4-2022

SibNet: Food instance counting and segmentation

Huu-Thanh. NGUYEN

Chong-wah NGO
Singapore Management University, cwngo@smu.edu.sg

Wing-Kwong CHAN

Citation
NGUYEN, Huu-Thanh.; NGO, Chong-wah; and CHAN, Wing-Kwong. SibNet: Food instance counting and segmentation. (2022). *Pattern Recognition*. 124, 1-11.
Available at: https://ink.library.smu.edu.sg/sis_research/6952

SibNet: Food instance counting and segmentation

Huu-Thanh Nguyen [a,*], Chong-Wah Ngo [b], Wing-Kwong Chan [a]

a Department of Computer Science, City University of Hong Kong, Hong Kong, 999077, China
b School of Computing and Information Systems, Singapore Management University, 178902, Singapore
Corresponding author: tnguyenhu1-c@my.city.edu.hk (H.T. Nguyen).

Published in Pattern Recognition, (2022 April), 124, 108470, 1-11. DOI: 10.1016/j.patcog.2021.108470

Abstract: Food computing has recently attracted considerable research attention due to its significance for health risk analysis. In the literature, the majority of research efforts are dedicated to food recognition. Relatively few works are conducted for food counting and segmentation, which are essential for portion size estimation. This paper presents a deep neural network, named SibNet, for simultaneous counting and extraction of food instances from an image. The problem is challenging due to varying size and shape of food as well as arbitrary viewing angle of camera, not to mention that food instances often occlude each other. SibNet is novel for proposal of learning seed map to minimize the overlap between instances. The map facilitates counting and can be completed as an instance segmentation map that depicts the arbitrary shape and size of individual instance under occlusion. To this end, a novel sibling relation sub-network is proposed for pixel connectivity analysis. Along with this paper, three new datasets covering Western, Chinese and Japanese food are also constructed for performance evaluation. The three datasets and SibNet source code are publicly available.

Keywords: Food counting, Food instance segmentation

1. Introduction

Understanding diet patterns is helpful for the analysis of long-term health trends. In clinical practices, food intake is usually logged manually by 24-hour recall or food frequency questionnaire. The paper-based logging process is cumbersome and time-consuming. In addition to specifying food type, a user is also requested to quantify the serving sizes of food taken. As reported in the clinical study [1], under and over estimation of consumption is common in food logging. With the rapid progress in deep learning, there have been various research devoted to automating food logging through image processing. These efforts include food segmentation [2], detection [3], ingredient recognition [4], food recognition [5], portion [6], weight [7], volume [8] or calories [9], [10] estimation, and recipe retrieval [11]. Some of these techniques have been deployed to mobile applications [12].[1]

This paper addresses the problem of counting and extracting food items on a plate. Counting is essential because the number of servings is defined upon countable units, such as “piece and “slice. Counting is essential because the number of servings is defined upon countable units, such as “piece and “slice. Extraction of food items, on the other hand, facilitates the estimation of serving size. In food industry, the correlation between area and weight of food is measurable [7]. Assuming that camera is calibrated, for example by using a fiducial marker, food area can be estimated by simply counting the number of pixels. Therefore, serving size, or more specifically the weight of a food item, is possible to be calculated by segmenting the image region corresponding to food. Most of nutrition databases, usually being referred to as food composition table (FCT) (e.g., CFC [13], USDA [14]), specify the nutrients and calories of food. By knowing the number of servings and their sizes through food counting and segmentation, as well as the names of food through dish recognition, the food consumption in terms of nutrition content and calories can be quantified by mapping this information to FCT.

The challenge of food counting and segmentation comes from diverse visual appearance, severe occlusion, perspective distortion and obscure food boundary. Fig. 1 shows four examples to illustrate these challenges. In Fig. 1a, each sushi is topped with different ingredients, resulting in different appearances. In Fig. 1b, some cookies are partially visible due to vertical stacking. In Fig. 1c, the egg tarts are not only occluded but also changed in size and shape due to camera perspective distortion. In Fig. 1d, the contour of each pancake slice is not clearly depicted, especially for slices that are farther away from camera. The existing state-of-the-art object detection techniques [15], [16], which bound objects in rectangular boxes, are incapable of dealing with food items in arbitrary shape

[1] https://github.com/alannguyencs/sibnet.

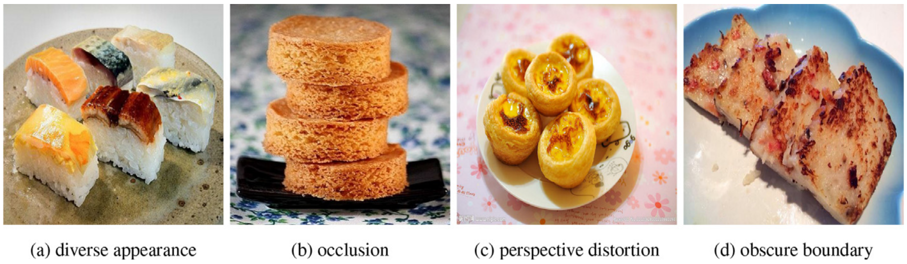


Fig. 1. Challenges in food instance counting and segmentation.

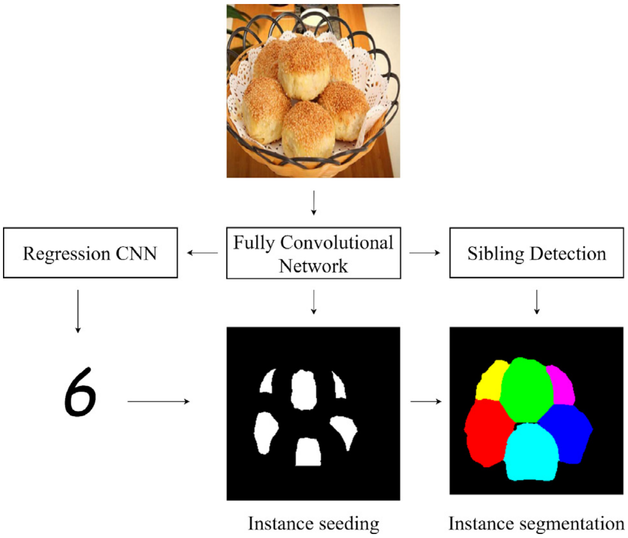


Fig. 2. The architecture of SibNet.

(e.g., Fig. 1d) or largely overlapped (e.g., Fig. 1b). Counting based on the number of detected bounding boxes in an image is likely to perform unsatisfactorily. On the other hand, semantic segmentation [17], which labels pixels based on semantic categories, requires post-processing to extract instances of food with the same category. Occlusion, such as those depicted in Fig. 1(b)-(d), which are common in food presentation, demands sophisticated image post-processing for instance segmentation.

This paper proposes a novel network architecture, named Sibling Network (SibNet), for counting and segmentation of food instances on a plate. Fig. 2 provides an overview of SibNet, which is a multi-task neural network. Given a food image, fully convolutional network (FCN) [17] first labels the pixels into food and non-food regions. Different from the conventional FCN, nevertheless, the generated segmentation map occupies partial regions of food instances only, aiming to reduce the overlapping between instances. This map inherently provides instance seeds as priori knowledge for regression-based convolutional neural network (CNN) to perform counting. Such architecture, by simultaneous instance seeding and counting, is effective in alleviating the counting problems depicted in Fig. 1. Another novelty of SibNet is the proposal of a sibling detection sub-network, which performs pixel connectivity analysis to complete a full instance segmentation map. The sub-network detects sibling relation between neighbouring pixels and assigns labels to pixels based on the number of instances found. SibNet systematically refines found instance identification from high level seed counting to the detailed separation between instances using the novel sibling relation.

The main contribution of this paper is the proposal of SibNet that addresses the unique challenges of counting and segmentation in food domain. To the best of our knowledge, there is no work yet to systematically explore this problem for food items of arbitrary shape, often with severe occlusion and obscure boundary. The existing works are mostly based on CRF [9], FCN [17] or saliency analysis [10], which can only deal with food items that are well separated. The rest of this paper are organized as follows. Section 2 discusses related works in object counting and food segmentation. Section 3 presents the baseline frameworks for counting based on single and multi-task learnings, respectively. Section 4 details the proposed SibNet, particularly the networks for generation of seed map and detection of pixel sibling relation. Sections Sections 5 and 6 describe the datasets and empirical findings to justify the merit of SibNet over other existing works. Finally, Section 7 concludes this paper.

