Machine Vision and Applications (2021) 32:56
https://doi.org/10.1007/s00138-021-01186-6

ORIGINAL PAPER

An end-to-end annotation-free machine vision system for detection of products on the rack

Bikash Santra[1] · Avishek Kumar Shaw[2] · Dipti Prasad Mukherjee[1]

Received: 1 May 2020 / Revised: 15 November 2020 / Accepted: 12 February 2021 / Published online: 10 March 2021

Abstract

Given single instance (or template image) per product, our objective is to detect merchandise displayed in the images of racks available in a supermarket. Our end-to-end solution consists of three consecutive modules: exemplar-driven region proposal, classification followed by non-maximal suppression of the region proposals. The two-stage exemplar-driven region proposal works with the example or template of the product. The first stage estimates the scale between the template images of products and the rack image. The second stage generates proposals of potential regions using the estimated scale. Subsequently, the potential regions are classified using convolutional neural network. The generation and classification of region proposal do not need annotation of rack image in which products are recognized. In the end, the products are identified removing ambiguous overlapped region proposals using greedy non-maximal suppression. Extensive experiments are performed on one in-house dataset and three publicly available datasets: Grocery Products, WebMarket and GroZi-120. The proposed solution outperforms the competing approaches improving up to around 4% detection accuracy. Moreover, in the repeatability test, our solution is found to be better compared to state-of-the-art methods.

Keywords Detection · Recognition · Grocery products · Single instance · Region proposal · Non-maximal suppression

1 Introduction

Humans effortlessly identify merchandise displayed on the racks in a supermarket. But the integration of such skill in a smart machine vision system poses many challenges [31]. A product detection system is designed for continuous monitoring of the arrangement of products in racks [21], checking compliance of the planogram (plan of display of products) [28], estimating stock of the products [16], enhancing the value-added service to the consumer [7] and assisting visually impaired shoppers [34].

Each product in a supermarket usually has one marketing image (also referred as product or template image) for promotion. The marketing images of the products are typically captured in a controlled studio environment. On the contrary, the rack images are clicked in an uncontrolled retail store environment. Thus, the quality, illumination and resolution of the rack image differ from those of marketing images (see Fig. 1). This poses an important challenge in detecting products on the rack image. The recognition of similar yet nonidentical (fine-grained) products is yet another challenge.

The bottom row of Fig. 1 shows a few examples of product images, while the top row of Fig. 1 shows some rack images. We also provide some examples of fine-grained products in the bottommost row of Fig. 1a.

In this paper, we introduce an end-to-end machine vision system for detecting the products displayed on a rack image. Note that the detection of products refers to the recognition and localization of products in one go.

The proposed solution is obtained under the following assumptions: (i) The physical dimension of each product template is available in some suitable unit of length. In case of absence of physical dimension of the products, we use the context information of retail store. The context information assume similar products or products of similar shapes are arranged together in rack for shopper's convenience. (ii) All

✉ Bikash Santra
bikash.santra@isical.ac.in

Avishek Kumar Shaw
shaw.avishek@tcs.com

Dipti Prasad Mukherjee
dipti@isical.ac.in

[1] Electronics and Communication Sciences Unit Indian Statistical Institute, Kolkata, India

[2] Tata Consultancy Services Limited, Kolkata, India

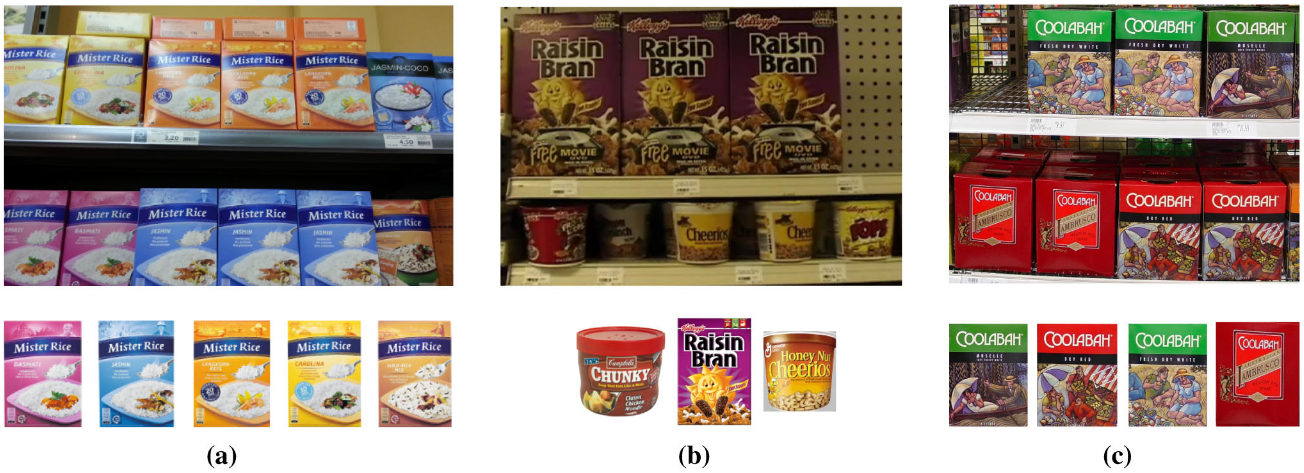


Fig. 1 Examples of rack (top row) and marketing images (bottom row) from **a** Grocery Products [6], **b** GroZi-120 [23] and **c** WebMarket [37] datasets

rack images are captured where the camera is nearly fronto-parallel with respect to face of the rack. Assume that $\mathtt{n}_f$ is the normal to the plane representing the face of the rack. Also assume that the normal to the image (camera) plane is $\mathtt{n}_c$. If $\mathtt{n}_f$ and $\mathtt{n}_c$ are either collinear or mutually parallel contained in a plane, say π, and the plane π is parallel to the ground plane (on which the cameraman and the racks are standing), then the image capturing position is fronto-parallel. This is now shown in Fig. 2.

The proposed solution first introduces a two-stage exemplar-driven region proposal algorithm using the information of individual product images. The region proposal is exemplar-driven as the proposals are generated based on individual example or image templates of products. Each proposal is then classified by a convolutional neural network (CNN) [14]. The product classification does not need any annotation of the rack image from where the individual products are identified. Subsequently, greedy non-maximal suppression technique [3] is implemented to remove the overlapping and ambiguous region proposals.

The paper is organized as follows. We present literature review and our contributions in Sect. 1.1. Section 2 describes the proposed method. The experimental analysis is performed in Sect. 3. Finally, Sect. 4 concludes the paper.

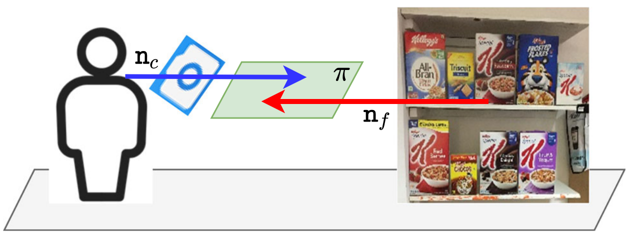


Fig. 2 Illustration of fronto-parallel imaging of a rack

1.1 Literature review and contributions

A recent comprehensive survey on identification of products displayed in rack images is available in [31]. In this section, we discuss approaches relevant to our proposal in order to highlight the gaps in the existing approaches and our contribution.

The advent of deep learning-based schemes like R-CNN [10], Fast R-CNN [9], YOLO [29] and Mask R-CNN [13] shows significant improvement in performance for object detection. These methods cannot be used straightway as they need annotated rack images for training. These annotations are difficult to obtain with frequent changes in promotional packages and product display plan. The exemplar-independent R-CNN [10] and its variants do not use the product template for generating region proposals. Rather they depend on the annotation of the rack image to learn shape and scale of different products. In order to rectify the generated product proposals, exemplar-independent R-CNN optionally implements *bounding box regression* which again demands annotated rack images. On the contrary, the proposed annotation-free system (which does not require annotated rack images) introduces an exemplar-driven (or exemplar-dependent) region proposal scheme to extract the region proposals around the products displayed on the rack using individual marketing images of the products.

