Pattern Recognition 47 (2014) 1941–1952

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

A mobile, lightweight, poll-based food identification system

Luciano Oliveira [a,*], Victor Costa [a], Gustavo Neves [b], Talmai Oliveira [b], Eduardo Jorge [b], Miguel Lizarraga [c]

[a] Intelligent Vision Research Lab, Federal University of Bahia, Brazil
[b] Reconcavo Institute of Technology, Brazil
[c] Samsung Institute of Development in Informatics, Brazil

ARTICLE INFO

Article history:
Received 30 October 2012
Received in revised form
4 December 2013
Accepted 11 December 2013
Available online 19 December 2013

Keywords:
Food identification
Multi-hypothesis segmentation
Multi-ranking classification
Mobile device

ABSTRACT

Even though there are many reasons that can lead to people being overweight, experts agree that ingesting more calories than needed is one of them. But besides the appearance issue, being overweight is actually a medical concern because it can seriously affect a person's health. Losing weight then becomes an important goal, and one way to achieve it, is to burn more calories than ingested. The present paper addresses the problem of food identification based on image recognition as a tool for dietary assessment. To the best of our knowledge, this is the first system totally embedded into a camera-equipped mobile device, capable of identifying and classifying meals – that is, pictures which have multiple types of food placed on a plate. Considering the variability of the environment conditions, which the camera will be in, the identification process must be robust. It must also be fast, sustaining very low wait-times for the user. In this sense, we propose a novel approach, which integrates segmentation and learning on a multi-ranking framework. The segmentation is based on a modified region-growing method which runs over multiple feature spaces. These multiple segments feed support vector machines, which rank the most probable segment corresponding to a type of food. Experimental results demonstrate the effectiveness of the proposed method on a cell phone.

1. Introduction

Let us imagine the following situation: a person goes to the nutritionist with the goal of losing weight; the professional asks them to take notes on how much calorie intake they ingest and the quality of each food. In practice, to accomplish this the person would need to: estimate the volume of food, identify the ratio of volume-calorie-food type, calculate and quantify, by using a volume-calorie table, the value of the intake, and finally, document the results in some kind of notebook or diary. Now consider a cell phone with a camera able of accomplishing, in few seconds, that same long and boring process of estimating, checking, calculating and documenting. Certainly time would be saved if there was an application that could do this quickly and automatically, requiring only a single picture of the meal from the user. From an intelligent image recognition method, all required steps would be automatically completed, attending the dietician's original request.

With the advance of mobile device technology, image pattern recognition algorithms can now be embedded in very small devices. In order for users to be able to take this diet-manager-on-a-chip anywhere, the image identification system would need to be robust to luminosity changes, as well as being capable of handling pictures taken with normal amounts of instability. Tackling all these issues, however, demands balancing the inherent trade-off between computational cost and image classification performance, which is definitely not a simple task; especially since an attractive software application should give accurate results within a few seconds in order to avoid bothering the end-user with long waits.

Food identification as an application of image pattern recognition is a very recent research field. As a matter of fact, there exists a limited number of previous research in this field, especially those that are focused on fast food identification [1–3]. To the best of our knowledge, our approach is the first one with the goal of recognising meals – that is, prepared plates of food – totally embedded in a camera-equipped mobile device. Here, the very first objective is to identify some previously trained food types from an overhead picture of the dish. Fig. 1 illustrates an example of food identification. In the figure, foods in the dish were segmented semi-automatically, with just the centroids of the segments given by the end-user. The food labels were achieved by a support vector machine (SVM) learner, integrated with the segmentation method.

* Corresponding author. Tel.: +55 71 3283 9472.
E-mail addresses: lrebouca@ufba.br (L. Oliveira), vfcosta@ufba.br (V. Costa), gustavo.neves@reconcavo.org.br (G. Neves), talmai@talmai.com.br (T. Oliveira), emjorge@reconcavo.org.br (E. Jorge), m.lizarraga@samsung.com (M. Lizarraga).

0031-3203/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2013.12.006

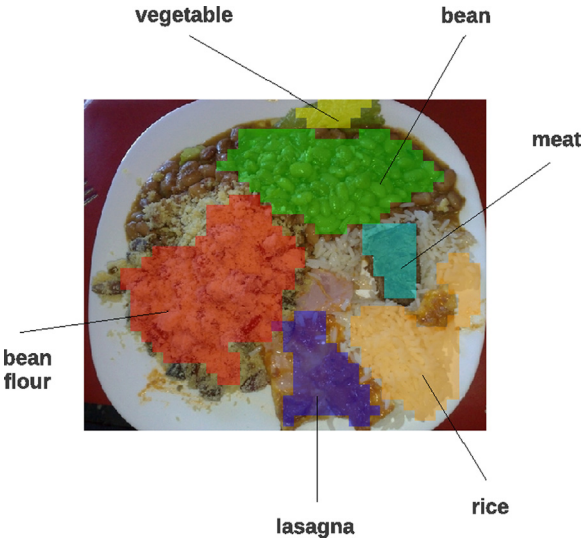

Fig. 1. Example of food segmentation and identification in a mobile device equipped with a high resolution camera.

Identifying prepared meals (in contrast with packaged or individual pictures of foods) from a single overhead image is a very challenging task because of the diversity of textures involved, as well as, a strong mixture of elements. To accomplish the task, we propose a new method which integrates image segmentation and multi-ranking classification. Particularly, the segmentation takes place by a modified region growing method over different feature spaces. The choice of the final segment is made by a polling method over gaussian SVMs applied in functional elements (patches) of the food image within all hypothesised segments. A patent, with the proposed method, was published in [4].

The rest of this paper is organised as follows: Sections 1.1 and 1.2 respectively describe related works and the outline of our proposed solution, preparing the reader to the next more thorough sections, describing each element of the framework, i.e., Sections 2 and 3. Section 4 discusses computational time aspects, in a cell phone, of the proposed system. Experimental results are addressed in Section 5. Finally, some discussion and remarks are drawn in Section 6.

1.1. Related work

Food image recognition is a relatively young research field with few published papers. One of the earlier works dates back to 1996, and proposes a system called VeggieVision, intended to ease the process of food checkout in supermarkets and grocery stories [5]. For that system, Bolle et al. [5] conceived a camera-equipped apparatus which makes use of a trainable recognition system to identify food images. The identification system is comprised of a segmentation module, which makes the work by extracting texture and colour features, and a nearest neighbour-based image classifier. According to the authors, on average, 84% of produce items were correctly identified, or 96% of produce items were the correct choice for the top four classifier hypotheses; this hit rate was achieved on a 1450 image set as the training and, at the same time, testing sets, using a leave-one-out cross validation strategy. The reason of the high classification rate could reside in the fact that the testing images have just contained just few food types, with very uncluttered backgrounds and colourful foods.

Shroff et al. proposed a cell phone-based system to identify fast food [1]. A neural network is used, relying on colour and size statistics of the food images as input; after classification, contextual information is applied considering the period of the day in which the food is being ingested. It also uses geolocalisation, that is, global positioning system (GPS) to further filter the food type considering the location where it was ingested. The average performance of that proposed system was 90%, over 200 images obtained via a cell phone; as in [5], performance evaluation was also accomplished by a leave-one-out cross-validation. After all, the food recognition process is performed in a remote server, sending the resulting information to the end-user's cell phone.

Chen et al. have introduced the first image dataset related to (fast) food recognition, called Pittsburgh Fast-Food Image Dataset (PFID) [2]. The PFID dataset contains three instances (images in restaurants, images in the lab and stereo pair images) of 101 meals, which were benchmarked by using scale invariant feature transform (SIFT) and colour histograms as features, as well as SVM as a classifier. The results indicated 90% of hit rate for salads, and less than 69% for the other food types. Making use of the PFID datasets, Yang et al. [3] addressed the fast food recognition problem by the computation of pairwise statistics between local features, which were calculated over a soft pixel-level food segmentation. This latter corresponds to a labelling task of all segmented pixels in the food image into nine categories (eight foods and the background). After that, the spatial relationship between the labelled pixels was computed by using the so-called pairwise statistics. The best results over PFID images were 80% of recognition rate on a major category-level confusion matrix (i.e., grouping food types into major categories).