2. Related works

Instance counting is prevalent for different applications, including counting crowd in surveillance videos [18,19], cells [20] and bacterial colonies [21] in medical images, maize tassels [22] and fruits-on-tree [23] for agriculture, and animal counting [24] for natural reserve. Counting everyday objects is also recently explored in [25]. These applications differ mainly in terms of camera viewpoint and visual property. For instance, the viewing angle of a camera is assumed to be perspective in crowd counting [18,19], bird view in cell counting [20] and front view in fruit counting [23]. Cells and fruits are treated as blobs of equal size, while crowds are treated as small dots of elliptical shape. In contrast to these applications, food images can be captured from any arbitrary viewing angles. For everyday objects [25], the shapes and visual presentations of food instances vary across categories and cannot be predefined. As the number of counts is much smaller than crowds, for example, the perspective distortion could exaggerate the appearance change in size and shape.

Glance-based counting, which predicts counts without the knowledge of instance locations, is widely adopted in different applications [20,21,23,25]. A solution can be as simple as a CNN with input as an image and output as a continuous number [20,23,25] or a counting category [21]. Nevertheless, when the count is a large number, precise prediction could be challenging. In surveillance applications, this problem is addressed by the prediction of density [18,22] and count maps [19]. Specifically, each pixel indicates a local density of population. Summing up all the pixel values in a map is equivalent to population counting. As density map has predefined assumption on instance size and camera angle, directly applying for counting food is expected to yield suboptimal performance. A more generalized approach is proposed in [25] for both the recognition and counting of everyday objects. Its key idea is to subitize small counts locally at image regions before integrating them into final counts. Two subitizing techniques, Aso-Sub and Seq-Sub, are proposed. The former estimates local count using regression-based CNN. The latter further considers the notion of spatial context by using a recurrent network to propagate the local counts to neighbouring image regions. Nevertheless, as the number of food items on a plate is usually small, using subitizing techniques may unnecessarily complicate the counting procedure.

In contrast to glance-based counting, detection-based approaches perform counting-by-detection. LC-FCN [24], implemented on top of fully convolutional network (FCN) [17], is proposed to count by producing a segmentation blob for each instance. To expedite annotations, each instance is marked with a dot for training. While performing excellently for counting a variety of objects, LC-FCN is not applicable for instance segmentation due to the absence of instance shape and size in its blob representation. In addition to LC-FCN, object detection techniques such as [15,16,26–28] have also been directly applied for counting. These techniques predict bounding boxes or octagon to localize instances by visual features such as objectness (Mask-RCNN [15], Yolact [26]), 4D object vectors (FCOS [16]), corners (CornerNet [27]) and extreme points (ExtremeNet [28]). Due to the assumption of predefined shape representation, nevertheless, these techniques are generally limited in detecting objects of irregular shapes and severely occluded, which are common in food images. To address this problem, Mask-RCNN [15] and Yolact [26] are equipped with semantic segmentation [17] to extract instances from boxes, at the expense of annotating instances masks for model training. The problem is addressed by Park et al. [29], which generates synthetic images and their instance masks using a computer graphics software for training Mask-RCNN. The work is applied for food instance segmentation to extract dishes on a tray, which are usually well separated.

SibNet is closely relevant to instance segmentation [30–36], which first labels pixels based on visual properties and then spatially cluster pixels into instances. Due to the bottom-up processing without shape assumption, these approaches can flexibly handle arbitrary shapes of food instances. These approaches are mostly based on FCN-like backbone and differ in ways in which visual features are exploited and how pixels are clustered. In [30], a watershed method is proposed by predicting distance transform energy to quantify the distance between a pixel and its instance boundary. The energy values are grouped into 16 levels from an instance centroid towards its border to facilitate pixel clustering. Terrace [36] models the shape of food as a terrace map with different contour levels of height. The height corresponds to object attention while the evolution of contours signifies the difficulty of segmentation. A multi-task learning neural network is proposed, showing a performance superior to the watershed method. SECB [31] performs spatial embedding such that a pixel value predicts the position of an instance centroid that the pixel belongs to. Each pixel is also predicted with a margin value indicating the instance size for clustering. Similar in spirit, TextMountain [32] also predicts a centre-boundary probability map and a direction-to-centroid map. For each instance, four directional vectors are used to generate the two ground-truth maps. The generalization of this method to handle arbitrary shapes of food instances is unclear. Instead of grouping pixels as watershed [30], PSENet [33] predicts the instance centroids as seeds and progressively expands the size of seeds towards the instance boundaries by breadth-first search. PAN [35] targets for real-time segmentation and replaces the network for progressive expansion of seeds in PSENet with a lightweight backbone. Pixel affinity learning is also explored in [34], by using a neural network to predict the 8-directional pixel connectivity as feature maps, which will be further fused for pixel clustering. SibNet shares some properties of these approaches, such as using seeding as PSENet [33] and learning pixel affinity as PixelLink [34]. Different from these algorithms, nevertheless, SibNet leverages counting to enhance the robustness of instance segmentation. Furthermore, SibNet is computationally more efficient than PixelLink and PSENet due to different learning procedures as will be elaborated in the latter section.

3. Food counting and segmentation

We begin by introducing a standard regression-based counting network. The network is then extended for multi-task learning such that food regions are segmented to constrain counting.

3.1. Single-task counting

Food items are generally large in size due to camera capturing angle. The shapes differ considerably not only due to variations across food categories, but also perspective distortion because of camera-to-food distance. A baseline method is by estimating counts based on the feature maps generated by CNN. Specifically, the output layer of CNN is replaced with one neuron for regression-based counting. Denote N as the number of training examples. The loss function of regression-based counting (RC) minimizes the mean absolute error between the actual (r_n) and predicted ($\hat{r}_n$) counts over training examples, as following:

$$L_{RC} = \frac{1}{N}\sum_{n=1}^{N}\left|r_n - \hat{r}_n\right| \tag{1}$$

3.2. Multi-task counting and segmentation

Single-task counting could be sensitive to background clutter. A more robust way is by segregating non-food regions from counting. We formulate this problem as a multi-task learning problem, with one path for food counting and the other for two-class semantic segmentation. The semantic segmentation targets generating a map that encloses food regions while masking out non-food regions.

Fully convolutional network (FCN) [17] is employed as the backbone network. The counting pathway is branched out from the last convolution layer of the FCN. Due to the sharing of the same layer with a large receptive field as the segmentation pathway, counting is expected to be beneficial by attending to only food regions. The entire network is end-to-end trained with both pathways being updated simultaneously. The loss function consists of two parts for counting and segmentation respectively, as follows:

$$L_{MUL} = \lambda_{RC}L_{RC} + \lambda_{SEG}L_{SEG} \tag{2}$$

where λ_{RC} and λ_{SEG} are trade-off parameters. The function L_{SEG} is based on the cross-entropy loss enumerated over all the pixels of a segmentation map. As pixels belonging to food regions are generally less than that of non-food regions, the loss of each pixel is weighted according to the proportion of food and non-food regions. Specifically, denote N and M as the number of training images and the number of pixels per image respectively, the proportion of pixels labelled as food is defined as follows:

$$\overline{W} = \frac{\sum_n^N \sum_m^M t_{n,m}}{NM} \tag{3}$$

where $t_{n,m} = \{0, 1\}$ is the ground-truth label of the mth pixel in the nth image, with the value of 1 indicates a pixel belonging to food category, and the value of 0 otherwise. Further denote $W_0 = \overline{W}$ and $W_1 = 1 - \overline{W}$ as the weights for food and non-food pixels respectively, the segmentation loss is defined as:

$$L_{SEG} = \frac{-\sum_n^N \sum_m^M W_s \log(\hat{t}_{n,m,s})}{NM} \tag{4}$$

where $\hat{t}_{n,m,s}$, produced by softmax activation function, represents the probability score of a pixel with ground-truth label as $s = \{0, 1\}$.

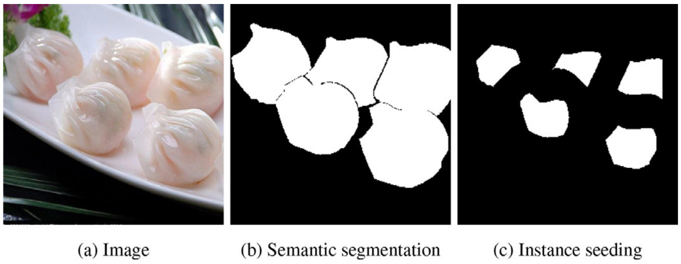


Fig. 3. The semantic segmentation (b) and seed map (c) of a food image (a). Food items are well-separated in (c).