Ray et al. [28] find out the potential locations of the products displayed on a shelf of a rack image by moving rectangular regions (often referred to as sliding windows) of different scales over the shelf. Consequently, they obtain the closest product class of a cropped region by matching score of the cropped regions and product templates. Finally, the products are detected by finding out the maximum weighted path of a directed acyclic graph of cropped regions.

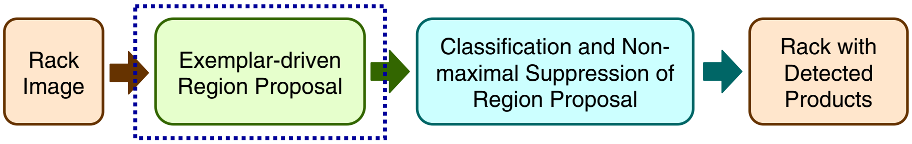

Rack Image
Exemplar-driven Region Proposal
Classification and Non-maximal Suppression of Region Proposal
Rack with Detected Products

Fig. 3 Flowchart of the proposed scheme; blue-colored dotted rectangle highlights our contribution (colour figure online)

Unlike [28], the authors of [21] automatically detect shelves from a rack image using a reference image of the rack captured earlier in the same store with a known resolution. Then at each shelf, the regions covered by sliding windows are extracted and matched with the product templates and the overlapped proposals are disambiguated using non-maximal suppression.

Merler et al. [23] compare three different object localization strategies for detecting products. They study sliding window-based histogram matching method, SIFT matching method [20] and Adaboost-based method [5] for detecting products. In [4], Franco et al. extract a number of rectangular regions at each corner points detected using Harris corner detector [12] and match those regions with the products.

George et al. [6] subdivide a rack image into a number of grids and estimate multi-class ranking of products by classifying each grid using discriminative random forest (DRF) [35]. Then a deformable spatial pyramid-based fast dense pixel matching [17] and genetic algorithm-based optimization scheme [11] are performed for identification of products. In [37], Zhang et al. implement Harris-Affine interest region detector [24] to find out important regions in an image. The SIFT features are extracted only from important regions for representing an image. They subdivide both product and rack images into sub-images and match them using SIFT features.

The contribution of the proposed work compared to state-of-the-art methods is twofold:

(a) We propose an automated approach to estimate the scale between product(s) and rack image. The previous attempts either move windows of different scales over the rack image [21,23,28] or divide the rack image into a number of grids of different resolutions [6,37] to find out potential regions in the rack.
(b) We introduce an exemplar-driven region proposal scheme for detecting objects in a scene crowded with products in contrast to annotation-based region proposal scheme (without taking the help of product images) using *bounding box regression* [10] in R-CNN or its variants.

2 Methods

Overall methodology of the proposed scheme is demonstrated in Fig. 3. Our solution takes care of both recognition and localization of products (including multiple products stacked horizontally or vertically) displayed on the rack. The following subsections present the modules of our scheme.

2.1 Exemplar-driven region proposal

The probable locations of products in a rack are determined through the proposed exemplar-driven regional proposal (ERP) scheme. As shown in Fig. 4, the proposed ERP takes a rack image as input and returns a number of region proposals for the rack (green rectangular boxes in Fig. 4). This region proposal scheme is a two-stage process: estimation of scale between the product image and the rack image, and the extraction of region proposal (see Fig. 4).

Let η be the number of individual products in a database of product templates $\mathbb{D}$. In other words, η is the number of product classes. We refer each product template as $\mathbb{D}_t$, $t = 1, 2, \ldots, \eta$. A few examples of $\mathbb{D}_t$ are displayed in the bottom row of Fig. 1. Let the physical dimensions of the products be available in some unit of length (say cm). Assume the given rack image is I which displays multiple products. The top row of Fig. 1 illustrates three examples of such I. However, the physical dimension of the rack I is unknown. Given this setting, our aim is to localize the products $\mathbb{D}_t$ in the rack I.

We first extract the BRISK [18,25] descriptors of all the template images of products and rack image. A BRISK descriptor defines a key-point in the image and a 64-dimensional feature vector at that point. In the feature vector, each feature value is a 8 bit number. Hence, the feature vector is a 512 bit number. Assume we obtain β and γ BRISK key-

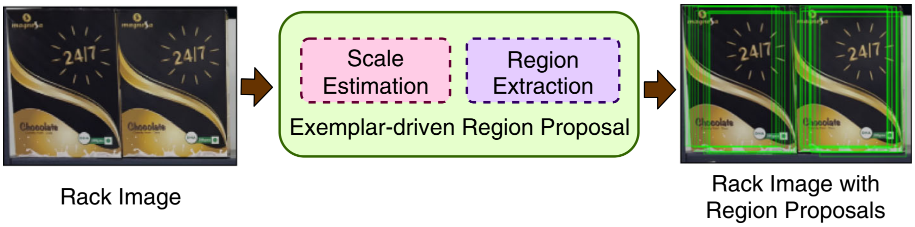

Rack Image
Scale Estimation
Region Extraction
Exemplar-driven Region Proposal
Rack Image with Region Proposals

Fig. 4 Flowchart of the proposed two-stage exemplar-driven regional proposal. Green rectangular boxes on the rack image are the regional proposals generated by the proposed ERP (colour figure online)

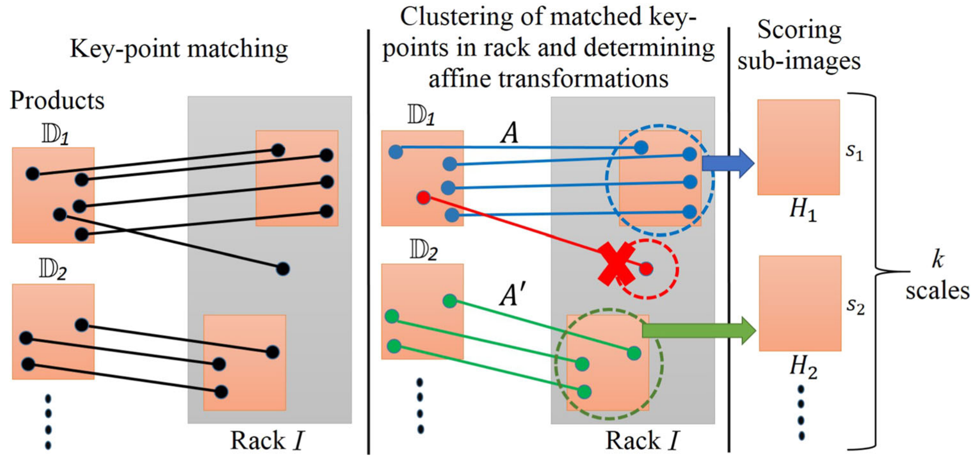


Fig. 5 Scale estimation procedure: black dots and lines represent key-points and matched correspondences. Blue, red and green circles indicate the clusters of matched key-points in rack. Correspondences of the clustered key-points in the product are also highlighted using the respective colors of clusters. s_1 and s_2 are the scores of sub-images H_1 and H_2 extracted from rack, respectively (colour figure online)

points in the tth product $\mathbb{D}_t$ and the rack I, respectively. Let (x_t^i, y_t^i) and f_t^i, $i = 1, 2, \ldots, \beta$ be the BRISK key-points and corresponding feature vectors of the tth product. Also let (x_I^j, y_I^j) and f_I^j, $j = 1, 2, \ldots, \gamma$ be the BRISK key-points and corresponding feature vectors of the rack image I. Using these BRISK key-points, we generate a set of region proposals for the rack I through the following two successive stages.

2.1.1 Stage-1: scale estimation

The scale between the products and rack plays an important role in extracting the potential regions. As mentioned earlier in Sect. 1.1, the computer vision practitioners in their previous attempts take care of the scale between the products and rack considering variable sizes of search windows [4,23,28] or grids [6,21,37] in the rack. On the contrary, we estimate k possible scales between the η #products $\mathbb{D}_t$ and rack image I using the physical dimensions of products.