As in [1], the work of Fengqing et al. [6] proposed a cell phone-based system to recognise fast food using a remote server to process the food recognition. The goal of that system is not only to recognise food types, but also to estimate the amount of food eaten; for that, the volume of food is estimated before and after food intake. Food image segmentation was performed fully automatic, with a hybrid method comprised of connected component analysis, active contours and normalised cuts. For food classification, Gabor filters were extracted, and then an SVM was used to classify these features. Volume estimation consisted of a camera calibration process and 3D reconstruction. Such heavy and long processes probably would lead to a system with a very computational cost, although nothing was said about system processing time. According to the experiments, over more than 3000 images, that proposed system achieved 84.2%, 91.7% and 95.8%, respectively having 10–90, 25–75 and 50–50 training and testing dataset distributions.

Although all those pioneers in food recognition have presented elegant and accurate systems, to the best of our knowledge, no one conceived a fully embedded mobile device framework to identify meals until now. The advantage of fully embedded mobile device systems is not only full ubiquity but also no dependency of network availability. Our proposed system has been inspired by recent advances in the image segmentation field [7–9] including comprehensive identification (structuring segmentation and learning) methods, colour-texture descriptors [10–12], and semantic information fusion [14]. Parallelisation techniques were applied in order to speed up the recognition system which led to a lightweight algorithm that could be executed on an energy and processor constrained mobile device.

1.2. Outline of the food identification system

Fig. 2 illustrates the proposed food identification structure. It is known that texture features are very dependent on the geometry of the image. In other words, depending on how close the camera is from the object, texture can be seen as different patterns. To cope with this problem, the user is instructed to highlight the food content using a virtual target overlay (a red circle on the mobile device screen); with that, it is possible to keep a predefined

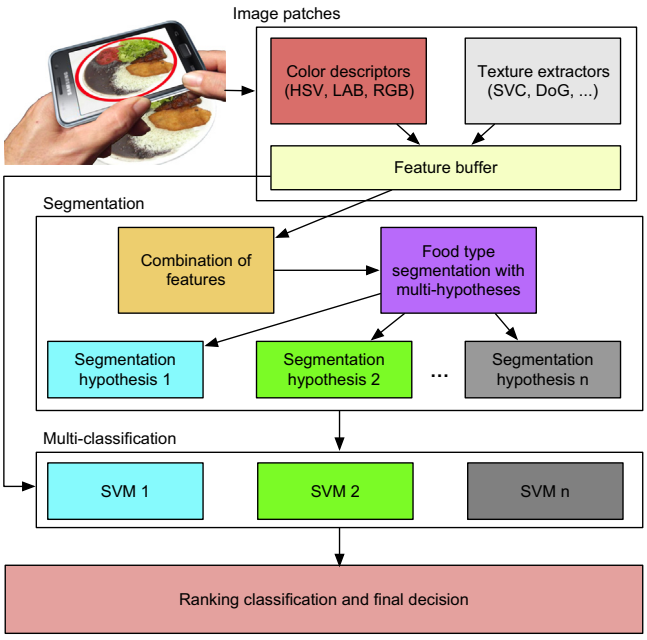

Fig. 2. Outline of the food identification system.

distance from the object. Given the overhead picture of the meal on the plate, the first step is to segment the food elements owing to the image plate; for that, an adaptive region growing segmentation method is performed over different feature spaces, in small patches (square image regions). The goal here is to cast multiple hypotheses of segmentation. Since the system was conceived to be used on any conditions of illumination environment and of use (obviously respecting certain limits), this stage supplies the classifier with more than one estimation of food area. As a matter of fact, by doing so, we are hopefully able to deal with a list of idiosyncrasies with respect to feature extraction (discussed in more details in Section 2), which will be affected by lightning conditions.

For each hypothesised segment, image patches, similar to the ones used in the segmentation step and totally bounded within the segment, are extracted. Those windows represent the functional elements where the image classifier will execute its function. To speed up the process of food identification, the size of those patches was chosen to be multiple of the segment patch (used by the region growing segmentation). This way, the extraction area for classification features contains, in fact, four window segments, cached into a special buffer to avoid recomputing features which are used whether in segmentation or in classification.

So once the SVM classifier is performed on each classification patch, it provides a score for each window which is converted into a probability value. To rank the final probabilities for each group[1] of patches within the segment, the average probability is calculated from the multi-classifier SVM.[2]

2. Image food segmentation

The segmentation of the image taken from each meal is initially based on a region growing approach.[3] The main idea of the method relies on grouping pixels or sub-regions into wider regions, based on some sort of criteria. The starting point of the algorithm consists of choosing a seed pixel, which is used as an initial criterion to make the regions to grow by adding homogeneous pixels at a certain neighbourhood. The initial seed is chosen by the user after taking the picture of the food.

We will now discuss the features relied upon in our adaptive region growing algorithm.

2.1. Features used as similarity properties of the growth

The segmentation method, as it was slightly aforementioned and will be described in the next section, makes use of some criterion to put similar pixels together. At the bottom of the segmentation pyramid, the choice of the features will influence how decisions are made in higher levels. In this sense, we have chosen features which could demonstrate robustness with regard to lighting changes and blurring effects.

Table 1 shows 10 types of features used in the process of feature evaluation. In the first column, there is an index for the feature; in the second column, a description of the feature used; in the third column, the similarity distance is shown for each one of those features, and, finally, in the last column, we have the thresholds to halt the region growing segmentation (the choice of this parameter is experimentally demonstrated in Fig. 7). The composite distances (i.e., spatial variation coefficient (SVC)[10, + colour descriptor]) are comprised of a balance between two single distances, weighted by an $\alpha=0.3$ parameter. The value of this α parameter was experimentally found with the goal of giving more emphasis to the SVC texture values, which have lately showed more efficient results in the segmentation (refer to Section 2.3 for more details). This way, colour spaces were kept to complement texture information. Single similarities were computed via Euclidean distances, except for HSV colour components, in which case the method in [15] was applied. It is important to note that the SVC values are calculated always over RGB colour channels [10].

Referring to Table 1, it is worth noting that colour space information works as an auxiliary feature to disambiguate different texture values. For completeness, information on how to compute SVC texture features will now be presented. If needed, the reader may obtain more information by reading the original manuscript found in [10].

Features based on SVC quantify a texture through spatial variations of colour and intensity. It takes into account not only the intensities of the pixels, but also the relationships among them. To compute SVC features, first one must define, for each colour channel, the square image patches where the set of pixel intensities are calculated according to the Manhattan distance, *MD*, in pixels, between a given pixel $p(x_p, y_p)$ and the centre pixel of the patch $c(x_c, y_c)$:

$$MD = |x_p - x_c| + |y_p - y_c|, \quad (1)$$

where (x_p, y_p) and (x_c, y_c) are the coordinates of the current and centre pixels, respectively.

Within a square image patch, the Manhattan distance is applied to find distance classes, which are later defined as the number of *MDs* from the centre (in pixels) that are taken. In other words, all the channel values far from the centre by one *MD* represent the class-one distance (grey pixel squares and → in Fig. 3), all those channel values far from the centre by two MDs represent the class-two distance (light grey pixel squares and ↠in Fig. 3), and so on.

After obtaining the distance classes, the next step is to calculate new colour intensities for each one of the colour channels, considering the other channels. For instance, considering a three-channel colour space with the following original intensities c_1, c_2, c_3, the SVC value for each channel, within the given image

[1] The number of groups is equal to the number of categories with no zero scores.

[2] This multi-classifier SVM provides scores for all food categories, having higher scores for those ones which are farther to the trained classification hyperplanes.

[3] For a survey of segmentation methods based on boundary and region refer to [16].

Table 1
Features for the segmentation process and their similarity distances (*SD*). Composite distances are arranged through an α value equal to 0.3. Single distances are Euclidean on each colour or texture components (i.e., XXXDist={SVCDist, RGBDist, LABDist, HSVDist}) or on their min–max normalised versions (*NORMLAB* or *NORMSVC*), except for HSV instances, where it was used a specific similarity distance method from [15].

