Rapid Mobile Object Recognition Using Fisher Vector

Yoshiyuki Kawano and Keiji Yanai
The University of Electro-Communications,Tokyo
1-5-1 Chofugaoka, Chofu-shi, Tokyo, 182-8585 Japan
Email: kawano-y@mm.inf.uec.ac.jp, yanai@cs.uec.ac.jp

***Abstract*—We propose a real-time object recognition method for a smartphone, which consists of light-weight local features, Fisher Vector and linear SVM. As light local descriptors, we adopt a HOG Patch descriptor and a Color Patch descriptor, and sample them from an image densely. Then we encode them with Fisher Vector representation, which can save the number of visual words greatly. As a classifier, we use a liner SVM the computational cost of which is very low. In the experiments, we have achieved the 79.2% classification rate for the top 5 category candidates for a 100-category food dataset. It outperformed the results using a conventional bag-of-features representation with a chi-square-RBF-kernel-based SVM. Moreover, the processing time of food recognition takes only 0.065 seconds, which is four times as faster as the existing work.**

I. Introduction

In recent years, smartphones such as iPhone and Android phones have become popular and obtained more computational power. Then object recognition on smartphones in a real-time way becomes possible. Currently, a quad-core CPU is common as a smartphone's CPU. Old-style mobile image recognition systems carry out image recognition processing on the server side with much computational resource. They require wireless connection, and the communication delay is unavoidable. With the server-side recognition, real-time object recognition is impossible. In addition, when the number of users increases, response time will be much degraded unless making computational resource increased.

On the other hand, client-side processing on object recognition on a common smartphone needs no wireless connection and no commutation delay. This is one of the requirements for mobile real-time object recognition. However, computational resource is relatively limited compared to server-side processing where many CPUs are available. To realize real-time object recognition in the client side on a common smartphone, a method which enables us to use its computational resource efficiently is needed. This is apparently a non-trivial task.

Recently, several feature image representations which are suitable for a faster linear classifier have been proposed. Past style image recognition is common to classify images with SVMs with non-linear kernels. For N training samples and the number of support vectors M, computational complexity of non-linear SVM is $\mathcal{O}(N^2) \sim \mathcal{O}(N^3)$ in training, and the processing time is proportional to M in testing, which are very high cost and requires large memory space. On the other hand, the computational costs of a linear SVM is $\mathcal{O}(1)$, which means one-time evaluation of a linear SVM requires one computation of an inner product, while its performance is lower that that of a SVM with a non-linear kernel.

To resolve this, some image representations are proposed recently, which can boost recognition performance in case of using a linear SVM. Especially, Fisher Vector [1], [2] is known as a high performance method among the recent image representations. it is suitable for a linear classifier, and was turned out that it can improved recognition accuracy more than popular combination of bag-of-features (BoF) and a non-linear SVM. Moreover, while BoF needs larger dictionary to improve recognition accuracy, larger dictionary brings increase of computational cost for searching nearest visual words. On the other hand, Fisher Vector is able to achieve high recognition accuracy with even small dictionary, and low computational complexity. This is an advantage for mobile devices.

Thus adopting Fisher Vector as encoding method is better in terms of recognition accuracy and processing time for a mobile object recognition. But a system on a smartphone does not exist so far that carries out rapid and high precision image recognition with Fisher Vector. Then we propose a recognition method for a smartphone which is rapid and accurate by making good use of computational resource of the smartphone with Fisher Vector.

In the experiments, we have achieved 79.2% classification rate within the top five candidates for a 100-category food dataset with ground-truth bounding boxes. It is better than the exist food recognition method [3] by 11.0%, and processing time of 100 kinds of image recognition takes only 0.065 second. It is four times as faster as the existing method [3]. Moreover, we achieved the better results than the exist food recognition method running on the server-side regarding classification accuracy on 100 food categories. Therefore, the proposed method is more efficient than existing image recognition methods for a smartphone.

The rest of this paper is organized as follows: Section II describes related work. In Section III, we explain the proposed method for rapid and accurate image recognition on a smartphone. In Section IV describes the experimental results, and in Section V we conclude this paper.

II. Related Work

In this section, we introduce some related works in terms of object recognition on a smartphone.

