IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 13, 2020, accepted June 22, 2020, date of current version July 3, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.3005150*

# Multi-Scale CNN for Fine-Grained Image Recognition

**CHEE SUN WON, (Member, IEEE)**
Department of Electronics and Electrical Engineering, Dongguk University, Seoul 04620, South Korea
e-mail: cswon@dongguk.edu

This work was supported in part by the Basic Science Research Program of the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant NRF-2018R1D1A1B07043542, and in part by the Dongguk University Research Fund.

**ABSTRACT** Most conventional fine-grained image recognitions are based on a two-stream model of object-level and part-level CNNs, where the part-level CNN is responsible for learning the object-parts and their spatial relationships. To train the part-level CNN, we first need to separate parts from an object. However, there exist sub-level objects with no distinctive and separable parts. In this paper, a multi-scale CNN with a baseline Object-level and multiple Part-level CNNs is proposed for the fine-grained image recognition with no separable object-parts. The basic idea to train different CNNs of the multi-scale CNNs is to adopt different scales in resizing the training images. That is, the training images are resized such that the entire object appears as much as possible for the Object-level CNN, while only a local part of the object is to be included for the Part-level CNN. This scale-specific image resizing approach requires a scale-controllable parameter in the image resizing process. In this paper, a scale-controllable parameter is introduced for the linear-scaling and random-cropping method. Also, a line-based image resizing method with a scale-controllable parameter is employed for the part-level CNNs. The proposed multi-scale CNN is applied to a food image classification, which belongs to a fine-grained classification problem with no separable object-parts. Experimental results on the public food image datasets show that the classification accuracy improves substantially when the predicted scores of the multi-scale CNN are fused together. This reveals that the object-level and part-level CNNs work harmoniously in differentiating subtle differences of the sub-level objects.

**INDEX TERMS** Convolutional neural network (CNN), fine-grained image classification, food recognition, image resizing.

## I. INTRODUCTION

Convolutional neural networks (CNN) with deep layers have contributed significantly to the performance improvement for the object-wise image classification problems. This success has encouraged researchers to solve more challenging problems with CNNs, the fine-grained image classification of recognizing the sub-level classes under an upper-level class. The main difficulty in the fine-grained image classification problem comes from the nature of the domain-specific sub-level images, which have large intra-class and small inter-class variances [1], [2]. Moreover, the domain-specific images often demand the involvement of the expertise for object labeling, which is an expensive task [2]. Although the crowdsourcing [3] can be an alternative, it often causes a noisy labelling problem.

The associate editor coordinating the review of this manuscript and approving it for publication was Kathiravan Srinivasan.

Note that the performance of the fine-grained classification depends considerably on the separability and the localizability of the sub-level object and its parts in the image. For example, if the upper-level object is a bird class and the problem is to classify a bird image into one of many sub-level bird types [3], then the CNNs trained with the prior knowledge of the bird parts such as the head and the tail can significantly improve the classification performance [4]. The upper-level objects such as a dog and a car also belong to the part-separable fine-grained images. However, there are domain-specific sub-level images with no separable parts. As shown in Fig. 1, the bird images in Fig. 1(a) have a couple of distinctive object-parts such as head and tail, while the food images in Fig. 1(b) have no explicit object-parts.

For the fine-grained images, researchers have acknowledged that the combined features of a global object and a local part yield a better classification performance. This leads the researchers to treat the overall appearance of the objects and

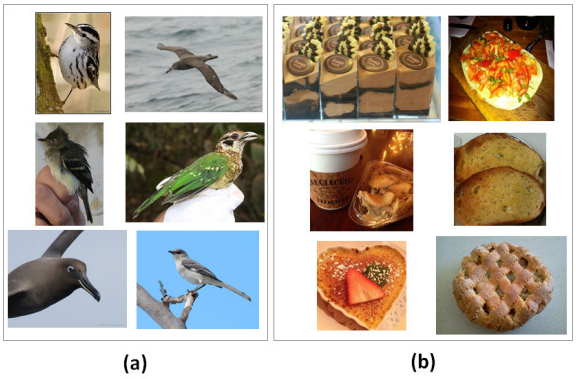


**FIGURE 1.** Image samples for the fine-grained classification from the datasets of: (a) CUB-200-2011 dataset [7] with separable parts, (b) Food-101 dataset [10] with no separable parts.

their parts separately in training CNNs. Unfortunately, not all domain-specific datasets have localizable and separable common parts. In this paper, we focus on such a weakly supervised fine-grained image classification problem that has neither object nor part annotations. Also, it is assumed that there are no object bounding boxes available for CNN training and testing. That is, we deal with a weakly supervised classification problem for a domain-specific problem, which has no explicit object-parts to learn among different sub-level objects.

In this paper, we deal with a classification problem for food images, which belongs to the domain-specific datasets with no distinctive object-parts. Having no explicit food parts to compare with different sub-level food images, the strategy adopted in this paper is to train multiple CNNs with different levels of food details. That is, each CNN is exclusively trained by a certain level of food details. This can be done by resizing a training image with different scale factors for different CNNs. Specifically, this paper adopts a multi-scale CNN with multiple levels of the food details, one for object-level and others for part-levels. The Object-level CNN is trained by resized images that cover global food appearances in them. For this, a linear scaling followed by a cropping [5] is used. This image resizing method can be also used for the Part-level CNN by controlling the scaling factors. Note that the image resizing for the Part-level CNN needs more pruning than scaling to fill the resized image only with a local part of the object. To this end, a line-based image resizing method [6] is also employed, which optimally selects the image lines to remove under the constraints of the target aspect ratio, the image saliency, and the target number of image-lines via a linear programming method. Here, the goal is that only a local detail of the food image is included in the resized image by setting the number of image-lines as a scale-constraint for the optimization.

The contributions of this paper can be summarized as follows

1) This work formulates a new problem for a set of domain-specific image datasets that cannot be benefited from existing fine-grained image recognition algorithms. The problem is how to effectively classify the sub-level objects with non-separable object-parts. The food classification belongs to this problem.
2) This work shows that a multi-scale CNN trained by a scale-controllable image resizing method can boost the recognition performance for food datasets substantially, proving that images resized in different scales help multi-scale CNNs learn different levels of object details without explicitly segmenting the object into parts.
3) A new sequential method of CNN training for a multi-scale CNN is proposed. First, the Object-level CNN learns the object appearance as a whole with the training images resized using an object-level scaling factor. Then, the trained Object-level CNN is used to fine-tune the Part-level CNNs with the resized images in various part-level scales. This sequential training process allows us to share the early-layers of the Object-level CNN with the Part-level CNNs, saving a lot of memory space for the trained coefficients.
4) The existing image resizing methods are reformulated to include a scale-controllable parameter. Specifically, the resizing-scale can be readily controllable by a parameter associated with the size-constraint in the line-based constraint optimization framework. Also, a formula with a scale-controllable parameter is introduced for the linear-scaling and random-cropping method. By setting the parameter appropriately, one can resize the images in different scales for the Part-level CNNs.
5) Only two scales for the proposed multi-scale CNN are enough to achieve state-of-the-art classification performance for three food datasets such as UECFood-256, Food101, and VireoFood-172 with the pre-trained ResNet50. Results with the pre-trained InceptionResNet-v2 also show that adding one more scale to have a three-scale CNN (i.e., one for the Object-level and the other two for the Part-level CNNs) improves the recognition performance substantially.

This paper is organized as follows. Related works, focusing on the fine-grained image classifications, are reviewed in Section II. The proposed two-scale CNN is introduced in Section III followed by experimental results in Section IV. Finally, Section V addresses the concluding remarks.