Feature index	Feature description	Distance metric	Threshold to stop
F1	HSV values + SVC texture values	$SD=\alpha SVCdist+(1-\alpha)HSVdist$	0.17
F2	Normalised LAB values + SVC texture values	$SD=\alpha SVCdist+(1-\alpha)normLABdist$	0.16
F3	Normalised SVC texture values	$SD=euclideanDist(NSVC_1, NSVC_2)$	0.16
F4	RGB + SVC texture values	$SD=\alpha SVCdist+(1-\alpha)RGBdist$	0.14
F5	LAB + SVC texture values	$SD=\alpha SVCdist+(1-\alpha)LABdist$	0.5
F6	RGB colour values	$SD=euclideanDist(RGB_1, RGB_2)$	0.13
F7	SVC texture values	$SD=euclideanDist(SVC_1, SVC_2)$	0.75
F8	HSV colour values	See [15] for details	0.18
F9	LAB colour values	$SD=euclideanDist(LAB_1, LAB_2)$	0.75
F10	Normalised LAB colour values	$SD=euclideanDist(NLAB_1, NLAB_2)$	0.13

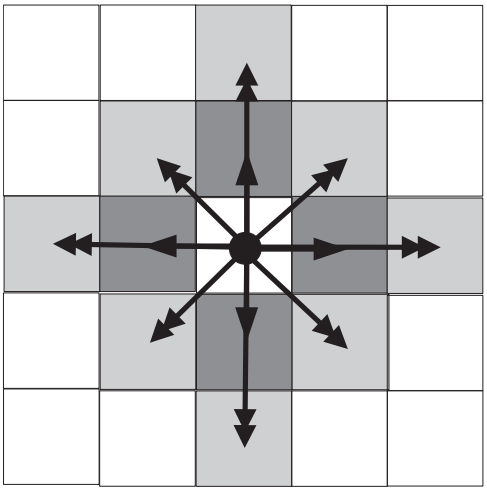

Fig. 3. 5×5 pixel window illustrating one (class-one distance, $\rightarrow$) and two (class-two distance, $\twoheadrightarrow$) Manhattan distances from the centre.

patch, is achieved by the following set of equations:

$$I_w=\alpha_w\beta_w, \tag{2}$$

where I represents the intermediary intensity of the colour channel w, and α_w and β_w are calculated, respectively, as follows:

$$\alpha_w=\sqrt{(c_i)^2+(c_j+1)^2}, \tag{3}$$

$$\beta_w=\arctan\left(\frac{c_i}{c_j+1}\right), \tag{4}$$

where c_i and c_j represent the original intensities of the other colour channels used to compute the intermediary of colour channel c_w. With the intermediary intensity I_w, now it is possible to calculate the new intensity, N, of the colour channel w according to

$$N_w=\gamma_w\delta_w, \tag{5}$$

where γ_w and δ_w are obtained by

$$\gamma_w=\sqrt{(I_i)^2+(I_j+1)^2}, \tag{6}$$

$$\delta_w=\arctan\left(\frac{I_i}{I_j+1}\right). \tag{7}$$

An *SVC* value for each class distance, SVC_{cd}, is found according to

$$SVC_{cd}=\arctan\left(\frac{\mu_{cd}}{\sigma_{cd}+1}\right)\sqrt{\mu_{cd}^2+(\sigma_{cd}+1)^2}, \tag{8}$$

where μ_{cd} and σ_{cd} are the average and standard deviation over all new intensities, N, of each colour channel c_i, c_j, c_w, within each class distance. The corresponding *SVC* of each square image window, for each channel, is obtained by also calculating (8), but now with the average and standard deviation of all SVC_{cd}. For case examples, refer to [10].

Table 2
Segmentation results over 10 different features and 17 different foods. A validation dataset was used with 20 food image samples. Best results were achieved with F1=HSV + SVC, F2=normalised LAB + SVC, and F3=Normalised SVC.

Food	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10
Lettuce	0.69	0.59	0.91	0.61	0.61	0.60	0.62	0.56	0.46	0.42
Rice	0.74	0.58	0.73	0.72	0.74	0.71	0.72	0.50	0.69	0.57
Potato	0.53	0.77	0.73	0.78	0.78	0.75	0.76	0.44	0.59	0.32
Brocolis	0.68	0.62	0.18	0.20	0.20	0.22	0.20	0.47	0.17	0.59
Meat	0.77	0.70	0.73	0.67	0.62	0.64	0.58	0.22	0.28	0.14
Carrot	0.58	0.51	0.60	0.53	0.56	0.47	0.54	0.56	0.25	0.25
Flour	0.78	0.64	0.78	0.78	0.78	0.64	0.78	0.72	0.69	0.69
Beans	0.61	0.62	0.53	0.59	0.57	0.56	0.55	0.42	0.59	0.53
Chicken	0.82	0.81	0.80	0.81	0.76	0.76	0.76	0.81	0.80	0.79
Lasagna	0.55	0.79	0.79	0.60	0.53	0.61	0.48	0.48	0.50	0.51
Spaghetti	0.60	0.65	0.49	0.56	0.40	0.54	0.43	0.58	0.53	0.75
Corn	0.74	0.78	0.74	0.72	0.72	0.74	0.72	0.53	0.65	0.74
Egg	0.17	0.06	0.16	0.06	0.05	0.17	0.05	0.18	0.16	0.02
Bread	0.71	0.74	0.72	0.68	0.68	0.67	0.55	0.27	0.77	0.65
Fish	0.37	0.54	0.54	0.37	0.47	0.43	0.44	0.32	0.26	0.27
Ham	0.39	0.43	0.32	0.33	0.19	0.35	0.19	0.63	0.42	0.42
Tomato	0.56	0.49	0.57	0.56	0.57	0.18	0.58	0.23	0.13	0.13
Average	**0.61**	**0.61**	**0.61**	0.56	0.54	0.53	0.52	0.47	0.47	0.46

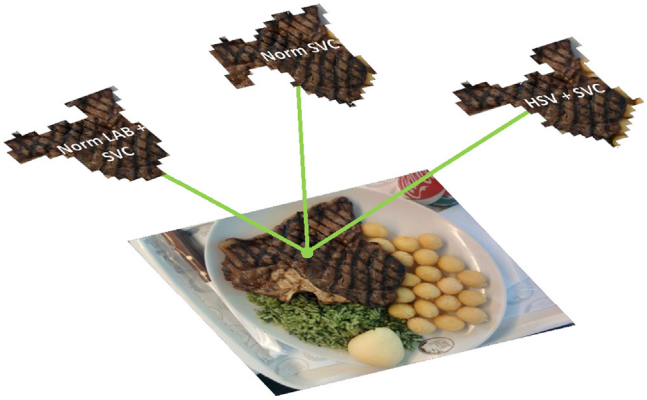

Fig. 4. Example of food segmentation by using the three best features, F1=HSV + SVC, F2=NORM*alized* LAB + SVC, and F3=NORM*alized* SVC, according to index *Q*, and results in Table 2.

Table 3
Elements of feature vector *V*.

Feature index	Feature representation	Feature description
$V_{1,2,3}$	$SVC\{DoG\{R\}, DoG\{G\}, DoG\{B\}\}$	A difference of Gaussian (DoG) is applied in each RGB channel, following by an SVC texture extraction in each DoG.
V_{4-6}	$\{\overline{R}, \overline{G}, \overline{B}\}$	Mean value of each RGB channel inside the window.
V_{7-9}	$\{\overline{L}, \overline{A}, \overline{B}\}$	Mean value of each LAB channel inside the window.
V_{10-12}	$SVC\{L, A, B\}$	SVC applied to each Lab channel value.

