An Automatic Calorie Estimation System of Food Images on a Smartphone

Koichi Okamoto Keiji Yanai

Department of Informatics,
The University of Electro-Communications, Tokyo
1-5-1 Chofugaoka, Chofu, Tokyo 182-8585 JAPAN
{okamoto-k,yanai}@mm.inf.uec.ac.jp

ABSTRACT

In recent years, due to a rise in healthy thinking on eating, many people take care of their eating habits, and some people record daily diet regularly. To assist them, many mobile applications for recording everyday meals have been released so far. Some of them employ food image recognition which can estimate not only food names but also food calorie. However, most of such applications have some problems especially on their usability. Then, in this paper, we propose a novel single-image-based food calorie estimation system which runs on a smartphone as a standalone application without external recognition servers. The proposed system carries out food region segmentation, food region categorization, and calorie estimation automatically. By the experiments and the user study on the proposed system, the effectiveness of the proposed system was confirmed.

1. INTRODUCTION

In recent years, due to a rise in healthy thinking on eating, many people take care of their eating habits, and some people record daily diet regularly. To assist them, many mobile applications for recording everyday meals have been released so far. Some of them employ food image recognition which can estimate not only food names but also food calories. However, in most of the cases, the estimated calories are just associated with the estimated food categories, or the relative size compared to the standard size of each food category which are usually provided by a user manually. Most of the applications does not estimate calories based on the amount of foods.

Then, in this work, we try to implement a mobile application which can estimate more accurate calories by simply taking one meal photo. The proposed application employs food region segmentation as well as food category recognition for a given meal photo.

In this work, we propose a mobile application for food calorie estimation from a signal photo which runs on Android smartphone. Taking into account usability and mobility of a system, we implement the system as a stand-alone mobile application which can be used even in an airplane or in the underground where the Internet connection is not available.

To estimate food calorie from a single image, a user needs to register a size-known reference object in advance and to take a food photo with the registered reference object. As a reference object, we assume a personal belonging which we are always carrying such as a wallet and a credit-card-size card in a wallet (except a smartphone for taking a meal photo). After taking a meal photo with a reference object, the system carries out segmentation of food items and the pre-registered reference object. Because the real size of the reference object is known (e.g. In case of a credit-card-size object, the size is 85.6mm x 54mm.), the system can estimate the real size of each detected food items. By using the estimated real size and the equations to calculate food calorie from their size, the system finally estimates the calorie of the food items in the real photo. Figure 1 shows an example of usage and a screen-shot of the proposed system.

In the rest of this paper, we describe related work in Section 2. We explain an overview of the proposed mobile system in Section 3 and the detail of the methods in Section 4. In Section 5, we report the experimental results including evaluation of food calorie estimation and user study on the proposed system. Finally we discuss the proposed system and the experimental results in Section 6, and conclude this paper in Section 7.

2. RELATED WORK

Various systems on automatic food calorie estimation have been proposed so far. We review them in this section.

Miyazaki et al. [1] proposed an image-search-based method to estimate food calories. They prepared a large number of food photos annotated with calorie values by dietitians, and they searched for the similar food photos to a given meal photo from their database. Although example-based calorie estimation was a novel approach, they did just search for similar looking images and transfer the annotated calo-