## II. RELATED WORKS

There are two main groups for the fine-grained sub-level object classifications. The first group makes use of the distinctive object-parts as well as the whole object appearance. So, for this approach, there should exist localizable and separable parts in the objects. The domain-specific datasets such as birds-200-2011 [7], Stanford dogs [8], and Stanford cars [9] belong to this group. The second group of the domain-specific datasets has no distinctive and localizable parts in the sub-level objects. So, since no part information can be

explicitly extracted and utilized for CNN training and testing, the classification problem becomes more challenging. The food classification problem with the datasets such as Food-101 dataset [10], UEC Food256 dataset [11], and Vireo Food-172 dataset [12] belongs to this group.

### A. FINE-GRAINED CLASSIFICATIONS WITH LOCALIZABLE PARTS IN OBJECTS

In order to exploit the part information for the sub-level object classification, above all, the sub-level objects and their parts should be separable. Some of the image datasets provide the location information of the objects with the bounding boxes. The classification performance certainly improves by making use of the object bounding boxes, especially in the CNN testing stage [13]. Of course, the bounding boxes can be also used for CNN training [14]. Once the object is localized, its distinctive parts can be separated from the object for the part-level learning. This leads to the two-level attention model, which simulates the human vision system that sees the object first and then its most discriminative parts later [1], [4], [15].

The part localization and annotation can be done manually for a training dataset. However, especially for a domain-specific dataset, the manual annotation demands an expensive involvement from the experts. A more realistic approach for the fine-grained image classification problems is not to use neither object nor part annotations in both training and testing stages. Under this weakly supervised environment, the localizations of the object and its parts can be done by a learned CNN model. That is, the part model can be learned in an unsupervised fashion [16] by a constellation model to localize parts of the object. Specifically, the local regions in the image that fire at the similar locations of the activation maps in a CNN are used to detect the object and its parts in the image. Then, the image patches proposed from the detected local regions can be used to establish the constellation model for the spatial relationship among the object-parts [1]. Also, the trained constellation model can be used to filter out the noisy object proposals by using the Selective Search method [17]. The patch images selected from an original image can be proposed from multi-scales and multi-views and, among them, noisy image patches that are not relevant to the object can be removed by the learned CNNs. Then, the main CNN model uses the remaining patches for both training and testing. However, this constellation model based method works well only when the sub-objects share some distinctive parts with a spatial relationship. The activeness (or objectness) of the activation map is also used for the attention models [4], [15], where the existence of object-part increases the objectness level. In [4], a two-level attention model with object-level and part-level attentions is used to aggregate the object and part features for the final classification. In [15], a graph analysis algorithm was used to find the object bounding box by considering the object co-localization from the similar images in the training dataset. Recently, more attention has been paid on the part-level features for the fine-grained classification by adopting end-to-end trainable part-localization network and part-classification network [18]. In [19] and [20], the attention model learns the image patches to be extracted, which are called glimpse, at varying sizes. The attention model of [20] is applied to the Stanford Dogs fine-grained categorization problem [8]. Recently, in [21], a neural model which learns attention from lower-level feature activations without requiring part annotations was proposed. In [22] a DCL (Destruction and Construction Learning) stream to learn from discriminative regions automatically was proposed. The idea of this approach is to disrupt the spatial layout of the training images by a block shuffling to guide the CNN to pay more attention to the discriminative local regions. The performance of this DCL-based method was tested for the part-separable image datasets such as birds, cars, and aircrafts. In [23] the part level features were learned by uniformly dividing input channels into several semantic groups. Then, a deep bilinear transformation (DBT) block was employed to learn their pairwise interactions. The performance of the DBT block was also evaluated by the part-separable datasets of birds, cars, and aircrafts.

### B. FINE-GRAINED CLASSIFICATIONS WITHOUT EXPLICITLY SEGMENTING OBJECT-PARTS

Most approaches introduced in Section II-A are based on the premise that there are distinctive and separable parts in sub-level objects. However, there is also another school of the fine-grained image categorization with no exclusively separable parts among different sub-level objects. In this case, since the sub-level objects typically share no specific parts with some spatial arrangements [24], detailed textural features in an object as well as its overall layout may help differentiate different sub-level classes. Food categorization belongs to this group. The shape of the dining plates cannot be the part information of the sub-level foods because, in reality, different sub-level foods are laid on the dining plates with a round shape and some of food contents such as the bread also have the round-shape [25]. Having no separable parts in food images, a specific food structure such as the vertical food trait has been exploited for the sub-food classification [26].

Note that the food images usually exhibit more complex layout with no clear-cut part-separation than the other fine-grained image datasets introduced in Section II-A. Also, the food datasets include the images with large inter but small intra variances [24]. Therefore, food image features extracted from a multi-scale and multi-view structure should help boost the recognition performance. For example, in [24], Multi-Scale Multi-View Feature Aggregation (MSMVFA) was proposed to fuse the features in different levels such as high-level semantic features, mid-level attribute features, and deep visual features into the food categorization. The MSMVFA consists of two-level fusion, namely multi-scale fusion for each type of features and multi-view aggregation for different types of features. In the MSMVFA, however, an ingredient network needs to be trained with multiple

ingredient labels, where the ingredient information should be available as an annotation.

## III. MULTI-SCALE IMAGE RESIZING

Having no explicit object-parts, the resized images in different scales can be alternatively used for the object-level and the part-level images. Specifically, the original image is resized by multiple scales and, then, the resized image for each scale is used to train a scale-specific CNN, yielding a multi-scale CNN. So, each convolutional pathway in the multi-scale CNN learns its own scale-specific object features in training images. The multi-scale CNN architecture was adopted in [27], where they employed it for a brain image segmentation. Each CNN pathway is trained with the down-sampled images with a given down-sampling factor.

The CNN architecture proposed in this paper is similar to the multi-scale CNN in [27]. In training the multi-scale CNNs, however, we adopt a sequential training process (see Fig. 2). That is, at the top-level, a publically available pre-trained CNN is transfer-learned for the highest object-level scale of Scale-1. Then, we use the trained weights in the object-level CNN as initial filter-coefficients and fine-tune the rest of the multi-scale CNNs (i.e., Scale-2,$\cdots$, Scale-N). Specifically, as shown in Fig. 2(a), the early layers from a pre-trained CNN and the new replaced layers are fine-tuned by the training images that are resized by setting the scale factor as Scale-1 for the highest objectness. This object-level CNN of Scale-1 serves as a baseline CNN for all remaining multi-scale CNNs and the next Scale-2 CNN is formed by assembling the fine-tuned early layers of the base-line CNN and its new replaceable layers. Then, it is fine-tuned with the training images resized by the next level of the scale factor, Scale-2. This fine-tuning process for the assembled CNN for different scale factors continues until we have the fine-tuned CNN for the finest local details of Scale-N. Since the weights of the baseline CNN for the object-level CNN are reused as initial weights for all multi-scale CNNs, their training convergences are fast. The multi-scale CNN can be viewed as a two-level hierarchical structure, where the top-level CNN represents the macroscopic object-level features and the bottom ones learn the microscopic part-level features in different scales. However, unlike the hierarchical set-subset relationship as the case of word semantics in [29], the relationship between the top-level object and the bottom-level parts are more like complementary.

Note that the learned weights of the entire CNN paths demand a lot of memory space. To save the memories, as shown in Fig. 2(b)), we keep the full CNN weights of Scale-1 (i.e., the object-level CNN), but only the fine-tuned replaced layers for the rest of scales (i.e., Scale-2,$\cdots$, Scale-N). Then, at the inference stage, the early layers of the base-line CNN of Scale-1 are assembled with the fine-tuned replaced layers for all scales of CNNs (see Fig. 3). Of course, this reuse of the early layers in the object-level Scale-1 CNN for all other prat-level CNNs of Scale-2,$\cdots$, Scale-N may deteriorate the recognition performance. However, since the early layers encode very primitive image features, the performance decline should be limited.