4. SibNet

By the multi-task approach, food regions may connect to each other, then hinder the counting and extraction of food instances. Therefore, a new network, named Sibling Network (SibNet) as shown in Fig. 2, is proposed. SibNet has an additional path to generate a multi-instance map, where each pixel is labelled with an identifier indicating the instance to which it belongs. The task is similar to panoptic segmentation [37], where the objective is to jointly label pixels (i.e., semantic segmentation) and extract object instances (i.e., instance detection).

4.1. Instance seeding

Despite effective in alleviating the influence of background noise, segmentation map does not help in separating food items that occlude each other. Seeding is proposed to locate the instance seeds by shrinking the mask of each food item towards its centre of mass. Fig. 3 shows an example where food items are well-separated and readily for counting when their masks are reduced by half. In this way, segmentation and counting take advantage of each other by simultaneous counting and generation of a reduced map. To this end, SibNet counts by learning to generate a seed map that well separates food items with partially visible areas. With this learning strategy, ideally, the network pays more attention to the centre of a food item and grows the mask to a size that facilitates counting. During learning, we fix the ground-truth size of a food item as 50% of its original size since this ratio basically guarantees all food items are separable. Multi-task architecture as described in Section 3.2 is employed by using the seed maps of positive examples for training.

The idea of instance seeding originates from Adams and Bischof [38]. Recently, PSENet [33] formulates the idea with neural networks to generate multiple scales of an instance seed for text detection. Progressive expansion of a seed from the lowest scale towards its entire instance size is required. The procedure dramatically increases the network design complexity. Different from [33], our proposed instance seeding is more computationally efficient by not considering multi-scale processing. Instead, the network for instance seeding is trained end-to-end simultaneously with instance counting for robustness consideration, rather than as a singleton network as in PSENet [33].

4.2. Sibling relation detection

SibNet generates a multi-instance map by analysis of sibling relation between pixels. For counting and segmentation, the task of sibling detection is branched out from the last convolutional layer of the FCN and followed by a deconvolution module. The design is the same as FCN [17] with deconvolution of 3 layers and skipped connections. The output is a sibling map of resolution 256 × 256 with C channels. Given a pixel pair p_i and p_j in the map, SibNet considers the sibling relationship, $S_{i,j}$, between them. Specially, $S_{i,j} = 1$ if both pixels belong to the same instance and $S_{i,j} = 0$ otherwise. As the number of pixel pairs can be huge, the sibling relations are recursively determined by considering the 8-neighbourhood configuration as shown in Fig. 4. Denote p_{i_b} as a neighbour of pixel p_i based on this configuration. The relation between p_i and any pixel p_j is recursively defined as:

$$S_{i,j} = \begin{cases} 1 & \text{if } (S_{i,i_b} = 1 \text{ and } S_{i_b,j} = 1) \text{ for any } b \in [1,8] \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

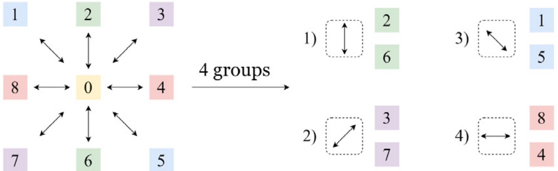


Fig. 4. The 8-neighbourhood configuration of a centre pixel (marked with 0). Four different types of neighbours are defined based on pairwise pixel direction.

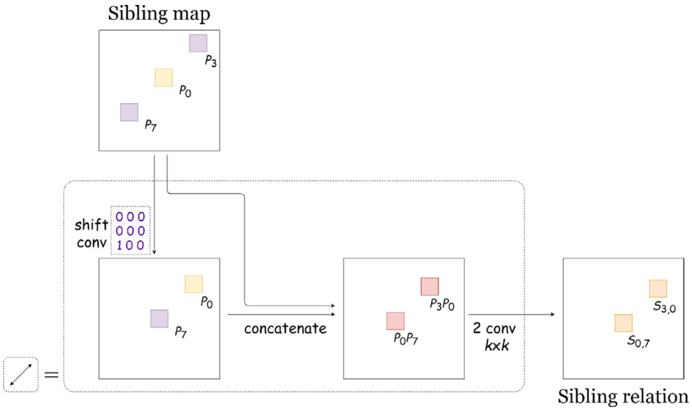


Fig. 5. Illustration of shifting (top right direction) and concatenation of feature maps to detect the sibling relations between pixel pairs (P_0, P_7) and (P_3, P_0).

SibNet models the sibling relation using convolutional-like network, producing a matrix where each element indicates the sibling relation, i.e., S_{i,i_b}, between two neighbouring pixels. Referring to Fig. 4, four different types of neighbours are defined based on the spatial direction between pixels p_i and p_{i_b}. The convolution is performed separately on each type of neighbour. For simplicity, we assume that each feature map only has one channel. Initially, an input feature map is shifted by a pixel distance to align p_i with p_{i_b} based on their type. For each type, the original feature map is stacked with its shifted version to form a 2-channel feature map. Referring to Fig. 5, after shifting to top-right, all neighbouring pixel pairs along this direction (e.g., p_0 and p_7, p_3 and p_0) share the same spatial location in their respective channels. With this alignment, a 3 × 3 convolution kernel is learned to detect the sibling relation $S_{0,7}$. The process is repeated for four different types of neighbours, where the sibling relations of a pixel with its 8 neighbours are altogether classified.

4.3. Instance extraction

Architecture. Fig. 6 shows the architecture design of SibNet as a multi-task learning network. Note that there are five branches rooted on the sibling map. The first four detect sibling relations in different spatial directions while the last branch performs semantic segmentation to generate a mask that labels each pixel as either food or non-food. The sibling relation between two pixels is represented by two scores produced after the softmax layer, corresponding to the probabilities of $S_{i,j} = 1$ and $S_{i,j} = 0$ respectively.

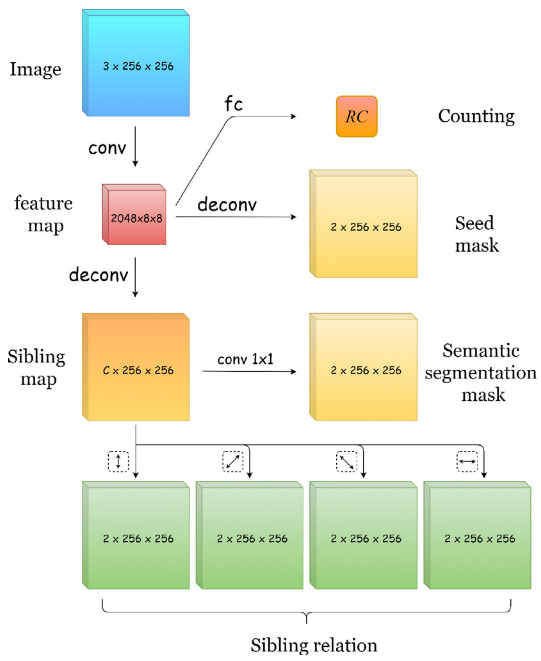


Fig. 6. SibNet architecture with multiple pathways for counting, seeding and sibling detection. The pathway for sibling detection consists of four branches for sibling relation (the arrow defines the direction of conv shifting operator to the top of deconv layer).

To model the difference between the predicted and ground-truth sibling relations, it is sufficient to use the weighted cross-entropy loss in Eq. (4) as the loss function. Nevertheless, as the number of sibling and non-sibling pixels are highly imbalanced, pixel-wise comparison like cross-entropy loss tends to produce a thick layer for non-sibling detection, which is a phenomenon also observed in [39]. In addition to cross-entropy loss, we also use Dice function to model the set-level (or image-level) for non-sibling loss as [39]. Dice loss models the difference between two sets of predicted probability $\hat{U}$ and ground-truth label U for non-sibling pixels (i.e., boundary pixels) as:

$$L_D = \frac{\sum_m (U_m^2 + \hat{U}_m^2)}{2\sum_m U_m \hat{U}_m} \tag{6}$$

where the subscript m refers to index of pixel which is declared as a non-sibling pixel either in ground-truth label or by SibNet. Similar to Deng et al. [39], the Dice loss is combined with cross-entropy loss L_C as:

$$L_{SD} = \alpha L_C + \beta L_D \tag{7}$$

where the trade-off parameters are empirically set as $\alpha = 0.01$ and $\beta = 1.0$ following the suggestion of [39].