As commercial services on image recognition for smartphones, Google Googles[1] is widely well-known. Google Googles is an Android-based application which can recognize logos, famous art, and famous landmarks and so on in photos

[1] http://www.google.com/mobile/goggles/

taken by users. However, recognition targets are limited to specific objects the appearance of which stays unchanged, which is different from this paper the target of which is generic objects. Kumar *et al.* [4] proposed a system which recognizes 184 plant species using curvature based shape feature peculiar leaf, after a user sends a leaf photo to server. These two systems basically employs server-side image recognition, while our goal is to implement a client-side generic object recognition system.

Next, we explain image recognition methods on run a smartphone. Lee *et al.* [5]. proposed a mobile specific object recognition system on a smartphone, which recognized registered specific objects in a real-time way. They train a template, which is decomposed into descriptors of intensity and gradient orientation, and employ matching method for real-time object detection and tracking. This is a mobile specific object recognition system. Maruyama *et al.* [6] proposed a recipe recommendation system, which adopts only color histogram based BoF and they are directly applied linear SVM. Thus it was very poor accuracy for generic object appearances of which are changeable. Kawano *et al.* [3] proposed a mobile food recognition system, which adopts SURF based BoF and color histogram applied χ^2 kernel feature map with linear SVM. SURF is popular robust local descriptor, but the method was relatively high cost for rapid image recognition on smartphone. And it was hard to recognize foods in real-time, if the number of categories increases. Moreover, conventional BoF is not suitable for rapid image recognition in terms of recognition accuracy and processing time.

In this paper, we propose a rapid and highly actuate image recognition method for a smartphone. It adopts HOG Patch and Color patch as light local descriptors, and Fisher Vector representation as encoding method. As a classifier, a linear SVM which is very efficient is employed.

III. Proposed Method

In this paper, we implement rapid and high accuracy image recognition system which can handle 100 kinds of categories with Fisher Vector based feature. We adopt HOG Patch and Color Patch as local descriptor, and each local descriptors are represented by Fisher Vector. Then they are classified with linear SVM.

A. Image Feature

In this paper, we use the following local descriptors for Fisher Vector encoding: HOG Patch and Color Patch.

1) HOG Patch: Histogram of Oriented Gradients(HOG) was proposed by N.Dalal *et al.* [7]. It is similar to SIFT is terms of how to describe local patterns which is based on gradient histogram. Since HOG description is very simple, it is able to describe much faster than popular local descriptor such as SIFT [8] and SURF [9]. This is important characteristic to carry out real-time recognition on a smartphone. In addition, it is able to extract local feature more densely. As a result, it improves recognition accuracy.

We extract HOG features as local features. We divide a local patch into $2times2$ blocks (totally four blocks), and extract gradient histogram regarding eight orientations from each block. Totally, we extract 32-dim HOG Patch features. Then the 32-dim HOG Patch is L2 normalized to an L2 unit length, not adopt HOG-specific normalization by sliding fusion. PCA is applied to reduce dimensions from 32 to 24.

2) Color Patch: We use mean and variance of RGB value of pixels as Color Patch feature. We divide a local patch into 2×2 blocks, and extract mean and variance of RGB value of each pixel within each block. Totally, we extract 24-dim Color Patch features. PCA is applied without dimension reduction. The dimension of a Color Patch feature are kept to 24-dim.

B. Fisher Vector

Fisher Vector [1], [2] can decrease quantization error than BoF [10] by using of a high order statistic. Moreover Fisher Vector is very highly efficient in recent encoding method [11], and almost higher rank team of recognition challenge used Fisher Vector [12].

For the number of local descriptors T, set of local descriptors $X = \{x_t, t = 1, ..., T\}$, Fisher Vector is defined as follows.

$$p(x|\theta) = \sum_{i=1}^{K} \pi_i \mathcal{N}(x|\mu_i, \Sigma_i) \tag{1}$$

where $p(X|\theta)$ is probability density function (pdf), $\nabla_\theta \log p(X|\theta)$ is gradient of log likelihood, F_θ is the Fisher information matrix, then F_θ is decomposed into $F_\theta^{-1} = L'_\theta L_\theta$.

Fisher Kernel $K(X, Y) = G_\theta^{X'} F_\theta^{-1} G_\theta^Y$ is an inner product of Fisher Vector. Thus it is able to classify images efficiently with a linear classifier.