### A. TWO-SCALE CNN ARCHITECTURE

For the food recognition problem, by setting $N = 2$ in Fig. 2 and 3, we have the most simple multi-scale CNN with only two scales of Object-level and Part-level CNNs (see Fig. 4 and 5). The Object-level CNN is responsible for learning the global appearance of the sub-level objects, whereas the Part-level CNN is to learn their local details. Following the sequential fine-tuning process in Fig. 2, a pre-trained CNN is transfer-learned to have the Object-level CNN first and, then, it is fine-tuned to have the Part-level CNN. Specifically, as shown in Fig. 4, a pre-trained CNN is transfer-learned by replacing the replaceable layers with new ones. Then, the early-layers of the transfer-learned Object-level CNN are used as the baseline of the two-scale CNN. So, they are combined with new replaceable last layers and are fine-tuned to have the Part-level CNN. To train the two CNNs the original training images are resized with different scales. For the Object-level CNN, the training images are resized such that they include the whole appearance of the sub-level object as much as possible. On the other hand, only a part of the object is to be included in the resized image for the Part-level CNN.

Note that the Object-level CNN is transfer-learned from a pre-trained CNN, which is trained by a general-purpose dataset, and the Part-level CNN is fine-tuned from the Object-level CNN. Therefore, the Object-level CNN needs more iterations for convergence than the Part-level CNN. Also, the image resizing for the Object-level CNN is relatively simple, because most of the original images already cover the whole appearance of the objects. We just need a linear scaling and a little bit of cropping for the Object-level CNN. So, in this paper, the linear-scaling and random-cropping with a light scale jittering [5] is used to resize training images for the Object-level CNN. On the other hands, to include only a local detail of the object for the Part-level CNN, a more sophisticated method of image resizing is needed. To this end, we may exploit the line-based optimal method [6] and the linear-scaling and random-cropping with a scale-controllable parameter [5]. More details about these methods will be addressed in Section III-B and Section III-C, respectively.

At the inference stage, a testing image is also resized with the two different methods used in the training. However, since the jittering effect is unnecessary for the testing, the scaling factors are fixed for the Object-level CNN and the Part-level CNN. Feeding the two resized images to the two-scale CNN as input images, we have two outputs of the score vectors from the two CNNs, which are fused for the final classification (see Fig. 5).

### B. LINE-BASED OPTIMAL IMAGE RESIZING FOR PART-LEVEL CNN

The two-scale CNN paths in Fig. 4 are expected to be complementary to each other in recognizing the sub-level objects.

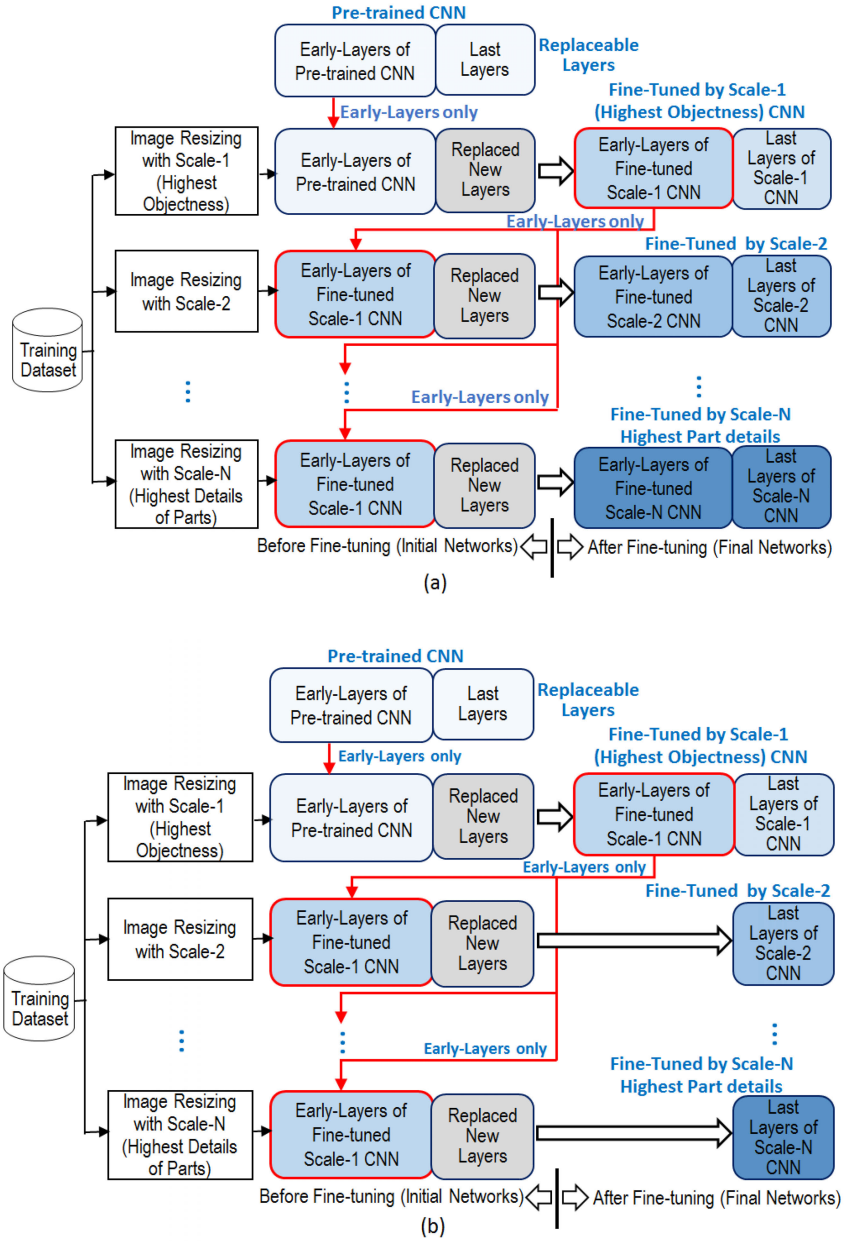


**FIGURE 2.** The sequential fine-tuning process for different scales with the common base-line early layers: (a) The fine-tuned early layers as well as the replaced ones are stored for the inference, (b) Only the fine-tuned replaced layers are stored for the inference.

That is, the Object-level CNN is to differentiate the macroscopic appearance among the sub-level objects, while the Part-level CNN is to detect the microscopic differences. Then, the original training images should be resized to provide microscopic local details of the objects to the Part-level CNN.

The line-based image resizing method based on the linear programming framework [6] is adopted for the Part-level CNN, where the image is resized by a line-by-line deletion for size-reduction or a line-by-line duplication for size-expansion depending on the size differences between the original and target images. The line-based image resizing method can be viewed as a binary decision problem such that each image line is to-be-remained (binary 1) or to-be-deleted (binary 0), which can be done by an optimal linear programming method. Followings are the brief explanation on the specific image resizing method of [6] used in this paper.

Suppose that the original image $I$ has a size of $M \times N$. Then, there are $M + N$ image lines and our goal is to make an optimal binary decision on each image line as either to-be-deleted (binary 0) or to-be-remained (binary 1). Let us denote $1 \times (M + N)$ row vectors of $f$ and $s$ as the cost function and the indicator of the binary decisions, respectively. Then, we