The scale estimation process starts by calculating the geometric transformations between the images of products and rack. Each transformation is assigned a classification score. Using transformations with top-k scores, we estimate the k possible scales between products and shelf. The overall process of estimating scale is demonstrated in Fig. 5. In the following paragraphs, we detail the entire process of estimation of scales.

Step 1: Matching of key-points The BRISK key-points of the tth product with those of the rack I are first matched using the approach presented in [20]. Note that the matching of the key-points refers to the matching of the feature vectors at those key-points. The procedure of matching the BRISK keypoints is clearly explained through the following example.

Assume we have a feature vector f_t (in the product $\mathbb{D}_t$) which we want to match with the two feature vectors f_I^1 and f_I^2 in the rack image I. Further assume the *Hamming distance* between f_t and f_I^1 is d_1, while the *Hamming distance* between f_t and f_I^2 is d_2 and let $d_1 > d_2$. Given this, we aim to find out the correct match of f_t in the rack I.

Since the BRISK feature vectors are 512-bit binary numbers, the Hamming distance between f_t and any one of f_I^1 and f_I^2 could be maximum 512-bit binary number. However, we assume that a potential match between two feature vectors is valid only when the distance between the two feature vectors is much less compared to their maximum distance (which is 512 bits). The distance between two feature vectors eligible to be a match is taken as $m = matchThreshold \times 512$, where *matchThreshold* $\in [0, 1]$.

Therefore, if both d_1 and d_2 are lower than m, then both f_I^1 and f_I^2 are eligible to be the potential matches of f_t. Next we move to another test (which is called *ratio test*) for finding out the correct match of f_t in the rack I.

Since $d_1 > d_2$, f_I^2 is the better match of f_t compared to the match of f_I^1 with f_t. If d_1 and d_2 values are close, this implies that f_I^1 and f_I^2 both are the potential matches of f_t. This results in an ambiguity. To address this, we allow only one match of f_t in the rack I for accurately identifying the unique correspondences between the key-points of $\mathbb{D}_t$ and I.

We remove this ambiguity by rejecting both f_I^1 and f_I^2 as a match of f_t if the ratio $\frac{d_2}{d_1}$ is greater than a threshold, *ratioThreshold* $\in [0, 1]$. Else the first nearest match f_I^2 is defined as the correct match for f_t in I.

Step 2: Clustering of matched key-points in rack The matching of BRISK key-points is performed to find out the probable locations of tth product $\mathbb{D}_t$ in the rack I. But the matching of key-points is not sufficient to correctly determine the locations of the $\mathbb{D}_t$ in I when I contains multiple instances of the $\mathbb{D}_t$. In other words, the matched key-points of $\mathbb{D}_t$ in I can represent multiple instances of $\mathbb{D}_t$. Thus, we cluster ζ #key-points in I to locate the multiple instances of $\mathbb{D}_t$. More formally, the matched key-points (x_I^m, y_I^m), $m = 1, 2, \ldots, \zeta$ in I are clustered using the DBSCAN [2,33] clustering technique.

The DBSCAN clustering method is implemented with two parameters $minimumPoints$ and $maximumRadius$. We require at least $minimumPoints$ number of points to form a cluster. And for each point (x_I^m, y_I^m) in a cluster, there exists at least one point $(x_I^{m'}, y_I^{m'})$ in the cluster such that $\|(x_I^{m'}, y_I^{m'}) - (x_I^m, y_I^m)\| \le maximumRadius$. This stage of our region proposal scheme aims to determine an affine transformation A of the tth product $\mathbb{D}_t$ with its potential match in the rack I. We set $minimumPoints = 3$ as we require exactly three point correspondences to determine an affine transformation. Note that the number of matches of the product $\mathbb{D}_t$ in I is the number of clusters determined by the DBSCAN algorithm. In the second block of Fig. 5, the clusters defined by blue and green circles are selected for the next step of our algorithm as these clusters have exactly or more than three point correspondences. The cluster denoted by red circle (containing less than three points) is discarded.

Step 3: Determining affine transformations of products Assume we obtain ρ_t #clusters of matched key-points (x_I^m, y_I^m) in the rack I for the tth product $\mathbb{D}_t$. In Fig. 5, $\rho_t = 2$. Let (x_I^n, y_I^n) and (x_t^n, y_t^n), $n = 1, 2, \ldots, \varphi$ be the key-points in the cth cluster and corresponding matched key-points in the tth product $\mathbb{D}_t$. Let $X_I^n = [x_I^n \;\; y_I^n \;\; 1]^T$ and $X_t^n = [x_t^n \;\; y_t^n \;\; 1]^T$ be the representations of the key-points. Given this, we aim to determine the affine transformation matrix A with six unknowns between $\mathbb{D}_t$ and I such that

$$X_I^n \equiv A X_t^n. \tag{1}$$

As $minimumPoints$ is set to 3 during clustering of key-points, we can have more than three correspondences of key-points between the tth product and cth cluster in I, i.e., $\varphi \ge 3$. In order to solve this overdetermined problem, we minimize the least squared sum $\mathfrak{S}(A)$ using Levenberg–Marquardt algorithm [19,22], where

$$\mathfrak{S}(A) \equiv \operatorname{argmin}_A \sum_{n=1}^{\varphi} \left\| X_I^n - A X_t^n \right\|_2^2. \tag{2}$$

Step 4: Extracting sub-images from rack Once the affine matrix A is obtained, we determine the transformation of the four corner points $(0, 0)$, $(w, 0)$, $(0, h)$, and (w, h) of the tth product template $\mathbb{D}_t$ in I using (1), where h and w are the width and height (in pixels) of $\mathbb{D}_t$. Using these four transformed points, we fit the largest possible rectangular bounding box and crop the region covered by the bounding box from I. Let the cropped sub-image be H. If the sub-image H matches (with high degree of confidence, i.e., with high classification score) with the product $\mathbb{D}_t$, the transformation is a potential transformation. Thus, the sub-image H determines the correctness of the transformation A. Examples of this sub-image extraction are illustrated in the third block of Fig. 5 (see H_1 and H_2).

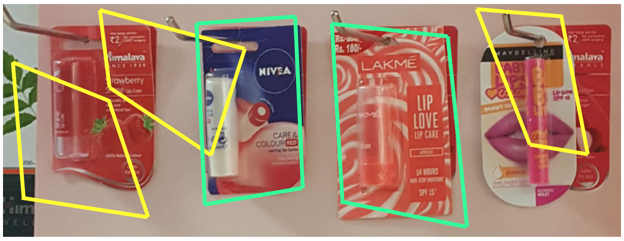


Fig. 6 Examples of transformed corner points (highlighted with the quadrilaterals) of products in rack using estimated affine transformations. The tight fitted rectangle, which covers a quadrilateral, is cropped and classified. Green and yellow rectangles provide correct and incorrect classifications, respectively (colour figure online)

Step 5: Scoring the sub-images As the sub-image H is obtained from $\mathbb{D}_t$, the class label of H must be $\mathbb{D}_t$. Thus, we match H with $\mathbb{D}_t$ in order to argue for the true label of H.

In order to match H with $\mathbb{D}_t$, we obtain the class label and classification score of H using our classification module. The classification adapting pre-trained CNN model [14] to this problem is detailed in Sect. 3. H is fed into the fine-tuned pre-trained CNN classifier (i.e., the pre-trained CNN which is adapted/fine-tuned to our product classification task) to obtain the class label and classification score of H. If the label of H is determined as $\mathbb{D}_t$, then H is considered as a valid sub-image and sent for further processing in estimating scale between the products and rack.

In this way, we perform this extraction and scoring of sub-images for all the clusters and all the products $\mathbb{D}_t$. Finally, we get a set of τ valid sub-images $\mathbb{H} = \{H_g\}$ with the set $\mathbb{L} = \{l_g\}$ of corresponding class labels and the set $\mathbb{S} = \{s_g\}$ of corresponding classification scores, where $g = 1, 2, 3, \ldots, \tau$. Figure 5 shows two valid sub-images H_1 and H_2 (along with the classification scores s_1 and s_2) extracted from the rack for the products $\mathbb{D}_1$ and $\mathbb{D}_2$, respectively.