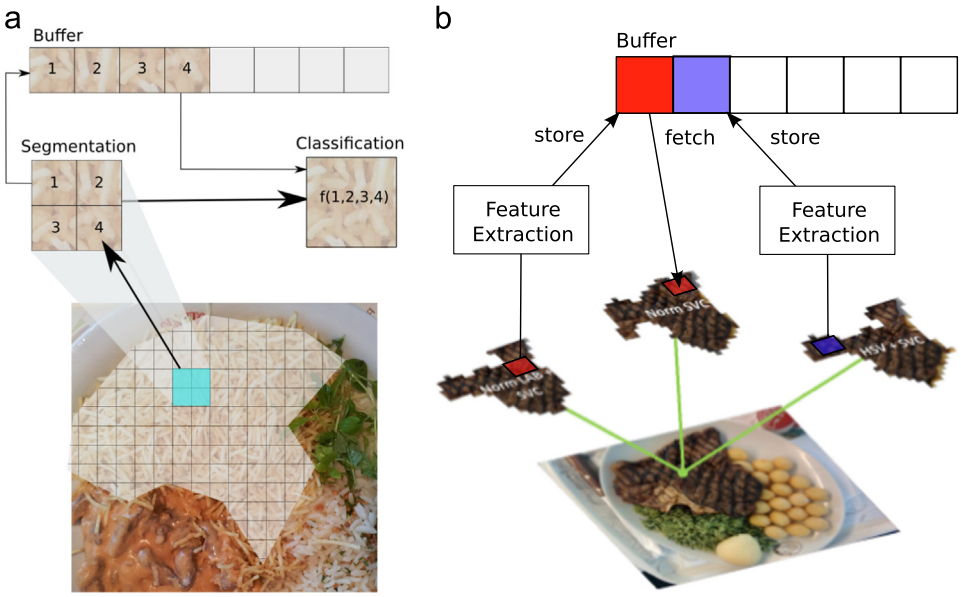

Fig. 5. Feature buffer mechanism. In (a), four patches in the segmentation step comprise a patch in the classification step; in (b), if a feature vector is not in the buffer, it is computed and stored in there; otherwise, the segmentation or classification steps fetch the feature from the buffer.

2.2. Adaptive region growing

In its basic form, region growing is a pixel-based image segmentation method which involves the choice of initial seed points to determine which pixels will be grouped. Since the goal is to implement our food identification system on a mobile device, with much less resources than a personal computer, we have made three types of modifications over the main concept: (i) the choice of the initial seeds is done by the mobile device end-user, although the way that the regions grow is fully automatic; (ii) rather than working with raw pixels, superpixels defined as continuous non-overlapped patches are used to make the regions to grow, and (iii) a floating average was applied in order to make the method adaptive to the tendency of growing the image segments.

The rationale for modification (i) covers the criteria of performance and accuracy of the system in a mobile device; because there are few resources to group neighbour pixels automatically, the system leaves the choice of the initial seed to the end-user, which can be temporarily given a role of an expert, and allowed to make the best subjective choice. With that, not only resources are set free for a more accurate food classification, but it will help guarantee more acceptable results to the end-user in the final identification process. As the system is totally embedded in a mobile device and will be employed in a variety of environments, the end-user selection of the initial seed brings more accuracy than the automatic version, with minimum interference in the whole process. Nevertheless, we have investigated a fully automatic version of the segmentation which might be embedded on a mobile device.

Table 4
Information about training and testing datasets.

Attributes	Training dataset	Testing dataset
Number of images	1500	2300
Number of patches in all image segments	19,900	34,500

Considering the fact that we want to achieve minimal delays and maximise the system computational time, modification (ii) offers the benefit of speeding up the process of grouping regions. Since the pixel-level clustering algorithm consumes much of the time from the system resources, by using regular superpixels, the system will be able to stop more quickly the growing of regions with no similarity.

In order to turn the segmentation method more adaptive to the image region growing, modification (iii) brings information about the tendency of growing. The average defines the next seed in order to calculate the similarity distances (see Table 1). By adding precision, and using float instead of integers to store the average, it is possible to track the tendency of clustering the neighbour superpixels (see line 8 in Algorithm 1). This way, the method becomes adaptive with respect to the segment size, which is later

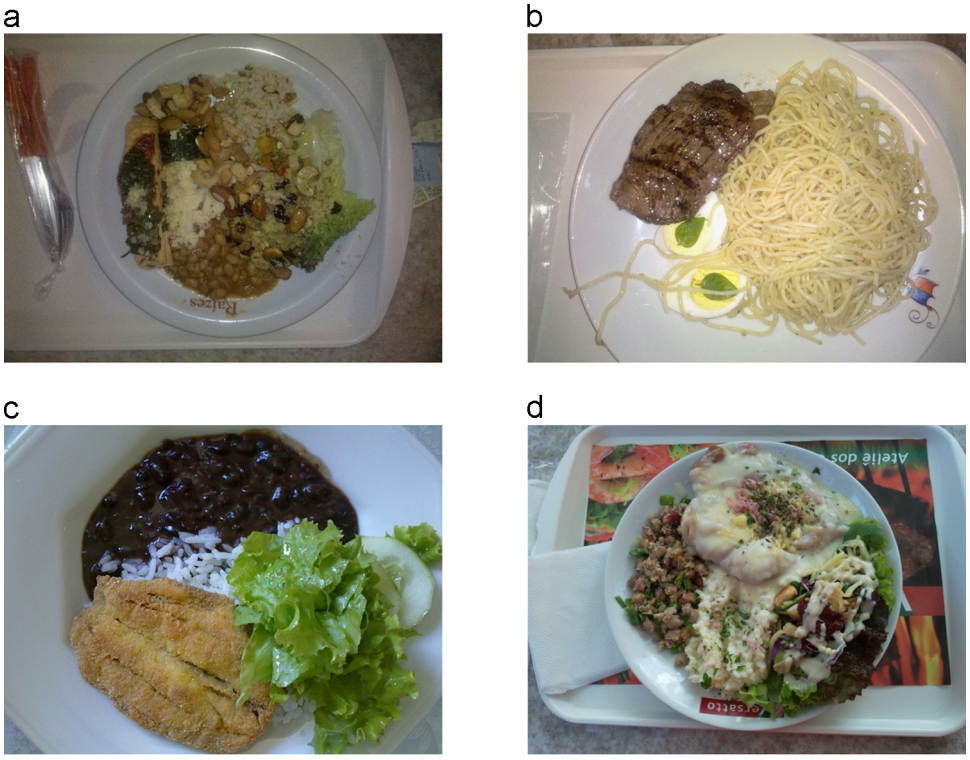

Fig. 6. Example of food plates used for testing in different lighting conditions.

defined by the previous selection of the regions. After the selection of the initial seeds of the food images, Algorithm 1 illustrates the necessary steps to perform the segmentation method.

Algorithm 1. Adaptive region growing segmentation.

input: Image, initSeedPoint
output: segment
1 $\mu_i^j \leftarrow \frac{C_j}{N_j}$, where C is the value of a colour channel j, and N the cardinality of each channel j inside the segment; μ_i is the average of the initSeedPoint;
2 $\mu^j \leftarrow \mu_i^j$;
3 **repeat**
4 | **for** *each one* G_r *of the* $4-$*connected neighborhoods* **do**
5 | | $d_r \leftarrow \text{calcDist}(\mu^j, G_r)$;
6 | **end**
7 | **if** $\min(d_r) < \text{thre}$ (*see Table* 1) **then**
8 | | segment $\leftarrow$ segment + addSuperpixel $(\min(d_r))$;
9 | | $\mu^j \leftarrow \frac{\mu^j * \text{regionSize} + \min(d_r)}{\text{regionSize} + 1}$ **end**
10 | **end**
11 **until** $\min(d_r) \geq \text{thre}$(*see Table* 1) *or image region* $\geq$ Image;

In Algorithm 1, a segment initiates as the initial seed point[4] (line 1). That segment grows by adding the superpixels closer to the segment (line 8), considering the distances for each colour or texture channel in Table 1. The floating average (line 9) adaptively controls the addition of superpixels into the segment, according to the calculated average of the previous image segment still growing. The segmentation stops when the distances to the 4-connected neighbourhoods are greater or equal to the *thresholds to stop* (see Table 1) or the input image.

[4] Initially, segment is equal to superpixel.

2.3. Segmentation with multi-ranking hypotheses

Considering the features in Table 1, an experiment was carried out to evaluate the segmentation quality. For that, an index was created with the goal of quantifying the segmentation performance by using our region growing method. A dataset with 20 food plate images, containing 17 different types of food, was manually annotated with the contours of the foods in the image. The index, Q, that measures feature performance in segmentation is based on

$$Q = \frac{A_{GT} \bigcap A_{RG}}{A_{GT} \bigcup A_{RG}}, \tag{9}$$

where A_{GT} and A_{RG} represent the ground truth and the region growing area, respectively. Q is in the interval [0; 1]. The greater the index value, the better the segmentation.