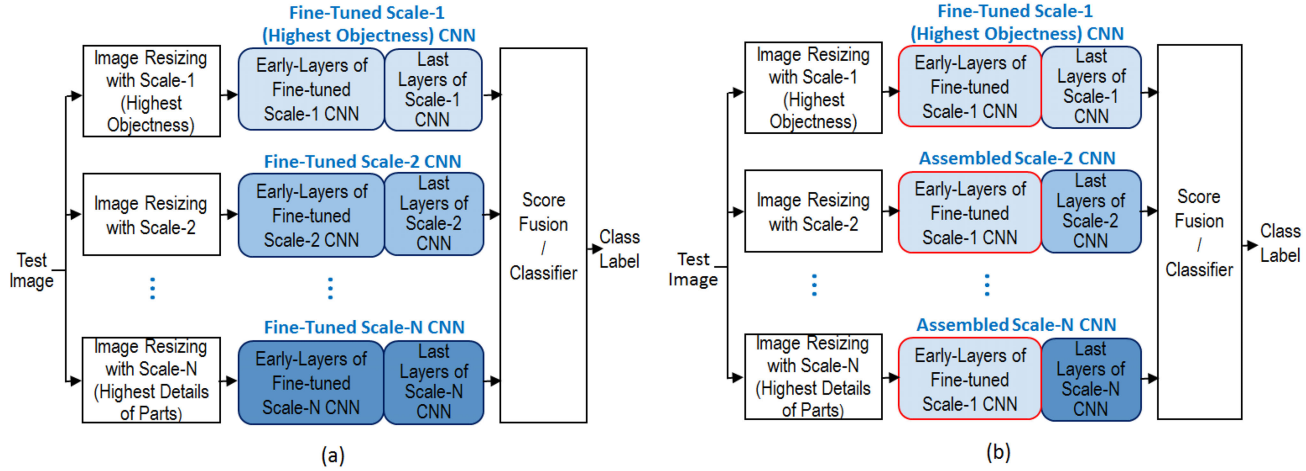


**FIGURE 3.** The multi-scale CNN for the inference process: (a) Fine-tuned early layers for all scales of Fig. 2(a) are used, (b) Only fine-tuned replaced layers are used and the early layers are from the object-level CNN.

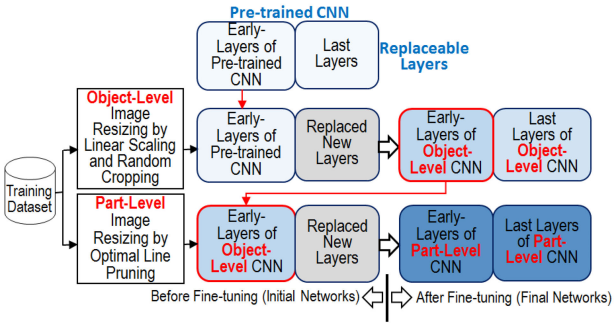


**FIGURE 4.** Two-scale training for Object-level and Part-level CNNs.

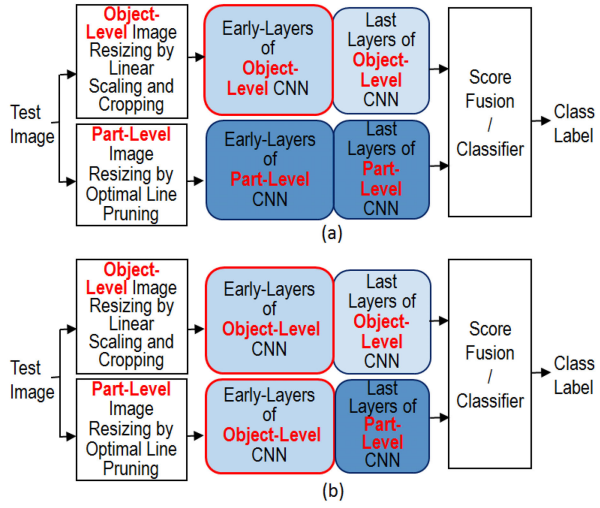


**FIGURE 5.** Two-scale inference with (a) the fine-tuned early layers of the Part-level CNN, (b) the fine-tuned early layers of the Object-level CNN.

have $f = [f(1)\cdots f(M)\ f(M+1)\cdots f(M+N)]$, where each element $f(k)$ represents the cost for deleting the image-line at $k$. The element of the binary indicator vector $s = [s(1)\cdots s(M)\ s(M+1)\cdots s(M+N)]$ takes $s(k) \in \{0, 1\}$, representing the binary decision of the line-keeping with $s(k) = 1$ or the line-deleting with $s(k) = 0$ at line $k$. Now, the optimal line selection can be made by following the linear programming framework with the objective function given by the inner product of $f$ and $s$, and with some constraints

$$s^* = \arg\min_s fs^T \tag{1}$$

such that

$$As \le b, \tag{2}$$

$$lb \le s \le ub, \tag{3}$$

where the associated matrices (or scalar) $A$ and $b$ are determined by the specific requirements for the optimization of (1). Note that, to yield binary values 0 and 1 for the element of $s$, the lower bound $lb$ and the upper bound $ub$ in (3) are set to zero vector and one vector with $M + N$ elements, respectively. For more details on the optimization of (1) via the linear programming technique and the various possible constrains (2) on the image resizing, refer to [6].

The training images for the two-scale CNN are resized from the original size of $M \times N$ to the target size of $M' \times N'$, where $M' \times N'$ fits the input of the CNN. Now, the original image is first resized to an intermediate size of $M_t \times N_t$ to have the similar aspect rstio of $M_t/N_t \approx M'/N' = \gamma$ by the linear programming framework based on (1)$\sim$ (3), then a simple linear-scaling technique is used to resize the intermediate image of $M_t \times N_t$ to have the final one of $M' \times N'$. The objective function to be minimized in (1) is given by the inner product between $f$ as in (4) and $s$ as the binary indicator vector. Among a set of requirements expressed by $A$ and $b$ in [6], the following constraints for the food classification problem are used in this paper to guarantee $M_t/N_t \approx M'/N' = \gamma$ with some tolerable margins as in (6) and (7)

$$f = [-E_r(1)G_r(1)\cdots - E_r(M)G_r(M) - E_c(1)G_c(1)\cdots - E_c(N)G_c(N)] \tag{4}$$

$$s = [s(1) \cdots s(M)\ s(M+1) \cdots s(M+N)] \tag{5}$$

$$A = \begin{bmatrix} \gamma & \cdots & \gamma & -1 & \cdots & -1 \\ -\gamma & \cdots & -\gamma & 1 & \cdots & 1 \\ 1 & \cdots & 1 & 1 & \cdots & 1 \end{bmatrix} \tag{6}$$

$$b = \begin{bmatrix} \beta \times |M-N| \\ \beta \times |M-N| \\ \alpha \times (M+N) \end{bmatrix} \tag{7}$$

$$lb = [0 \cdots 0]^T, \quad ub = [1 \cdots 1]^T \tag{8}$$

$E_r(i)$ and $E_c(j)$ in (4) are obtained by accumulating the normalized edge magnitudes along the row $i$ and column $j$, respectively. The accumulated edge magnitudes are weighted by Gaussian functions of $G_r(i) = \exp(-(i-i_c)^2/2\sigma^2)$ and $G_c(j) = \exp(-(j-j_c)^2/2\sigma^2)$ centered at a pixel $(i_c, j_c)$, which force the optimization to choose the to-be-deleted lines faraway from the center of the image. Top two rows in (6) and (7) are to meet the requirement of the target aspect ratio $\gamma$ (i.e., $\gamma = \frac{N'}{M'}$), where the tolerable margin toward the exact aspect ratio $\gamma$ is given by $\beta \times |M-N|$ as in (7). The parameter $\beta$, which is less than 1, controls the amount of tolerable image lines as a margin toward the exact $\gamma$. The third rows in (6) and (7) are to set the lower bounds for the number of image lines such that the total number of remaining lines should be less than $\alpha \times (M+N)$. So, $\alpha$ controls the minimal number of image lines to be deleted. Therefore, the above linear programming process optimally selects at least $(1-\alpha) \times (M+N)$ image lines to be deleted such that the image size becomes $M_t \times N_t$ with the approximate aspect ratio, $\gamma$, by a margin of $\pm\beta \times |M-N|$. In this paper, the parameter $\alpha$ is used as a scaling factor and can be chosen randomly within a pre-determined range in training, giving a jittering effect.