MADiMa'16, October 16 2016, Amsterdam, Netherlands

ISBN 978-1-4503-4520-0/16/10. . . $15.00

DOI: http://dx.doi.org/10.1145/2986035.2986040

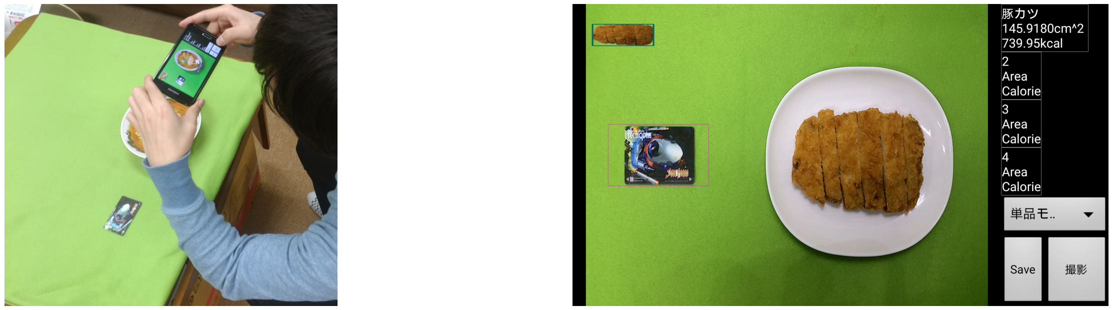

Figure 1: (Left) A user is taking a food photo with the pre-registered reference object (a used pre-paid card). (Right) A screen-shot

rie values. Therefore the estimation accuracy was not high. Most of the other works performed 3D volume estimation or at least 2D size estimation for accurate food calorie estimation.

Chen et al. [2] proposed an image-based food calorie estimation method using RGB-D images captured by depth cameras such as Kinect. Since a depth camera is not common and consumer smartphones have no depth cameras, it was difficult to implement the method on mobile devices.

Kong et al. [3] proposed a mobile application to estimate food calories from multiple images, "DietCam". They carried out segmentation and food item recognition, and in addition reconstructed 3D volumes of food items and calculate food calories based estimated volumes. 3D reconstruction was performed with SIFT-based keypoint matching and homography estimation which were a standard method of 3D stereo vision. They supported recognition and calorie estimation of leftover foods as well. Their system were a fully client-side system. Our work is similar to this work except for using only single images. In addition, we use deep-learning-based food recognition.

Xu [4] employed 3D model based food volume estimation from a single food image. They classify food items into some types of representative shapes such as a cylinder and a box, and they fitted corresponding 3D models to food items for estimating food volumes. They claimed that 3D model fitting was effective for most of the food items but not for some foods such as vegetables and breads. They also examined multiple view food volume estimation using the shape-from-silhouettes methods, which required many different-view images (e.g. 10 images).

Pouladzadhe et al. [5] proposed a food calorie estimation system which needed two dish images taken from the top and the side and used a thumb of a user as a reference object. Their method to estimate volumes were calculated by multiplying the size of food items estimated from the top-view image by the height estimated from the side-view image, which was relatively a straight-forward way. Different from their work, in our work, we assume that the height of food portion correlates with the size of foods and food categories, and we estimate calories of food items directly from the food sizes estimated from the top-view image. In our system, as a reference object, we use a personal belonging which we are always carrying such as a credit-card-size card or a wallet, and we use only a single mean photo taken from the top with a reference object.

Although it is reasonable to use multiple images [3, 5] to estimate food volumes more accurately which is sometimes troublesome, we prioritize usability as a mobile application rather than accurate estimation of food calories.

Myers et al. [6] proposed "Im2Calories" which was a food calorie estimation application for Android smartphones. Although this projects were carried out by the Google Research and they made the press-release on this project about one year ago, Android applications as well as the dataset which they announced they would make to the public in their paper have not been released yet. They employed state-of-the-art CNN-based segmentation [7] and CNN-based 3D volume estimation from 2D single images [8] in addition to CNN-based food category recognition. However, the experiments described in the paper was small-scale, which seemed far from practical use for common consumers.

All the above-mentioned systems employ food dish recognition to estimate food calorie. On the other hand, Okamoto et al. proposed GrillCam [9] which monitored eating actions of a user continuously. When a user conveys foods to his/her mouth, the system detects the moment, recognizes food categories and estimates rough volumes and rough calorie intake from the video stream. The estimated food calorie intake are accumulated during the meal, and after finishing the meal the user can get to know categories of eaten food items, their amounts and the total amount of calorie intake. Although this is based on the very different approach from the other works, it has drawback that the estimation biases tend to be accumulated.