According to [2], we encode local descriptors into Fisher Vector. We choice probability density functions as Gaussian mixture model (GMM). Then pdf is given as follows.

$$p(x|\theta) = \sum_{i=1}^{K} \pi_i \mathcal{N}(x|\mu_i, \Sigma_i) \tag{2}$$

where x is a local descriptor, K is number of component of Gaussian, $\theta = \{\pi_i, \mu_i, \Sigma_i, i = 1, ..., K\}$ is a parameter of GMM. π_i is the mixing coefficient, μ_i is mean vector and Σ_i is the covariance matrix. At this point, we assume that the covariance matrix is diagonal and diagonal elements are presented variance vector σ^2.

The probability of x_t is belong component i (estimated posterior probability) is given as follows.

$$\gamma_t(i) = \frac{\pi_i \mathcal{N}(x_t|\mu_i, \Sigma_i)}{\sum_{j=1}^{N} \pi_j \mathcal{N}(x_t|\mu_j, \Sigma_j)} \tag{3}$$

Then the gradient with respect to the mean and variance is defined as follows,

$$\mathcal{G}_{\mu,i}^X = \frac{1}{T\sqrt{\pi_i}} \sum_{t=1}^{T} \gamma_t(i) \left(\frac{x_t - \mu_i}{\sigma_i} \right) \tag{4}$$

$$\mathcal{G}_{\sigma,i}^X = \frac{1}{T\sqrt{2\pi_i}} \sum_{t=1}^{T} \gamma_t(i) \left[\frac{(x_t - \mu_i)^2}{\sigma_i^2} - 1 \right] \tag{5}$$

Finally, gradient $\mathcal{G}^X_{\mu,i}$ and $\mathcal{G}^X_{\sigma,i}$ are calculated for all the Gaussian. Fisher Vector $\mathcal{G}^X_\theta$ is their concatenation. Therefore Fisher Vector is $2KD$-dimensional.

In this paper, the number of component of Gaussian is 32 and local descriptors reduced to 24 dimensions by PCA. Thus each feature vector is 1536-dimensional. To improve recognition accuracy, we apply power normalization ($\alpha = 0.5$) and L2 normalization [2].

C. Classification

As a classifier, we use a linear kernel SVM, and we adopt the one-vs-rest strategy for multi-class classification. Linear kernel is defined as the inner product of two vectors.

An object recognition method for a smartphone should be low cost and low memory space. In case of linear SVM, the inner product of a support vector and the weight of the corresponding support vector are able to be computed in advance. Then a linear SVM score is given by the inner product $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$ between input data vector $\mathbf{x}$ and weight vector $\mathbf{w}$. Therefore when N is the dimension of feature vector, calculation of a SVM score requires $\mathcal{O}(N)$ operations and $\mathcal{O}(N)$ memory space. We train SVMs with LIBLINEAR [13] in off-line.

In the experiments, since the number of the given categories was 100, we trained 100 linear SVM classifiers. We trained linear SVMs for each feature. Then we combine the output values of both linear SVMs for HoG features and Color features in the late fusion manner

D. Implementation

We implemented a system as a multi-threaded system for using multiple CPU cores effectively. In the experiments, we use a common 4 cores CPU and 4 threads smartphone. We parallelized extraction for HOG Patch and Color Patch feature. Extract descriptor, reduce to dimension by PCA, encoding into Fisher Vector, power normalization, L2 normalization and classify with SVMs are carried out over 2 cores in parallel for each feature, totally over 4 cores in parallel.

In advance, we computed gradient magnitude and orientation for an image. Then we extract HOG Patch descriptors very fast.

The gradient of Fisher Vector with respect to the mean Eq.(4) is deformation as follows to decrease the number of operation. The number of local descriptors T is much bigger than the number of GMM component K and the dimension of local descriptor D, and calculate Eq.(4) for each component, so effectively.

$$\mathcal{G}^X_{\mu,i} = \frac{1}{\sqrt{\pi_i}\sigma_i} \frac{1}{T} \sum_{t=1}^{T} \gamma_t(i)(x_t - \mu_i) \tag{6}$$

Moreover in advance, we computed the term of calculate posterior probability by GMM and the gradient with respect to the mean and sigma in off-line, and we create the lookup table for acceleration (using for calculate posterior probability $\log \pi_i - 0.5 \times \log |\Sigma_i|$, $1/\sqrt{2\pi_i}$ and $1/\sigma^2$ of Eq.(5) and $1/\sqrt{\pi_i}\sigma_i$ of Eq.(6)).