### C. IMAGE RESIZING BY LINEAR-SCALING AND RANDOM-CROPPING FOR PART-LEVEL CNNs

Image resizing for the Object-level CNN takes a sequential execution of linear scaling followed by random cropping. This resizing method can be used for the Part-level CNN as well. Then, we need a scale-controllable parameter $S$. In [5], the scaling factor $S$ was chosen uniformly from a fixed range, say $S \in [256, 512]$, regardless of the sizes of the original images. In this paper, the sizes of the original image and the target size are considered in the random selection of $S$ as follows

$$S = \lceil (O_{min} - T_{min}) \times \eta + (\frac{O_{min} + T_{min}}{2}) \times \zeta \rceil \tag{9}$$

where $O_{min} = \min\{M, N\}$ and $T_{min} = \min\{M', N'\}$. $\eta$ is a random number generated from a normal distribution, where the mean is 0 and the standard deviation is $\sigma_S$. Because of the Gaussian random number $\eta$, the scaling factor $S$ determined by (9) will be different for every resizing attempt, giving a jittering effect. $\zeta$ is a predetermined constant, which is associated with the expected value of $S$. For example, since the mean of the Gaussian distribution is set to zero, the scaling factor S is mostly chosen in between the original image size $M \times N$ and the target size $M' \times N'$ (i.e., $(O_{min} + T_{min})/2$) for $\zeta = 1$. In this case the half of the size gap between original $M \times N$ and the target $M' \times N'$ is handled by the linear scaling and the remaining half to the target size is treated by the cropping. The level of the jittering effect can be controlled by the standard deviation $\sigma_S$ of $\eta$. That is, by setting $\sigma_S$ with a small value, the scaling factors chosen from (9) will be in a narrow range and crowded around $(O_{min} + T_{min})\zeta/2$. In this paper, we set $\sigma_S = 0.5$. Note that, to avoid the degenerate case, if $S$ chosen from (9) is less than $T_{min}$, we set $S = T_{min}$.

Note that $\zeta$ in the offset term $(O_{min} + T_{min})\zeta/2$ of (9) can be used to change the expected size of $M_t \times N_t$. For example, if $\zeta > 1$, then the expected value of $S$ increases and is closer to $O_{min}$ for $O_{min} > T_{min}$. This implies that the amount of size reduction by the linear scaling becomes smaller than the case of $\zeta = 1$, which means that the subsequent image cropping has more freedom to choose the image patch of $M' \times N'$ from the wider area of the linearly-scaled image of $M_t \times N_t$. As a result, for $\zeta > 1$, it is expected that the final cropped images of $M' \times N'$ are filled with the part-level local details. The opposite will happen for $\zeta < 1$.

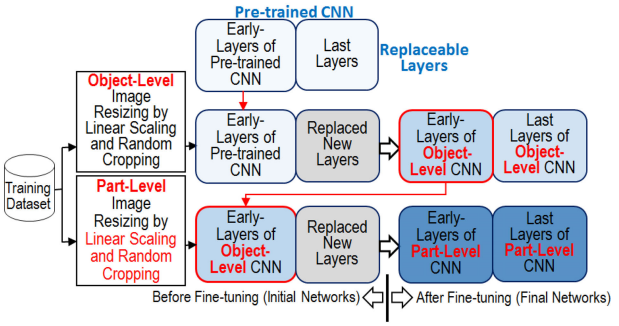


**FIGURE 6.** The linear-scaling and random-cropping method can be used for both Object-level and Part-level CNNs by setting $\zeta$ in (9) differently. For example, we can set $\zeta = 1$ for the Object-level CNN and $\zeta = 4/3$ for the Part-level CNN.

As an alternative to the line-based image resizing of Section III-B, the linear-scaling and random-cropping can be also used for the proposed multi-scale CNNs with $\zeta > 1$ in (9). That is, as shown in Fig. 6, the image resizing method of the linear-scaling and random-cropping can be used for both the Object-level and Part-level CNNs. Here, the scaling factors are chosen from (9) with $\zeta = 1$ and with $\zeta > 1$ (say, $\zeta = 4/3$) for the Object-level CNN and the Part-level CNN, respectively. At the inference stage, the scaling factors for the Object-level and the Part-level CNNs are fixed as $S = (O_{min} + T_{min})\zeta/2$ by setting $\eta = 0$. Of course, the two-scale CNN can be extended to have a three-scale CNN, where we may set $\zeta = 1.25$ for the first Part-level (Mid-level) CNN and $\zeta = 1.5$ for the second Part-level CNN.

## IV. EXPERIMENTS

Pre-trained ResNet50 [28] and InceptionResNet-v2 [30] are used as the baseline CNN for the proposed multi-scale CNN and they are transfer-learned for the Object-level CNN as

in Fig. 2. The sizes of the CNN input for the ResNet50 and the InceptionResNet-v2 are fixed as $224 \times 224$ (i.e., $M' = N' = 224$ and $\gamma = 1$) and $299 \times 299$ (i.e., $M' = N' = 299$ and $\gamma = 1$), respectively.

For the Part-level CNN of Fig.4, we first resize the original $M \times N$ image to $M_t \times N_t$ by the line-based linear programming optimization method using (4) $\sim$ (8). Then, a linear scaling is used to get the final size of $M' \times N'$. To give a jittering effect, the centers of Gaussian weighting $G_r(i)$ and $G_c(j)$ in (4) are chosen randomly as $i_c + 0.5 \times (\xi - 0.5) \times M$ and $j_c + 0.5 \times (\xi - 0.5) \times N$, respectively, where $i_c = M/2$ and $j_c = N/2$ are the centers of the two sides of the image and $\xi$ is a random number in $[0, 1]$. Also, considering the square input for the ResNet50 and the InceptionResNet-v2, we set $\sigma = \min(M, N)/5$ for both the Gaussian weights of $G_r$ and $G_c$, which forces the optimization to remove more lines at the longer side of the image. There are two parameters $\alpha$ and $\beta$ in (7). In this paper, $\beta$ is fixed as $\beta = 0.1$, but $\alpha$ is chosen randomly from $\alpha \in [0.45,\ 0.65]$. The range $[0.45,\ 0.65]$ for $\alpha$ is determined experimentally, which forces the optimization to remove as many image lines as possible for the resulting $M_t \times N_t$ image to have only a local part of the object in the image.

At the inference stage, the scaling factor for the Object-level CNN is fixed as $S = \lceil T_{min} \times 0.1 + T_{min} \rceil$ and the cropping is done at the center of the linearly scaled image to have the final size of $M' \times N'$. For the Part-level CNN, the centers of Gaussian weighting factors of $G_r(i)$ and $G_c(j)$ in (4) are fixed at the center of the original image (i.e., $i_c = M/2, j_c = N/2$). Also, $\alpha$ and $\beta$ in (7) are fixed as 0.55 and 0.1, respectively.

The datasets used in this paper are Food-101 dataset [10], UEC Food256 dataset [11], and Vireo Food-172 dataset [12]. Among the three datasets, the Food-101 provides separate training and testing datasets. For the other two datasets, the images in the datasets are randomly divided into three sets of training and testing pairs with 80% of the original datasets for training and the remaining 20% for testing. The final mean accuracies (mAP) are obtained by averaging the results of the three splits.