Step 6: Estimating k possible scales A number of sub-images $H_1, H_2, H_3, \ldots$ with respective classification scores are obtained after Step 5. Top k number of sub-images H_g sorted based on descending classification scores are selected to define k different scales between sub-images H_g and corresponding products $\mathbb{D}_t$.

Let $\mathbb{S}' \subseteq \mathbb{S}$ containing top-k scores. Hence, we obtain the sets $\mathbb{H}' \subseteq \mathbb{H}$ and $\mathbb{L}' \subseteq \mathbb{L}$ corresponding to $\mathbb{S}'$. For each $H \in \mathbb{H}'$, we obtain the label l of H from $\mathbb{L}'$ and determine the cm-to-pixel ratio of width (or x-scale) and height (or y-scale) using the physical dimensions of lth labeled product $\mathbb{D}_l$ and pixel dimensions of the sub-image H of I. Consequently, we get k cm-to-pixel x-scales, sc_x^u and y-scales, sc_y^u.

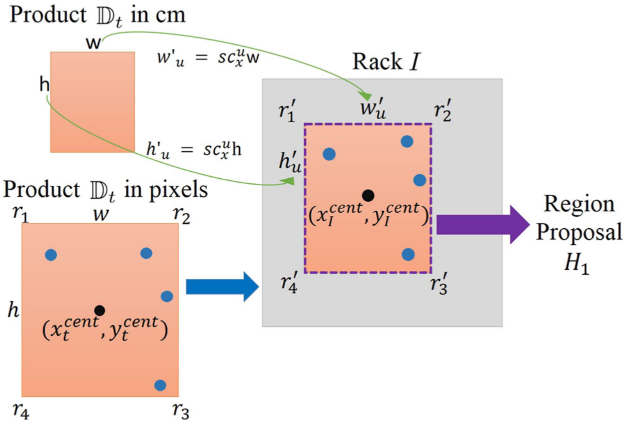


Fig. 7 Process of generating region proposals. Blue dots represent the key-points. Black dots show the centroid of the key-points (colour figure online)

2.1.2 Stage-2: region extraction

Figure 6 presents a few examples of transformed corner points of the products in a rack using the affine matrices as described in Stage-1 of our scheme (refer Sect. 2.1.1). We can see the incorrect transformations of corner points (which produce yellow quadrilaterals in the figure) of the products in rack. These incorrect transformations extract many sub-images which either do not cover any product or display a tilted/skewed image of true product. Furthermore, the affine transformations are not calculated in Stage-1 when there exists only one or two matches of the key-points between the products and rack. Which is why Stage-2 of our region proposal algorithm is introduced. Moreover, Stage-2 of the proposed region proposal scheme exhaustively searches for the potential regions around the product displayed on the rack.

Similar to Stage-1, key-points of the products and rack are matched followed by clustering of key-points in the rack as shown in 1st and 2nd block of Fig. 5. But the *minimumPoints* parameter of DBSCAN clustering algorithm is set to 1. Because in extracting potential regions from the rack, we do not want to lose a single proposal even if there exists only one match of key-points between the products and rack.

Let us assume that we obtain ρ_t #clusters of the matched key-points in the rack image I. In Fig. 7, we show only one cluster (say, cth cluster) for the tth product $\mathbb{D}_t$. As shown in Fig. 7, let (x_I^n, y_I^n) and $(x_t^n, y_t^n), n = 1, 2, 3, 4$ be the matched key-points in the cth cluster of the rack I and the tth product $\mathbb{D}_t$, respectively. For the cth cluster, we now extract potential regions from I using the geometric alignment of the matched key-points and k estimated cm-to-pixel scales. In the procedure of extracting such potential regions, we first calculate the centroids $(x_I^{\text{cent}}, y_I^{\text{cent}})$ and $(x_t^{\text{cent}}, y_t^{\text{cent}})$ of the matched key-points (x_I^n, y_I^n) in the cth cluster of I and (x_t^n, y_t^n) in the tth product (see the black dots in Fig. 7).

In Sect. 2.1.1, we have already derived k cm-to-pixel x-scales sc_x^u and y-scales sc_y^u between the products and rack, $u = 1, 2, \ldots, k$. However, rest of the process is described for only uth scale considering $k = 1$. Let h and w be the height and width in pixels of $\mathbb{D}_t$, while let h and w be the height and width in cm of $\mathbb{D}_t$. For uth cm-to-pixel scale sc_x^u and sc_y^u, the transformed width and height of $\mathbb{D}_t$ in I are $w'_u = sc_x^u$ w pixels and $h'_u = sc_y^u$ h pixels, respectively (see Fig. 7).

Algorithm 1: Exemplar-driven Region Proposal

```
Input  : The rack image $I$
Output: The region proposals $H$
for each product $\mathbb{D}_t$ in the database $\mathbb{D}$ do
    Find the BRISK key-points of $\mathbb{D}_t$;
end
Find the BRISK key-points of the rack image $I$;
/* Stage-1: Scale Estimation                    */
for each $\mathbb{D}_t$ in $\mathbb{D}$ do
    Match the key-points of $\mathbb{D}_t$ with that of $I$;
    Find the clusters of matched key-points in $I$;
    for each cluster do
        Calculate affine matrix $A$ between the of matched
        key-points of $\mathbb{D}_t$ and $I$ using (1);
        Extract a sub-image $H$ of $I$ applying $A$ on $\mathbb{D}_t$;
        Find the class label and classification score of $H$ from
        CNN;
    end
end
Find top-$k$ classification scores and corresponding labels
among that of all $H$;
Find $k$ cm-to-pixel scales between the products and rack using
physical dimensions of the products in the database $\mathbb{D}$;
/* Stage-2: Region Extraction                   */
for each $\mathbb{D}_t$ in $\mathbb{D}$ do
    Match the key-points of $\mathbb{D}_t$ with that of $I$;
    Find the clusters of matched key-points in $I$;
    for each cluster do
        for each cm-to-pixel scale do
            Extract potential region $H$ from $I$ using (3);
        end
    end
end
```

As shown in Fig. 7, let r_1, r_2, r_3 and r_4 be the four corner points of the tth product $\mathbb{D}_t$. So, $r_1 = (0, 0), r_2 = (w, 0), r_3 = (w, h)$, and $r_4 = (0, h)$. Therefore, the centroid $(x_t^{\text{cent}}, y_t^{\text{cent}})$ must lie within the rectangle formed by these four corner points. Let the four corner points be transformed to r'_1, r'_2, r'_3 and r'_4 in the rack I for the cth cluster (see Fig. 7). Then for the uth scale, the coordinates of the transformed points in I are determined as follows:

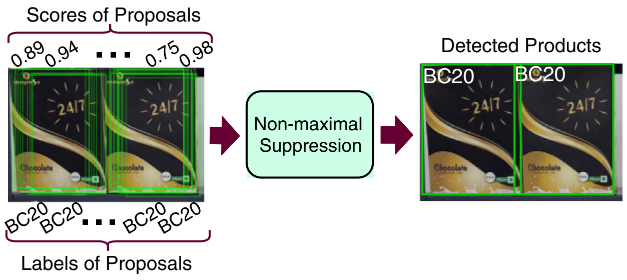


Fig. 8 Flowchart of the non-maximal suppression scheme. BC20 and 0.89, 0.94, . . . , 0.98 are the class labels and classification scores of the proposals, respectively, obtained from the pre-trained CNN [14] adapted to our product classification task

$$
\begin{aligned}
r'_1 &= \left(x_I^{\text{cent}} - \frac{w'_u}{w}\, x_t^{\text{cent}},\, y_I^{\text{cent}} - \frac{h'_u}{h}\, y_t^{\text{cent}} \right)\\
r'_2 &= \left(x_I^{\text{cent}} - \frac{w'_u}{w}\, x_t^{\text{cent}} + w'_u,\, y_I^{\text{cent}} - \frac{h'_u}{h}\, y_t^{\text{cent}} \right)\\
r'_3 &= \left(x_I^{\text{cent}} - \frac{w'_u}{w}\, x_t^{\text{cent}} + w'_u,\, y_I^{\text{cent}} - \frac{h'_u}{h}\, y_t^{\text{cent}} + h'_u \right)\\
r'_4 &= \left(x_I^{\text{cent}} - \frac{w'_u}{w}\, x_t^{\text{cent}},\, y_I^{\text{cent}} - \frac{h'_u}{h}\, y_t^{\text{cent}} + h'_u \right). \qquad (3)
\end{aligned}
$$

