International Journal of Machine Learning and Cybernetics (2022) 13:1847–1860
https://doi.org/10.1007/s13042-021-01492-5

ORIGINAL ARTICLE

Candidate region acquisition optimization algorithm based on multi-granularity data enhancement

Chen Dong[1] · Miao Duoqian[1] · Cairong Zhao[1] · Hailong Zhou[2]

Received: 25 May 2021 / Accepted: 6 December 2021 / Published online: 27 January 2022

Abstract
Given the deepening network hierarchy of deep learning, improving the accuracy of the candidate region acquisition algorithm can help save time and improve operational efficiency in subsequent work. Since the traditional methods overly rely on single-grain size, color and texture features of images, which can easily lead to candidate frames cutting off the foreground object when acquiring candidate regions, this paper proposes a multi-granularity selective search algorithm (MGSS) for candidate region acquisition by extracting the main features such as outline, texture and color of images with multiple grain sizes and improving the subgraph similarity calculation method.This paper mainly compares the performance of previous common algorithms on Pascal VOC 2012 and 2007 datasets, and the experiments show that the method used in this paper maintains the Mean Average Best Overlap (MABO) values of 0.909 and 0.890, which is 9.55% and 2.05% better than the Selective Search (SS)"Fast" and SS "Quality" results, respectively. The experiments show that both R-CNN and Fast R-CNN algorithms improve mAP (mean Average Precision) values by 1.5, 0.8 and 0.6 % with MGSS respectively, over with the traditional SS algorithm and RPN algorithm.

Keywords Multi-granularity · Object detection · Watershed algorithm · Perceptual hashing

1 Introduction

Accurately determining the candidate region proposal (RP) builds an important basic step in the field of image identification, object detection or image classification. If we can find the bounding box as close as possible to the ground truth at the initial time, we can provide good basic data for the later bounding box regression algorithm, and more importantly, we can greatly reduce the time cost of the program.

✉ Miao Duoqian
dqmiao@tongji.edu.cn

Chen Dong
1910691@tongji.edu.cn

Cairong Zhao
zhaocairong@tongji.edu.cn

Hailong Zhou
longzh@sit.edu.cn

[1] Tongji University, Zhixin Hall, Jiading Campus, 4800 Cao'an Road, Shanghai, China

[2] Information Technology Center, Shanghai Institute of Technology, Shanghai, China

Currently, the more commonly used algorithms for object detection of artificial intelligence, such as R-CNN [1, 2], SPP-Net [3, 4], and Fast R-CNN [5–7] are based on Selective Search (SS) algorithms, and SS also provides the preliminary work of Faster R-CNN [8] and R-FCN [9] a useful reference.

The above methods are classical object detection models. Object detection is to identify which objects are in the picture and the position (coordinate position) of the objects. The main purpose of object detection is to solve localization and recognition at the same time. It is multi-task learning with two output branches. One branch is used for image classification, that is, full connection + softmax to judge the object category. The difference from simple image classification is that there is another"background" class. The other branch is used to judge the object position, that is, to complete the regression task and output the position of four digital marker bounding boxes (such as the abscissa and ordinate of the center point and the length and width of the bounding box). The output result of this branch can only be used when the classification branch is judged not to be "background".

The following three methods [10] are commonly used to obtain RP: (1) silding window [11], (2)regular block [12], (3)selective search (SS) [13]. SS is still a relatively common and reliable method to obtain RP [10], but there are some shortcomings of the SS method that need to be compensated in terms of the image data used. SS is used to process the original image at a single granularity to obtain RP.

SS is not friendly to small and medium-sized objects in the picture, because SS method always uses the same granularity to extract features and calculate similarity. In this paper, we focus on overcoming the over-reliance of the original candidate region generation algorithm on pixel-level image information by enriching the image granularity for generating candidate regions. With the help of the new algorithm, The features of candidate region with different granularity can be extracted and the similarity can be calculated.

