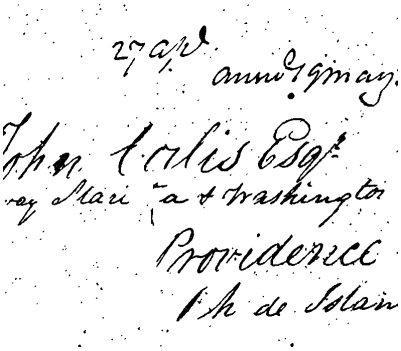

Imagem 5.5.5 HW6 método 11

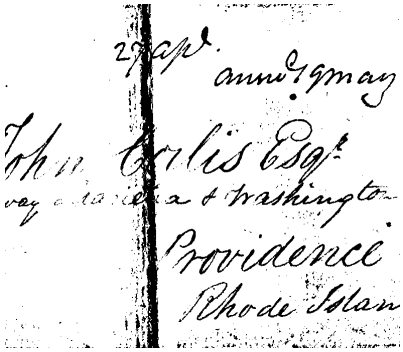

Imagem 5.5.6 HW6 escolhida por hom, map max, bin max

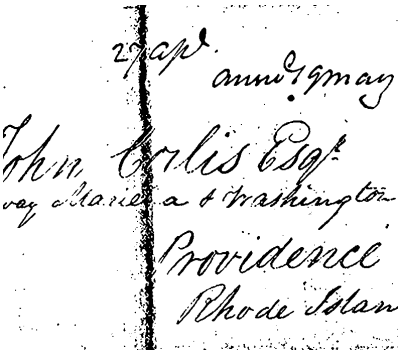

Imagem 5.5.7 HW6 escolhida por map mmin

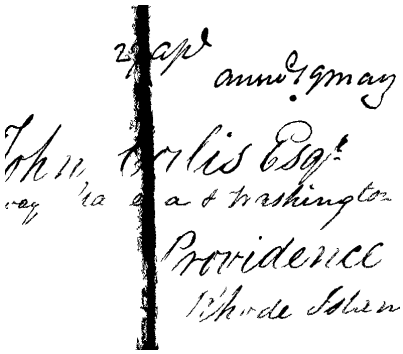

Imagem 5.5.8 HW6 escolhida por bin mmin

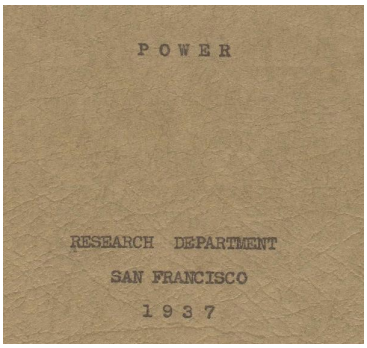

Imagem 5.5.9 PR7 original

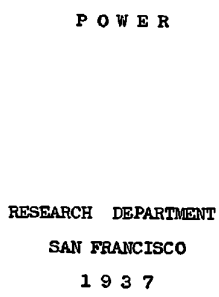

Imagem 5.5.10 PR7 *ground truth*

Imagem 5.5.11 PR7 método 10

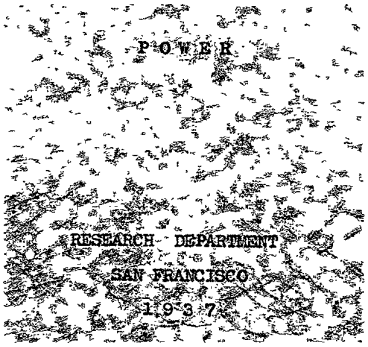

Imagem 5.5.12 PR7 método 8

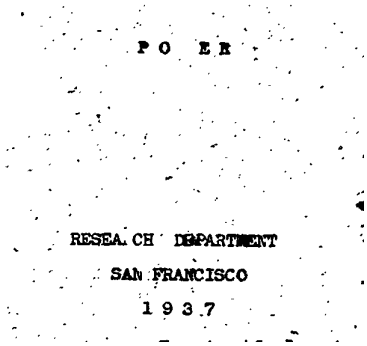

Imagem 5.5.13 PR7 método 11

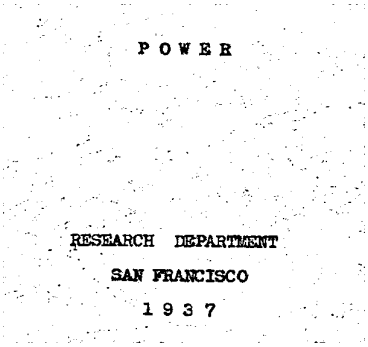

Imagem 5.5.14 PR7 escolhida por hom

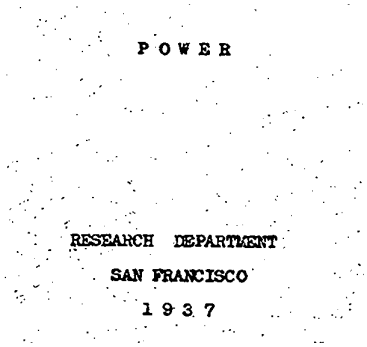

Imagem 5.5.15 PR7 escolhida por bin max

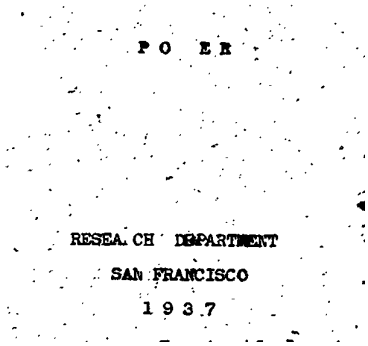

Imagem 5.5.16 PR7 escolhida por map max, map mmin, bin mmin

5.6. DIBCO 2011 – Validação da Avaliação

Os resultados obtidos da simulação utilizando os métodos, métricas e forma de avaliação do DIBCO 2011 foram validados com os dados publicados em [4]. As diferenças maiores do que 4 foram marcadas na tabela a seguir.

Método	Posição	Pontuação
10	1	316
8	2	351
11	3	428
6	4	467
4	5	486
5	6	515
7	7	535
9	8	596
1a	9	614