Q measures effectively how much of the region growing based segment intersects the ground truth. Table 2 shows the results obtained with the experiment. Three features were highlighted with best segmentation performances, F1 = HSV + SVC, F2 = normalised LAB + SVC, and F3 = Normalised SVC. The reason for the selection of those three features was the commitment to the computational cost, while still keeping the recognition performance high. In other words, if the 10 features were used to build more segmentation hypotheses, the computational time for the overall process of recognition would get hindered, since the segmentation results must be classified, in the next phases. This way, three features keep multiple alternatives for the final integration between segmentation and learning, without compromising the response for the end-user (a discussion about the system computational time is accomplished in Section 4).

In Fig. 4, an example of the segmentation by using the best three features is depicted.

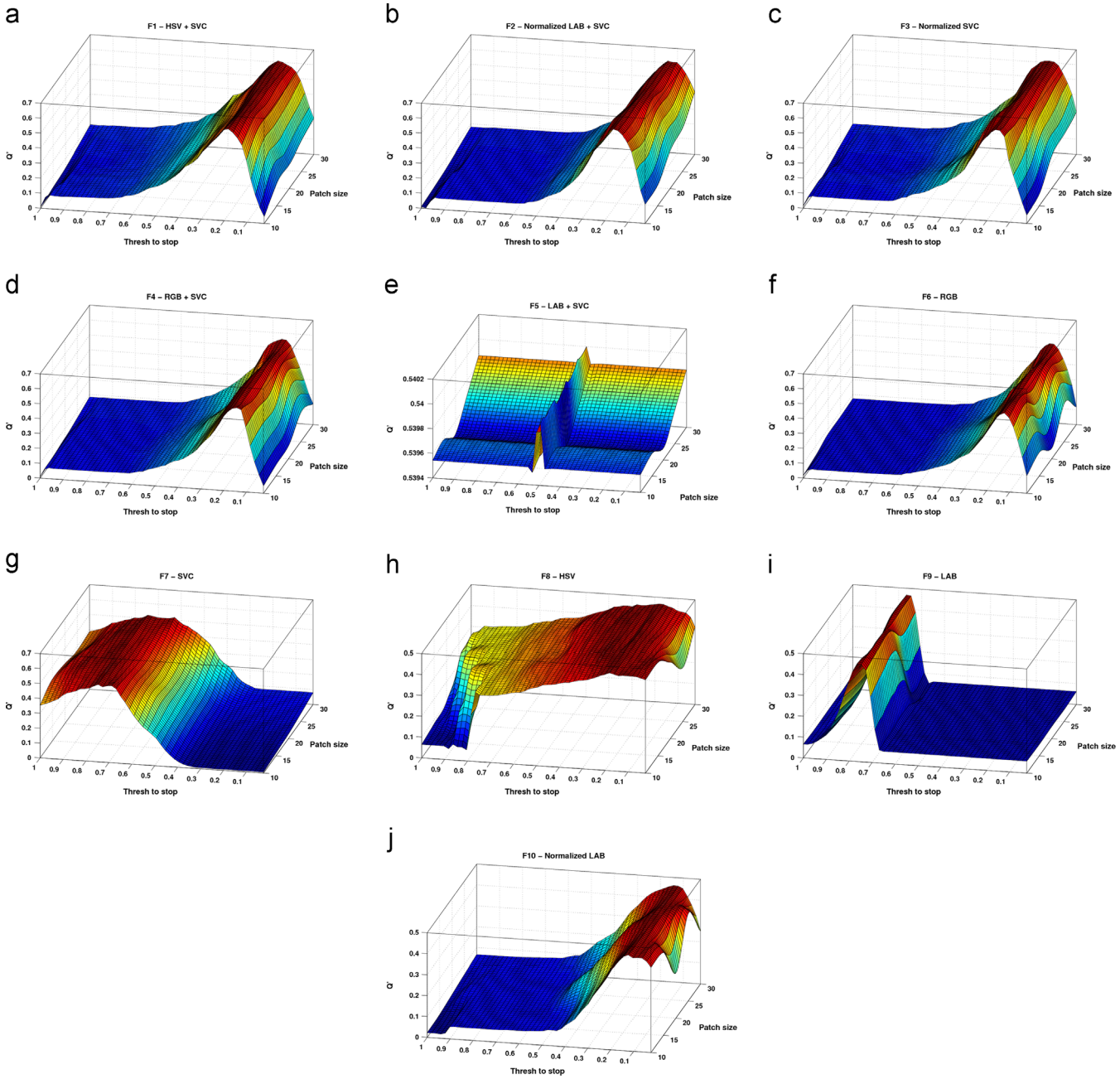

Fig. 7. Evaluation of the segmentation performance and its parameters. Q' denotes the average of Q index over all foods. The best thresholds for each feature are summarised on Table 1, while the average of the values of Q can be seen on Table 2.

3. Food classification

With the goal of automatically ranking the three final segments for each food type, an SVM was applied, following the general idea presented in Fig. 2.

Let us consider segments s_i, $\{i\}_1^N$, obtained by the region growing segmentation method described in the last section. Completely inside each segment, non-overlapped square windows $(\omega_j,\ \{j\}_1^M)$ are extracted with a 12-element feature vector V, according to Table 3.

For the first 3 elements of the vector V, a Difference of Gaussian (DoG) is applied over each RGB channel. With that result, for each DoG-RGB channel, an SVC texture is extracted. The rest 9 vector elements are comprised of RGB mean, LAB mean and SVC over LAB colours, respectively. Next, V feeds a gaussian SVM with a kernel K of the type:

$$K(x_i, x_j) = e^{-\gamma \| x_i - x_j \|^2}, \tag{10}$$

where γ is a kernel parameter, and x_i and x_j are input feature vectors.

With K, the score ς for food classification is given by

$$\varsigma(x) = \sum_{i=1}^{\mathcal{L}} w_i K(x, x_i) + b, \tag{11}$$

where w_i are the weight vector obtained in the training stage, b is a bias parameter used to normalise the decision function, and $\mathcal{L}$ is the number of support vectors. Rather than classifying binarily the