Let H_1 be the rectangular region of I covered by these four points r'_1, r'_2, r'_3 and r'_4 (refer Fig. 7). So H_1 is a potential region proposal having some productness. The process is repeated for all k #scales and ρ_t #clusters for the tth product. The #clusters varies from one product to another. Thus, for the η #products, the total number of clusters can be determined as $e = \sum_{t=1}^{\eta} \rho_t$. By iterating the process for η #products, our exemplar-driven region proposal scheme generates $\chi = \eta k e$ number of region proposals $H_z, z = 1, 2, \cdots, \chi$. Algorithm 1 summarizes the proposed region proposal scheme. Next we discuss the classification and non-maximal suppression of the region proposals.

2.2 Classification and non-maximal suppression

Figure 8 demonstrates the initial set of overlapping region proposals obtained from the previous step. These initial proposals with their classification scores and labels (obtained from the pre-trained CNN [14] classifier which is adapted/fine-tuned to our product classification task as described in Sect. 3) are input to greedy non-maximal suppression (greedy-NMS) [3] scheme. Consequently, the greedy-NMS provides the final output with detected products as shown in Fig. 8.

Note that we interchangeably use the words proposal, region and detection. The goal of greedy-NMS is to retain at most one detection per group of overlapping detection. The *intersection-over-union* (IoU) [10] between two regions defines the overlap amount (between those two regions). The overlapping group of any proposal H_z is a set of proposals which overlap with H_z by an amount more than a preset IoU value $IoUthresh$. Example group of overlapping detection can be seen in Fig. 8. The greedy-NMS retains at most one detection (having highest classification score) per overlapping group of regions. In other words, greedy-NMS eliminates all regions having lower classification scores and overlapping with a region with the higher score. Next section presents various experiments and their results.

3 Experiments

The proposed solution is implemented in python and tested in a computing system with the following specifications: 64 GB RAM, Intel Core i7-7700K CPU @ 4.2GHz×8 and TITAN XP GPU. The following paragraphs detail the experimental settings and implementation details of the methods.

Exemplar-driven Region Proposal In Stage-1 of our algorithm (see Sect. 2.1.1), we choose $ratioThreshold = 0.73$ and $matchThreshold = 0.45$ for matching the key-points between the products and rack through the following experiment. Initially, we assume that the two key-points can be considered as a match if their distance is lower than the 50% of maximum possible distance, i.e., lower than $matchThreshold \times 512$, where $matchThreshold = 0.5$. On the other hand, $ratioThreshold$ is initially set to 0.8 following [20]. With this initialization of the parameters and extensive experimentation with 180 product templates, we obtain the best result for $matchThreshold = 0.45$ and $ratioThreshold = 0.73$.

As mentioned in Sect. 2.1.1, $minimumPoints$ of the DBSCAN algorithm is always 3 and $maximumRadius$ of the same is empirically set to 60 pixels for clustering the matched key-points in the rack. We consider one estimated scale ($k = 1$) for generating region proposals. In Stage-2 of our algorithm (see Sect. 2.1.2), the values of $ratioThreshold$ and $matchThreshold$ parameters are same as these are in Stage-1. For clustering the key-points in rack, $minimumPoints = 1$ (as mentioned in Sect. 2.1.2) to obtain the exhaustive set of proposals and $maximumRadius = 60$ pixels to execute DBSCAN.

Classification of Regions Here, we successively discuss the standardization of data, data augmentation, training of pre-trained CNN or domain adaptation of CNN for classification of regions.

Data Standardization By design [27], the pytorch implementation of pre-trained ResNet-101 CNN model requires input images of size 224 × 224 × 3 pixels. That is, input images are color images having 3 RGB channels, each of size 224 × 224. Therefore, we have resized all the product images into the size of 224 × 224 × 3 pixels. We transform the product image into a fixed size of 224 × 224 × 3 pixels without altering the aspect ratio (see Fig. 9).

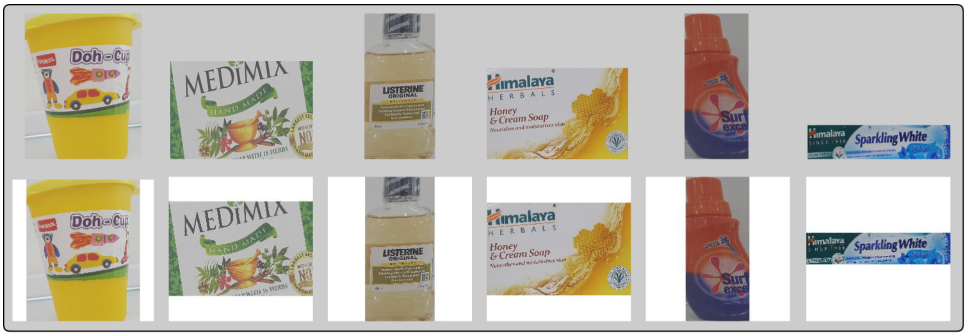


Fig. 9 Original (top row) versus standardized (bottom row) product images

A product image of $w \times h \times 3$ pixels is first resized to $224 \times 224\frac{h}{w} \times 3$ pixels if $w > h$, else $224\frac{w}{h} \times 224 \times 3$. The resized product image is then superimposed in a white frame of $224 \times 224 \times 3$ pixels such a way so that the smaller side of the image is aligned in the middle of the frame, while the other side perfectly fits the frame. Figure 9 demonstrates some examples of product images and corresponding transformed product images. Consequently, the transformed product image is standardized by normalizing each color channel R, G and B of the image. Assume C denotes any color channel R, G or B. Let C' be the normalized version of C such that $C' = \frac{C-\mu J}{\sigma}$, where μ and σ are the mean and standard deviation of all pixel intensities of C over all training samples, and J is the 224×224 *all-ones* matrix.

Data Augmentation The detection of products is a problem where only one template image is available per product class. But to train a learning-based scheme like CNN, we require a huge amount of training samples per class. Thus, using various photometric and geometric transformations, we generate $\sim 10^4$ training samples from the single template of a product. The training samples are augmented using three python libraries: keras,[1] augmentor,[2] and imgaug.[3] Considering supermarket like scenario, the synthesis process applies the following photometric transformations: random contrast adjustment, random brightness adjustment, noise (salt and pepper and Gaussian) addition and blurring (Gaussian, mean and median). Consequently the geometric transformations such as rotation, translation, shearing and distortion are applied on the photometrically transformed synthesized images. These augmented training samples are used for training the CNN.

Domain Adaptation of CNN The classification of region proposal into any one of the product classes uses PyTorch [26] implementation of the ResNet-101 CNN model [14]. The final layer (i.e., last fc layer) of the ResNet-101 network has 1000 nodes for 1000-way classification of ImageNet [1] dataset. For our problem, let a product dataset has η #product classes. In order to adapt the network to our task, the last fc layer of the ResNet-101 network is replaced by a newly introduced fc layer having η nodes. The weights of the new connections are initialized with random values drawn from $[-1, 1]$. Now the entire network is trained with our augmented training samples.