2	10	622
16	11	639
3	12	645
12	13	672
17	14	679
13	15	711
1b	16	792
14	17	819
15	18	1044

Tabela 5.6.1 – posições por imagem da simulação

Método	Posição	Pontuação
10	1	309
8	2	346
11	3	429
6	4	470
4	5	489
5	6	515
7	7	532
9	8	600
1a	9	616
2	10	620
16	11	630
3	12	649
12	13	676
17	14	682
13	15	715
1b	16	792
14	17	835
15	18	1045

Tabela 5.6.2 – posições por imagem publicados em [4]

O erro quadrático médio para as pontuações é de 30,28 , que é menor do que 10% do valor mínimo de pontuação; e a distância euclidiana, 23,34 , menor que 7,6%. O motivo dá-se pela implementação do DRD, como visto na tabela a seguir.

Imagem	10	8	11
HW1	26,0	54,4	60,6
HW2	5,5	6,7	6,9
HW3	7,2	7,9	7,1
HW4	10,0	11,8	11,3
HW5	6,6	7,1	18,5
HW6	7,9	25,4	15,5
HW7	7,1	8,3	13,9
HW8	5,0	6,0	5,0
PR1	10,0	14,0	12,0
PR2	51,0	36,0	78,3
PR3	7,1	9,2	5,3
PR4	7,9	13,8	8,0
PR5	9,7	8,2	5,9
PR6	2260,8	12,1	36,6
PR7	3937,2	715,6	41,2
PR8	14,5	15,2	13,1

Tabela 5.6.3 – DRD dos 3 primeiros lugares, obtidos pelo software

Imagem	10	8	11
HW1	6,6	13,8	15,3
HW2	1,4	1,7	1,7
HW3	1,8	2,0	1,8
HW4	2,5	3,0	2,8
HW5	1,6	1,8	4,6
HW6	2,0	6,3	3,9
HW7	1,7	2,0	3,4
HW8	1,3	1,5	1,3
PR1	2,5	3,5	3,0
PR2	12,8	9,0	19,6
PR3	1,8	2,3	1,3
PR4	2,0	3,5	2,0
PR5	2,4	2,0	1,5
PR6	575,0	3,1	9,3
PR7	1052,7	191,3	11,0
PR8	3,6	3,8	3,2