The color feature of an image is indeed the primary feature of an image, but multi-granularity color space should be chosen so that the color dissimilarity in the original image increases [13], and the image texture should be distinguished and extracted separately, excluding the effect of the color space. Secondly, in general, the difference of textures is much smaller than the difference of colors. Before extracting the image textures, it is necessary to process the original image to eliminate the color differences and to amplify the texture differences before performing the calculation of image texture similarity.

Using the original image with the watershed algorithm, we can obtain a coarse-grained image.We use coarse-grained images to obtain the outline features of the image. In the processed original image, the noise is removed first, and then the watershed algorithm is used to mark the outline of the foreground image, while the foreground and background levels of the image are separated, and finally RP of the complete foreground is outlined by using the "coherence" of the outline in the foreground image with the perceptual hashing algorithm.

The feature extraction method using the multi-granularity idea can take into account both the pixel-level details of the image and the highly abstract semantic information.

After using the MGSS algorithm, we obtained a smaller number of candidate regions with more accurate locations. We feed the newly generated candidate regions into the object detection model under the deep learning framework, which not only improves the accuracy but also saves the time of running the model. The MGSS method gives a good data base for objection detection models under deep learning and is a fundamental work after optimization. MGSS algorithm mainly calculates the similarity from three aspects: outline, color and texture. For specific operation details, see the main structure diagram and the algorithm description in **Algorithm 1** and Fig. 5.

2 Related work

After a long period of development, three main methods for acquiring regional proposals have gradually emerged in the field of computer vision [10].

(1) **Sliding window** is essentially an exhaustive method, using different scales and aspect ratios to enumerate all possible large and small blocks, and then send them for identification, and those with high probability of identification will stay [11].

The concept of sliding window is similar to convolution. It slides on the image by using a certain frame with fixed length and width , and the step size can be set arbitrarily. In the sliding process, each image block is classified to determine whether it contains objects. The trained classifier is used to classify each image block of the picture, so as to detect whether there are preset categories in the picture block. By using windows with different sizes and aspect ratios to search the whole picture completely, the target location can be found correctly in theory. The sliding window method needs a lot of sliding judgment on the whole map, and the calculation efficiency is relatively low. In addition, the size selection of sliding window needs human intervention, and the selection of length-width ratio is also related to the size of the object. Obviously, sliding windows produce a large number of redundant image blocks, resulting in a lot of redundant candidate regions, which takes up a lot of time and is not feasible in reality.

(2) **Regular blocks** are pruned on the basis of exhaustive method, and only fixed size and aspect ratio are selected. Exhaustive method is very effective in some specific application scenarios, such as the real application of Chinese character detection, because the Chinese characters are square and square, and the aspect ratio is mostly consistent, so it is a more appropriate choice to use rule block for region nomination. However, for ordinary target detection, rule blocks still need to access a lot of locations, with high complexity [12].

(3) **Selective search (SS)** From the perspective of machine learning, the accuracy of the previous methods is not satisfactory, so the core of the problem is how to

effectively remove redundant candidate regions. In fact, most of the redundant candidate regions overlap. Selective search uses this fact to merge adjacent overlapping regions from bottom to top, so as to reduce redundancy [13].

Firstly, Felzenszwalb segmentation algorithm proposed in the paper effective graph—based image segmentation is used to obtain the initial region R, and initialize the similarity measure set S. Then traverse the entire region set to get all adjacent region pairs sets (r_i, r_j). Then traverse all region pair sets (r_i, r_j), calculate the similarity measure of region r_i and region r_j, and add this similarity measure to S.

Then, when S is not empty, cycle processing is performed. In the loop, first find the corresponding region pair with the greatest similarity (r_i, r_j). Then, the region r_i and the region r_j are combined and recorded as r_t. Then delete the similarity measurement of other regions adjacent to region r_i and r_j from S. Then calculate the similarity measure between the region and its adjacent region and add it to the S. Finally, add the region r_t to R [14].

