FussyFood Melhorado

Isabel Souza de Carvalho, Luiza Helena Vieira

Ciência da Computação – Universidade Federal do Maranhão (UFMA) – Campus Bacanga

65080-805 – São Luís – MA– Brasil

bel.souza07@gmail.com, luizahelenavieira92@gmail.com

Abstract. *This article describes proposals for improving the methodology used by Bagadia and Mundra (2015) in the development of FussyFood application. The main changes to be established are related to the steps of image processing and OCR application, present the methodology proposed by the authors mentioned above. It is expected that the proposals in this paper extend the amount of recognized ingredients properly and contribute to a better feedback to the user.*

Resumo. *O presente artigo descreve propostas para melhoria da metodologia utilizada por Bagadia e Mundra (2015) no desenvolvimento do aplicativo FussyFood. As principais modificações a serem estabelecidas estão relacionadas às etapas de processamento das imagens e aplicação de OCR, presentes na metodologia proposta pelos autores acima citados. Espera-se que as propostas apresentadas no presente trabalho ampliem a quantidade de ingredientes reconhecidos corretamente e contribuam para um melhor feedback ao usuário.*

1. Introdução

Em 2015, Sameep Bagadia e Rohit Mundra desenvolveram um aplicativo chamado FussyFood que tem como objetivo o reconhecimento de ingredientes a partir de imagens de embalagens de alimentos. A aplicação auxilia pessoas com restrições nutricionais ou alergias alimentares na escolha de alimentos apropriados para o consumo. Apesar de suprir boa parte das necessidades do usuário, o aplicativo ainda necessita de melhorias em alguns aspectos. No presente artigo, serão apresentadas propostas para incrementar o funcionamento do aplicativo FussyFood.

2. Propostas apresentadas

A metodologia utilizada por Bagadia e Mundra (2015) é composta por 6 passos principais, conforme a Figura 1:

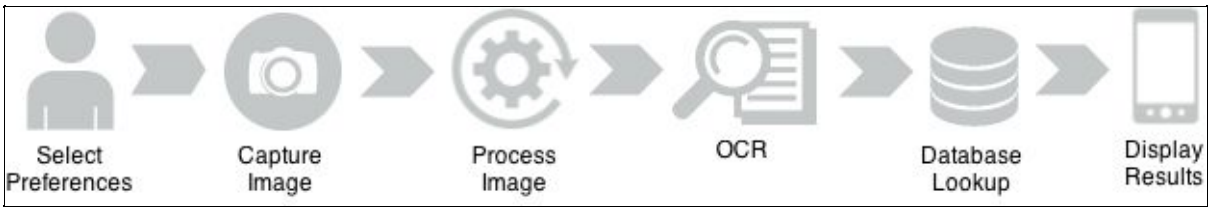

Figura 1: Fluxo de Processamento no FussyFood

Fonte: Bagadia e Mundra (2015)

As propostas para melhoria da metodologia utilizada pelos autores citados não visam a modificação do fluxo de processamento mostrado na Figura 1, mas algumas mudanças na forma como cada passo do fluxo é executado. Alguns os passos do fluxo de processamento serão analisados e terão propostas para melhoria em sua execução, conforme relatado nas seções a seguir. São eles: seleção de preferências, captura da imagem, processamento da imagem, *Optical Character Recognition* (OCR), busca na base de dados e resultados.

2.1 Seleção de Preferências

A primeira etapa do processamento no aplicativo FussyFood é a seleção de preferências de acordo com as restrições alimentares do usuário. O aplicativo contêm 6 (seis) opções de restrições, conforme mostrado na Figura 2:

Figura 2: Painel de Preferências

Fonte: adaptado de Bagadia e Mundra (2015)

De acordo com a Figura 2, percebe-se que 2 restrições são tratadas em conjunto, são elas Laticínios e Ovos (*Dairy e Eggs*). Dessa forma, usuários que possuem apenas uma dessas restrições (ou a laticínios, ou a ovos) terão um *feedback* restrito de alimentos liberados para o consumo.

A proposta para essa etapa seria a separação das preferências alimentares que são analisadas em conjunto, de modo que cada restrição alimentar seja tratada separadamente.

2.2 Captura da Imagem

A segunda etapa consiste na captura da imagem. O aplicativo FussyFood foi projeto para capturar imagens de superfícies planas, conforme mostrado na figura 3 abaixo.