3. OVERVIEW OF THE SYSTEM

In this paper, we propose a food calorie estimation system running on a consumer smartphone. In the proposed system, a user needs to take only a single meal photo from the top (right overhead) with a pre-registered reference object. After taking photo, the system segments out food items, classify them into the pre-defined food categories, and finally estimates food calories of each of the detected food items.

The processing steps of the proposed system are as follows (see Figure 2 as well):

1. Take a meal photo from the top with a reference object.
2. Extract regions of food items and the reference object.
3. Recognize food categories of the detected food items.
4. Calculate the real sizes of the foods and food calories based on the pre-trained relation between sizes and calories.

In our mobile system, we assume that a meal photo is taken from the overhead, which enables us to omit correction of trapezoidal distortion. We assume that background of food dishes are not textured but uniform for making segmentation easy. In addition, we also assume that the size of the reference object is known. In fact, in the application we implemented, a user can register the size of a reference object which is expected to the user's personal belonging such as a credit-card-size card in the wallet or a wallet. Note that we do not recommend to use a credit card as a reference object for the security reason. Instead we recommend to use a used pre-paid card such as a pre-paid card for a pre-paid mobile phone, Internet shopping, and Internet games.

To extract regions of food items and a reference object, firstly we estimate rough position of dishes based on edge detection results, and secondly we apply color-pixel-based k-means clustering for estimating bounding box of food regions. Finally we apply GrabCut [10] with the detected bounding box as a segmentation seed.

For food classification, we use a Convolutional Neural Network (CNN) based recognition engine which run on a consumer smartphone [11] with high accuracy. It takes only 0.2 seconds on a consumer Android phone.

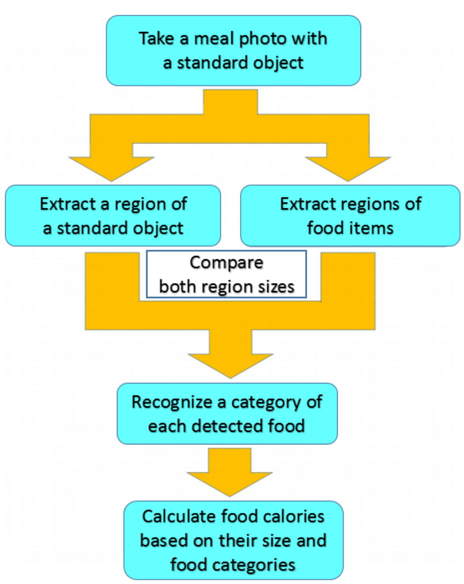

Figure 2: The flow of the proposed method.

4. DETAIL OF THE METHODS

In this section, we explain the detail of each step.

4.1 Food Region Segmentation

In this subsection, we describe the method to extract food regions. Firstly, we estimate rough location of dishes by edge detection, secondly we extract a bounding box of a food region by color-pixel-based k-means clustering, and finally we apply GrabCut [10] using the estimated bounding box to obtain an accurate food region. Figure 3 shows the flow of these processing.

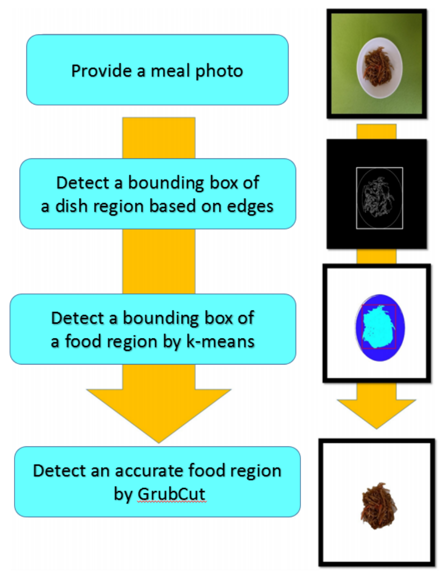

Figure 3: The processing steps of food region extraction.