Faster R-CNN [8] uses a neural network called RPN (region proposal network) to generate regional proposals. RPN can be directly inserted into Fast R-CNN, so the whole detection framework of Faster R-CNN is end-to-end, which can be easily optimized globally. R-FCN [9] generates a position sensitive score map, which contains the relative position relationship information between different object types, and then uses PSROI pooling (position sensitive ROI pooling) to extract the regional features containing relative spatial position information, so as to realize the calculation and sharing in regional classification.

FPN [15] combines the feature maps of different layers in the neural network, so that the features of low layers contain high-level strong semantic information, and are predicted on the feature maps of different layers. The final detection results need to be fused with multiple prediction results on different layers. However, these methods will produce a large number of redundant candidate boxes and waste a lot of computing resources.

At present, SS is still a common and reliable method to obtain region proposal [10], but SS is not friendly to small and medium-sized objects in the picture, because SS method always uses the same granularity to extract features and calculate similarity. Meanwhile, in terms of image data, SS method has some defects that need to be remedied:

(1) Excessive dependence on single color space to determine image similarity;

(2) The processing of texture similarity is too rough;

(3) The outline of the object in the image is not considered;

(4) The calculation method of subgraph similarity is single.

3 Multi-Granularity Selective Search Algorithm

We creatively propose a coarse-grained outline feature of the original image. Compared with the fine-grained features of the image, the coarse-grained features contain more abstract semantic information, so they are more suitable for the candidate region generation algorithm. Also in this paper, we improve the extraction and calculation of color and texture features. Firstly, more multi-granularity color space is tried, and secondly, Laplace operator is used for the extraction of image texture while ignoring the effect of noise.

In this paper, for these reasons, and in order to reduce the number of bad candidate region proposal and ease the computational effort of the subsequent work, we propose a multi-granularity selective search algorithm (**MGSS** for short)that integrates the application of three features, color texture and outline, to obtain RP. With the help of multi granularity image, MGSS can mine different levels of detail features [16],which is more effective for small and medium-sized objects than SS method, so as to obtain more accurate region proposal positioning.

The MGSS method gives a good data base for objection detection models under deep learning and is a fundamental work after optimization. MGSS algorithm mainly calculates the similarity from three aspects: outline, color and texture. For specific operation details, see the main structure diagram and the algorithm description in **Algorithm 1** and Fig. 5.

Algorithm 1 Similarity Calculating Algorithm

Input: Images;
Output: Similarity of subgraphs and region proposals; Obtain initial regions $R = \{r_1, \ldots, r_n\}$ using [13];

1: **for** each neighbouring region pair (r_i, r_j) **do**
2: calculate $s(r_i, r_j)$:

$$
\begin{aligned}
&s_{\text{outlines}}(r_i, r_j) = \text{pHash}(watershed(r_i), watershed(r_j)) \\
&s_{\text{texture}}(r_i, r_j) = \sum_{k=1}^{n} \min(fine)\left(t_i^k, t_j^k\right) + \cdots + \sum_{k=1}^{n} \min(coarse)\left(t_i^k, t_j^k\right) \\
&s_{\text{color}}(r_i, r_j) = \sum_{k=1}^{n} \min(fine)\left(c_i^k, c_j^k\right) + \cdots + \sum_{k=1}^{n} \min(coarse)\left(c_i^k, c_j^k\right) \\
&s_{\text{fill}}(r_i, r_j) = 1 - \frac{\text{size}(BB_{ij}) - \text{size}(r_i) - \text{size}(r_j)}{\text{size}\ (im)} \\
&s_{\text{size}}(r_i, r_j) = 1 - \frac{\text{size}(r_i) + \text{size}(r_j)}{\text{size}\ (im)} \\
&s(r_i, r_j) = \sqrt{as_{\text{outlines}}^2(r_i, r_j) + bs_{\text{texture}}^2(r_i, r_j) + cs_{\text{color}}^2(r_i, r_j) + ds^2{}_{\text{fill}}(r_i, r_j) + es_{\text{size}}^2(r_i, r_j)} \\
&S = S \cup s(r_i, r_j)
\end{aligned}
$$