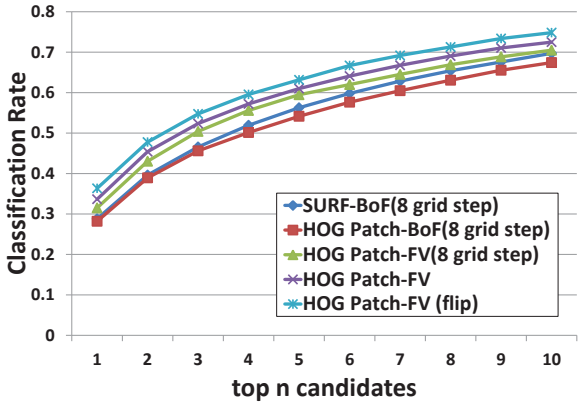

Fig. 2. Classification rate by SURF based BoF, HOG Patch based BoF and FV with 6 or 8 grid step

We trained SVMs in off-line. And all the parameter values using recognition steps are loaded on main memory (eigenvalue and eigenvector for PCA, created lookup table, mean of GMM, weight vectors of SVMs). Although all the values can be stored on main memory in advance, Fisher Vector is able to bring better recognition result with even smaller dictionary than that of BoF. We also set the dimension of feature vectors smaller reduced by PCA. As a result, memory space required for Fisher Vector is smaller than the space for codebook for conventional BoF, and in respect of memory Fisher Vector is also superior to BoF.

Finally, we implemented an Android application and integrate our recognition method into this application. It is characterized as follows. When a user point a smartphone camera toward food items, the system real-time recognizes 100 kinds of food items. Then a fixed time later, update list of recognition result. Finally, the user selects food items from the food candidate list and record daily foods.

IV. Experiments

In this section, we describe experimental results regarding recognition accuracy and processing time. In the experiments, we prepared one hundred categories food image dataset which has more than 100 images per categories and all the food item in which are marked with bounding boxes. The total number of food images in the dataset is 12,905. Figure 1 shows all the category names and their sample photos.

We set the validation data and the test data for each category as 20 images. The rest is the train data, and evaluated classification rate 5 trials, randomly changing the images in the five-fold cross validation manner. Next we measure processing time of image recognition. We use Samsung Galaxy NoteII (1.6GHz 4 cores, 4 tureads, Android4.1) which is the same as [3].

In this experiments, we compare with 2 types of the existing food recognition system. One is a client-side recognition system by Kawano *et al.* [3], and the other is a server-side recognition system by Matsuda *et al* [14].

At this point, we refer [3] which carries out real-time recognition on smartphone, and encoded into BoF representation.

Fig. 1. 100 kinds of food images which are recognition targets in the paper

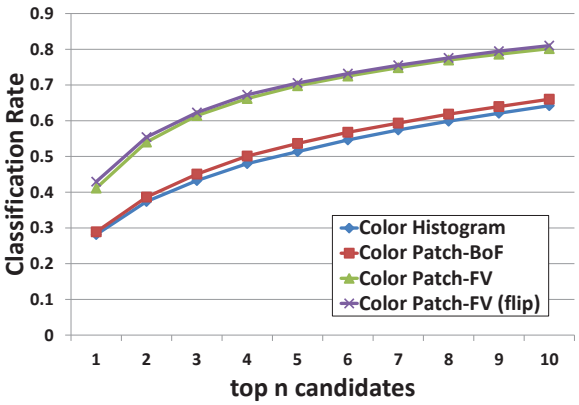

Fig. 3. Classification rate by Color Histogram, Color Patch based BoF and Color Patch based FV