Tabela 5.6.4 – DRD dos 3 primeiros lugares, obtidos em [4]

Como pode ser visto, a diferença entre os números é grande, mas, pelo menos para estes valores, as implementações concordam com a ordem relativa dos métodos em cada imagem. A causa desta diferença não foi descoberta, nem se existe alguma relação entre os números. Para as outras métricas, os valores são quase idênticos para MPM (possível diferença de implementação na detecção de bordas e transformada distância), e F-Measure e PSNR são idênticos, pois baseiam-se somente no tamanhos das partições TP, TN, FP e FN, que é simples de implementar e está muito bem definido.

5.7 Implementação do Software

O software de avaliação desenvolvido é um aplicativo de linha de comando que lê três arquivos de entrada, um contendo o nome dos arquivos das imagens (GT, imagens a ser binarizadas, e GT esqueleto, se houver), outro com o nome dos métodos a ser testados e o outro com o nome das métricas que avaliarão o resultado. Os conjuntos de métricas, métodos e dados podem ser alterados diretamente nos arquivos de parâmetro, ou simplesmente alterando o nome de um parâmetro, o que o torna bem flexível.

Como saída, cria um diretório com os resultados. Neste diretório, são criados subdiretórios para cada imagem de entrada, onde serão colocados as imagens resultantes de cada binarização, e um arquivo de texto com o resultado das métricas para aquela imagem. No diretório principal, são criados arquivos de texto com a pontuação pela média das métricas ou por cada imagem, e as médias dos resultados das métricas para cada método são colocados no seu próprio subdiretório.

O software em si foi desenvolvido numa linguagem orientada a objetos, Java. Graças ao polimorfismo, é fácil adicionar um novo método ou métrica sem modificar qualquer parte do código existente. Ele pode servir como um conjunto de ferramentas minimalista para criação de alguns novos métodos e métricas, sem precisar alterar uma linha do que está escrito, ou adicionando funcionalidades necessárias, como mais transformações de imagens.

A arquitetura do software define uma hierarquia de imagens, com seus tipos definidos por interfaces, e uma interface para os métodos e métricas. A hierarquia de imagens têm todas suas implementações padrão feitas e não expostas. Alguns padrões de projeto [21] foram aplicados quando estes simplificavam a estrutura do código, como:

- O tipo de contraste, e de certa forma os métodos e métricas, são *Estratégias* (do original, *Strategy*) pois encapsulam um algoritmo e utilizam o polimorfismo para troca de algoritmos.
- Todas as estratégias não parametrizadas (quase todas as métricas e o método de Otsu, por exemplo) são *Singleton*, por não possuírem estado.
- Superclasses abstratas esqueletais existem para várias interfaces, e estas possuem um ou vários *Métodos Modelo* (do original, *Template Methods*), sendo a única parte do código que um cliente precisa implementar para cumprir o contrato da interface.
- Como as imagens são imutáveis, elas são criadas com um *Montador* (do original, *Builder*).
- Para encapsular o módulo de entrada e saída, é utilizado um *Adaptador* (do original, *Adapter*). Ele também é usado para fazer transformações em imagens.
- Janelas, imagens complementares e inversas são implementadas como *Decoradores* (do original, *Decorator*), pois alteram o comportamento de um objeto dado sem que ele tenha conhecimento disso, encaminhando as mensagens adiante da forma adequada.

Foi planejado algum tipo de interface gráfica, ou de integração com algum aplicativo mais robusto, como o ImageJ [22], mas não foi levado adiante por ter sido necessário um maior foco no desenvolvimento do método novo, que virou a seleção atual.

As bibliotecas utilizadas foram:

- API do ImageJ [22] para transformações binárias da métrica MPM
- ImageJIO (plugin do ImageJ) [23] para a leitura e escrita de arquivos de imagens.