Due to the randomly chosen parameters associated with the image resizing methods, the resized image for the same training image looks different for every epoch. This gives a jittering effect to overcome the overfitting problem in training [5]. To give a sense of how different they are, Fig. 7 shows five consecutively resized images for the Object-level and the Part-level CNNs. The leftmost images in the figures are the original images chosen from three food classes in UEC Food256 dataset (Fig. 7(a)) and Food-101 dataset (Fig. 7(b)). As shown in the figures the resized images tend to include a whole object with the plate for the Object-level CNN, while only a local part of food for the Part-level CNN. However, because of the randomness, the resized images for the object-level sometimes have part-level flavor, and vice versa. To some extent, this may be necessary to overcome the

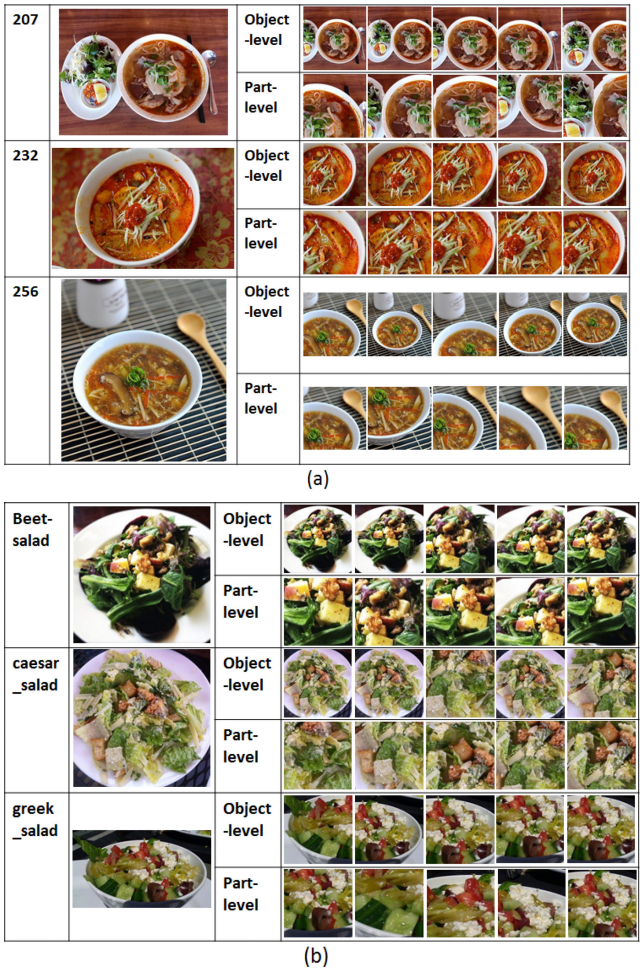


**FIGURE 7.** Five consecutively resized images for the Object-level and the Part-level CNN: (a) UEC Food256 dataset, (b) Food-101 dataset.

overfitting problem. It is worth mentioning that the resized images for every epoch are not stored and are not added to the original training dataset.

The stochastic gradient descent with momentum optimizer is used to minimize the cross-entropy loss for the CNN fine-tuning. The hyper-parameters are fixed as follows: the momentum is 0.9, the mini-batch size is 16, the maximum epoch is 10 for the Object-level CNN and 7 for the Part-level CNN, the initial learning rate is 0.001, the learning rate drop factor is 0.5, and the learning rate drop period is 3. Also, the training data is shuffled before each training epoch. No other image augmentation methods such as the rotation is used for the experiments except the mean subtraction for each color channel.

Since the ResNet50 [28] is the most widely used CNN architecture for food recognition problems, there are many results available for comparisons. So, the performance of the proposed two-scale CNN is compared with the previous methods [31]–[34] that are based on the ResNet50. ResNet-50 is a convolutional neural network with 50 layers

**TABLE 1. Comparisons of the proposed two-scale CNN with the ResNet50-based methods in terms of Top-1 and Top-5 accuracies.**

| DataSet | Method | Sub-Model | mAP Top1 | mAP (Fused) | |
|---|---|---|---|---|---|
| | | | | Top 1 | Top 5 |
| UECFood 256 | Ciocca, et al. [31] | NA | NA | 71.70 | 91.33 |
| | Fu, et al. [32] | NA | NA | 71.2 | 91.0 |
| | Aguilar, et al. [33] | NA | NA | 65.54 | - |
| | Proposed Two-Scale CNN (Fig 5(b)) | Object-level CNN | 67.48 | 70.36 | 90.71 |
| | | Part-level CNN | 57.70 | | |
| | Proposed Two-Scale CNN (Fig 5(a)) | Object-level CNN | 67.48 | **71.75** | **91.49** |
| | | Part-level CNN | 62.02 | | |
| Food101 | Ciocca, et al. [31] | NA | NA | 82.54 | 95.79 |
| | Fu, et al. [32] | NA | NA | 78.5 | 94.1 |
| | Aguilar, et al. [33] | NA | NA | 82.82 | - |
| | Li, et al. [30] | NA | NA | 82.6 | 95.3 |
| | Proposed Two-Scale CNN (Fig 5(b)) | Object-level CNN | 84.24 | 85.33 | 96.96 |
| | | Part-level CNN | 76.02 | | |
| | Proposed Two-Scale CNN (Fig 5(a)) | Object-level CNN | 84.24 | **86.21** | **97.19** |
| | | Part-level CNN | 78.34 | | |
| VireoFood 172 | Ciocca, et al. [31] | NA | NA | 85.86 | 97.32 |
| | Proposed Two-Scale CNN (Fig 5(b)) | Object-level CNN | 88.01 | 88.87 | 98.27 |
| | | Part-level CNN | 82.38 | | |
| | Proposed Two-Scale CNN (Fig 5(a)) | Object-level CNN | 88.01 | **89.72** | **98.40** |
| | | Part-level CNN | 84.87 | | |

and it has a fixed input size of 224 × 224. The network pretrained on more than a million images from the ImageNet database [35] is available for the transfer-learning. The pre-trained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and animals. In this paper, the pre-trained ResNet50 on the ImageNet is used for fine-tuning the object-level CNN in Fig. 4. The comparative results for the ResNet50-based methods are shown in Table 1. First of all, for the proposed two-scale CNN, the fused results of the Object-level and Part-level CNNs yield substantially better results than the separated ones. This implies that the object-level and part-level CNNs learn distinctive features in their own scales, to some extent, exclusively. Next, the results of the two types of the inference in Fig. 5 are compared. When the early layers of the Object-level CNN are reused for the Part-level CNN (i.e., Fig. 5(b)), the performance loss is only around 1%, while the memory increase for the replaced last layers of the Part-level CNN is only fractional and we can save a lot of memory space. For example, for the ResNet50 trained by the Food101 dataset, the Part-level CNN parameters including all layers as Fig. 5(a) take 86.579MB, but the last layers of them in Fig. 5(b) need only 0.754MB. Similar memory savings can be observed for other datasets. Finally, comparing to all the previous results based on the ResNet50, the proposed two-scale CNN outperforms the existing methods and achieves the state-of-the-art results for all datasets. Interestingly, the performance gains for Food-101 and VireoFood-172 are considerably higher than UEC Food256. A possible cause for this is that the average image sizes of Food-101 and VireoFood-172 are bigger than that of UEC Food256. Specifically, the average numbers of pixels (i.e., $M \times N$) for Food-101, VireoFood-172, and UEC Food256 are 235140, 406050, and 210900, respectively, and the UEC Food256 is the smallest. Note that, as the size of the image increases, we need to delete more lines to have the fixed target image size, say 224 × 224. This in turn demands a sophisticated image resizing method to choose the image lines to-be-deleted more wisely. In this regard, it is expected that the optimal method of the linear programming framework in (4) ∼ (8) works more effectively.

Although fusion is an important step to combine the outputs of the multi-scale CNNs, since it is not the main topic of this paper, no exhaustive trials for all possible methods have been conducted except two simple score-level fusions. Specifically, the normalized output-scores (i.e., class-probabilities) of all multi-scale CNN outputs are combined via a class-wise multiplication or a class-wise addition. Then, the class label that gives a maximum among all fused scores is selected as the classified result. Experiments show that the class-wise multiplication yields slightly better results than the class-wise addition. So, all experimental results shown in the tables were obtained by the class-wise multiplication.