4.1.1 Dish Region Detection

In the first, we detect dish regions from a given meal photo. We apply edge detection to a given image after slight smoothing, and extract bounding boxes which surrounds edges as shown in Figure 4. Note that we expect that all the dishes and the reference object are separated and do not touch with each other. After this step, all the processing are applied to each bounding box detected in this step.

By applying the GrabCut method with this bounding box, we can extract dish regions correctly as shown in the right side of Figure 5. Without the bounding box detected by edges, we obtain incomplete regions as shown in the left side of Figure 5.

4.1.2 Food Region Detection

In the previous step, we extracted not food regions but dish regions. As the next step, we need to detect food regions which are expected inside the detected dish region for food calorie estimation. To do this, we adopt two step processing consisting of rough segmentation by color-pixel-based k-means clustering and GrabCut.

Figure 4: An example of the detected result of the bounding box surrounding edges.

Figure 5: (Left) GrabCut results without the bounding box. (Right) GrabCut results using the detected dish bounding box.

Although a graph-based segmentation method, GrabCut [10], is a highly effective method to extract a specific object, it needs to be given a bounding box which shows rough location of the target object. Since we aim at automatic calorie estimation, we need to estimate a bounding box by other methods before applying GrabCut. To do that, we adopt a simple region segmentation method which is based on k-means clustering [12].

We simply perform k-means with $k = 3$ for all the pixels within the target dish bounding box, which means we expected all the pixels into three kinds of the regions consisting of “foods”, “dish”, and “background”. Note that we apply median smoothing before applying k-means in order to prevent small noise regions from being generated. We assume the order of the regions from the center to the outside are always “foods”, “dish”, and “background”. Figure 6 shows an example of the segmentation result by k-means.

4.1.3 Segmentation with GrabCut

In the previous step, we obtained a food region by k-means. Since this is a simple method, the result tends to be incomplete. Therefore, for more accurate estimation, we adopt GrabCut [10] which is graph-based region segmentation and employs energy minimization based on graph cut. Since it was originally proposed as an interactive region segmentation method, minimum supervision is needed. Here, as a supervision signal, we use a bounding box which indicates that the target object is inside it.

We estimate the bounding box inscribed with the detected food region as a seed of GrabCut, and apply GrabCut with this seed. We can expect more accurate food region is ex-

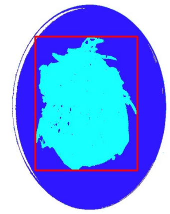

Figure 6: An example of a segmentation result by k-means. (Skyblue: food region, Blue: dish region, White: background)

tracted. Accurateness of food region extraction is important for image-based food calorie estimation.

Figure 7: (Right) The final food region estimated by the three-step processing. (Left) Original meal photo.

4.2 Extraction of a Reference Object

To estimate a real size of a food region, we assume that a size-known reference object is taken in a meal photo which we like to estimate food calories at the same time. In this work, we assume that a meal photo is taken from the overhead, which enables us to omit correction of trapezoidal distortion. Therefore, we can use not only a rectangular object such as a card but also any shaped objects such as a wallet and a smartphone case.

To enable us to extract any shaped objects from an image, we use GrabCut in the same way as food region extraction. Figure 8 shows a wallet as an example of a reference object, and the result of the detected region. In the current implementation, a reference object is assumed to be located in the left of a given photo.

4.3 Food Category Recognition

The proposed system employs a Convolutional Neural Network based food recognition engine which completes all the processing inside a smart phone and does not needs to communicate with external recognition servers.

The proposed CNN-based food recognition engine employs “Network in Network (NIN)” [13]. As basic CNN architectures for object recognition, AlexNet [14], Network-in-Network (NIN) [13], GoogLeNet [15] and VGG-net [16]

Figure 8: (Left) An example of a reference object, wallet. (Right) The extracted region of the wallet.

are commonly used. Because both AlexNet and VGG-net have fully-connected layers which requires a huge number of weight parameters, 59M for AlexNet and 129M for VGG-16, they are not appropriate for mobile implementation. Although both GoogLeNet and NIN has around 7M parameters, GoogLeNet is quite complicated which consists of many "Inception modules" each of which consists of 5x5, 3x3, 1x1 conv layers and pooling layers. With these reason, we selected NIN as an architecture for mobile food recognition. Regarding mobile implementation on CNN-based recognition engine, we followed the work on efficient mobile implementation [11].