Capítulo 6 – Conclusão

Os resultados mostrados levam às seguintes conclusões:

1. A seleção consegue um bom desempenho em média, frequentemente superior a todos os métodos individualmente, e até mesmo considerando imagem a imagem, o que garante não somente a aceitabilidade, mas a superioridade da escolha. Não consegue ser melhor sempre pois, ao tentar achar um resultado aceitável, a estimativa dá mais ênfase ao que maioria dos métodos considera correto, que às vezes é abaixo do ótimo. Isto aconteceu em quase todas as imagens do H-DIBCO 2010.
2. Apesar de ser paramétrica, a seleção sai-se bem para a maioria dos parâmetros utilizados, o que não a torna sensível à estimativa inicial. Isto significa que a escolha dos parâmetros é possui um certo grau de arbitrariedade, principalmente ao variar entre as classes estabelecidas nesta avaliação. Alguns outros parâmetros foram experimentados (janelas maiores e menores, probabilidades diferentes), mas estas 5 classes acabaram sendo as melhores para estes conjuntos de dados. Na prática, a aplicação que utilizar a seleção deve escolher a classe que mais se identifica com suas imagens de entrada, e, se realmente necessário, alterar os parâmetros da classe.
3. Alguns casos bem patológicos acontecem em certas imagens, em que métodos muito bons tem péssimo desempenho. Como foi mostrado nos exemplos, a seleção consegue evitar estes casos escolhendo imagens melhores de métodos que podem ser considerados inferiores para o resto do conjunto de imagens.
4. A avaliação está de acordo com o padrão estabelecido pelos DIBCO. Embora não possa ter sido testada nos anos anteriores da mesma forma como em 2011, as 3 métricas validadas estão presentes em todos os anos, e a NRM e o Pseudo F-Measure só dependem do tamanho das partições TP, TN, FP e FN, o mesmo do F-Measure e PSNR. A métrica DRD não é idêntica, mas preserva a ordem relativa dos resultados de forma satisfatória. As formas de avaliação e a pontuação também estão de acordo.

Conclusões 1 e 3 dão uma certa garantia de consistência e confiabilidade à binarização, cumprindo a meta estabelecida ao propor esta seleção. A conclusão 2 diminui ainda mais a necessidade de supervisão humana, pois a parametrização não precisa ser cuidadosamente ajustada, nem alterada com frequência.

As seções seguintes exploram as possibilidades de evolução tanto da seleção quanto do software.

6.1 Evolução da Seleção

A seleção ainda tem vários ramos de desenvolvimento, como:

- novas formas de estimar o *ground truth* para poder utilizar outras métricas, podendo também classificar os pixels em fundo, texto e indefinido;
- diferentes níveis de confiança para os métodos da seleção, priorizando os que se sabe ter melhores resultados, mas mantendo outros para o caso de eles falharem (Lamiroy e Sun [11] indicam que seu estudo vai explorar esta possibilidade);
- modelagem diferente da uniforme para os pixels, pois o texto e o fundo não têm mesma frequência (observações empíricas podem sugerir outro modelo);
- outras estratégias para a estimativa inicial que melhor localizam a região textual;
- utilizar um segundo nível de seleção, em que cada seleção do primeiro nível possuem conjuntos diferentes (não obrigatoriamente disjuntos), e depois uma das seleções é escolhida;
- generalizando o caso anterior para incluir as combinações de métodos, como a de Su [10] entre outras, criando uma árvore de binarização, na qual os nós são combinações e seleções e as folhas são os métodos (para evitar reexecução de métodos, sugere-se que estes binarizem a entrada *a priori* e somente seus resultados sejam passados para os nós da árvore);
- estimar parâmetros através da seleção através de um método iterativo, em que o mesmo método é executado com diferentes parâmetros, descartando os mais discrepantes e inserindo novos resultados com parâmetros derivados dos que sobrarem até convergir numa resposta;
- obter uma forma de mensurar a incerteza da escolha, de forma que a própria seleção possa distinguir quando uma certa escolha é insegura (podendo o software gerar algum tipo de aviso para o usuário, para que a seleção não precise ser manualmente verificada a não ser que emita o aviso);
- escalar o impacto da estimativa inicial, pois conforme o número de métodos aumenta, menos importante se torna a estimativa num primeiro momento (no entanto, conforme os resultados vão sendo descartados, a situação volta para o caso de poucos métodos, e ela tem que ter um peso condizente novamente).

