An Automatic Calorie Estimation System of Food Images on a Smartphone

Koichi Okamoto and Keiji Yanai
The University of Electro-Communications, Tokyo
Introduction (1)

- Recording everyday meals is important.

Dietary health control
  e.g. Foodlog

Nutritional study
• Smartphone-based Food logging services

Foodlog

CaloNavi

Need to teach the amount of foods by selecting items

Food category recognition and manual vol. estimation

Dietary advices by professional nutrition human cost → Pay service
Objective

• Automatic calorie estimation system: **New system: CalorieCam**
• Standalone mobile system
  - very rapid recognition (less than one second)
  - all the processing inside a smartphone

*For better usability*  
e.g. user can give up eating

<table>
<thead>
<tr>
<th>dishes</th>
<th>Calorie Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pork cutlet</td>
<td>528kcal</td>
</tr>
<tr>
<td>Rice</td>
<td>172kcal</td>
</tr>
<tr>
<td>Miso soup</td>
<td>43kcal</td>
</tr>
<tr>
<td>Salad</td>
<td>27kcal</td>
</tr>
</tbody>
</table>

We focus on calorie intakes rather than carbohydrate.
Related works (1)

- Some food recognition systems can estimate food calories in the simplified way.
  - Indicating the amount of food manually
  - Counting the number of the food pieces

FoodCam
(Kawano et al. 2014)

GrillCam
(Okamoto et al. 2015)
Related works (2)

- Several work on automatic food volume estimation
  - 3D volume reconstruction by multiple images
  - 3D volume estimation by CNN
  - Using the size-known reference objects

*2 The TADAProject www.tadaproject.org
Related works (3)

- **Im2Calories [Myers et.al. 2015]**
  - All the processing inside a smartphone
  - CNN-based methods (categorization, 3D vol. est.)
  - Not yet released as a mobile application
    - Just presented some ideas for image-based calorie estimation.
Design of the proposed system

• All the processing inside a smartphone
  • Very quick calorie estimation from a food image

• Simple 2D-size-based calorie estimation

• Use CNN only for food categorization
  • Use the “DeepFoodCam” food recognition engine, while use conventional methods for segmentation.
Conditions to be assumed

- Uniform background
- Size-known reference obj.
- Taken from top (top-view photo)

8.5\text{cm} \quad 5.5\text{cm} \quad 46.8\text{cm}^2
Processing flow

1. Take a photo with a reference object
   - Extract a region of a ref. object
   - Extract regions of food items

2. Recognize a category of each detected food
   - Compare both region sizes
   - Pre-registered size: 46.8cm²
     - Pixels: 43149

3. Calculate food calories based on their relative size and food categories
   - Pixels: 68066

Food category and Calorie value

(size vs calorie)
Step 1: region extraction

1. Take a photo with a reference object
2. Extract a region of a ref. object
3. Extract regions of food items
4. Recognize a category of each detected food
5. Compare both region sizes
6. Calculate food calories based on their relative size and food categories
7. Food category and Calorie value
Dish region detection

1. Provide a meal photo
2. Detect a bounding box of a dish region based on edges
3. Detect a bounding box of a food region by k-means
4. Detect an accurate food region by GrubCut
Extraction of a given reference

- In the same way as food region extraction
- Any shape is possible, since only the size is important for estimating actual size of foods.

\[ 46.8\text{cm}^2 \]

\[ 180.5\text{cm}^2 \]
Step 2: food classification

1. Extract a region of a ref. object
2. Extract regions of food items
3. Recognize a category of each detected food

pre-registered size: $46.8cm^2$
pixels: 43149

Compare both region sizes
pixels: 68066

Calculate food calories based on their relative size and food categories

Food category and Calorie value

``Kinpira Burdock'']
CNN-based mobile food recognition engine (1)

- Use the “DeepFoodCam” mobile food rec. engine
- CNN: Network in Network (NIN)
  - No fully connected layer
  - AlexNet 60million ⇔ NIN 7.5million

CNN-based standalone mobile food recognition engine (2)

- Pre-training: ILSVRC1000 classes + 1000 food-related classes in ImageNet
- Finetuning: UECFOOD-100 (100 kinds of Japanese foods)

- Performance:
  78.8% (top1)
  95.2% (top5)

- Time:
  55.7ms / img (227x227) (iPhone7+)
  26.2ms / img (160x160) (iPhone7+)
Step 3: food calories estimation

1. Take a photo with a reference object
2. Extract a region of a ref. object
3. Extract regions of food items
4. Recognize a category of each detected food
5. Compare both region sizes
6. Calculate food calories based on their relative size and food categories
7. Food category and Calorie value
Estimation of real size of foods

- 2D-Size-based calorie estimation

Reference Object
#pixel: 43149
Real size: 46.8 cm$^2$ (known)

“Kinpira burdock”
#pixel: 68066
Real size: ?? cm$^2$

\[
s_{\text{food}} = \frac{s_{\text{ref}}}{#\text{pix}_{\text{ref}}} \times #\text{pix}_{\text{food}}
\]

\[
78.82 \text{ cm}^2
\]
Quadratic curve based calorie estimation

Use quadratic curve based estimation

\[ cal = a_i \times size_{food}^2 + b_i \times size_{food} + c_i \]

where \( a_i, b_i, c_i \) are trained params for \( i \)-th categories.

```
“Kinpira burdock”
#pixel: 68066
Real size: 78.8 cm²
a=0.001, b=0.5, c=37
```

```
“Beef bowl”
#pixel: 21043
Real size: 208 cm²
a=1.8, b=0.4, c=190
```

“Kinpira burdock”
74.4 kcal

“Beef rice bowl”
744.0 kcal
User Interface

- Implemented the proposed system as Android app.
- Took only less than 1 seconds for one food image.