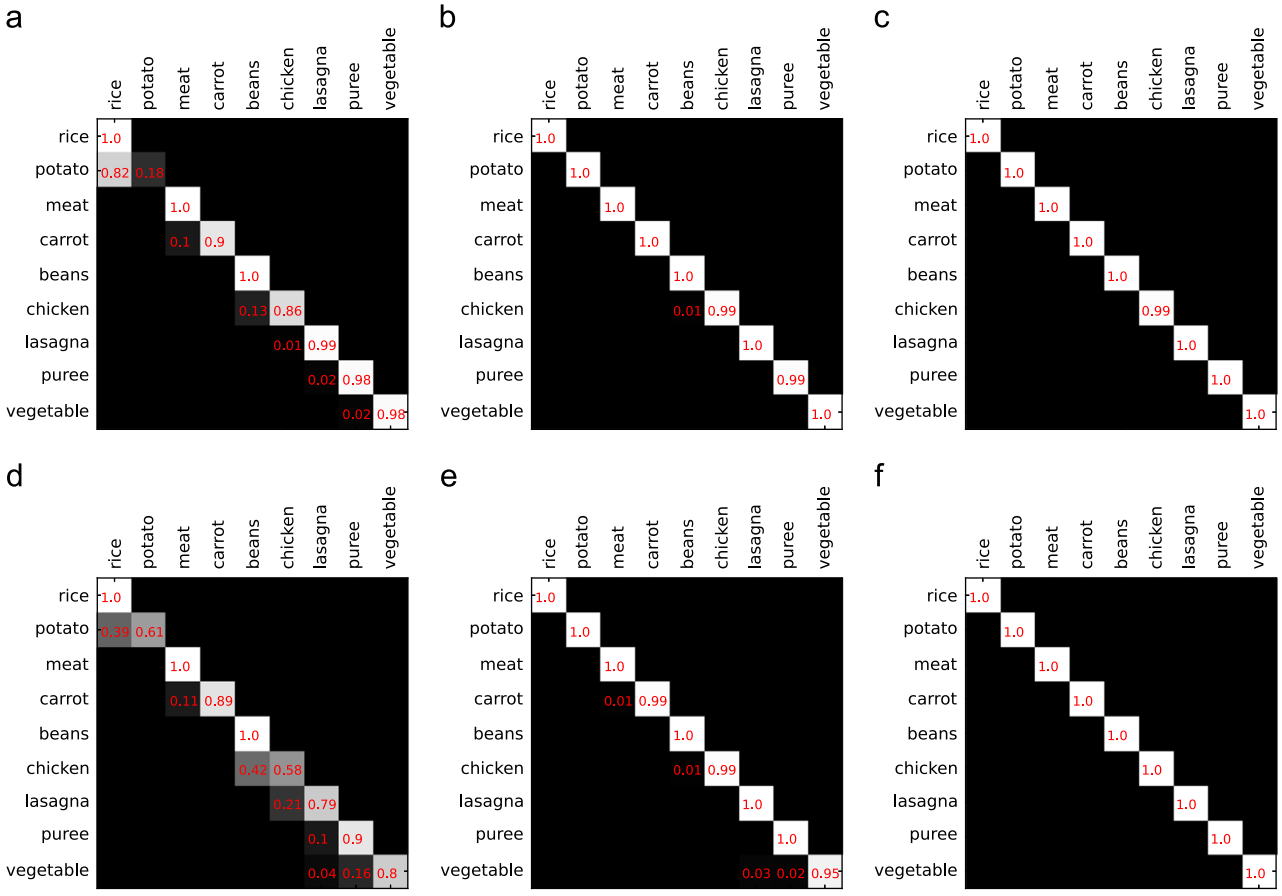

Fig. 8. Top 1–3 confusion matrices of the 9 types of food for (top-down row enumeration): SVM and Adaboost (with 1000 weak classifiers). (a) Top 1 (SVM), (b) Top 2 (SVM), (c) Top 3 (SVM), (d) Top 1 (Adaboost), (e) Top 2 (Adaboost) and (f) Top 3 (Adaboost).

feature vector extracted from the square windows, an one-against-one strategy is employed in order to multi-classify each window. For that, $\kappa(\kappa-1)/2$ binary classifiers are trained, where κ is the number of classes. At the end, the classification system is composed of 2-by-2 combinations of classifiers.

Normalising the score ς to the interval [0; 1] aids the process of making the final decision. The normalisation works by converting the classification score into a probability, $p(s)$, through a logistic regression function:

$$p(\varsigma) = \frac{1}{1-e^{A\varsigma+B}}, \tag{12}$$

where A and B are obtained in the training stage according to [17].

With the probabilistic values of each square window within a segment calculated, the final segment probability of being a certain class, p_{seg}, is given by

$$p_{seg} = \max_i \left(\sum_{i,j} \overline{p}_i(\omega_j) \right), \tag{13}$$

where $i = 1, 2, \ldots, R$, with R being the number of food classes, and $j = 1, 2, \ldots, M$, with M being the number of patches for each class inside the segment. The average probability for each food class is represented by $\overline{p}$.

The final segment comes from the best probability out of the three segments according to

$$p_{final} = \max_g (p_g), \tag{14}$$

where p_g is the probability for each one of the three segments g.

With that, we have a multi-ranking, multi-classification system. It is endorsed by different types of features used whether in segmentation, or in classification, which provide various ways of coping with food identification in different environments without incurring high computational costs to the process.

4. Computational time optimisation with buffer of features

Considering the system as it was sequentially described until then, it would take more than a minute to identify all the foods in a plate. To overcome this issue, we designed our system on the basis of a buffer of features. A patch in the classification step is then comprised of four patches in the segmentation step (see Fig. 5a). The rationale here is to cache the features in such a way to avoid recomputing those that are used whether in segmentation or in classification.

As seen in Fig. 5a and b, the mechanism behind the buffer of features works in this way: First, a process (performed in the segmentation or classification steps) searches in the buffer for precomputed features; if a feature vector within a patch does not exist yet in the buffer, the feature extraction is performed and the result set is stored and indexed by patch position; otherwise, the feature set is fetched from the buffer and used in the desired step, avoiding a redundant call to the feature extractors. This way, the computational complexity of the overall feature extraction remains in $O(n)$ (n=total number of patches), despite the number of hypotheses returned in the segmentation step. After applying the buffer of features, the time to output the results of food

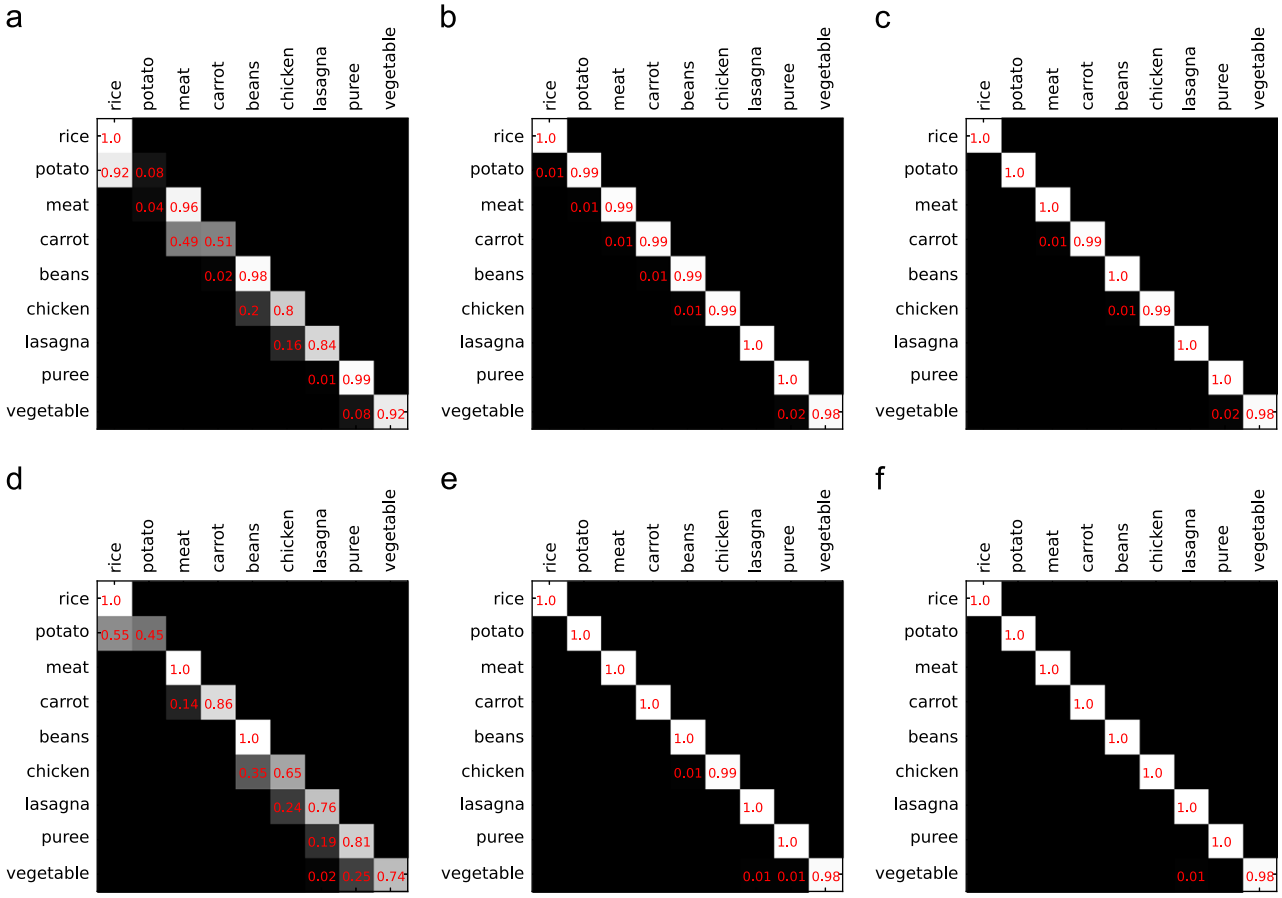

Fig. 9. Top 1–3 confusion matrices of the 9 types of food for (top-down row enumeration): MLP (with 128 hidden neurons) and LS-SVM. (a) Top 1 (MLP), (b) Top 2 (MLP), (c) Top 3 (MLP), (d) Top 1 (LS-SVM), (e) Top 2 (LS-SVM) and (f) Top 3 (LS-SVM).

identification was reduced, on average, from more than a minute to less than 10 s, running on a Samsung Galaxy SII cell phone.