6.2 Evolução do software

O software não chegou num estado final de entrega, pois boa parte do tempo foi gasta na elaboração de um novo método de binarização, que provou ser infrutífera, por fim levando à seleção proposta. Por isso, ele é somente um aplicativo de linha de comando que possui uma entrada fixa e uma saída um tanto verbosa, usada somente para experimentos. Da forma que está, pode até ser usado para execução de um método específico (basta dar um arquivo de métricas vazio), mas a entrada é estranha (qual arquivo é vazio, qual arquivo só tem um método escrito...) e a saída desnecessariamente grande (uma pasta para cada arquivo, copia GT e imagem original, cria um arquivo de texto com as métricas sem avaliação alguma).

Outro problema é o tempo de execução, pois a implementação direta da avaliação leva tempo igual à soma do tempo de execução de todos os métodos. Além disso, a seleção foi implementada como um método, e pelo principio do encapsulamento, não teria como obter os resultados dos métodos já executados. Como geralmente o conjunto de escolha é igual ao conjunto de concorrentes, isto significa que o tempo de execução dobra ao inserir a seleção. Para executar 5 seleções, como foi feito, o tempo é sextuplicado! Tudo isso poderia ser resolvido se a seleção (e possivelmente combinações) fossem feitas como árvores de binarização, e que os resultados das folhas pudesse ser compartilhado entre os nós.

Estes problemas **devem** ser solucionados caso o software venha a ser finalizado. Além disso:

- algum tipo de interface gráfica ajudaria bastante, mas mais importante que ter uma interface própria seria integrar este software a algum ambiente maior já largamente utilizado (o candidato mais adequado seria o ImageJ [22], dada sua ampla utilização pela comunidade de processamento de imagens e estar sempre em desenvolvimento);
- perfilamento para fazer as adaptações necessárias para tornar o programa mais eficiente, pois isto tornaria seu uso mais agradável (como boa prática de programação, primeiro foi escrito o mais claro e conciso que se pode, e só depois considerar a eficiência real, só descoberta através do perfilamento, do inglês, *profiling*);
- algum tipo de interface para Matlab, pois esta parece ser a linguagem mais adotada por pesquisadores deste meio, mesmo que isto reduza bastante a portabilidade do software, pois serviria como medida improvisada para poder integrar código legado até sua reescrita em uma linguagem da JVM (Java, Scala, Groovy, Clojure...);
- código mais robusto para injeção de dependências dos métodos e métricas, através de um formato padronizado (provavelmente XML), já que o atual utiliza uma simples entrada de texto e necessita de uma classe auxiliar para carregar cada um deles;
- maior cobertura de testes unitários, pois só há testes para as estruturas de dados (que são a maior parte do programa), faltando para os métodos e métricas;
- mais métodos implementados, sejam clássicos, como o Kittler e Illingworth [12], ou algum dos DIBCO (mais um estava sob desenvolvimento, mas suas várias etapas tinham uma alta curva de aprendizado e necessitavam de vários detalhes implementados, como solucionar um problema de mínimos quadrados, não dando tempo para terminar);
- implementação de combinações de métodos e suporte para elas;
- criar um serviço Web a partir dele, como [18] foi para segmentação de vídeo.

Como projeto pessoal, planejo tentar terminar o software, ou até mesmo reescrevê-lo para melhorar ainda mais sua estrutura.

Bibliografia

[1] GATOS, Basilis; NTIROGIANNIS, Konstantinos; PRATIKAKIS, Ioannis. An Objective Evaluation Methodology for Document Image Binarization Techniques. In: IAPR INTERNATIONAL WORKSHOP ON DOCUMENT ANALYSIS SYSTEMS, 8., 2008, Nara, Japan. *Proceedings of the...* Los Alamitos: IEEE Computer Society, 2008. p. 217-224.

[2] _____. ICDAR 2009 Document Image Binarization Contest (DIBCO 2009). In: INTERNATIONAL CONFERENCE ON DOCUMENT ANALYSIS AND RECOGNITION, 10., 2009, Barcelona, Spain. *Proceedings...* [Washington DC]: IEEE Computer Society, 2009. p. 1375–1382.

[3] _____. H-DIBCO 2010 – Handwritten Document Image Binarization Competition. In: INTERNATIONAL CONFERENCE ON FRONTIERS IN HANDWRITING RECOGNITION, 12., 2010, Kolkata, India. *Proceedings of the...* Los Alamitos: IEEE Computer Society, 2010. p. 727–732.