Table 2 shows the comparisons among the state-of-the-art Top-1 and Top-5 accuracies reported in [26], [36]–[39], which adopts other neural networks besides ResNet50. This time, as an updated version of the ResNet50, InceptionResNet-v2 [30] was adopted as the baseline CNN of the proposed multi-scale CNN. Inception-ResNet-v2 is a convolutional neural network with 164 layers and it has a fixed input size of 299 × 299 (i.e., $M' = N' = 299$). The Inception-ResNet-v2 network pretrained on the ImageNet database [35] is available for the transfer-learning and is used as the early-layers of the pre-trained CNN in Fig. 4. Also, to make sure the performance improvement with additional Part-level CNNs, one more Part-level CNN dubbed as Mid-level CNN is added to have a three-scale CNN (see Fig. 8). The Mid-level CNN in Fig. 8 is trained to learn the intermediate features between the object appearance and the part details. To do that the line-based image resizing method of the linear programming framework of (4) ∼ (8) is employed to provide resized training images with the randomly chosen scale parameter $\alpha \in [0.25,\ 0.45]$. Recall that the scale parameter $\alpha$ for the

**TABLE 2. Comparisons of the proposed two-scale and three-scale CNNs with the state-of-the-art results of other networks beyond the ResNet50 for food recognition.**

| DataSet | Method | Model | | mAP Top1 | mAP (Fused) | |
|---|---|---|---|---|---|---|
| | | | | | Top 1 | Top 5 |
| UECFood 256 | Hassannejad, et al. [36] | Inception V3 | | NA | 76.17 | 92.58 |
| | Martinel, et al. [26] | WISeR (w/ BB) 10-crop testing | | NA | **83.15** | **95.45** |
| | | WISeR (w/o BB) 10-crop testing | | NA | 72.71 | **93.78** |
| | Proposed Two-Scale CNN (Fig5(b)) | Inception ResNet V2 | Object-level CNN | 70.33 | 70.48 | 91.56 |
| | | | Part-level CNN | 53.73 | | |
| | Proposed Two-Scale CNN (Fig 5(a)) | Inception ResNet V 2 | Object-level CNN | 70.33 | 73.47 | 92.91 |
| | | | Part-level CNN | 65.15 | | |
| | Proposed Two-Scale CNN (Fig 6) | Inception ResNet V 2 | Object-level CNN | 70.33 | 73.20 | 92.74 |
| | | | Part-level CNN | 68.95 | | |
| | Proposed Three-Scale CNN (Fig 8) | Inception ResNet V2 | Object-level CNN | 70.33 | **74.11** | 93.17 |
| | | | Part-level CNN | 65.15 | | |
| | | | Mid-level CNN | 69.43 | | |
| Food101 | Hassannejad, et al. [36] | Inception V3 | | NA | 88.28 | 96.88 |
| | Martinel, et al. [26] | WISeR 10-crop testing | | NA | **90.27** | **98.7** |
| | Proposed Two-Scale CNN (Fig 5(b)) | Inception Resnet V 2 | Object-level CNN | 87.21 | 87.15 | 97.61 |
| | | | Part-level CNN | 75.84 | | |
| | Proposed Two-Scale CNN (Fig 5(a)) | Inception ResNet V 2 | Object-level CNN | 87.21 | 88.45 | 97.97 |
| | | | Part-level CNN | 81.20 | | |
| | Proposed Two-Scale CNN (Fig 6) | Inception ResNet V 2 | Object-level CNN | 87.21 | 87.62 | 97.70 |
| | | | Part-level CNN | 84.57 | | |
| | Proposed Three-Scale CNN (Fig 8) | Inception ResNet V2 | Object-level CNN | 87.21 | **88.84** | **98.08** |
| | | | Part-level CNN | 81.20 | | |
| | | | Mid-level CNN | 85.87 | | |
| VireoFood 172 | Aguilar, et al. [37] | RUMTL | | NA | 85.19 | - |
| | Metwalli, et al. [38] | DenseFood | | NA | 81.23 | 95.44 |
| | Proposed Two-Scale CNN (Fig 5(b)) | Inception ResNet V2 | Object-level CNN | 89.87 | 89.99 | 98.63 |
| | | | Part-level CNN | 82.85 | | |
| | Proposed Two-Scale CNN (Fig 5(a)) | Inception ResNet V2 | Object-level CNN | 89.87 | 90.86 | 98.76 |
| | | | Part-level CNN | 87.00 | | |
| | Proposed Two-Scale CNN (Fig 6) | Inception ResNet V 2 | Object-level CNN | 89.87 | 90.85 | 98.76 |
| | | | Part-level CNN | 87.52 | | |
| | Proposed Three-Scale CNN (Fig 8) | Inception ResNet V2 | Object-level CNN | 89.87 | **91.34** | **98.87** |
| | | | Part-level CNN | 87.00 | | |
| | | | Mid-level CNN | 89.69 | | |

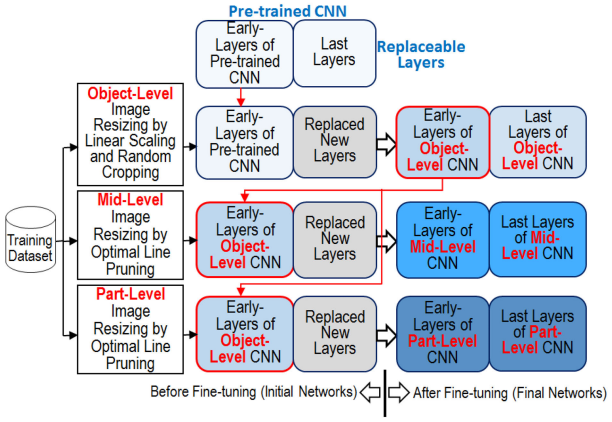


**FIGURE 8. Three-scale CNN with Object-level, Mid-level, and Part-level CNNs.**

Part-level CNN was chosen randomly from $\alpha \in [0.45, 0.65]$. At the inference stage $\alpha$ is fixed as $\alpha = 0.35$ for the Mid-level CNN. For the UEC Food256 dataset, the WISeR (WIde-Slice Residual Network) in [26] held the state-of-the-art result. However, the result of 83.15% in [26] was obtained by making use of the ground-truth (i.e., the bounding box) information provided by the dataset. Since the proposed multi-scale CNN takes no prior information about the dataset except the class labels, it will be fair to compare the results without the bounding boxes, which is 72.71% and is inferior to 73.47% of the two-scale CNN and 74.11% of the three-scale CNN. It is unclear whether 76.17% accuracy of [36] was obtained by making use of the bounding box information or not. For the Food101 dataset, there is no bounding box information available. Now, excluding the WISeR 10-crop testing [26], the result from the three-scale CNN achieved the state-of-the-art for the Food101 dataset. As shown in Table 2, for VireoFood 172 dataset, both implementations of the proposed three-scale CNN and the two-scale CNN broke the state-of-the-art accuracy. It is worth mentioning the comparative results of the three implementations, namely, the two-scale CNN with the line-based resizing (Fig. 5(a)), the two-scale CNN with the linear-scaling and random-cropping (Fig. 6), and the three-scale CNN (Fig. 8). The performance gains of the two-scale CNN in Fig. 5(a) compared to the two-scale CNN in Fig. 6 are rather small. On the other hand, there are noticeable performance improvements of the three-scale CNN (Fig. 8) over the two-scale CNN (Fig. 5(a)). One thing to note is that the result of the WISeR [26] was obtained by aggregating the results of 10-crop images [5], [40]. This is inefficient as it requires the network to re-compute the predictions for ten times. The results of the proposed multi-scale CNN were obtained by running each path of the multi-scale CNN only once. Therefore, we need only two resized images for a given test image and each resized image is applied as an input for each path of the two-scale CNN.

## V. CONCLUSIONS

Multi-scale CNN with an Object-level CNN and multiple Part-level CNNs has been proposed for a fine-grained image classification with no explicitly separable object-parts. The basic approach to solve this domain-specific problem is to resize training images with different scales. Specifically, for the Object-level CNN the training image is resized such that the global appearance of the object is to be included as much as possible. A linear scaling followed by a random cropping is adopted for the object-level image resizing. For the Part-level CNNs, we need an image resizing method with a scale-controllable parameter. So, the existing line-based image resizing method is updated to control the scaling factor by a parameter associated with the number of image-lines in the constraint of the constraint optimization framework. Also, the linear-scaling and random-cropping method is modified such that the scaling factor can be determined by a parameter in a new formula. Experimental results show that the proposed two-scale CNN achieved the state-of-the-art accuracies for the food recognition based on ResNet50 for all datasets of Food-101, UEC Food256, and Vireo Food-172. Also, the two-scale and the three-scale CNN broke the record of the food recognition for Vireo Food-172 dataset with the InceptionResNet v2. The proposed multi-scale CNN can be applied to other fine-grained image datasets to boost the classification performance. It is especially suitable for the datasets with no separable object-parts and no object bounding boxes.