We trained the NIN with the ILSVRC 2012 dataset consisting of one million images with 1000 categories from scratch, and fine-tuned it with the UEC-FOOD100 dataset [17] using the Caffe framework [18]. The recognition engine can recognize a 227x227 image with 250ms on SAMSUNG Galaxy S5 (QuadCore 2.5GHz). The recognition accuracy was much improved compared to non-CNN-based mobile food recognition. Our CNN-based engine achieved 75.0% and 93.7% for the top-1 and top-5 accuracy, respectively, while non-CNN based "FoodCam" [19] achieved 65.3% and 86.7%.

4.4 Calorie Estimation

Although several existing works on food calorie estimation [3, 4, 5] carried out 3D reconstruction of food shapes for estimating calories, they required multiple-view images to do that. On the other hand, we use a single-view image for calorie estimation. Being different from the prior works, in our work, we assume that the height of food portion correlates with the size of foods and food categories, and we estimate calories of food items directly from the food sizes estimated from the top-view image. To do that, we use not simple linear estimation but quadratic curve estimation from the 2D size of foods to their calories. The quadratic curve of each food category is estimated based on the training data annotated with real food calories independently.

The real size of food regions is estimated based on the size of the reference object. Since the real size of reference object is known, the real size of foods, F_r, can be obtained by the following equation:

$$F_r = S_r * \frac{F_p}{S_p}, \tag{1}$$

where F_p represents the number of pixels of the region of the target food item, S_p represents the number of pixels of the region of the reference object, and S_r represents the real size of the top view of the reference object which is expected to be registered in advance.

After calculating the real size of the target food item, we estimate its calorie value. To do that, we need to convert the 2D top-view size of the food to the calorie value. For the planar-shaped foods such as a stake and a hot-cake, we can assume that food calorie is proportional to the size of a food item. However, for the foods the shape of which are 3-D such as rice bowl and miso soup, this assumption does not fit. Therefore, in this work, we use not simple linear estimation but quadratic curve estimation from the 2D top-view size of foods to their calories. The quadratic curve of each food category is estimated based on the training data annotated with real food calories independently. The calorie value is approximately calculated by the following equation:

$$C = a_i * F_r^2 + b_i * F_r + c_i, \tag{2}$$

where a_i, b_i, c_i are constant values for the i-th food category which are trained from the calorie-annotated food image dataset in advance. In the experiments, we estimated a_i, b_i, c_i using a polynomial curve fitting method in a least-squares sense.

5. EXPERIMENTS

5.1 Food Dataset Annotated with Calorie Values

To make experiments for image-based food calorie estimation, at first, we prepared a food image dataset in which each image was annotated with a food calorie value.

To create a calorie-known dataset, we bought 20 kinds of Japanese foods sold in the packages indicating food calories at supermarkets, convenience stores and lunch box shops. We prepared two packages for each food category, because we needed to create a training dataset and a test dataset independently. We dish up them in three kinds of the amounts (large, medium, small) on plates or bowls, and took food photos with uniform backgrounds. Figure 9 and 10 shows three size dishes of "croquette" and "meat and potato mix" ("Nikujaga" in Japanese). The amounts of foods to place on dishes were measured by a kitchen scale. We assumed a food calorie was proportional to the weight of a food, and calculated food calories of three different amounts of each food item according to its weight and the total calorie shown on the package. Totally we collected totally 120 food images containing 20 categories with three different amounts. Half of 120 images were used for training of Equation 2 (estimating a_i, b_i, c_i), and the rest were used for evaluation.