Figura 3: Imagem a ser processada

Fonte: adaptado de Bagadia e Mundra (2015)

A proposta para essa etapa diz respeito à captura de imagens de superfícies cilíndricas, pois há diversas embalagens de alimentos em formato de cilindro e pessoas com algum tipo de restrição alimentar não se limitam em consumir produtos com embalagem plana.

Em imagens onde o as linhas de texto são curvadas e as palavras próximas das extremidades laterias são mais estreitas, os sistemas OCR apresentarão falhas. Bieniecki (*et al*, 2007) propõem uma forma para processar imagens não lineares. O primeiro passo para a correção é traçar curvas que delimitem as bordas superior e inferior da área onde encontra-se o texto, conforme as Figuras 4 e 5.

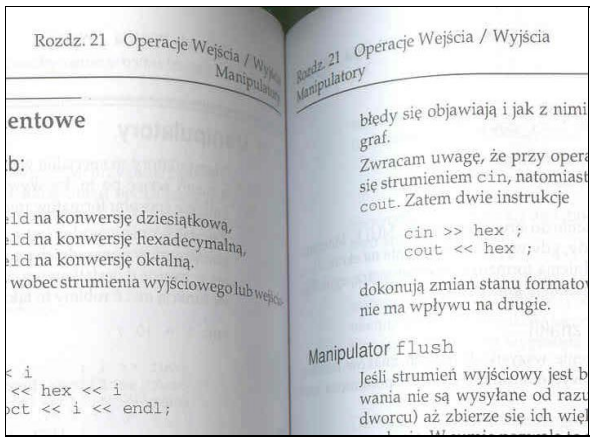

Figura 4: Imagem escaneada de um livro aberto.
Fonte: adaptado de Bienieck (et al, 2007)

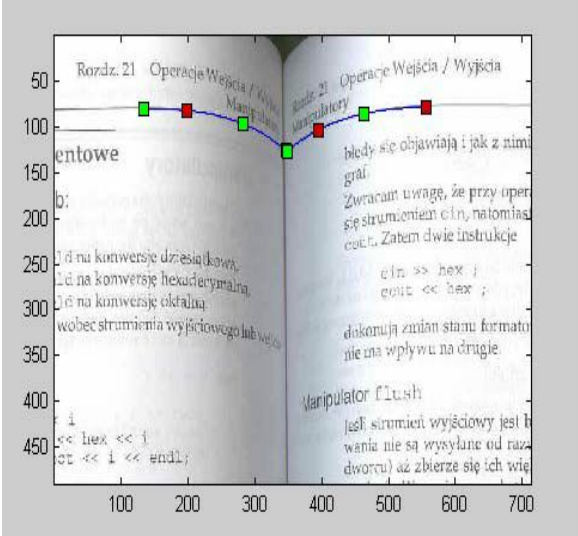

Figura 5: Processo de ajustes na curva de delimitação
Fonte: adaptado de Bienieck (et al, 2007)

Encontradas as curvas superior e inferior, o procedimento de correção é executado. Cada ponto da curva é descrito pelo parâmetro t ∈[0,1]. Para um valor específico t, uma linha reta entre a curva superior e inferior é avaliada. O comprimento da linha passa pelos pontos $[x_u(0), y_u(0)]$ e $[x_1(0), y_1(0)]$, que será a altura da nova imagem. Cada linha $[x_u(t), y_u(t)]$ é transformada em uma linha perpendicular usando as equações de interpolação, como resultado é criada a imagem retangular, conforme as Figuras 6 e 7.

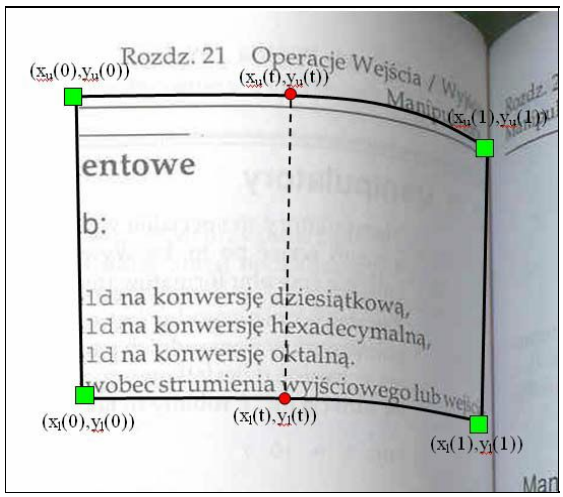

Figura 6: Encontrando as curvas de delimitação para a área de interesse.
Fonte: adaptado de Bienieck (et al, 2007)