To this end, SibNet defines four different types of loss functions as follows:

$$\begin{aligned} L_{SIB} &= \lambda_{RC} L_{RC} + \lambda_{SEED} L_{SEED} \\ &\quad + \lambda_{SEG} L_{SEG} + \tfrac{\lambda_{SD}}{4} \textstyle\sum_i^4 L_{SD_i} \end{aligned} \tag{8}$$

where L_{SD_i} stands for the sibling loss in one spatial direction, and L_{SEED} quantifies the loss in generating a seed map. The terms L_{SEG} and L_{SEED} are based on the weighted cross-entropy loss computed by Eq. (4), while L_{RC} is based on the mean absolute counting error computed by Eq. (1). Each function is associated with a tradeoff parameter. In the experiment, we set $\lambda_{RC} = \lambda_{SEED} = \lambda_{SEG} = \lambda_{SD} = 1.0$ for simplicity. Note that SibNet is different from other panoptic segmentation algorithms such as [40] that measures Kullback Leibler (KL) divergence on all pairs of pixels to generate multi-instance map. SibNet evaluates sibling relation of a pixel with its 8 neighbours, resulting in lower memory consumption and faster training time.

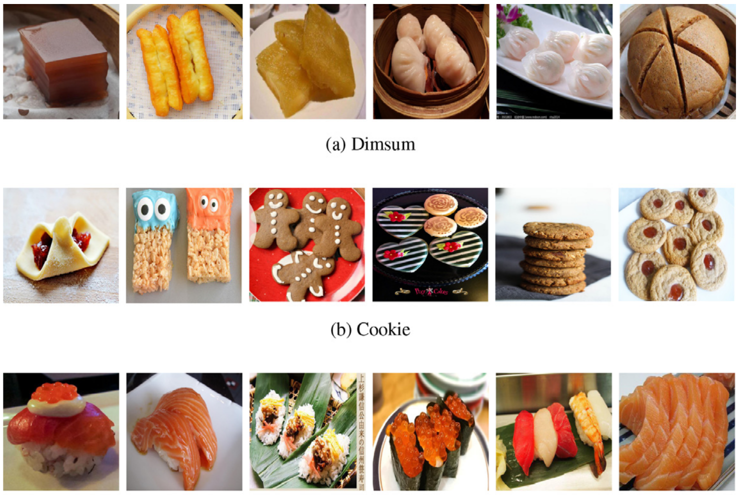
Algorithm. Based on the architecture, the multiple pathways in SibNet are jointly exploited for the extraction of food instances. Specifically, the sibling relations give clue for connected component analysis, where each component refers to a food item. With the aid of semantic segmentation mask, non-food pixels are excluded from analysis. Our connected component analysis starts from the seeds, which are referred to as the centre-of-mass (CoM) of each food instance, provided by the seed map. Note that the number of seeds is not necessarily equal to the count predicted by the counting pathway. In general, the seed map is susceptible to noises, forming small clusters of pixels as candidate seeds. In the algorithm, the predicted count is leveraged to suppress those small clusters from being considered as seeds. In particular, the potential food instances in a seed map are sorted in descending of their sizes. The number of instances being considered as seed candidates is set not larger than the predicted count by the counting pathway, and the seeds are selected based on the sorted order. To this end, we have identified the set of seeds $\mathbf{\Phi} = \{\Phi_i\}$ for the desired food instances.

The basic idea of instance extraction is by propagating the labels of pixels to their neighbours as in Eq. (5). The number of labels directly corresponds to the food counts. As summarized in Algorithm 1, the label propagation takes the set $\mathbf{\Phi}$, sibling relation S and food segmentation map F as inputs, and returns a multi-instance map L. The algorithm is iterative and starts by assigning labels i to the CoMs of items $\{\Phi_i\}$ in the multi-instance map. These seeding pixels are then pushed into a queue Q which works in first-come-first-serve basis. Next, every pixel in Q is retrieved iteratively for label propagation. In the first iteration, the pixel p at the head of the queue propagates its label to the neighbours p_b who are classified as siblings $S_{p,p_b} = 1$. This process is repeated, specifically, pixels having labels take turns to propagate the labels to their 8-neighbour siblings in each iteration. Note that, during the entire process, pixels masked out by the semantic segmentation mask, $F[p_b] \neq 1$, will not receive labels. On the other hand, pixels that reside in the map but receive no label will be re-classified as non-food pixels.

Algorithm 1 Instance label propagation via Sibling relation.

```
Require: seeds: $\mathbf{\Phi}$, sibling relation: $S$, semantic segmentation: $F$
Ensure: Multi-instance map $L$
function InstanceLabelling($\mathbf{\Phi}$, $S$, $F$)
    $L \leftarrow \emptyset$; $Q \leftarrow \emptyset$
    for $\Phi_i \in \mathbf{\Phi}$ do        ▷ Label and push seeding pixels to queue
        for $p \in \Phi_i$ do
            $L[p] = i$
            Enqueue($Q$, $p$)
        end for
    end for
    while $Q \neq \emptyset$ do
        $p \leftarrow$ Dequeue($Q$)
        for $b \in [1, 8]$ do        ▷ Propagate label to sibling neighbours
            if $p_b \notin L$ and $S_{p,p_b} = 1$ and $F[p_b] = 1$ then
                $L[p_b] = L[p]$
                Enqueue($Q$, $p_b$)
            end if
        end for
    end while
    return $L$
end function
```

Deep learning based connectivity analysis has been investigated in PixelLink [34] and PSENet [33] for text detection. Like SibNet,



(a) Dimsum

(b) Cookie

(c) Sushi

Fig. 7. Sample examples from three datasets.

Table 1
Statistics on the number of food categories and images in three food datasets: Cookie, Dimsum, Sushi.

	Dimsum	Cookie	Sushi
Food Category	27	100	11
Counting Category	6	9	6
Images	3760	5920	2877
Training	2700	4050	2700
Testing	768	1152	768
Validation	408	612	408

PixelLink [34] models sibling relations using CNN. Nevertheless, as instance seeding is not considered, the connectivity analysis in PixelLink starts from the instance border and the procedure could be error-prone, thereby a post filtering of segmentation being required to suppress noises. PSENet [33], on the other hand, considers seeding but requires progressive analysis of pixel connectivity for multi-scale expansion of an instance mask. The process is computationally expensive. SibNet, inheriting the merits of PixelLink and PSENet, is lightweight for efficient single-scale processing and robust by leveraging the counting result and seeds for segmentation.

5. Experimental setup

5.1. Data collection

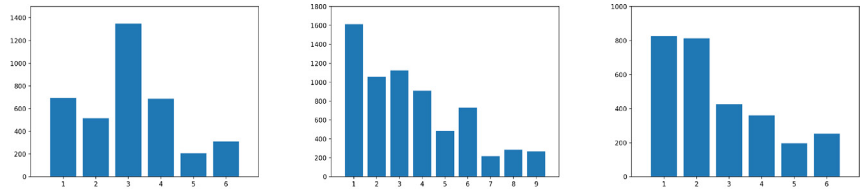
As there is no public dataset available for food counting, we collect three datasets covering different kinds of cookie, dimsum and sushi for experiments. These datasets, respectively, represent Western, Chinese and Japanese food, which are popular worldwide. Table 1 summarizes the numbers of images and food categories, and the proportions of images for training, validation and testing for each dataset. Fig. 7 further shows sample images of different datasets, highlighting the difficulties of counting and segmentation due to challenges such as size variation, occlusion and food stacking. Cookie dataset has food items in a variety of rigid shapes ranging from star, heart, and animal contour. The food items in Dimsum dataset are not as rich as cookies in texture and shape. Counting becomes a challenge, in particular, when items occlude or connect with each other. The items in Sushi dataset are usually rice wrapped in different shapes and topped with ingredients rich in colour and texture. The shape of item is not rigid as cookie but deformable and will exhibit larger appearance change under different camera perspectives. The samples in three datasets represent various challenges, for example, counting of food with various shapes (Cookie dataset) and varying visual appearances under different camera perspectives (Sushi), and segmenting occluded food instances lacking texture pattern (Dimsum).