The training of the CNN is performed by calculating cross-entropy loss and optimizing the loss using mini-batch stochastic gradient descent (SGD) [8,30]. We uniformly sample 2^5 augmented images to form a mini-batch in each iteration of SGD. The SGD is started at a learning rate 0.01 and with a momentum of 0.9. After each 10 epoch, we update the learning rate by a factor of 0.1. However, the parameters of the network have been trained for 200 epoch. The adapted ResNet-101 propagates (in forward direction) the standardized product images through its layers and the output of last fc layer is then passed through the *softmax* [15,36] function to obtain the class probabilities for any region proposal. This classification strategy in association with our exemplar-driven region proposal (ERP) scheme is referred to as **ERP + CNN**.

Non-maximal Suppression The greedy non-maximal suppression (greedy-NMS) are performed on the proposals with the classification score above the threshold $scoreThresh = 0.5$. In greedy-NMS, we empirically choose $IoUthresh = 0.07$ for detecting products displayed on a rack. Next we present the results and comparisons.

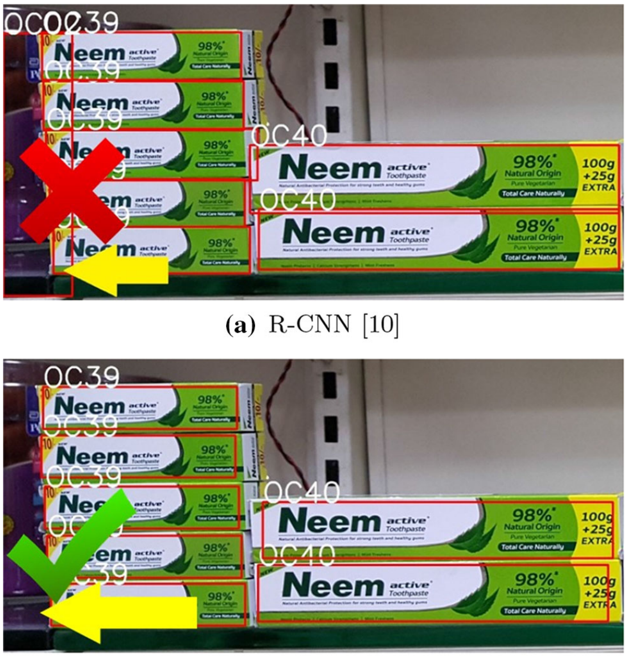
3.1 Results and analysis

Competing Methods In order to perform experimental analysis and a comparative study, we have implemented the competing methods in [4,6,10,21,23,28,32,37] which we have discussed in Sect. 1.1. The methods in [6,10,28,32] are referred to as R-CNN, U-PC, G-NMS and MLIC, respectively. The authors of [4,21,23,37] present more than one method. The histogram of oriented gradients (HOG)- and bag of words (BoW)-based schemes are implemented from [21], while color histogram matching (CHM)-based scheme is reproduced from [23]. In case of [37], we implement its best scheme S1. From [4], we implement both bag of

[1] https://github.com/keras-team/keras accessed as on 04/2020.

[2] https://github.com/mdbloice/Augmentor accessed on 04/2020.

[3] https://github.com/aleju/imgaug accessed as on 04/2020.



(a) R-CNN [10]

(b) ERP+CNN

Fig. 10 Example output of **a** exemplar-independent region proposal (in R-CNN) and **b proposed** exemplar-driven region proposal (in **ERP + CNN**) schemes. The red cross mark highlights the incorrect/false (see yellow arrow in **a**) detection by R-CNN, while that false detection is removed by our **ERP + CNN** (see the green tick mark in **b** pointed with yellow arrow) (colour figure online)

words (BoW) and deep neural network (DNN)-based methods which are referred to as GP (BoW) and GP (DNN), respectively.

Evaluation Indicators In [31], we notice that different evaluation indicators are used in different state-of-the-art methods for validating the solutions. Keeping retail context in mind, in this paper, the efficiency of the methods are evaluated by calculating F_1 score. This score implicitly encapsulates the standard metrics (for measuring the accuracy of object detectors) recall, precision or intersection-over-union (IoU). Let a product P is present in the rack I. If the center of any detected bounding box (i.e., detection) lies within P in the rack and the bounding box is detected as P, the count of TP (true positive) of the rack I is increased by 1. If the center of the detection lies within P in the rack, but it is not detected as P, the count of FP (false positive) of the rack I is increased by 1. Also if the center of a detection does not lie within any true product in I, the count of FP of the rack I is increased by 1. If there does not exist any detection whose center lie within the P in the rack, the count of FN (false negative) of the rack I is increased by 1. Given this, for the rack I, $F_1 = 2\frac{\text{recall precision}}{\text{recall+precision}}$, where recall $= \frac{\text{TP}}{\text{TP+FN}}$, precision $= \frac{\text{TP}}{\text{TP+FP}}$.

Datasets The experiments are carried out on one in-house and three publicly available benchmark datasets: GroZi [23], WebMarket [37] and Grocery Products [6]. The in-house dataset consists of six categories of products. In all these datasets, the products are captured in a controlled environment, while the racks are imaged in the wild. So the product images differ from rack images in scale, pose and illumination. Note that the benchmark datasets do not provide physical dimensions of the products. So we use context information of retail store that similar products and the products of similar shapes are put together for shopper's convenience. Because of the context, we can assume that the physical dimensions of the products displayed in a rack are almost similar. Consequently assume that the physical dimension is equivalent to the dimension in pixels of a product. Next we successively brief each dataset with the experimental results.

The **in-house dataset** is comprised of 457 rack images of 352 products captured in various supermarkets. The products are collected from six categories of products such as *breakfast cereals* (BC) (72 products and 151 racks), *deodorant* (DEO) (55 products and 100 racks), *lip care* (LC) (20 products and 80 racks), *oral care* (OC) (51 products and 30 racks), *personal wash* (PW) (82 products and 36 racks) and *mixed* (MIX) (72 products and 60 racks). Note that the dataset includes only one template image per product. In order to calculate the accuracy of different proposals, we manually annotate all the rack images labeling the products with tight rectangular bounding boxes.

Table 1 presents the average F_1 scores of various methods for all the racks of six categories of the dataset. Table 1 shows that our proposed ERP-based scheme **ERP** + **CNN** yields $\sim$ 4% higher F_1 score than the best among other competing methods for five categories (BC, LC, OC, PW, MIX). An example of the qualitative results of exemplar-independent region proposal-based scheme (in R-CNN) and our exemplar-driven region proposal-based scheme (in **ERP** + **CNN**) are compared in Fig. 10. In case of exemplar-independent R-CNN, as shown using yellow arrow in Fig. 10a, the false detection is marked by red cross. The green tick mark (pointed with yellow arrow) in Fig. 10(a) highlights that the proposed exemplar-driven region proposal scheme in **ERP** + **CNN** does not identify that false detection from background regions as objects.

Repeatability Test We also perform the repeatability test of the proposed exemplar-driven region proposal scheme in **ERP** + **CNN** and exemplar-independent region proposal scheme in R-CNN on the six categories of the in-house dataset. For each category of the dataset, we collect images of a same rack with variations in illumination, scale and pose and group them together. For each of such groups, we determine standard error of the mean (SEM) $= \frac{\sigma_{F1}}{\sqrt{\vartheta}}$ of the F_1 scores of rack images in the group, where σ_{F1} defines the standard deviation of F_1 scores of ϑ #rack images. Consequently, we find out average SEM over all such groups for

Table 1 Results of various methods on in-house dataset

Methods	F_1 score (%) on categories of in-house dataset					
	BC	DEO	LC	OC	PW	MIX
R-CNN [10]	82.04	83.76	87.99	79.72	88.05	73.16
G-NMS [32]	88.43	83.96	89.91	86.70	91.03	79.17
U-PC [28]	84.77	52.59	86.29	55.65	81.15	65.49
HOG [21]	62.00	28.52	49.37	28.73	44.06	50.62
BoW [21]	65.05	45.10	70.72	53.31	71.23	59.91
CHM [23]	48.04	33.56	60.12	30.77	36.40	44.74
GP (BoW) [4]	72.07	49.98	68.29	46.79	77.22	53.41
GP (DNN) [4]	82.12	55.49	82.55	51.32	87.64	61.98
MLIC [6]	64.45	50.08	54.91	40.23	59.97	48.76
S1 [37]	41.01	45.21	47.87	48.08	54.91	49.01
ERP + CNN	90.86	83.76	92.49	89.80	92.12	82.98