5. Experimental results

To validate the proposed system, all the images used for training and testing (forming two disjoint sets) were gathered from a camera-equipped cell phone. Particularly, the food recognition system was embedded in a Samsung Wave and a Galaxy SII, which come with an auto-focus camera.[5]

Table 4 summarises information about training and testing datasets. The table shows the number of images used in training and test, and also the total number of patches within all segments in the 1500 training images and 2300 testing images. Each patch in the segmentation has the size of 15 $\times$ 15 pixels, while, in the classification, size of patches was 30 $\times$ 30 pixels. Indeed, the choice for the segmentation patch size and the threshold to stop the segmentation were experimentally found by varying these two parameters and evaluating the average of index Q (from Eq. (9)); these results are illustrated in Fig. 7. Although the size of the patch did not influence considerably the segmentation performance, the choice for the size 15 $\times$ 15 pixels was considered because of the classification process; smaller sizes would increase the computational cost, greater size could not appropriately take the gist of the food elements. It is noteworthy to highlight that, in order to decide which category of food pertains to a given segment, in the ranking phase it is necessary that the great majority of the patches within the segment be classified by the target food type. In lieu of that, the classification method must be consistent over each patch in order to avoid that the system arrives at conclusions with no clear criterion. Indeed, we will now present results that point to this correct direction.

The validation setup takes into consideration different scenarios for shooting the pictures. This was of underlying importance since the proposed method to identify foods must be applied indoor, thus in different illumination environments, and prone to not only shadow interference but also lighting spots. Fig. 6 illustrates some examples of images used for the testing stage. These different lightning conditions are tackled by the texture and colour features chosen, and ranked by the different classifiers with the aim of coping with that variation.

For testing, 9 types of food were chosen for evaluation: rice, potato, meat, carrot, beans, chicken, lasagna, puree and many different types of vegetables. These types correspond to a regular occidental diet. All food images were taken from the overhead view of the plate using our virtual targeting system (see Fig. 2), in order to minimise variation in texture information.

In order to properly evaluate the classification performance of the proposed method, two steps were followed: (i) different classifiers were applied over each feature vector of Fig. 4 and confusion matrices were build to highlight the hit rate on each food, and (ii) with the best classifier, the proposed system was compared with the two approaches proposed in [2]. Figs. 8 and 9

[5] Camera without auto-focus is out of the scope of the system, since it brings additional challenges not foreseen in the system design.

Table 5
False positive rate of Top 1–3 according to the confusion matrices of Figs. 8 and 9.

Classifier	Ranking	False positive rate									
		Rice	Potato	Meat	Carrot	Beans	Chicken	Lasagna	Puree	Vegetable	Avg error
SVM	Top1	0.00	0.82	0.00	0.10	0.00	0.13	0.01	0.02	0.02	0.12
	Top2	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.00	0.00
	Top3	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00
Adaboost	Top1	0.00	0.39	0.00	0.11	0.00	0.42	0.21	0.10	0.20	0.16
	Top2	0.00	0.00	0.00	0.01	0.00	0.01	0.00	0.00	0.05	0.01
	Top3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MLP	Top1	0.00	0.92	0.04	0.49	0.02	0.20	0.16	0.01	0.08	0.21
	Top2	0.00	0.01	0.01	0.01	0.01	0.01	0.00	0.00	0.02	0.01
	Top3	0.00	0.00	0.00	0.01	0.00	0.01	0.00	0.00	0.02	0.00
LS-SVM	Top1	0.00	0.55	0.00	0.14	0.00	0.35	0.24	0.19	0.27	0.19
	Top2	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.02	0.00
	Top3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.00

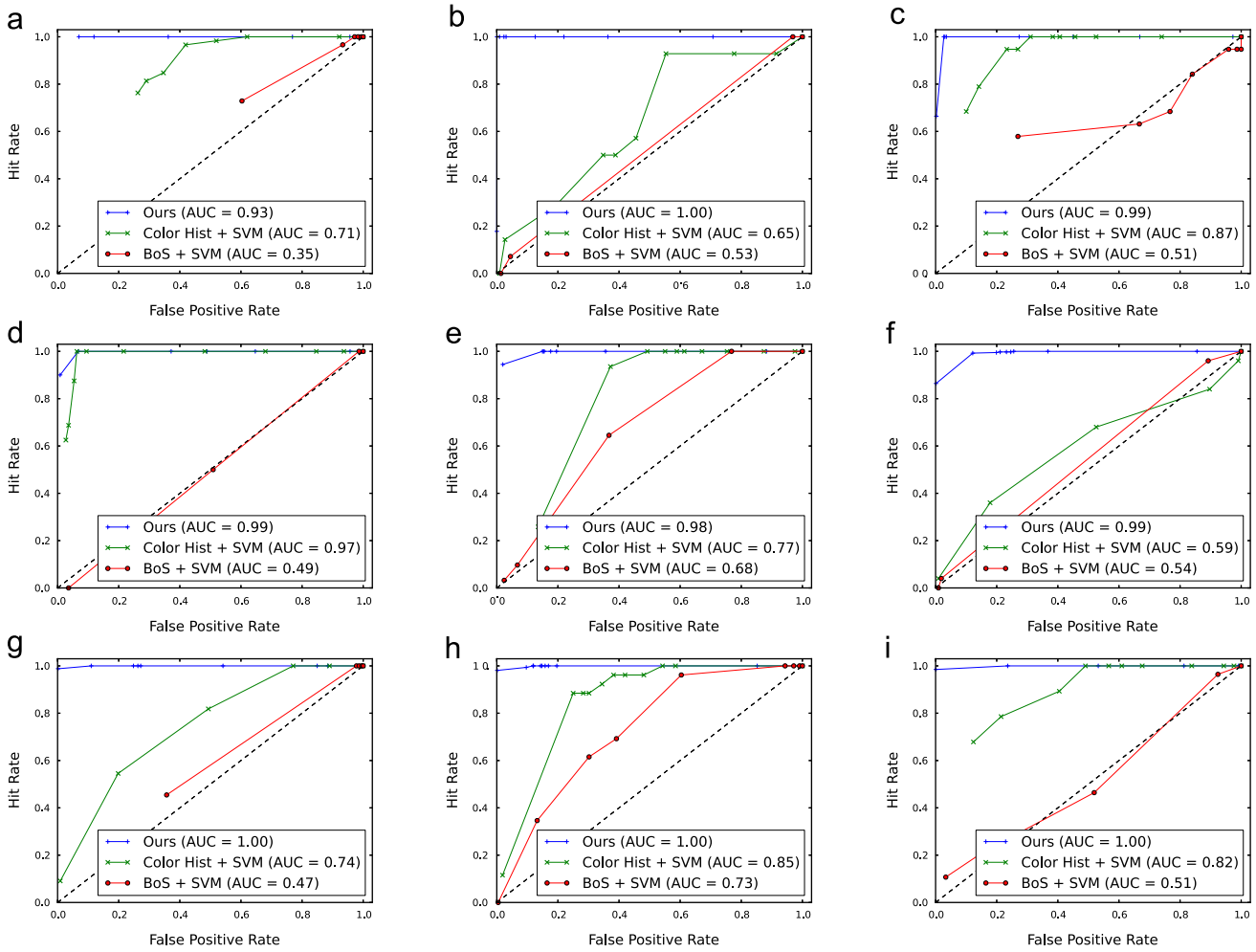

Fig. 10. ROC curves of the 9 types of food with comparative results among our proposed method and the methods used in [2] – colour histogram + SVM and a bag of 1000 sifts + SVM. In the legends, the area under curve (AUC) shows the performance of each method. (a) Rice, (b) potato, (c) meat. (d) carrot, (e) beans, (f) chicken, (g) lasagna, (h) puree. and (i) vegetable.

illustrate the results. Four classifiers (with parameters which presented best results in the experiments) were explored: standard SVM with a Gaussian kernel (radial basis function), Adaboost with 1000 weak classifiers, a multi-layer perceptron (MLP) trained with a Levenberg–Marquardt algorithm and 128 hidden neurons, and, finally, an SVM regularised by a least-square method (LS-SVM) [18], RBF kernel and trained with a simplex optimisation algorithm to tune the hyperparameters of the model with respect to the given performance measure. All classifiers were trained with one-versus-all multi-classification strategy.