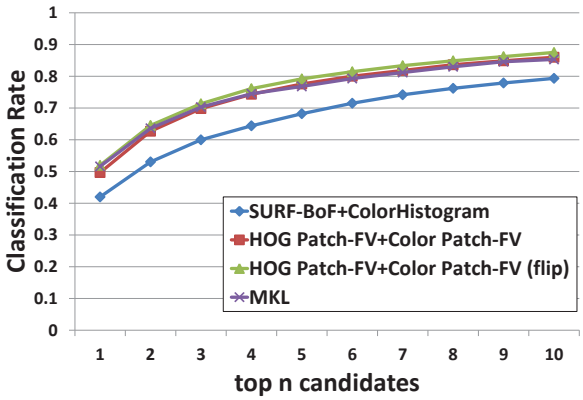

Fig. 4. Classification rate by proposed method, client side [3] and server side [14]

We build a 500 dimension codebook by k-means clustering, and we apply soft assignment [15], then we apply χ^2 kernel feature map [16]. Finally it is a 1500 dimension vector. And we don't apply PCA for local descriptors of BoF representation.

Figure 2 shows the comparison with SURF-BoF [9], HOG Patch-BoF and HOG Patch-FV. First, we set that a step of dense grid sampling is every 8 pixels. The difference top1 and top5 classification rate from SURF-BoF and HOG Patch-BoF is only 0.62% and 2.1%, respectively, which means thath

TABLE I. CLASSIFICATION RATE THE TOP1 AND TOP5 CANDIDATES BY PROPOSED METHOD, CLIENT SIDE [3] AND SERVER SIDE [14]

method	top1	top5
SURF-BoF+ColorHistogram [3]	42.0	68.2
HOG Patch-FV+Color Patch-FV	49.7	77.6
HOG Patch-FV+Color Patch-FV(flip)	51.9	79.2
MKL [14]	51.6	76.8

TABLE II. AVERAGE PROCESS TIME

	time[sec]	ncategory
SURF-BoF+ColorHistogram [3]	0.26	50
HOG Patch-FV+Color Patch-FV	0.065	100

SURF is slight better. However, HOG Patch-FV achieved higher performance than SURF-BoF, differ is 2.7% and 3.22%. And then HOG Patch extraction is very fast than SURF extraction. In this paper, we set that a step of dense grid sampling is every 6 pixels. Then improved classification rate and in case of adding horizontally flipped images for training data, we achieved 36.3% and 63.2% classification rate with top1 and top5 candidates. The classification rate is 7.52% and 6.9% higher than SURF-BOF.

Next, we evaluated Color Patch feature. Figure 3 shows the color histogram, Color Patch-BoF and Color Patch-FV. Color histogram divides a given image into 3×3 blocks, and extract a 64-bin RGB color histogram from each block. Totally, we extract a 576-dim color histogram. Then apply χ^2 kernel feature map. Finally we build a 1728 dimension. The difference top1 and top5 classification rate from Color Patch-BoF and color histogram is only 0.76% and 2.28%, Color Patch-BoF is slight better. But in case of Color Patch-FV, classification rate is much improved, and top1 and top5 classification rate is 13.0% and 18.4% higher than color histogram. In case of adding horizontally flipped images for training data, we achieved 43.0% and 70.6% classification rate.

Then we compare recognition accuracy with exist image recognition system. Our proposed method (HOG Patch-FV+Color Patch-FV) and (HOG Patch-FV+Color Patch-FV(flip)), client side recognition method [3](SURF-BoF+color histogram) and server side recognition method [14]. [14] adopt total 5 features including hard assignment BoF and global feature based feature and classify with nonlinear $\chi^2 RBF$ kernel MKL-SVM, which is very high computational cost.

Figure 4 shows classification rate of each recognition method and Table I shows classification rate with top1 and top5. Our HOG Patch-FV+Color Patch-FV achieved 49.7% and 77.6% classification rate with top1 and top5 and in case of adding flipped images, it achieved 51.9% and 79.2% classification rate with top1 and top5. In case of using only Color Patch feature, we achieved better classification rate than [3], which means that Fisher Vector representation for Color Patch feature is very effective for food recognition. And our approach is better than [14] which server side very high cost recognition system. Therefore we show efficacy our recognition method and it suggests that it is able to carry out high precision image recognition on smartphone.

Next, we measured recognition time by repeating 20 times and averaging them. Table II shows results of average recognition time. We also show experiment value of [3] for 50 kinds of image recognition. Our approach takes 0.065 seconds for 100 kinds. On the other hand, [3] takes 0.26 seconds for 50 kinds. Thus proposed method is very fast than [3], and showed proposed method suitable for rapid image recognition.