Figure 9: Croquette. (Left: small, Middle: medium, Right: large)

Figure 10: Meat and potato mix (Japanese food "Nikujaga"). (Left: small, Middle: medium, Right: large)

5.2 Evaluation on Calorie Estimation

We made experiments for evaluating the proposed method of image-based calorie estimation using 60 food images in the test dataset. Table 1 shows the average absolute errors, the relative average absolute errors and standard deviations of all the estimated results for 60 test images.

Table 1: Absolute and relative average errors and standard deviations (s.d.) of all the estimated results.

avg. absolute err.	abs. s.d.	relative avg. err.	rel. s.d.
52.231	40.401	0.213	0.823

In the experiments for 20 kinds of the target foods, the absolute error was 50kcal, and the relative error was 20%, which were promising results.

5.3 User Study

We made user study on calorie estimation on a real setting and the usability of the proposed system as well. As a baseline system, we used a mobile food recognition application, "FoodCam", proposed by Kawano et al. [19]. Although "FoodCam" can recognize food categories, it cannot estimate the amount of foods. Instead, on "FoodCam" we can adjust the amount of the target foods by using the slider on the GUI manually, and get to know the food calorie corresponding to the given amount.

We asked 12 subjects who had no special knowledge about nutrition to use both our system and the baseline system to estimate calories for three kinds of actual foods. We used "beef rice bowl", "croquette", and "salad" as target food items. We evaluated errors between estimated calories and real calories and compared the results by two systems. Table 2 shows errors and standard deviations on the results by both the systems.

Although the average error of the estimated calorie of "beef rice bowl" by the propose system became larger than the baseline, regarding "croquette" and "salad" the proposed system reduced the average error greatly compared to the baseline.

In addition, we asked the subjects to evaluate usability of both systems in five step evaluation. The larger values means the better evaluation. Table 3 shows the results which means that the larger value is the better evaluation.

Table 3: Five-step evaluation on usability for recording meals with calorie information by 12 subjects.

	How easy to take meal records
Baseline	2.83 ± 0.80
Proposed	4.25 ± 0.72

As results, the proposed system was evaluated as much easier to use with a large margin than the baseline system. This is partly because our system can estimate food calories automatically simply by taking a meal photo from the top, while the baseline needs to adjust a slider which indicates relative amounts of the target foods.

Table 2: Estimation errors by the baseline system, "FoodCam" and the proposed system.

		baseline		proposed	
food name	real value (kcal)	avg. err.	SD	avg. err.	SD
beef rice bowl	962	-53.25	209.79	-242	55.10
croquette	552	-242	91.26	-47.08	52.52
salad	14	54.83	36.28	4.86	11.87

6. DISCUSSION

In the experiments, the absolute average error was 52.231kcal, and the relative average error was 21.3%. This 21.3% error was not a bad result, because the ministry of consumer in the Japanese Government defines 20% error as an allowable error on calorie values shown in the food packages. Basically calorie estimation by looking is difficult task even for experienced dietitians. Therefore, 20% error can be regarded as promising, allowable, and probably useful from the practical point of view.

In this work, we have prepared only 120 calorie-annotated food photos. This is not enough. However, it is not easy to prepare calorie-annotated food photos, although there are so many food photos on Web which have no information on calorie. To build such dataset, we have to prepare real foods and take their photos, which is extremely difficult to do this in the large-scale way. Although Google announced they would release a calorie-annotated food photo dataset in their paper [6], they have not released such dataset yet half year after publishing the paper. (It is completely unacceptable that they cheated both reviewers and readers.) It is the biggest issue how to build a large-scale calorie-annotated food photo dataset.

7. CONCLUSIONS

In this paper, we proposed an image-based calorie estimation system which runs on a consumer smartphone without external recognition servers. The system estimate food calories automatically by simply taking a meal photo from the top with a pre-registered reference object.