The lists of food categories are compiled based on their popularity. These names were issued as keywords to commercial search engines including Google and Baidu. We used different languages as keywords for different kinds of food, Chinese for dimsum and English for cookie and sushi. The crawler, icrawler[2], was employed to download 500 images per food category. The collected data was cleaned by removing images of resolution lower than 300 × 300. In addition, we manually screened every image under a food category to get rid of false positives, cartoon images and advertisements. Finally, the number of food items in a picture was manually annotated.

Fig. 8 shows the distributions of images across different counting categories. The distributions are unbalanced. For example, Cookie dataset is peaked at images with one item and Dimsum dataset is peaked at images with three items. As machine learning with unbalanced data distribution remains an open problem and will create bias in estimation, we used 646 images per counting category in the experiment. The sample images for a counting category with more than 646 images were randomly drawn. On the other hand, new images were augmented by flipping and rotation and then added to the categories with less than 646 images. For fair comparison, we made sure that an image and its augmented versions were kept in one folder only, either training, validation or test set.

In addition to manual labelling of counting categories, the food instance masks were semi-automatically generated. We created three kinds of masks: pixel-wise instance mask, polygon mask and bounding box. For training data, a polygon was created by first having an annotator to mark the corners of a food instance. The corners were then connected with straight lines as a polygon mask. The seed mask of an instance was generated by shrinking the boundary of a polygon mask towards its centre of mass. Shrinking was performed automatically by reducing the shortest distance between a boundary pixel and the centre by half. For testing data, nevertheless, precise instance masks were created by

[2] https://pypi.org/project/icrawler/.



(a) Dimsum (b) Cookie (c) Sushi

Fig. 8. Distribution of images across counting categories, with *x*-axis as category and *y*-axis indicates the number of images.

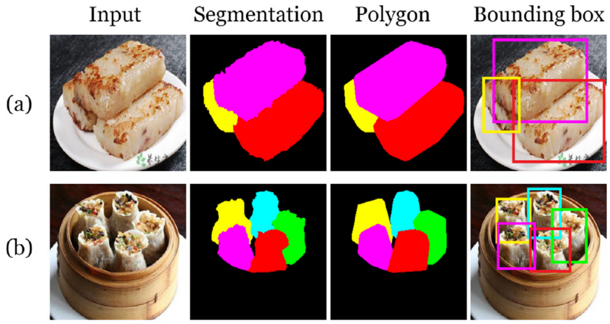


Fig. 9. Different instance masks created for experiment.

pixel labelling with the aid of the GrabCut algorithm [41]. For experiment comparison, a bounding box is also created for each instance. Fig. 9 shows the three types of masks.

5.2. Performance evaluation

Instance counting. We employ mean absolute error (MAE) for evaluation. Denote $\hat{p}_{cj}$ as the predicted count for a sample j belonging to category c. MAE measures the estimation error as following:

$$\text{MAE} = \frac{1}{C}\sum_{c=1}^{C}\sum_{j=1}^{N_c}\frac{|\hat{p}_{cj} - c|}{N_c} \tag{9}$$

where C is the number of counting categories and N_c is the number of testing samples under category c. Note that $\hat{p}_{cj}$ can be a real number with floating point precision.

Instance segmentation. We employ Panoptic Quality (PQ) [37] for evaluation. PQ first performs one-to-one matching between ground-truth and segmented instances, and then measures the intersection over union (IoU) of two matched instances. A match is qualified as true positive (*TP*) if the IoU between two instances is more than 0.5. Otherwise, the ground-truth instance is regarded as a false negative (*FN*), and a segmented instance is treated as false positive (*FP*). Specifically, denoting p and g as the segmented and ground-truth instances respectively, PQ of an image is defined as:

$$\text{PQ} = \frac{\sum_{(p,g)\in TP}\text{IoU}(p,g)}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|} \tag{10}$$

where the denominator is to penalize the result with missing and falsely detected instances. The function IoU measures the percentage of overlapping pixels. The value of PQ is then averaged over all the testing images. Note that this formula is slightly different from Kirillov et al. [37], where the average is taken over the instances of all images.

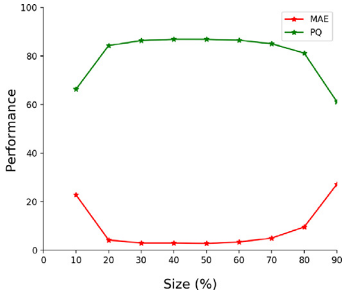


Fig. 10. The varying sizes of instance seed (*x*-axis) in impacting the performance of SibNet (*y*-axis) on Dimsum dataset. The *x*-axis shows the proportion of size w.r.t the original instance in percentage. Note that MAE is scaled up 20 times for better visualization.

5.3. Model training

All the proposed models were trained using ResNet-50 [42] as backbone. These models were pre-trained on ImageNet[3] dataset. Inspired by [43], a “cosine strategy was employed to adjust the learning rate in the ranges of $[10^{-6}, 10^{-4}]$ and $[10^{-6}, 8 \times 10^{-4}]$ for single and multi-task respectively. The cycle length was set to 8 times higher than the batch size per epoch. All the models were trained with Adam optimizer and the batch size of 16. In the experiment, the model training was stopped after 128 epochs when training loss converged.

SibNet was implemented by first end-to-end training the multi-task counting and seeding network. The two tasks were then detached from the model, and the architecture for both sibling detection and semantic segmentation was plugged in for the second round of model training. Finally, all the tasks were integrated and the model parameters were simultaneously updated for the third round of training.

6. Experimental results

6.1. Ablation and variant study

We first investigate the impact of instance seeding. Fig. 10 shows the performances of instance counting and segmentation when seeds varying between 10% to 90% of their original instance sizes were used as training examples. The result indicates that seed size should be kept within the range of 20% to 70% for satisfactory performances. When food instances are highly crowded, the predicted seeds could remain close in proximity if the seed size is reduced by less than 30% during training. On the other hand, when the size of an instance is small due to either occlusion or perspective distortion, further shrinking its size by more than 70% will result in a dot of few pixels, which increases the difficulty of predicting seeds during testing. In the remaining experiments, we set the seed size at 50%. Varying the seed size will not significantly

[3] http://www.image-net.org/.

Table 2
MAE for three variants of instance counting methods in SibNet.

Method	Dimsum	Cookie	Sushi
Counting branch	**0.14**	**0.11**	**0.14**
Segmentation branch	0.17	0.18	0.20
Counting + Segmentation	0.15	0.13	0.15

impact the performances adversely so long as the size is within a reasonable range.

Next, we investigate the options for counting in SibNet. Note that there are three variants of counting methods. With reference to Fig. 6, the first method is based on the regression counting performed by the two-task network for counting and seeding. The second method is carried out by enumerating the number of instances outputted by the instance extraction algorithm. The third method is a hybrid approach adopted by SibNet, which leverages the count regressed by the counting branch to safeguard the number of seeds to be considered for pixel connectivity analysis. The result is listed in Table 2, where regression approach outperforms the remaining two methods across three datasets. The instance segmentation approach is susceptible to small clusters of pixels due to decoration and shadow surrounding food. The hybrid approach, by using the regressed count as a prior, is effective in noise removal.

6.2. Instance counting

We compare SibNet to five major groups of approaches: glance-based [25], density map [18], count map [19], object detection [15,16,24,26–28] and instance segmentation [30,31,33–36]. Glance-based approaches perform regression counting on either image or region level. The comparison is made against two subitizing techniques (Aso-Sub and Seq-Sub) [25] for region-level counting and the two baselines (single and multi-task counting in Section 3) for image-level counting. There is a variety of approaches using density map generation for crowd counting. We compare SibNet to the classic approach in [18], which generates a density map by representing each object instance as a Gaussian distribution. The number of instances is counted by integrating the pixel values of a map. Instead of using density map, a count map, which is produced by performing block-wise classification of counts at region level, is proposed in [19]. As impressive performance is demonstrated, comparison is also made against this method (B-classification). Object detection techniques can be directly applied for counting by enumerating the number of instances being detected. We compare to the recent approaches which locate objects with bounding boxes [15,16,26–28] and blobs (LC-FCN) [24]. Different from object detection, instance segmentation generates a mask for each object instance. We compare SibNet to different approaches that are based on spatial embedding (SECB [31]), seeding (PSENet [33], PAN [35]), watershed [30] or terrace [36] algorithms and pixel affinity learning (PixelLink [34]).