Table 2 Performances of various methods on benchmark datasets

Datasets	F_1 score (%) of the methods										
	R-CNN [10]	G-NMS [32]	U-PC [28]	HOG [21]	BoW [21]	CHM [23]	GP (BoW) [4]	GP (DNN) [4]	MLIC [6]	S1 [37]	ERP + CNN
Grocery products	78.99	80.21	76.20	58.11	59.91	51.2	74.34	73.09	59.07	58.39	81.05
WebMarket	72.01	75.50	67.79	43.03	55.15	52.81	65.59	71.13	53.33	49.19	78.76
GroZi	40.91	44.81	40.10	28.33	26.83	24.70	39.66	43.99	33.10	31.71	47.49

each category of the dataset. The lower the average SEM is, the higher repeatable the method is. In Fig. 11, the average SEM of the exemplar-independent and our exemplar-driven schemes are shown on six categories of in-house dataset. The figure infers that our exemplar-driven region proposal scheme is more repeatable than exemplar-independent region proposal scheme in detecting retail products.

The **Grocery Products dataset** [6] includes 680 rack images displaying 3235 products. The rack images display many similar yet nonidentical (i.e., fine-grained) products (see Fig. 1). The classification of fine-grained products is a challenge for this dataset. The dataset also provides the ground truth for the rack images defining bounding boxes of the products. Out of 680 rack images, we find 74 which comply with our assumptions on physical dimensions discussed in 3rd paragraph of this section. These 74 rack images display 184 products. The first row of Table 2 presents results of our proposed scheme and other competing methods on this dataset. The proposed **ERP + CNN** scheme outperforms R-CNN by ∼ 2% and yields better results than other competing methods.

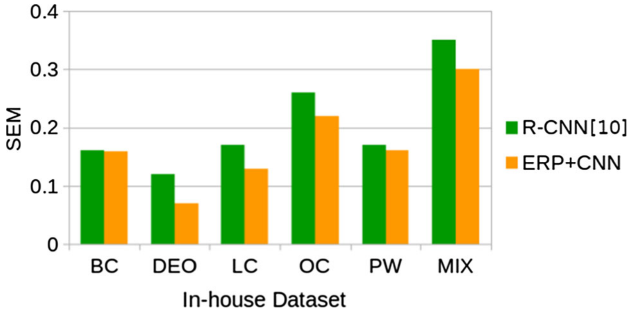


Fig. 11 Repeatability test: the average SEM of exemplar-independent (R-CNN) and exemplar-driven region proposal scheme (**ERP + CNN**) on the in-house dataset

The **WebMarket dataset** [37] comprises of 3153 images of racks collected from 18 shelves in a retail store. We select 36 rack images which meet our assumptions mentioned in 3rd paragraph of this section. These 36 racks include 155 products. Dataset provides the template images of only 100 products. The template images for rest 55 products are extracted from racks. The dataset does not provide annotations for the bounding boxes of the products in the rack images. We have manually annotated the rack images. The second row of Table 2 lists the results of various methods on this dataset. The proposed scheme **ERP + CNN** is found to be the winner by at least a margin of 3%.

The **GroZi dataset** [23] consists of 29 videos of racks displaying 120 products collected from various supermarkets. We select 28 frames, i.e., rack images from the videos satisfying our assumptions described in 3rd paragraph of this section. These 28 rack images display 20 products. All rack images are manually annotated with bounding boxes of the products. Out of 2 to 14 available templates per product, we

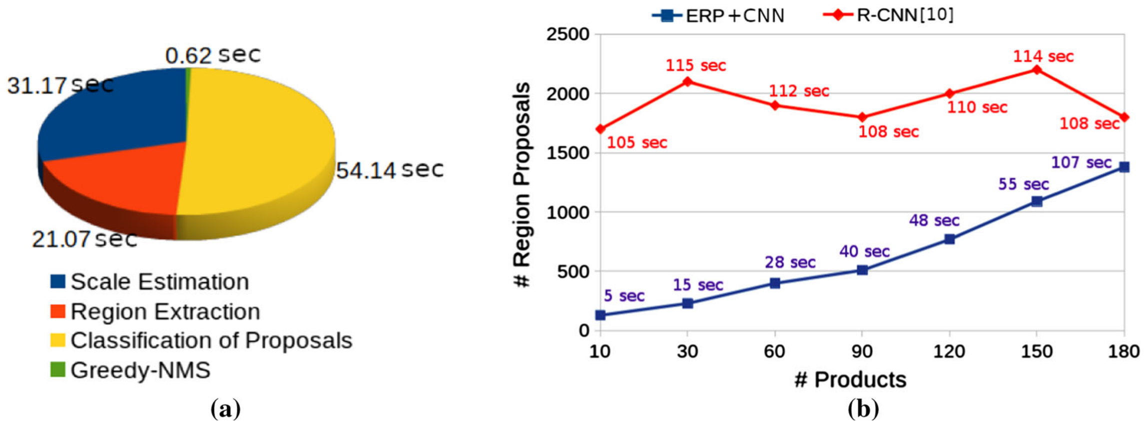


Fig. 12 **a** Execution time of different modules of the proposed **ERP** + **CNN** for processing one rack with 180 products in the dataset, **b** #products versus #region proposals (with corresponding execution time for processing one rack) generated by R-CNN and the proposed **ERP** + **CNN**

choose one template per product for the experiments. Since the template images are collected from web, the products in rack mostly differ from the templates. This is the primary challenge of this dataset. The performances of various methods on this dataset are compared in the third row of Table 2. Like previous two benchmark datasets, the proposed scheme shows its **superiority** over other methods for this dataset.

Time Analysis We also present an analysis on execution time of the proposed scheme **ERP** + **CNN**. Figure 12a shows time taken by each module of our scheme for detecting products displayed on a rack when the dataset contains 180 products. In that case, the entire process of detecting products on the rack takes about 107 s out of which scale estimation, region extraction, classification of region proposals and non-maximal suppression of regions consume 31.17, 21.07, 54.14 and 0.62 s, respectively. Thus, the proposed ERP, which includes scale estimation and region extraction procedures, takes about $31.17 + 21.07 = 52.24$ s. It can clearly be seen that the classification module of our scheme runs for a longer period of time than other modules. Our analysis finds that the data standardization process in the classification module is responsible for it.

However, the total execution time of our proposed scheme mainly depends on the number of region proposals. The number of proposals again depends on the number of products in a dataset. This is clearly observed in Fig. 12b. For example, running time of the proposed scheme is 40 s for processing 500 proposals in a rack populated with 90 products in the dataset. The same is 48 s for handling 800 proposals generated with 120 products in the dataset. We have tested our scheme for at most 180 products, and the corresponding execution times are plotted in Fig. 12b. We find that our exemplar-driven region proposal-based scheme generates much lesser number of proposals and takes lesser time than exemplar-independent region proposal-based scheme R-CNN. This is clearly shown in Fig. 12b. The next section concludes the paper.

4 Conclusions

We have introduced an exemplar-driven region proposal scheme. Since the physical dimensions of the products are available for in-house dataset, the scale between templates of the products and the image of rack is correctly estimated. For benchmark datasets, the context information of retail store is used for estimating the scale. The context information sometimes leads to inferior result compared to using physical dimension of product template. We plan to improve context information using a model similar to Markov process. There also exists some confusion between the choice of best (geometrically) fitted region proposal versus region proposal with higher classification score. We plan to pose this region selection as a cost optimization problem in our next work. Incorrect classification of (very) similar but nonidentical products is yet another challenge that needs attention.

