Relatório da Fase 1: Pré-processamento

Daniel C. Valério[1], **Henrique S. Pinheiro**[1], **Sávio Camacam**[1]

[1]Departamento Acadêmico de Computação
Universidade Tecnológica Federal do Paraná (UTFPR)
Caixa Postal 271 – 87301-899 – Paraná – PR – Brazil

{danielvalerio,henriquepinheiro,saviocamacam}@utfpr@alunos.edu.br

Resumo. *O objetivo deste trabalho é documentar a abordagem utilizada no pré-processamento da base de dados de escritores árabes Writer Identification Arabic.*

1. Base de Dados

Para esse trabalho foi utilizada a base de dados de escritores árabes *Writer Identification Arabic*. Esta base contém 100 classes, representando 100 escritores distintos, com duas amostras para cada classe, totalizando 200 imagens com escritas em árabe.

2. Análise da Base de Dados

Na (Figura 1) é mostrada uma amostra original da base de dados sem nenhum pré-processamento e, através de análises visuais foi percebido que as imagens foram capturadas em boas condições e não apresentavam ruídos significativos. Diante disso, foi decidido que o pré-processamento seria feito através do recorte das imagens, para eliminação dos espaços em branco excedentes, e da binarização das imagens para eliminar variações nos dados devido a diferenças no tom de cor da caneta utilizada para escrita.

3. Pré-processamento

Nesta seção são descritas com detalhes as etapas de pré-processamento realizadas nas imagens da base de dados utilizada.

3.1. Binarização das imagens

Para que as diferenças no tons das cores das canetas utilizadas pelos escritores da base de dados fossem eliminadas, foi realizada a binarização das imagens. A etapa de binarização transforma uma imagem em tons de cinza para uma imagem preta e branca segundo um valor limiar que indica em qual ponto uma intensidade deve ser considerada branca ou preta. O valor limiar, obtido empiricamente, que melhor que preservou os dados de escrita foi de 0.90. Os resultado pode da limiarização pode ser observado na (Figura 2).

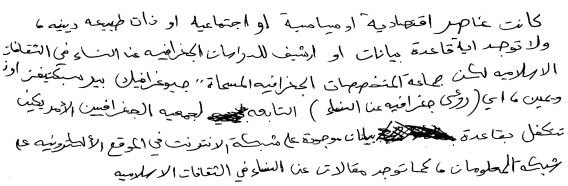

Figura 2. Imagem final binarizada e recortada

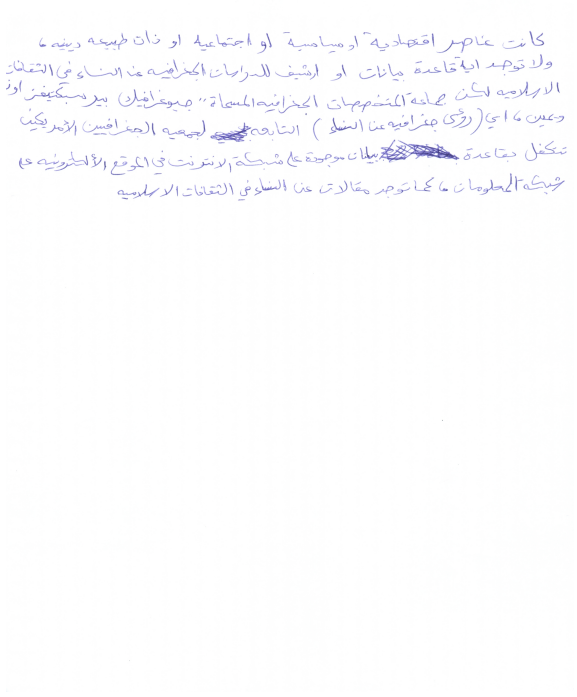

Figura 1. Imagem original sem pré-processamento

3.2. Recorte das bordas brancas das imagens

Como as escritas da bases de dados foram realizadas em folhas sem pautas, encontramos divergências no posicionamento e alinhamento dos blocos de texto. Dessa maneira, torna-se mais complexa a extração exata da região efetiva de texto, podendo ocasionar problemas em uma etapa mais adiante, como a extração de características da escrita, uma vez que o espaço em branco pode acabar sendo considerado uma característica do escrito dependendo pelo método utilizado.