[4] _____. ICDAR 2011 Document Image Binarization Contest (DIBCO 2011). In: INTERNATIONAL CONFERENCE ON DOCUMENT ANALYSIS AND RECOGNITION, 11., 2011, Beijing, China. *Proceedings...* [S.I.]: IEEE Computer Society, 2011. p. 1506–1510.

[5] OTSU, Nobuyuki. A threshold selection method from gray level histogram. *IEEE Transactions on Systems, Man and Cybernetics*, v. 19, n. 1, p. 62–66, January 1978.

[6] NIBLACK, Wayne. *An Introduction to Digital Image Processing*. 1.ed. New Jersey: Prentice-Hall, 1986. p. 115–116.

[7] SAUVOLA, Jaakko; PIETIKAINEN, Matti. Adaptive document image binarization. *Pattern Recognition,* v. 33, n. 2, p. 225–236, January 2000.

[8] WHITE, J. M.; ROHRER, G. D.. Image thresholding for optical character recognition and other applications requiring character image extraction. *IBM Journal of Research and Develepment,* v. 27, n. 4, p. 400–411, July 1983.

[9] SU, Bolan; LU, Shijian; TAN, Chew Lim. Binarization of historical document images using local maximum and minimum filter. In: IAPR INTERNATIONAL WORKSHOP ON DOCUMENT ANALYSIS SYSTEMS, 9., 2010, Boston, USA. *Proceedings...* New York: ACM, 2010. p. 159–166.

[10] _____. Combination of Document Image Binarization Techniques. In: INTERNATIONAL CONFERENCE ON DOCUMENT ANALYSIS AND RECOGNITION, 11., 2011, Beijing, China. *Proceedings...* [S.I.]: IEEE Computer Society, 2011. p. 22-26.

[11] LAMIROY, Bart; SUN, Tao. Precision and Recall Without Ground Truth. In: IAPR INTERNATIONAL WORKSHOP ON GRAPHICS RECOGNITION, 9., 2011, Seoul, Korea. *Proceedings...* [S.I.: s.n.], 2011. p. [?].

[12] KITTLER, J.; ILLINGWORTH, J.. On threshold selection using clustering criteria. *IEEE Transactions on Systems, Man and Cybernetics*, v. 15, n. [?], p. 652–655, 1985.

[13] TRIER, Oivind Due; JAIN, Anil Kumar. Goal-directed evaluation of binarization methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 17, n. 12, p. 1191-1201, 1995.

[14] SEZGIN, Mehmet; SANKUR, Bulent. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, v. 13, n. 1, p. 146–165, January 2004.

[15] BAUMANN, Axel et al. A Review and Comparison of Measures for Automatic Video Surveillance Systems. *EURASIP Journal on Image and Video Processing*, v. 2008, n. [?], p. [?]

[16] STATHIS, Pavlos; KAVALLIERATOU, Ergina; PAPAMARKOS, Nikos. An Evaluation Technique for Binarization Algorithms. *Journal of Universal Computer Science*, v. 14, n. 18, p. 3011-3030, 2008.

[17] LU, Haiping; KOT, Alex C.; SHI, Yun Q.. Distance-Reciprocal Distortion Measure for Binary Document Images. *IEEE Signal Processing Letters*, v. 11, n. 2, p. 228-231, 2004.

[18] YOUNG, David P.; FERRYMAN, James M.. PETS metrics: on-line performance evaluation service. In: JOINT IEEE INTERNATIONAL WORKSHOP ON VISUAL SURVEILLANCE AND PERFORMANCE EVALUATION OF TRACKING AND SURVEILLANCE, 2., 2005, Beijing, China. *Proceedings of the...* [S.I.: s.n.], 2005. p. 317-324.

[19] AGUILERA, Josep et al. Evaluation of Motion Segmentation Quality for Aircraft Activity Surveillance. In: JOINT IEEE INTERNATIONAL WORKSHOP ON VISUAL SURVEILLANCE AND PERFORMANCE EVALUATION OF TRACKING AND SURVEILLANCE, 2., 2005, Beijing, China. *Proceedings of the...* [S.I.: s.n.], 2005. p. 293-300.