Table 3 lists the counting performance of different approaches. SibNet outperforms all the approaches across the three tested datasets. Density map as well as count map, which are proven successful for crowd and vehicle counting [18,19], turn up to be suboptimal in enumerating food items of varying shapes and sizes. The glance-based approaches, either count at image or region level, generally perform better. For food images with crowded items, as shown in Fig. 11a, subitizing techniques do not perform as well as the single and multi-task baselines. When there are multiple partial portions of different items in an image region, the regressed count is fuzzy (see Seq-Sub [25] in Fig. 11a) and even human has problem judging the correct mass of count.

Table 3
MAE performance of different branches of counting approaches.

Method		Dimsum	Cookie	Sushi
Glance-based	Single-task	0.21	0.18	0.26
	Multi-task	0.19	0.17	0.22
	Aso-Sub [25]	0.24	0.23	0.27
	Seq-Sub [25]	0.24	0.23	0.26
Density map	Hydra CCNN [18]	0.30	0.33	0.37
Count map	B-classification [19]	0.39	0.39	0.51
Object detection	CornerNet [27]	0.42	0.77	0.56
	FCOS [16]	0.33	0.25	0.43
	Mask R-CNN [15]	0.29	0.38	0.48
	Yolact [26]	0.30	0.55	0.39
	ExtremeNet [28]	0.52	0.64	0.66
	LC-FCN [24]	0.46	0.92	0.73
Instance Segmentation	Watershed [30]	0.56	0.36	0.56
	PixelLink [34]	0.53	0.38	0.54
	PSENet [33]	0.25	0.30	0.45
	PAN [35]	0.58	0.45	1.06
	SECB [31]	0.33	0.28	0.46
	Terrace [36]	0.17	0.17	0.19
Ours	SibNet	**0.15**	**0.13**	**0.15**

As reported in other works [25], localization-based approaches, which count by explicitly detecting object instances, do not perform better than simpler glance-based approaches. These approaches are sensitive to food presentation, and hence performances vary across different datasets. For example, the bounding-box-based instance detection performs worse on Cookie and Sushi than Dimsum due to the diverse non-rectangle shapes of items. Predicting blobs, as by LC-FCN [24], does not show advantage over predicting boxes. LC-FCN tends to over-count when food items are decorated and under-count when food items differ in size due to camera capturing angle. Labelling pixels for instance segmentation and counting also does not necessarily result in better performance. Among them, SECB [31], PSENet [33] and Terrace [36] are the most competitive approaches to SibNet. SECB, which assigns a score to every pixel based on its offset vector to the predicted instance centroid, is sensitive to obscure boundary effect. Hence, the result is relatively poor on Sushi dataset. PSENet, although performing seeding as SibNet, is susceptible to noisy clusters produced by seeding. Attributed to counting branch, SibNet performs relatively more robustly than PSENet. The performance of Terrace depends on the number of contour levels. While more levels are helpful for locating complex shapes, it also leads to false positive counting. Fig. 11b shows examples contrasting various localization-based approaches.

6.3. Instance segmentation

We compare the performance of SibNet against proposal-based methods (Mask-RCNN [15], Yolact [26], ExtremeNet [28]), and various clustering approaches based on spatial embedding (SECB [31]), seeding (PSENet [33], PAN [35], Terrace [36]), affinity learning (PixelLink [34]) and watershed [30]. Table 4 shows their performance in terms of panoptic quality (PQ). Proposal-based approaches rely heavily on the accuracy of object detection and hence, in general, are not as superior as clustering-based approaches in dealing with diverse instance shapes. The bottom-up strategy by grouping extreme and centre points as the quadrangles of instances, as adopted by ExtremeNet [28], also does not offer advantage. Clustering-based approaches, by bottom-up pixel-level analysis, are more capable of tackling the problems of shape variation and occlusion. Similar to the result of counting, SECB [31], PSENet [33] and Terrace [36] are the most competitive approaches of Sib-

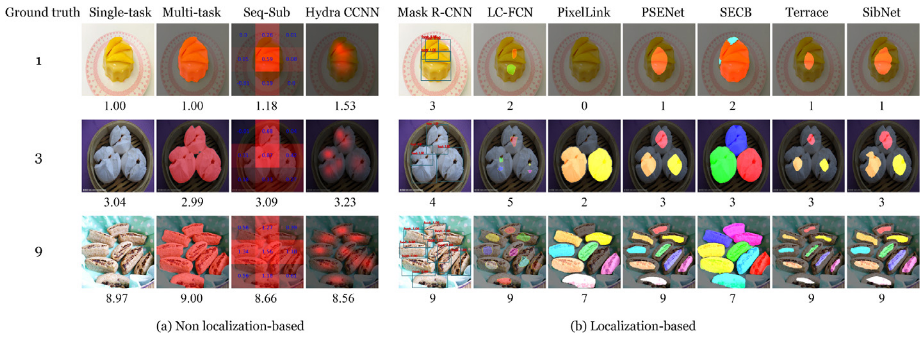


Fig. 11. Comparison of SibNet with (a) non-localization based and (b) localization based approaches. The number shown below an image is the predicted count. The segmented (multi-task, PixelLink, SECB) or detected (Mask R-CNN) food regions, seeds (SibNet, PSENet, Terrace), mass (Seq-Sub) and density (Hydra CCNN) are highlighted with overlaid colour or local count.

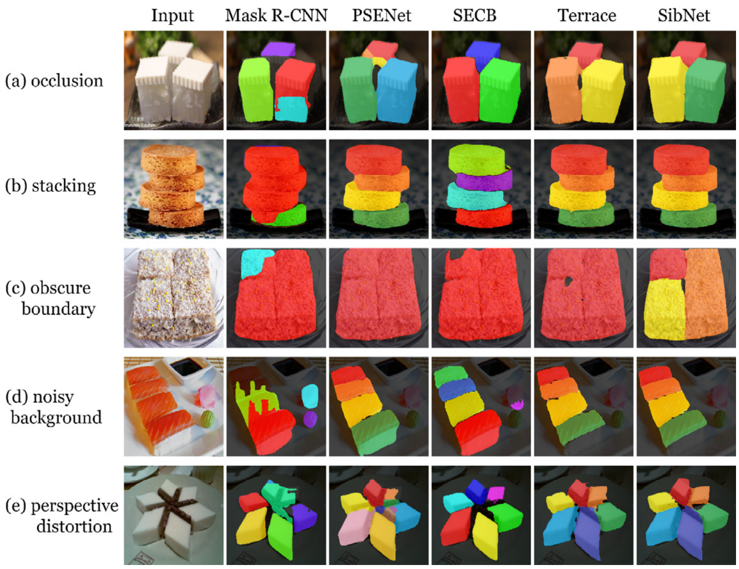


Fig. 12. Instance segmentation results on various scenarios.

Table 4
Instance segmentation performance (PQ).

Method	Dimsum	Cookie	Sushi
Mask R-CNN [15]	82.81%	81.10%	78.87%
Yolact [26]	82.34%	81.83%	78.15%
ExtremeNet [28]	82.65%	79.75%	73.03%
Watershed [30]	78.62%	85.36%	76.00%
PixelLink [34]	76.06%	82.60%	74.55%
PSENet [33]	85.71%	87.75%	81.17%
PAN [35]	81.52%	85.52%	74.57%
SECB [31]	84.38%	87.88%	80.22%
Terrace [36]	87.29%	89.15%	84.98%
SibNet	**88.06%**	**89.83%**	**85.51%**

Net. These four methods start by predicting the candidate instance centroids and then expanding the instances. The differences in performances rely on the stability of the underlying algorithms for instances of varying sizes and complex shapes. Fig. 12 contrasts the performances of different approaches by visualizing the results of instance segmentation for five different common scenarios. Compared with other three methods, PSENet is not always robust in centroid prediction, resulting in either false or miss instance detection. SECB and Terrace, on the other hand, suffer from imprecise detection of boundary pixels. The prediction of margin value as cluster size in SECB is prone to error when the shape is irregular, while using a fixed number of layers makes Terrace not reliable in boundary detection across different shapes and sizes of instances. SibNet, which locally considers pixel connectivity in different directions and uses Dice loss to penalize the misclassification of boundary pixels specifically, is empirically shown to be resilient to various challenges in food segmentation.

6.4. Speed efficiency

We further compare the computational efficiency of different approaches, as shown in Fig. 13. Except ExtremeNet whose backbone network is Hourglass, the other approaches are based on ResNet-50. Among these approaches, SibNet, Terrace [36], PAN [35] and Yolact [26] are considerably faster with processing speed of 18–19 frames per second. SibNet, although is slightly slower than PAN and Yolact, shows much better accuracy in MAE and PQ. Compared to PSENet [33] which performs multi-scale seeding and pixel connectivity analysis, SibNet is 8 times faster. Compared to SECB [31], SibNet is also superior in both speed and accuracies.