Para realizar o recorte utilizamos a seguinte abordagem: (i) foi realizado o somatório das colunas e das linhas da imagem, com os resultados sendo armazenados em um vetor M para as linhas, e N para as colunas; (ii) foi definido um tamanho, $slice$, que representa um corte da imagem; (iii) dividimos $M/slice$ e $N/slice$ para obtermos o tamanho das regiões da imagem; (iv) são criados dois vetores de tamanho $slice$ para guardar o somatório das intensidades encontradas nas regiões; (v) é definido um valor de limiar $threshold$ que representa um valor mínimo para uma região ser considerada como um possível bloco de texto; (vi) os vetores de tamanho $slice$ são percorridos comparando os resultados do somatório das regiões com o $threshold$ de ruído para detecção do início e fim da região de texto. Na (Figura 3) esta ilustrado a abordagem utilizada para o calculo do recorte das imagens.

4. Dificuldades e problemas no pré-processamento

Durante a aplicação do recorte das imagens foi observado que, para algumas imagens, o nosso algoritmo não consegue recortar corretamente o final de um bloco de texto. Na

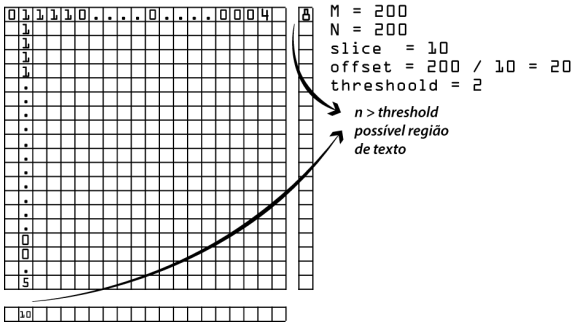

Figura 3. Processo do calculo do recorte da imagem

(Figura 4) é possível perceber que o inicio do bloco de texto foi detectado e recortado corretamente enquanto que o final não pode ser detectado corretamente. Acreditamos que isso ocorra devido a presença de ruídos em uma quantidade maior que o valor limiar mínimo para que um dado seja considerado uma linha, fazendo com que o algoritmo entenda erroneamente que a região com ruído trata-se de um texto.

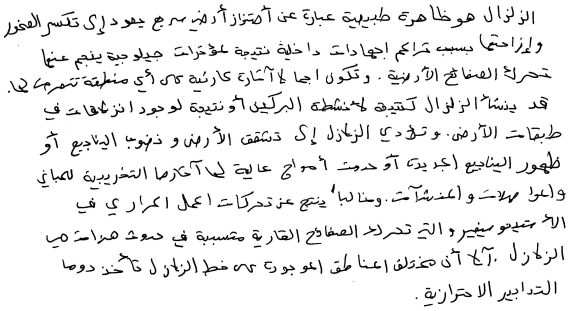

Figura 4. Imagem final com falha na detecção do final do bloco de texto

5. Conclusão

Para início dos trabalhos, foi feita uma observação manual de todas as cartas que determinou como deveria ser o pre-processamento, a exemplo do processo de encurtamento das margens de cada carta pela detecção das regiões que continham texto ou não. O processo foi feito com um script em Octave, que fazia a binarização e, a partir de um limiar de aceitação para ruídos, detectava regiões com altos índices de preto que denotavam conter texto. O *script* teve um nível de acerto, mas não funcionou em absolutamente todas as cartas, a correção será aplicada entre esta entrega e a fase de segmentação e recorte das imagens.