## REFERENCES

[1] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 842–850.

[2] S. Xie, T. Yang, X. Wang, and Y. Lin, "Hyper-class augmented and regularized deep learning for fine-grained image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2645–2654.

[3] P. Welinder and P. Perona, "Online crowdsourcing: Rating annotators and obtaining cost-effective labels," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (Workshops)*, Jun. 2010, pp. 25–32.

[4] Y. Peng, X. He, and J. Zhao, "Object-part attention model for fine-grained image classification," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1487–1500, Mar. 2018.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[6] C. S. Won, "Constrained optimization for image reshaping with soft conditions," *IEEE Access*, vol. 6, pp. 54823–54833, 2018.

[7] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 dataset," California Inst. Technol., CA, USA, Tech. Rep. CNS-TR-2011-001, 2011.

[8] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li, "Novel dataset for fine-grained image categorization: Stanford dogs," in *Proc. CVPR Workshop Fine-Grained Vis. Categorization (FGVC)*, 2011, vol. 2, no. 1, pp. 1–2.

[9] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D object representations for fine-grained categorization," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Sydney, NSW, Australia, Dec. 2013.

[10] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101–mining discriminative components with random forests," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, in Lecture Notes in Computer Science, vol. 8694. Cham, Switzerland: Springer, 2014, p. 446–461, doi: 10.1007/978-3-319-10599-4_29.

[11] Y. Kawano and K. Yanai, "Automatic expansion of a food image dataset leveraging existing categories with domain adaptation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, Sep. 2014, pp. 3–17.

[12] J. Chen and C.-W. Ngo, "Deep-based ingredient recognition for cooking recipe retrieval," in *Proc. 24th ACM Int. Conf. Multimedia*, 2016, pp. 32–41.

[13] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear CNN models for fine-grained visual recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1449–1457.

[14] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based R-CNNs for fine-grained category detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2014, pp. 834–849.

[15] H. Yao, S. Zhang, C. Yan, Y. Zhang, J. Li, and Q. Tian, "AutoBD: Automated bi-level description for scalable fine-grained visual categorization," *IEEE Trans. Image Process.*, vol. 27, no. 1, pp. 10–23, Jan. 2018.

[16] M. Simon and E. Rodner, "Neural activation constellations: Unsupervised part model discovery with convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1143–1151.

[17] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Apr. 2013.

[18] J. Han, X. Yao, G. Cheng, X. Feng, and D. Xu, "P-CNN: Part-based convolutional neural networks for fine-grained visual categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Aug. 6, 2019, doi: 10.1109/TPAMI.2019.2933510.

[19] P. Sermanet, A. Frome, and E. Real, "Attention for fine-grained categorization," 2014, *arXiv:1412.7054*. [Online]. Available: http://arxiv.org/abs/1412.7054

[20] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," 2014, *arXiv:1412.7755*. [Online]. Available: http://arxiv.org/abs/1412.7755

[21] P. Rodriguez, D. Velazquez, G. Cucurull, J. M. Gonfaus, F. X. Roca, and J. Gonzalez, "Pay attention to the activations: A modular attention mechanism for fine-grained image recognition," *IEEE Trans. Multimedia*, vol. 22, no. 2, pp. 502–514, Feb. 2020.

[22] Y. Chen, Y. Bai, W. Zhang, and T. Mei, "Destruction and construction learning for fine-grained image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5157–5166.

[23] H. Zheng, J. Fu, Z. J. Zha, and J. Luo, "Learning deep bilinear transformation for fine-grained image representation," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2019, pp. 4279–4288.

[24] S. Jiang, W. Min, L. Liu, and Z. Luo, "Multi-scale multi-view deep feature aggregation for food recognition," *IEEE Trans. Image Process.*, vol. 29, pp. 265–276, 2019.

[25] A. Singla, L. Yuan, and T. Ebrahimi, "Food/Non-food image classification and food categorization using pre-trained GoogLeNet model," in *Proc. 2nd Int. Workshop Multimedia Assist. Dietary Manage. (MADiMa)*, 2016, pp. 3–11.

[26] N. Martinel, G. L. Foresti, and C. Micheloni, "Wide-slice residual networks for food recognition," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 567–576.

[27] K. Kamnitsas, C. Ledig, V. F. J. Newcombe, J. P. Simpson, A. D. Kane, D. K. Menon, D. Rueckert, and B. Glocker, "Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation," *Med. Image Anal.*, vol. 36, pp. 61–78, Feb. 2017.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[29] J. Yu, M. Tan, H. Zhang, D. Tao, and Y. Rui, "Hierarchical deep click feature prediction for fine-grained image recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jul. 30, 2019, doi: 10.1109/TPAMI.2019.2932058.

[30] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 1–12.

[31] G. Ciocca, P. Napoletano, and R. Schettini, "CNN-based features for retrieval and classification of food images," *Comput. Vis. Image Understand.*, vols. 176–177, pp. 70–77, Nov. 2018.

[32] Z. Fu, D. Chen, and H. Li, "Chinfood1000: A large benchmark dataset for chinese food recognition," in *Proc. Int. Conf. Intell. Comput.* Cham, Switzerland: Springer, 2017, pp. 273–281.
[33] E. Aguilar and P. Radeva, "Class-conditional data augmentation applied to image classification," in *Proc. Int. Conf. Comput. Anal. Images Patterns*. Cham, Switzerland: Springer, 2019, pp. 182–192.
[34] Z. Li, X. Zhu, L. Wang, and P. Guo, "Image classification using convolutional neural networks and kernel extreme learning machines," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 3009–3013.
[35] *ImageNet*. Accessed: Jun. 5, 2020. [Online]. Available: http://www.image-net.org/
[36] H. Hassannejad, G. Matrella, P. Ciampolini, I. De Munari, M. Mordonini, and S. Cagnoni, "Food image recognition using very deep convolutional networks," in *Proc. 2nd Int. Workshop Multimedia Assist. Dietary Manage. (MADiMa)*, 2016, pp. 41–49.
[37] E. Aguilar, M. Bolaños, and P. Radeva, "Regularized uncertainty-based multi-task learning model for food analysis," *J. Vis. Commun. Image Represent.*, vol. 60, pp. 360–370, Apr. 2019.
[38] A.-S. Metwalli, W. Shen, and C. Q. Wu, "Food image recognition based on densely connected convolutional neural networks," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Feb. 2020, pp. 027–032.
[39] W. Min, S. Jiang, L. Liu, Y. Rui, and R. Jain, "A survey on food computing," *ACM Comput. Surv.*, vol. 52, no. 5, pp. 1–36, Sep. 2019.
[40] Y. Liu, B. Yin, J. Yu, and Z. Wang, "Image classification based on convolutional neural networks with cross-level strategy," *Multimedia Tools Appl.*, vol. 76, no. 8, pp. 11065–11079, Apr. 2017.

**CHEE SUN WON** (Member, IEEE) received the B.S. degree in electronics engineering from Korea University, Seoul, South Korea, in 1982, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Massachusetts, Amherst, MA, USA, in 1986 and 1990, respectively.

From 1989 to 1992, he was a Senior Engineer with Goldstar Company, Ltd. (LG Electronics), Seoul. In 1992, he joined Dongguk University, Seoul, where he is currently a Professor with the Division of Electrical and Electronics Engineering. He has been a Visiting Professor with Stanford University, Stanford, CA, USA, and McMaster University, Hamilton, ON, Canada. His research interests include computer vision, deep neural networks, video signal processing, and image feature detection and matching.