6.5. Generalization

To evaluate the generalizability of instance counting and segmentation techniques in processing food images outside of training data, a new dataset “Extra is constructed for testing. The dataset

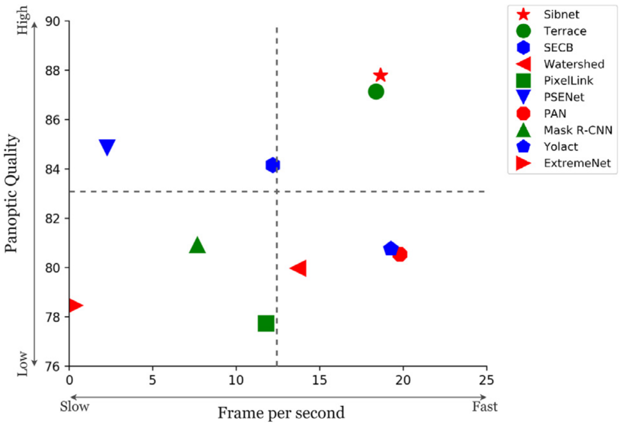


Fig. 13. Speed versus counting and segmentation accuracies for various approaches.

Table 5
Performance of Terrace [36] and SibNet for within and cross-dataset generalization test. "Extra" is a novel dataset without Dimsum, Cookie and Sushi items."All" pools all the examples in Dimsum, Cookie and Sushi for training.

Training	Testing	Counting (MAE)		Segmentation (PQ)	
		Terrace [36]	SibNet	Terrace [36]	SibNet
Dimsum	Dimsum	0.17	0.15	87.29%	88.06%
Cookie		0.62	0.47	75.28%	78.64%
Sushi		0.60	0.51	75.19%	76.63%
All		0.18	**0.14**	87.35%	**88.99%**
Dimsum	Cookie	0.67	0.56	77.51%	79.72%
Cookie		0.17	**0.13**	89.15%	**89.83%**
Sushi		0.84	0.92	75.21%	74.94%
All		0.22	0.14	88.48%	89.75%
Dimsum	Sushi	0.85	0.58	63.97%	68.93%
Cookie		0.72	0.63	63.75%	68.14%
Sushi		0.19	**0.15**	84.98%	85.51%
All		0.27	**0.15**	84.11%	**86.53%**
Dimsum	Extra	0.67	0.49	74.08%	77.48%
Cookie		0.91	0.46	72.92%	78.71%
Sushi		0.75	0.62	74.19%	74.93%
All		0.53	**0.36**	78.24%	**81.68%**

comprises 180 images uniformly distributed across 60 new food categories not existed in the three presented datasets. We compare the generalization of SibNet with Terrace [36], the most competitive approach in terms of MAE, PQ and speed. Table 5 shows their performances on four datasets, including "Extra", using the models trained by different training sets.

For both SibNet and Terrace, the models trained using the examples having the same food categories (within-dataset) are considerably better than the models trained using examples of different categories (cross-dataset). In contrast to SibNet, Terrace records a significant drop in performance in cross-dataset evaluation, even when the training examples from all three datasets are pooled for model training. The result indicates that SibNet is less dataset-dependent in both counting and instance segmentation compared to Terrace. Particularly, considering their performances on the novel "Extra" dataset, SibNet exhibits much higher accuracies consistently across the models trained with different examples. When using "All" as training dataset, SibNet outperforms Terrace by 17% in MAE and 3.5% in PQ on "Extra" dataset. Fig. 14 shows three examples demonstrating SibNet in conquering the general challenges of obscure boundary, occlusion, background clutter, diverse size and shape in the novel dataset.

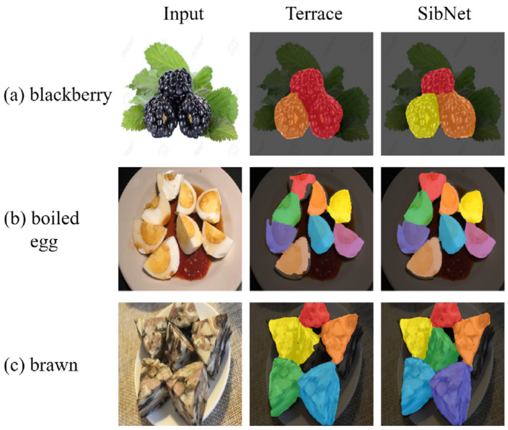


Fig. 14. Examples showing generalizability effectiveness of SibNet compared to Terrace [36] in performing instance segmentation for raw and cooked food outside of the food categories in the training datasets. SibNet produces the near-perfect segmentation and correct counting for all the three examples.

7. Conclusion

We have presented SibNet along with three food datasets to evaluate the performance. Empirical studies show that SibNet outperforms the existing approaches in both food instance counting and segmentation. Seeding is a critically important step in reducing the adverse effects due to instance occlusion and obscure boundary. When learning end-to-end together with counting, seeding can be implemented efficiently using single-scale image processing. Furthermore, the result of counting is also empirically shown to be effective in removing noisy seeds for instance segmentation. Further through the pixel connectivity analysis for sibling relation detection, SibNet provides three essential information (count, seed, affinity), resulting in a highly efficient algorithm for instance segmentation. With considerably better performance in counting and segmentation, the speed is comparable to the fastest algorithms (PAN, Yolact) in the literature. Last but not least, SibNet also generalizes well to food images of novel categories compared to Terrace.

While the results are encouraging, the current work can be extended in two directions. First, fractional counting, such as two and a half pieces of biscuits, is currently not considered. The issue is not trivial for requiring the correction of perspective distortion or even the estimation of food volume. Second, this paper considers homogeneous food counting and segmentation. Specifically, each image is assumed to contain one food category. A realistic extension of the current work is to summarize an image with food categories along with quantity information. SibNet may also be revised by extending from 2-class to multi-class semantic segmentation. The algorithm stated in Section 4.3 also needs to be revisited to consider not only pixel connectivity but also semantic coherency when propagating instance labels to neighbouring pixels.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] A. Goris, M. Westerterp-Plantenga, K. Westerterp, Undereating and underrecording of habitual food intake in obese men: selective underreporting of fat intake, Am. J. Clin. Nutr. (2000) 130–134, doi:10.1093/ajcn/71.1.130.

[2] F. Zhu, M. Bosch, N. Khanna, C.J. Boushey, E.J. Delp, Multiple hypotheses image segmentation and classification with application to dietary assessment, IEEE J. Biomed. Health Inf. (2015) 377–388, doi:10.1109/JBHI.2014.2304925.

[3] E. Aguilar, B. Remeseiro, M. Bolaños, P. Radeva, Grab, pay, and eat: Semantic food detection for smart restaurants, IEEE Trans. Multimedia (2018) 3266–3275, doi:10.1109/TMM.2018.2831627.

[4] S. Jiang, W. Min, L. Liu, Z. Luo, Multi-scale multi-view deep feature aggregation for food recognition, IEEE Trans. Image Process. (2020) 265–276, doi:10.1109/TIP.2019.2929447.

[5] X. Wang, N. Thome, M. Cord, Gaze latent support vector machine for image classification improved by weakly supervised region selection, Pattern Recognit. (2017) 59–71, doi:10.1016/j.patcog.2017.07.001.

[6] J. Lei, J. Qiu, F.P.-W. Lo, B. Lo, Assessing individual dietary intake in food sharing scenarios with food and human pose detection, in: Pattern Recognition. ICPR International Workshops and Challenges, 2021, pp. 549–557, doi:10.1007/978-3-030-68821-9_45.

[7] Y. He, C. Xu, N. Khanna, C.J. Boushey, E.J. Delp, Food image analysis: segmentation, identification and weight estimation, in: IEEE International Conference on Multimedia and Expo (ICME), 2013, pp. 1–6, doi:10.1109/ICME.2013.6607548.

[8] J. Dehais, M. Anthimopoulos, S. Shevchik, S. Mougiakakou, Two-view 3D reconstruction for food volume estimation, IEEE Trans. Multimedia (2017) 1090–1099, doi:10.1109/TMM.2016.2642792.

[9] A. Myers, N. Johnston, V. Rathod, A. Korattikara, A. Gorban, N. Silberman, S. Guadarrama, G. Papandreou, J. Huang, K. Murphy, Im2Calories: towards an automated mobile vision food diary, in: IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1233–1241, doi:10.1109/iccv.2015.146.