In the step to detect food regions, we used edge-based dish localization and k-means-based dish bounding box estimation, and finally we applied GrabCut for accurate food region estimation. To recognize each of the detected food regions, we adopted the CNN-based food recognition engine which recognizes food photos in around 0.2 seconds. To estimate calorie, we used not simple linear estimation but quadratic curve estimation from the 2D size of foods to their calories. The quadratic curve of each food category is trained based on the training data annotated with real food calories independently. As results, in the experiments, we achieved the absolute average error, 52.231kcal, and the relative average error, 21.3%, regarding food calorie estimation with 60 test images annotated with real food calorie values. In addition, we obtained positive evaluation by the subjects regarding the usability of the proposed system compared to the baseline system.

In this work, we adopted relatively simple segmentation methods because of taking into account mobile implementation. Therefore, it is sometime difficult to treat a meal photo with non-uniform background shown as Figure 11. For future work, we like to introduce more sophisticated segmentation methods, hopefully state-of-the-art CNN-based methods such as [20]. In addition, we like to introduce planar calibration by using a rectangular card as a reference object, which relaxes the assumption of taking a meal photo from the overhead.

Figure 11: A meal photo in the real setting with non-uniform background.

8. REFERENCES

[1] T. Miyazaki, De. S. G. Chamin, and K. Aizawa, "Image-based calorie content estimation for dietary assessment," in *IEEE International Symposium on Multimedia*, pp. 363–368, 2011.

[2] M. Chen, Y. Yang, C. Ho, S. Wang, S. Liu, E. Chang, C. Yeh, and M. Ouhyoung, "Automatic chinese food identification and quantity estimation," in *Proc. of SIGGRAPH Asia Technical Briefs*, p. 29, 2012.

[3] F. Kong and J. Tan, "Dietcam: Automatic dietary assessment with mobile camera phones," in *Proc. of Pervasive and Mobile Computing*, pp. 147–163, 2012.

[4] C. Xu, Y. He, N. Khannan, A. Parra, C. Boushey, and E. Delp, "Image-based food volume estimation," in *Proceedings of the international workshop on Multimedia for cooking & eating activities*, pp. 75–80, 2013.

[5] P. Pouladzadeh, S. Shirmohammadi, and R. Almaghrabi, "Measuring calorie and nutrition from food image," *IEEE Transactions on Instrumentation and Measurement*, pp. 1947–1956, 2014.

[6] A. Meyers, N. Johnston, V. Rathod, A. Korattikara, A. Gorban, N. Silberman, S. Guadarrama, G. Papandreou, J. Huang, and K. P. Murphy, "Im2Calories: Towards an automated mobile vision food diary," in *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 2015.

[7] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," 2014.

[8] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pp. 2650–2658, 2015.

[9] K. Okamoto and K. Yanai, "Grillcam: A real-time eating action recognition system," in *Proc. of International Conference on Multimedia Modelling (MMM)*, 2016.

[10] C. Rother, V. Kolmogorov, and A. Blake, "GrabCut: Interactive foreground extraction using iterated graph cuts," in *Proc. of ACM SIGGRAPH*, pp. 309–314, 2004.

[11] K. Yanai, R. Tanno, and K. Okamoto, "Efficient mobile implementation of a cnn-based object recognition system," in *Proc. of the ACM International Conference on Multimedia*, 2016.

[12] J. A. Hartigan and M. A. Wong, "A K-means clustering algorithm," *Applied Statistics*, vol. 28, pp. 100–108, 1979.

[13] M. Lin, Q. Chen, and S. Yan, "Network in network," *In Proc. of International Conference on Learning Represenation Conference Track*, 2013.

[14] A. Krizhevsky, S. Ilya, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 1097–1105. 2012.

[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. of IEEE Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[17] "UEC FOOD 100," http://foodcam.mobi/dataset.html.

[18] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proc.of ACM International Conference Multimedia*, pp. 675–678, 2014.

[19] Y. Kawano and K. Yanai, "FoodCam: A real-time food recognition system on a smartphone," *Multimedia Tools and Applications*, vol. 74, no. 14, pp. 5263–5287, 2015.

[20] W. Shimoda and K. Yanai, "CNN-based food image segmentation," in *Proc. of International Workshop on Multimedia Assisted Dietary Management (MADIMA)*, 2015.