For each matrix in Figs. 8 and 9, it is shown the top 3 choices of the multi-classification results. Considering a mobile device embedded system, it is highly acceptable that the user be presented with a list of at least three of the most probable food categories which the expert system suggested. By suggesting a list, instead of a single choice of food for each segment, we are able to

increase the chance to be accepted as an expert system, indeed. As a matter of fact, by considering the top 3 classification results, we were able to achieve perfect results with three of the four classifiers evaluated (SVM, Adaboost and LS-SVM). With the notable exception of MLP, it is possible to state that we can apply any of the best three classifiers. By analysing the results of each confusion matrix still deeper, the best result at all was obtained by the standard SVM with RBF kernel with the lowest average error (false positive rate) in top 1–3, as summarised in Table 5.

The next step was to compare the best classification system (standard SVM) with those ones used in [2]. For that, we computed the performance over all types of food, comparing our proposed method with a colour histogram (64 dimensional vector for each RGB channel) and a bag of 1000 SIFTs (scale invariant feature transform), both set of features being classified by a standard SVM. Fig. 10 shows the receiver operating characteristic (ROC) curves of all 9 food types with comparative results. In the legends, it is notable that the area under ROC (AUC) of our method is the highest in all situations. It can be explainable due to prepared food is eminently textured and coloured, and with those two kinds of features (colour and interesting points) in [2] are not able to deal with the diversity of image information; our method become thus more appropriate for that situation, presenting the best performance in our experimental analysis.

Considering all the results, quantitative and qualitatively speaking, we can conclude that the proposed system had a very interesting performance even taking the fact that it is embedded in a mobile device.

6. Conclusion

In this paper, we presented a semi-automatic system to recognise prepared meals which is lightweight and can be easily embedded on a camera-equipped mobile device. The multi-category system relies on several texture and colour features which are ranked by multiple classifiers. A very effective strategy, capable of making the system suitable to be embedded, was a buffer of features which shares these between the segmentation and classification steps. This proposed strategy reduces in approximately 90% the time required to run the classification system.

The next step will be to transform the semi-automatic segmentation into an automatic one, even if this means that our solution will need to deal with a much higher computational cost. Consequently, we will need to come up with additional strategies to deal with these new challenges in a similar way as our buffering of features. Also, we are already investigating using additional feature spaces which can disambiguate some image situations.

Conflict of interest statement

None declared.

Acknowledgements

Part of the results presented in this paper were obtained through research activities sponsored by Samsung Eletronica da Amazonia Ltda. under the terms of Brazilian federal law No. 8.248/91.

References

[1] G. Shroff, A. Smailagic, P. Siewiorek, Wearable context-aware food recognition for calorie monitoring, in: Proceedings of IEEE International Symposium on Wearable Computers, 2008, pp. 119–120.
[2] M. Chen, K. Dhingra, W. Wu, L. Yang, R. Sukthankar, J. Yang, PFID: Pittsburgh fast-food image dataset, in: Proceedings of IEEE International Conference on Image Processing, 2009.
[3] S. Yang, M. Chen, D. Pomerleau, R. Sukthankar. Food recognition using statistics of pairwise local features, in: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2010, pp. 2249–2256.
[4] L. Oliveira, E. Jorge, A. Filho, V. Costa, G. Neves, T. Oliveira, G. Anunciacao, E. Santos. Method for automatic recognition of food through portable devices equipped with digital cameras, US Patent 12/981,634, July, 12, 2012.
[5] R. Bolle, J. Connell, N. Haas, R. Mohan, G. Taubin, VeggieVision: a produce recognition system, in: Proceedings of IEEE Workshop on Applications of Computer Vision, 1996, pp. 2–4.
[6] Z. Fengqing, M. Bosch, W. Insoo, K. SungYe, C. Boushey, D. Ebert, E. Delp, The use of mobile devices in aiding dietary assessment and evaluation, in: IEEE Journal of Signal Processing, 2010, pp. 756–766.
[7] X. Ren, J. Malik, Learning a classification model for segmentation, in: Proceedings of IEEE International Conference on Computer Vision, 2003, pp. 10–17.
[8] S. Alpert, M. Galun, R. Basri, A. Brandt, Image segmentation by probabilistic bottom-up aggregation and cue integration, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.
[9] F. Li, J. Carreira, C. Sminchisescu, Object recognition as ranking holistic figure-ground hypotheses, in: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2010, pp. 1712–1719.
[10] E. Nunes, A. Conci, Segmentaç ao por textura e localizaç o do contorno de regi oes em imagens multibandas, IEEE Lat. Am. Trans. 5 (3) (2007) 185–192.
[11] K. van de Sande, T. Gevers, C. Snoek, Evaluating color Descriptors for object and scene recognition, IEEE Trans. Pattern Anal. Mach. Intell. 32 (9) (2010) 1582–1596.
[12] D. Ilea, P. Whelan, Image segmentation based on the integration of colour-texture descriptors – a review, Pattern Recognit. 44 (10–11) (2011) 2479–2501, Elsevier.
[14] L. Oliveira, U. Nunes, P. Peixoto, M. Silva, F. Moita, Semantic fusion of laser and vision in pedestrian detection, Pattern Recognit. 43 (10) (2010) 3648–3659, Elsevier.
[15] S. Sural, G. Qian, S. Pramanik, Segmentation and histogram generation using the HSV color space for image retrieval, in: Proceedings of International Conference on Image Processing, 2002, pp. 589–592.
[16] J. Freixenet, X. Muñoz, D. Raba, J. Martí, X. Cuf, Yet another survey on image segmentation: region and boundary information integration, in: European Conference on Computer Vision, 2002, pp. 408–422.
[17] J. Platt, Probabilistic outputs for support vector machines and comparison to regularized likelihood methods, in: A. Smola, P. Bartlett, B. Scholkopf, D. Schuurmans (Eds.), Advances in Large Margin Classifiers, MIT Press, 2000.
[18] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, Least Squares Support Vector Machines, World Scientific, Singapore, 2002.

Luciano Oliveira received his BSc in Computer Science and MSc in Mechatronics degrees from Federal University of Bahia, Brazil, in 1997 and 2005, respectively. In 2010, he received his PhD in Electrical and Computer Engineering at Coimbra University, Portugal. He is the head of the Intelligent Vision Research Lab, at Federal University of Bahia, and has been working in several research projects sponsored by Samsung, Federal Research Network of Brazil and Petrobrás. He is author or co-author of several patents integrating Computer Vision and mobile devices for Samsung.

Victor Costa received his BSc Computer Science from Catholic University of Salvador, Brazil, in 2004 and his MSc in Mechatronics from Federal University of Bahia, Brazil, in 2009. He is now a PhD student in Mechatronics at Federal University of Bahia. He has participated as a researcher in projects in the field of Computer Vision and Image Pattern Recognition, mainly focused on camera-equipped mobile systems. He worked as author and co-author in several patents for Samsung.

Gustavo Neves received his BSc in Computer Science degree from Catholic University of Salvador, Brazil, in 2010. Since 2012, he is a master student in Mechatronics at Federal University of Bahia. He has worked in several research projects in the field of Computer Vision for camera-equipped mobile systems, as well as, he is co-author of various patents for Samsung.

Talmai Oliveira received his BSc in Computer Science and MSc in Mechatronics degrees from Federal University of Bahia, Brazil, in 2002 and 2008, and PhD in Computer Science and Engineering at the University of Cincinnati. His research area includes dependable heterogeneous wireless networks and dealing with the inherently massive uncertainty issues. He has also worked for Samsung in projects involving Image Pattern Recognition, being co-author of many patents.

Eduardo Jorge received his MSc in Computer Science from the Federal University of Campina Grande, Brazil, in 2001. In 2012, he received his PhD from Federal University of Bahia, in the field of Knowledge Dissemination, with focus on Ontologies for Conceptual Modeling. He leads several innovative technological and research project, as Samsung partner, using mobile devices.

Miguel Lizarraga received his degree in Electrical Engineering from the State University of Campinas (UNICAMP), Brazil in 1994, and also his MSc and PhD in the fields of image processing and pattern recognition, in 1996 and 2000. Since 2005, he works at Samsung, making the interface between academic research conducted by universities and the market needs. He also develops projects in the area of innovation and intellectual property management.