[10] T. Ege, Y. Ando, R. Tanno, W. Shimoda, K. Yanai, Image-based estimation of real food size for accurate food calorie estimation, in: IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), 2019, pp. 274–279, doi:10.1109/MIPR.2019.00056.

[11] M. Carvalho, R. Cadène, D. Picard, L. Soulier, N. Thome, M. Cord, Cross-modal retrieval in the cooking context: learning semantic text-image embeddings, in: International ACM SIGIR Conference on Research & Development in Information Retrieval, ACM, 2018, pp. 35–44, doi:10.1145/3209978.3210036.

[12] L. Oliveira, V. Costa, G. Neves, T. Oliveira, E. Jorge, M. Lizarraga, A mobile, lightweight, poll-based food identification system, Pattern Recognit. (2014) 1941–1952, doi:10.1016/j.patcog.2013.12.006.

[13] National Institute for Nutrition and HealthChinese Center for Disease Control and Prevention, China food composition tables, Beijing: Peking University Medical Press, 2019.

[14] J.K. Ahuja, A.J. Moshfegh, J.M. Holden, E. Harris, Usda food and nutrient databases provide the infrastructure for food and nutrition research, policy, and practice, J. Nutr. (2012) 241–249, doi:10.3945/jn.112.170043.

[15] K. He, G. Gkioxari, P. Dollár, R.B. Girshick, Mask R-CNN, in: IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980–2988, doi:10.1109/ICCV.2017.322.

[16] Z. Tian, C. Shen, H. Chen, T. He, FCOS: fully convolutional one-stage object detection, in: IEEE International Conference on Computer Vision (ICCV), 2019, pp. 9626–9635, doi:10.1109/ICCV.2019.00972.

[17] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3431–3440, doi:10.1109/TPAMI.2016.2572683.

[18] D. Oñoro-Rubio, R.J. López-Sastre, Towards perspective-free object counting with deep learning, in: European Conference on Computer Vision (ECCV), 2016, pp. 615–629, doi:10.1007/978-3-319-46478-7_38.

[19] L. Liu, H. Lu, H. Xiong, K. Xian, Z.-G. Cao, C. Shen, Counting objects by blockwise classification, IEEE Trans. Circuits Syst.Video Technol. (2019), doi:10.1109/TCSVT.2019.2942970.

[20] J. Cohen, G. Boucher, C. Glastonbury, H. Lo, Y. Bengio, Count-ception: counting by fully convolutional redundant counting, in: IEEE International Conference on Computer Vision Workshop (ICCVW), 2017, pp. 18–26, doi:10.1109/ICCVW.2017.9.

[21] A. Ferrari, S. Lombardi, A. Signoroni, Bacterial colony counting with convolutional neural networks in digital microbiology imaging, Pattern Recognit. (2017) 629–640, doi:10.1016/j.patcog.2016.07.016.

[22] H. Lu, Z.-G. Cao, Y. Xiao, B. Zhuang, C. Shen, TasselNet: counting maize tassels in the wild via local counts regression network, Plant Methods (2017), doi:10.1186/s13007-017-0224-0.

[23] S. Chen, S. Skandan, S. Dcunha, J. Das, E. Okon, C. Qu, C. Taylor, V. Kumar, Counting apples and oranges with deep learning: a data driven approach, IEEE Rob. Autom. Lett. (2017) 781–788, doi:10.1109/LRA.2017.2651944.

[24] I.H. Laradji, N. Rostamzadeh, P.O. Pinheiro, D. Vazquez, M. Schmidt, Where are the blobs: Counting by localization with point supervision, in: The European Conference on Computer Vision (ECCV), 2018, doi:10.1109/CVPR.2019.00904.

[25] P. Chattopadhyay, R. Vedantam, R.R. Selvaraju, D. Batra, D. Parikh, Counting everyday objects in everyday scenes, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, doi:10.1109/CVPR.2017.471.

[26] D. Bolya, C. Zhou, F. Xiao, Y.J. Lee, YOLACT: real-time instance segmentation, in: The IEEE International Conference on Computer Vision (ICCV), 2019, doi:10.1109/ICCV.2019.00925.

[27] H. Law, J. Deng, CornerNet: detecting objects as paired keypoints, Int. J. Comput. Vis. (2019), doi:10.1007/s11263-019-01204-1.

[28] X. Zhou, J. Zhuo, P. Krahenbuhl, Bottom-up object detection by grouping extreme and center points, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 850–859, doi:10.1109/CVPR.2019.00094.

[29] D. Park, J. Lee, J. Lee, K. Lee, Deep learning based food instance segmentation using synthetic data, CoRR, 2021.

[30] M. Bai, R. Urtasun, Deep watershed transform for instance segmentation, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2858–2866, doi:10.1109/CVPR.2017.305.

[31] D. Neven, B.D. Brabandere, M. Proesmans, L.V. Gool, Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, doi:10.1109/CVPR.2017.471.

[32] Y. Zhu, J. Du, TextMountain: accurate scene text detection via instance segmentation, Pattern Recognit. (2021) 107336, doi:10.1016/j.patcog.2020.107336.

[33] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, S. Shao, Shape robust text detection with progressive scale expansion network, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 9328–9337, doi:10.1109/CVPR.2019.00956.

[34] D. Deng, H. Liu, X. Li, D. Cai, PixelLink: detecting scene text via instance segmentation, The Association for the Advancement of Artificial Intelligence (AAAI), 2018, doi:10.1109/CVPR.2017.305.

[35] W. Wang, E. Xie, X. Song, Y. Zang, W. Wang, T. Lu, G. Yu, C. Shen, Efficient and accurate arbitrary-shaped text detection with pixel aggregation network, in: The IEEE International Conference on Computer Vision (ICCV), 2019, pp. 8439–8448, doi:10.1109/ICCV.2019.00853.

[36] H.-T. Nguyen, C.-W. Ngo, Terrace-based food counting and segmentation, in: The Association for the Advancement of Artificial Intelligence (AAAI), 2021, pp. 2364–2372.

[37] A. Kirillov, K. He, R. Girshick, C. Rother, P. Dollar, Panoptic segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, doi:10.1109/CVPR.2019.00963.

[38] R. Adams, L. Bischof, Seeded region growing, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, 1994, pp. 641–647, doi:10.1109/34.295913.

[39] R. Deng, C. Shen, S. Liu, H. Wang, X. Liu, Learning to predict crisp boundaries, in: European Conference on Computer Vision (ECCV), 2018, pp. 570–586, doi:10.1007/978-3-030-01231-1_35.

[40] Y. Hsu, Z. Xu, Z. Kira, J. Huang, Learning to cluster for proposal-free instance segmentation, in: International Joint Conference on Neural Networks (IJCNN), 2018, pp. 1–8, doi:10.1109/IJCNN.2018.8489379.

[41] C. Rother, V. Kolmogorov, A. Blake, GrabCut: interactive foreground extraction using iterated graph cuts, in: ACM SIGGRAPH, 2004, pp. 309–314, doi:10.1145/1186562.1015720.

[42] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, doi:10.1109/CVPR.2016.90.

[43] I. Loshchilov, F. Hutter, SGDR: stochastic gradient descent with warm restarts, in: International Conference on Learning Representations (ICLR), 2017.

Huu-Thanh Nguyen is currently working towards a PhD degree at VIREO Group, Department of Computer Science, City University of Hong Kong. He was a visiting researcher with NExT++ research centre at National University of Singapore in 2018 and 2019. His research interests include deep learning and computer vision. His works focus on food recognition, counting, detection, segmentation, and recipe retrieval.

Chong-Wah Ngo is a Professor with the School of Computing and Information Systems, Singapore Management University. His research interests include large-scale multimedia information retrieval, video computing, multimedia mining, and visualization. Prof. Ngo was an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA (2011–2014), Conference Co-Chair of MM 2019 and ICMR 2015, Program Co-Chair of MMM 2012 and ICMR 2012, Chairman of ACM (Hong Kong Chapter) from 2008 to 2009.

Wing-Kwong Chan is an Associate Professor at City University of Hong Kong. His current main research interest is software engineering, program analysis and software infrastructure for AI-based systems. Currently, he is a Special Issues Editor of Journal of Systems and Software, associate editor of Software Testing, Verification and Reliability and International Journal of Creative Computing, program chairs of COMPSAC 2021 and AITest 2021.