Acknowledgements We would like to thank TCS Limited for partially supporting this work. We would also like to thank NVIDIA Corporation for donating one Titan XP GPU used for this research.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: IEEE Con-

ference on Computer Vision and Pattern Recognition, 2009. CVPR 2009, pp. 248–255. IEEE (2009)
2. Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. Kdd **96**, 226–231 (1996)
3. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1627–1645 (2010)
4. Franco, A., Maltoni, D., Papi, S.: Grocery product detection and recognition. Expert Syst. Appl. **81**, 163–176 (2017)
5. Freund, Y., Schapire, R., Abe, N.: A short introduction to boosting. J. Jpn. Soc. Artif. Intell. **14**(771–780), 1612 (1999)
6. George, M., Floerkemeier, C.: Recognizing products: a per-exemplar multi-label image classification approach. In: European Conference on Computer Vision, pp. 440–455. Springer (2014)
7. George, M., Mircic, D., Soros, G., Floerkemeier, C., Mattern, F.: Fine-grained product class recognition for assisted shopping. In: Proceedings of the IEEE International Conference on Computer Vision Workshops, pp. 154–162 (2015)
8. Ghassabeh, Y.A., Moghaddam, H.A.: Adaptive linear discriminant analysis for online feature extraction. Mach. Vis. Appl. **24**(4), 777–794 (2013)
9. Girshick, R.: Fast r-cnn. arXiv preprint arXiv:1504.08083 (2015)
10. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587 (2014)
11. Goldberg, D.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, New York (1989)
12. Harris, C., Stephens, M.: A combined corner and edge detector. In: Alvey Vision Conference, vol. 15, pp. 10–5244. Manchester, UK (1988)
13. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2980–2988. IEEE (2017)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
15. Hu, Y., Lu, M., Lu, X.: Driving behaviour recognition from still images by using multi-stream fusion cnn. Mach. Vis. Appl. **30**(5), 851–865 (2019)
16. Kejriwal, N., Garg, S., Kumar, S.: Product counting using images with application to robot-based retail stock assessment. In: 2015 IEEE International Conference on Technologies for Practical Robot Applications (TePRA), pp. 1–6. IEEE (2015)
17. Kim, J., Liu, C., Sha, F., Grauman, K.: Deformable spatial pyramid matching for fast dense correspondences. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2307–2314 (2013)
18. Leutenegger, S., Chli, M., Siegwart, R.Y.: Brisk: inary robust invariant scalable keypoints. In: 2011 IEEE International Conference on Computer Vision (ICCV), pp. 2548–2555. IEEE (2011)
19. Levenberg, K.: A method for the solution of certain non-linear problems in least squares. Q. Appl. Math. **2**(2), 164–168 (1944)
20. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
21. Marder, M., Harary, S., Ribak, A., Tzur, Y., Alpert, S., Tzadok, A.: Using image analytics to monitor retail store shelves. IBM J. Res. Dev. **59**(2/3), 3–1 (2015)
22. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. J. Soc. Ind. Appl. Math. **11**(2), 431–441 (1963)
23. Merler, M., Galleguillos, C., Belongie, S.: Recognizing groceries in situ using in vitro training data. In: IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR'07, pp. 1–8. IEEE (2007)
24. Mikolajczyk, K., Schmid, C.: Scale and affine invariant interest point detectors. Int. J. Comput. Vis. **60**(1), 63–86 (2004)
25. Mukherjee, D., Wu, Q.J., Wang, G.: A comparative experimental study of image feature detectors and descriptors. Mach. Vis. Appl. **26**(4), 443–466 (2015)
26. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: NIPS-W (2017)
27. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, R. Garnett (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019). http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
28. Ray, A., Kumar, N., Shaw, A., Mukherjee, D.P.: U-pc: unsupervised planogram compliance. In: European Conference on Computer Vision, pp. 598–613. Springer (2018)
29. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
30. Robbins, H., Monro, S.: A stochastic approximation method. The annals of mathematical statistics, pp. 400–407 (1951)
31. Santra, B., Mukherjee, D.P.: A comprehensive survey on computer vision based approaches for automatic identification of products in retail store. Image Vis. Comput. **86**, 45–63 (2019)
32. Santra, B., Shaw, A.K., Mukherjee, D.P.: Graph-based non-maximal suppression for detecting products on the rack. Pattern Recogn. Lett. **140**, 73–80 (2020). https://doi.org/10.1016/j.patrec.2020.09.023
33. Shen, J., Hao, X., Liang, Z., Liu, Y., Wang, W., Shao, L.: Real-time superpixel segmentation by dbscan clustering algorithm. IEEE Trans. Image Process. **25**(12), 5933–5942 (2016)
34. Winlock, T., Christiansen, E., Belongie, S.: Toward real-time grocery detection for the visually impaired. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 49–56. IEEE (2010)
35. Yao, B., Khosla, A., Fei-Fei, L.: Combining randomization and discrimination for fine-grained image categorization. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1577–1584. IEEE (2011)
36. Yu, J., Yow, K.C., Jeon, M.: Joint representation learning of appearance and motion for abnormal event detection. Mach. Vis. Appl. **29**(7), 1157–1170 (2018)
37. Zhang, Y., Wang, L., Hartley, R., Li, H.: Where's the weet-bix? In: Asian Conference on Computer Vision, pp. 800–810. Springer (2007)

Bikash Santra is currently a Senior Research Fellow at the Electronics and Communication Sciences Unit, Indian Statistical Institute, pursuing PhD in Computer Science. He received the M. Tech., M. Sc. and B. Sc. degrees from Indian Institute of Engineering Science and Technology (2016), West Bengal University of Technology (2010) and Vidyasagar University (2007), respectively. His primary research interest is in computer vision, deep learning and image processing. Mr. Santra authored many peer-reviewed research papers and has two patent applications pending. He had held project-linked scholar positions in the research projects at the Indian Institute of Engineering Science and Technology (2010-2011) and Indian Statistical Institute (2013-2016). He has been associated with the conferences like ICVGIP, ICAPR and NCVPRIPG in reviewing the papers and various organizational activities. He serves as a Reviewer of the IEEE Transactions on Image Processing, IEEE Access, and IET Image Processing. Mr. Santra is one of the recipients of 2013 Young IT Professional Award from the Computer Society of India. He is also one of the finalists of Qualcomm Innovation Fellowship (QIF) 2020, India.

Avishek Kumar Shaw is currently a Data Science Engineer at Tata Consultancy Services Limited. He received the B. Sc. (Honours) degree in Computer Science from the West Bengal State University (2013). His primary research interest is in computer vision, deep learning and image processing. Mr. Shaw has two peer-reviewed research papers. He holds a patent (USA and Australia) and also has two patent applications pending. Mr. Shaw serves as a Reviewer of IET Image Processing.

Professor Dipti Prasad Mukherjee is currently a Professor at the Electronics and Communication Sciences Unit, Indian Statistical Institute. His primary research interest is in computer vision and applications of machine learning. He has guided several Master's and PhD level dissertations. He has written two books on Computer Graphics and more than hundred thirty peer-reviewed research articles. He had held visiting faculty positions at the Oklahoma State University (1998-99), the University of Virginia (2002, 2013), Alcorn State University (2011), USA, the University of Alberta, Canada (2006, 2007, 2008, 2009, 2020), the University of Guanajuato, Mexico (2015, 2016), and the University of Aegean, Greece (2018). Prior to this, in 1992, Dr. Mukherjee is the recipient of the pre-doctoral UNDP fellowship at the Robotics Research Group, University of Oxford, and the UNESCO-CIMPA fellowship to INRIA, Sophia-Antipolis, France in 1991, 1993, 1995 and to ICTP, Trieste, Italy in 2000. In 2010, he had received Japan Society for the Promotion of Science (JSPS) Invitation fellowship to the Department of Radiology, Graduate School of Medicine, Osaka University and continued to visit Osaka University in the summer of 2012. He is a Fellow of the Computer Society of India, West Bengal Academy of Science and Technology and the Institution of Engineers (India). He had served as an Associate Editor of the IEEE Signal Processing Letters, SADHANA, Springer journal of the Indian Academy of Sciences, the IEEE Transactions on Image Processing and IET Image Processing journal.