[20] ERDEM, C.E.; SANKUR, Bulent. Performance Evaluation Metrics for Object-Based Video Segmentation. In: EUROPEAN SIGNAL PROCESSING CONFERENCE, 10., 2000, Tampere, Finland. *Eusipco 2000...* [S.I.: s.n.], 2000. p. 917–920.

[21] GAMMA, Erich; HELM, Richard; JOHNSON, Ralph; VLISSIDES, John. *Design Patterns: Elements of Reusable Object-Oriented Software*. 1.ed. Massachusetts: Addison-Wesley, 1995.

[22] ImageJ website: http://rsb.info.nih.gov/ij

[23] ImageJ I/O Plugin website: http://ij-plugins.sourceforge.net/plugins/imageio

Parte II – Parte Subjetiva

Capítulo 1 – Desafios e Frustrações

1.1 Em relação ao trabalho realizado

Este trabalho começou com uma pequena conversa com meu orientador em busca de um tema. Na época, embora tivesse uma boa ideia de em que gostaria de trabalhar, não conseguia pensar em nada que justificasse o trabalho de um ano de desenvolvimento ou pesquisa. Nesta conversa foi sugerido a mim o tema de buscar um novo método de binarização dado o contexto das competições DIBCO. Mesmo tendo pouquíssimo conhecimento na área de processamento de imagens, resolvi aceitar, pensando nas possibilidades de aprendizado. Além disso, meu orientador tinha me avisado que este seria um trabalho de implementação, e isto me agradou bastante.

A partir desta conversa, muita pesquisa foi feita para entender o domínio e o problema. Vagarosamente consegui avançar na área, conseguindo implementar uma ou outra coisa, explicar os intrincados detalhes de um método vencedor ou a obscura forma de avaliação de alguma métrica.

O software de avaliação, originalmente proposto apenas uma ferramenta para tentar implementar métodos e métricas, foi concebido tendo em mente vários conceitos de programação extensível e sujeita a mudanças, já pensando em torná-lo um produto do trabalho com utilidade real, e não mais uma ferramenta que todo pesquisador precisa implementar toda vez que vai publicar um resultado. Serviria também para experimentar padrões de projeto, boas práticas de programação, teoria de orientação a objetos, refatoração, e tudo mais relacionado ao desenvolvimento de um software.

Sendo assim, no primeiro semestre, apesar das dificuldades, o projeto progredia de forma previsível, sem grandes problemas. Mas o tal método a ser desenvolvido, a razão de ser de todo o esforço, não tinha sido concebida. As poucas boas ideias não chegavam nem de longe no desempenho dos melhores das competições. Poderia atingir uma posição razoável, mas nesta posição já existem vários. Não faria diferença alguma.

No início do segundo semestre, após implementar o método de Su, que apesar da excelente performance, é simples e não possui muitos passos, o desenvolvimento do software teve de ser deixado de lado, por isso nenhum outro método pode ser implementado (um ficou pela metade, por ser grande, vagamente descrito e muito complicado). Criar um novo método era a prioridade.

Depois de muito tempo e de muita leitura, acabei por tentar mudar um pouco o foco: não mais um método excelente de uso geral, mas um que fosse bom para um certo subdomínio caracterizado, como documentos com manchas e iluminação desigual. Entretanto, não pude progredir muito, pois todas as ideias tidas não pareciam ter qualquer propriedade especial que solucionasse muito bem um problema específico. Além disso, a maioria dos métodos das competições possui várias etapas, cada utilizando conhecimento avançados de áreas que eu desconheço, juntamente com várias heurísticas obtidas durante anos de observação. Parecia um terreno bastante desfavorável para um iniciante.

Lá pelo meio do segundo semestre, enquanto finalizava os detalhes do software que faltavam e ainda lia à procura de novos caminhos, me veio a ideia de selecionar métodos. Descobri rapidamente que existem formas de combinar métodos, das mais simples às muito complexas, mas nenhuma seleção foi encontrada.