Download (only Japanese version)
http://foodcam.mobi/calorie/
How to use the system

The processing time is less than 1 second!
Experiments

- Evaluation on calorie estimation accuracy
  - on PC

- User Study
  - using Android app.
Calorie estimation

• Dataset: 120 images (60 for training, 60 for eval)
  • 20 kinds of Japanese dishes with 3 different sizes
  • Prepare all the dishes and take photos in our lab
  • Use for training of calorie estimation parameters
    \((a_i, b_i, c_i: \text{quadratic curve parameters})\)

Chill shrimp 268kcal
French fries 454kcal
Spring rolls 428kcal
Sweet pork 292kcal
Takoyaki 241kcal
Gyoza 246kcal
Yakitori 165kcal
Boiled chikuzen 85kcal
Evaluation on Calorie Estimation

20 dishes for 3 images for each (totally 60 images)

<table>
<thead>
<tr>
<th>Mean avg err.</th>
<th>Mean SD</th>
<th>Mean relative err.</th>
<th>Mean relative SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>52.2 kcal</td>
<td>± 40.4 kcal</td>
<td>21.3%</td>
<td>± 0.82</td>
</tr>
</tbody>
</table>

Error: average absolute value of the difference
Relative error : average relative value of the difference
The cases of good estimation

<table>
<thead>
<tr>
<th>Input image</th>
<th>dish region</th>
<th>GT</th>
<th>Estimation.</th>
<th>Error</th>
<th>Relative err.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pork cutlet</td>
<td>586 kcal</td>
<td>559 kcal</td>
<td>27 kcal</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>Beef bowl</td>
<td>1322 kcal</td>
<td>1417 kcal</td>
<td>95 kcal</td>
<td>0.07</td>
<td></td>
</tr>
</tbody>
</table>
The cases of bad estimation

<table>
<thead>
<tr>
<th>Input image</th>
<th>dish region</th>
<th>GT</th>
<th>Estimation.</th>
<th>Error</th>
<th>Relative err.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Niku Jaga</td>
<td>170 kcal</td>
<td>122 kcal</td>
<td>48 kcal</td>
<td>0.28</td>
</tr>
<tr>
<td></td>
<td>Yakisoba</td>
<td>425 kcal</td>
<td>519 kcal</td>
<td>94 kcal</td>
<td>0.22</td>
</tr>
</tbody>
</table>
User Study

• Subjects: 12 students who have no knowledge on nutrition

• Task: estimate food calories with two systems: FoodCam (baseline) and CalorieCam (proposed)

• Target foods: beef rice bowl, croquette, salad
Baseline system: FoodCam (Kawano et al. 2015)

- Standard calorie amount depending on the dish categories
# Estimation by users

<table>
<thead>
<tr>
<th>Dish</th>
<th>GT</th>
<th>FoodCam Avg. err</th>
<th>FoodCam Avg. SD</th>
<th>Proposed Avg. err</th>
<th>Proposed Avg. SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beef bowl</td>
<td>962</td>
<td>-53.25</td>
<td>±209.79</td>
<td>-242</td>
<td>±55.10</td>
</tr>
<tr>
<td>Croquette</td>
<td>552</td>
<td>-242</td>
<td>±91.26</td>
<td>-47.08</td>
<td>±52.52</td>
</tr>
<tr>
<td>Salad</td>
<td>14</td>
<td>54.83</td>
<td>±36.28</td>
<td>4.86</td>
<td>±11.87</td>
</tr>
</tbody>
</table>

The average errors were reduced except for a beef bowl.

The standard deviation (SD) were also reduced, which means the proposed app achieved more stable estimation than FC.
Evaluation of usability by users

• 5-step evaluation on usability of the system

<table>
<thead>
<tr>
<th></th>
<th>Usability (5-step)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FoodCam</td>
<td>2.83 ± 0.80</td>
</tr>
<tr>
<td>Proposed (CalorieCam)</td>
<td>4.25 ± 0.72</td>
</tr>
</tbody>
</table>

• CalorieCam was much simpler, since it is an automatic calorie estimation system. (Foodcam is a manual system).
Conclusions

• We proposed an automatic calorie estimation mobile system implemented as an Android app which enable calorie estimation within one second.

• The system achieved automatic region extraction of dishes and a reference object.

• Avg. error 52 kcal, relative error 20% for 20 kinds of foods

• Higher usability than the baseline manual system.
Future work

• CNN-based region segmentation (+3D volume estimation)
  • For complicated background
• Add more food category
  • It is not practical to prepare and take photos of hundreds kinds of foods.
  • Use calorie-annotated recipe data on the Web