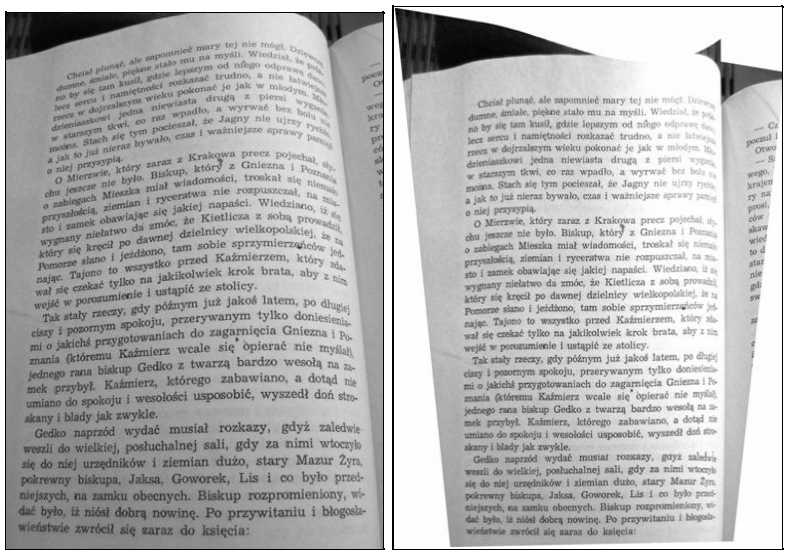

Figura 7: Resultado da restauração de uma imagem.
Fonte: adaptado de Bienieck (et al, 2007)

Portanto, pretende-se utilizar a metodologia propostas por Bieniecki (*et al*, 2007) para o processamento de imagens cilíndricas.

2.3 Processamento da imagem

Essa etapa seguirá a metodologia proposta por Bagadia e Mundra (2015). As alterações ocorrerão apenas em caso de tratamento de imagem de embalagens cilíndricas, se a imagem capturada pelo usuário for cilíndrica, ocorrerá primeiramente o processamento descrito na seção 2.2, com a aplicação da metodologia proposta por Bienieck (*et al*, 2007).

Capturada a imagem, serão aplicados os métodos utilizados no trabalho de Bagadia e Mundra (2015): o *thresholding* adaptativo local, a transformada probabilística de Hough, a remoção do *background* e a detecção de *bounding box*. O *thresholding* adaptativo local transformará a imagem com padrão RGB para escalas de cinza e ainda será aplicado o Método de Otsu nos blocos onde a variância for elevada. Por sua vez, a transformada probabilística de Hough será utilizada para rotacionar imagens que foram captadas de forma inclinada.

Para remover o *background*, utiliza-se o método de detecção de bordas Canny como uma forma de eliminação de ruído. Após esse processo, ocorre a aplicação de um filtro - o qual não foi citado por Bagadia e Mundra (2015) - para destacar os *pixels* brancos nas regiões de borda detectadas. Para remover os ruídos restantes, aplica-se um *threshold* e faz-se uma equalização.

Após a remoção do *background*, é necessária a detecção de *background* porque nem todo texto contido na imagem capturada é relevante, assegurando um melhor processamento da imagem. Para isso, dilata-se a imagem até encontrar a menor área com mais texto. Geralmente, essa área é a que contem a lista de ingredientes. É utilizado a técnica de F-*score* para detectar a *bounding box* e maximizar os *pixels* brancos sem comprometer a imagem, onde a precisão é a

região da imagem não contida na *bounding box* e o *recall* é a fração de *pixels* brancos contidos na *bounding box*.

2.4 Reconhecimento ótico de caracteres e busca no banco de dados

Depois de todo processamento e tratamento da imagem, a parte a ser utilizada para assimilar as restrições consiste na parte dos ingredientes.