Agora que tinha algo que valia a pena desenvolver, precisava de alguma forma de selecionar imagens, que encontrei, implementei e me desesperei. Era pior que o pior! Depois de muito analisar os números estimados pela seleção, consegui as soluções mostradas neste trabalho. Lentamente, a seleção foi se tornando algo útil, até alcançar resultados que eu já nem mais esperava.

Gostaria de ter tido mais tempo para fazer o software que almejava a princípio, mas já é bastante satisfatório ter conseguido algo que produz um resultado útil, uma vez que passei boa parte do ano sem saber como entregaria uma monografia de um trabalho sem resultados.

1.2 Em relação ao curso

Entrei no BCC sem saber do que se tratava, sem ter programado na vida, mas gostei desde o princípio, apesar de não ser muito afim com a matemática toda. Mas, com um ano, passei a desanimar com a dificuldade do curso, e não sabia mais se era mesmo o que queria fazer. Parecia gostar de programar, mas não era algo tão agradável assim.

Foi no terceiro ano que consegui ânimo de novo, em grande parte por ter começado um estágio. Não muito rigoroso, era fácil de conciliar com os estudos, e ao mesmo tempo me dava uma visão da real utilidade de tudo aquilo que era ensinado: tudo que produzia era por que era realmente necessário e iria ser usado logo quando possível. Tive contato com as tecnologias usados em ambiente de trabalho, que são muito interessantes, e mais importante do que tecnologias, conheci o que era orientação a objetos e seu inestimável valor durante o desenvolvimento de um software. Se não achava tão agradável antes, passou a ser extremamente satisfatório.

Graças a isso, consegui levar o curso adiante, pois sabia das mais formidáveis possibilidades que teria com todo aquele conhecimento para o qual teria que estudar bastante. Parte do que deixava insatisfeito era esta falta de comprometimento que sentia com o que estudava. Isto era ainda mais presente em trabalhos em grupo, pois toda aquela interação parecia inútil, e geralmente, mesmo quando se leva o trabalho a sério, nem todos o fazem.

Capítulo 2 – Disciplinas Relevantes

Bom, é difícil decidir quais são relevantes, porque praticamente todas são, então vou tentar falar das que mais lembro:

- Introdução à computação é a primeira impressão que se tem de como é programar, e gostei muito de como dada, o que me manteve no curso até hoje, apesar do desânimo.
- Princípios de Desenvolvimento de Algoritmos é uma grande introdução a vários temas importantes, e também é essencial para cativar o interesse dos ingressantes.
- Estrutura de Dados ensina lições valiosas sobre como dados são armazenados e qual o impacto de uma escolha de uma estrutura tem no andamento geral de algoritmo.
- Sistemas Operacionais e Arquitetura de Computadores são duas matérias que dão uma visão geral de tudo o que geralmente é abstraído durante a programação. Este conhecimento do que está por trás de cada comando escrito e a enorme quantidade de coisas que ocorrem em cada pequena coisa feita é imprescindível para poder lidar com todo tipo de situação que pode ocorrer quando um sistema funciona numa situação real.
- Análise de Algoritmos, Algoritmos em Grafos, Métodos Numéricos de Álgebra Linear e Autômatos são matérias da parte mais científica que é sempre útil saber. Máquinas de estados, modelagem de grafos, problemas NP, e resolução de mínimos quadrados aparecem quando menos se espera, e ignorar estes conhecimento costuma ser um erro.
- Álgebra I e II ajudam a pensar de forma metódica, a construir teoremas a partir de pequenos axiomas. Sem essa maturidade matemática, costuma-se encarar tudo na programação de forma muito heurística, o que pode levar a muitas contradições.
- Banco de Dados é tão útil que é difícil de descrever o porquê. Não imagino que tipo de situação não haveria um banco de dados, desde pesquisa científica, desenvolvimento de software, programação em nuvem, etc. E por isso mesmo, saber de como eles funcionam e como gerenciá-los e projetá-los é mais útil que muito conhecimento em programação.

  Existem outras matérias importantes que não falei, como Criptografia e Segurança de Dados, Programação para Redes, e Programação Concorrente.

Agradecimentos

Gostaria de agradecer à minha namorada que me ajudou muito nesses anos todos, à minha família por sempre ter me ajudado quando precisei e ao meu orientador por ter possibilitado a realização deste trabalho. Muito obrigado.