Regarding memory space, [3] adopt 1500-dim SURF-BoF and 1728-dim color histogram. On the other hand, our adopt 1536-dim HOG Patch-FV and 1536-dim Color Patch-FV. The feature vector is more compact but more dense. And then many values are loaded on memory to encode Fisher Vector faster, sum of these require only less one fifth in case of HOG Patch and less one fourth in case of Color Patch than memory space of codebook for BoF, respectively. Java heap (mainly except image processing) and native heap (mainly image processing) the implemented application required were about 16MB and 3MB. We realize the mobile system with low memory space. Therefore we are able to increase the number of recognition target and feature dimension.

According to the experiments of recognition accuracy and processing time, we showed efficacy our proposed method. And we achieved better result than server side very high cost recognition method [14]. Therefore we showed rapid image recognition and high precision on smartphone is possible.

V. Conclusion

We proposed image recognition method for run on a common smartphone that HOG Patch and Color Patch descriptor as local feature, then they are Fisher Vector represented, finally classify with linear SVM.

In the experiments, we have achieved 79.2% classification rate with the top five candidates when ground-truth bounding boxes are given. It is higher 11.0% than recognition method carries out on a smartphone that adopt color histogram and SURF-BoF with soft assignment and apply χ^2 kernel feature map. And it is also higher than very high cost server side recognition method. In addition, processing time is 0.065 second for 100 kinds of target. It is faster 75.0% than 0.26 of [3]. We showed efficacy of proposed method and rapid and high precision image recognition on a smartphone.

As a feature works, we plan to extend the method and system regarding the following issues. Increase recognition targets by using crowd-sourcing and run large scale image recognition on a smartphone. Then efficient classify by hierarchical classify. Not only run rapid image recognition, but also run rapid interest object detection.

Note that Android application of the implemented mobile food recognition system can be downloaded from http://foodcam.mobi/.

References

[1] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *Proc. of IEEE Computer Vision and Pattern Recognition*, 2007.

[2] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Proc. of European Conference on Computer Vision*, 2010.

[3] Y. Kawano and K. Yanai, "Real-time mobile food recognition system," in *Proc. of IEEE CVPR International Workshop on Mobile Vision (IWMV)*, 2013.

[4] N. Kumar, P. Belhumeur, A. Biswas, D. Jacobs, W. Kress, I. Lopez, and J. Soares, "Leafsnap: A computer vision system for automatic plant species identification," in *Proc. of European Conference on Computer Vision*, 2012.

[5] T. Lee and S. Soatto, "Learning and matching multiscale template descriptors for real-time detection, localization and tracking," in *Proc. of IEEE Computer Vision and Pattern Recognition*, 2011.

[6] T. Maruyama, Y. Kawano, and K. Yanai, "Real-time mobile recipe recommendation system using food ingredient recognition," in *Proc. of ACM MM Workshop on Interactive Multimedia on Mobile and Portable Devices(IMMPD)*, 2012.

[7] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. of IEEE Computer Vision and Pattern Recognition*, 2005.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[9] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[10] G. Csurka, C. Bray, C. Dance, and L. Fan, "Visual categorization with bags of keypoints," in *Proc.of ECCV Workshop on Statistical Learning in Computer Vision (SLCV)*, 2004, pp. 59–74.

[11] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman, "The devil is in the details: an evaluation of recent feature encoding methods," in *Proc. of British Machine Vision Conference*, 2011.

[12] D. Jia, B. Alex, S. Sanjeev, S. Hao, K. Aditya, and L. Fei-Fei, "Imagenet large scale visual recognition challenge 2012 (ILSVRC2012)," 2012, http://www.image-net.org/challenges/LSVRC/2012/index.

[13] R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin, "LIBLINEAR: A library for large linear classification," *The Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[14] Y. Matsuda, H. Hoashi, and K. Yanai, "Recognition of multiple-food images by detecting candidate regions," in *Proc. of IEEE International Conference on Multimedia and Expo*, 2012.

[15] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Lost in quantization: Improving particular object retrieval in large scale image databases," in *Proc. of IEEE Computer Vision and Pattern Recognition*, 2008.

[16] A. Vedaldi and A. Zisserman, "Efficient additive kernels via explicit feature maps," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.