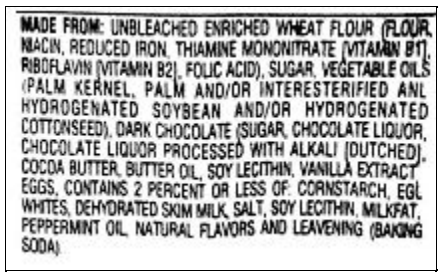
MADE FROM: UNBLEACHED ENRICHED WHEAT FLOUR (FLOUR, NIACIN, REDUCED IRON, THIAMINE MONONITRATE [VITAMIN B1], RIBOFLAVIN [VITAMIN B2], FOLIC ACID), SUGAR, VEGETABLE OILS (PALM KERNEL, PALM AND/OR INTERESTERIFIED ANL HYDROGENATED SOYBEAN AND/OR HYDROGENATED COTTONSEED), DARK CHOCOLATE (SUGAR, CHOCOLATE LIQUOR, CHOCOLATE LIQUOR PROCESSED WITH ALKALI [DUTCHED], COCOA BUTTER, BUTTER OIL, SOY LECITHIN, VANILLA EXTRACT EGGS, CONTAINS 2 PERCENT OR LESS OF: CORNSTARCH, EGL WHITES, DEHYDRATED SKIM MILK, SALT, SOY LECITHIN, MILKFAT, PEPPERMINT OIL, NATURAL FLAVORS AND LEAVENING (BAKING SODA)

Figura 8: Imagem final a ser processada

Fonte: adaptado de Bagadia e Mundra (2015)

Após separar as palavras que se referem aos ingredientes, o aplicativo faz uma busca no banco de dados e compara com as preferências do usuário. A proposta de melhoria nessa etapa seria o uso de um corretor ortográfico, para tratar nomes de ingredientes que foi reconhecido com algum erro, como, por exemplo, chooclate. Não houve proposta de melhoria para a base de dados, pois os autores não especificaram que base de dados eles utilizaram na aplicação.

Como a aplicação, FussyFood, foi desenvolvida para Android, um corretor ortográfico a ser utilizado pode ser o *framework* desenvolvido pela plataforma, *Spelling Checker Framework for Android*. O corretor usa as palavras obtidas da imagem como entrada no *framework*, gerando assim uma série de sugestões de palavras corretas.

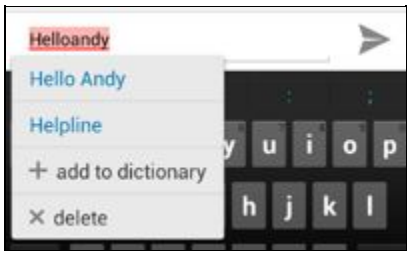

Figura 9: Spell checker em uso
Fonte: Spelling Checker Framework Documentation (2016)

O corretor pode ser requerido durante o funcionamento da aplicação e, quando o usuário terminar a operação, o corretor também é fechado. Se necessário, o próprio aplicativo pode fechar o corretor. A Figura 10 ilustra o funcionamento da ferramenta.

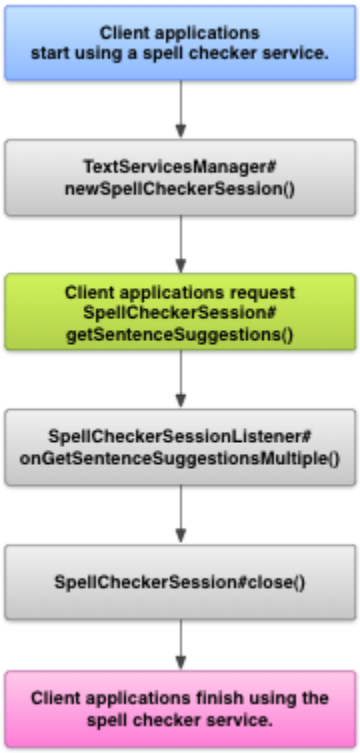

Figura 10: Interação do Spell Checker com a aplicação
Fonte: Spelling Checker Framework Documentation (2016)

De acordo com a Figura 10, ao detectar uma palavra com erro, o SpellCheckerService inicia, ele cria uma sessão. Ao ser iniciada, a aplicação solicita uma sessão para retornar sugestões para o usuário. Se o cliente decidir ignorar o corretor, o serviço é fechado.

Referências

Bagadia, S. e Mundra, R. “FussyFood: An Application to Navigate Food Allergies Dietary Restrictions”. Stanford University, California, 2015.

Bieniecki, W., Grabowski, S and Rozenberg, W. “Image preprocessing for improving ocr accuracy,” in Perspective Technologies and Methods in MEMS Design, 2007. MEMSTECH 2007. International Conference on, pp. 75–80, IEEE, 2007.

Spelling Checker Framework Documentation. Disponível em <http://developer.android.com/guide/topics/text/spell-checker-framework.html> Acesso em: 31 de março. 2016