3: **end for**
4: **while** $S \neq \varnothing$ **do**
5: Get highest similarity $s(r_i, r_j) = \max(S)$
6: Merge corresponding regions $r_t = r_i \cup r_j$
7: Remove similarities regarding $r_i : S = S \backslash s(r_i, r_*)$
8: Remove similarities regarding $r_j : S = S \backslash s(r_*, r_j)$
9: Calculate similarity set S_t between r_t and its neighbours

$$
\begin{aligned}
S &= S \cup s_t(r_i, r_j) \\
R &= R \cup r_t
\end{aligned}
$$

10: **end while**
11: return S and R
12: Extract object location boxes L from all regions in R.

4 Coarse-grained images for outline feature

In this paper, "watershed algorithm" is used to obtain coarse-grained image feature, distinguish the outline of foreground objects in the image. The watershed algorithm is a classical algorithm for image segmentation, which is a mathematical morphological segmentation method based on topological theory [6] and is good at segmenting adherent objects. Watershed algorithm can be combined with distance transform to find "catchment basin" and "watershed boundary", so as to segment the image [17].

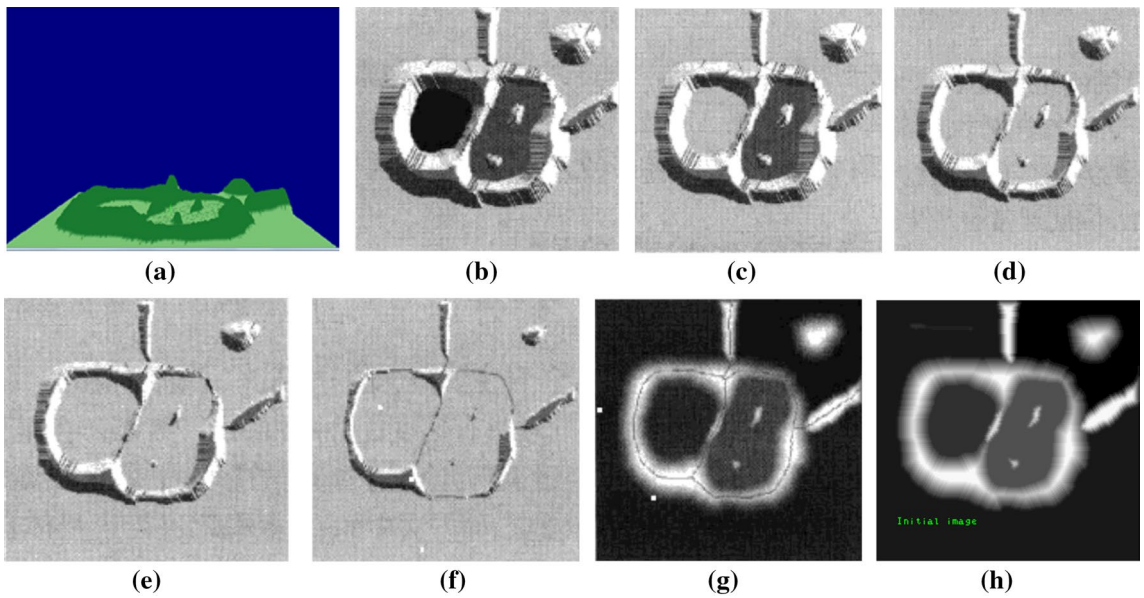

Fig. 1 Schematic diagram of watershed algorithm

4.1 Watershed Algorithm

At present, there are two well-known and widely used watershed segmentation algorithms: (1) Bottom-up algorithm to simulate flooding; (2) Top-down algorithm to simulate precipitation.This paper uses a bottom-up algorithm to simulate flooding.

The basic idea of the simulated flooding algorithm is to assume that a hole is made at the minimum of each area and allow water to gush out of the hole at a uniform rate of rise, flooding the entire terrain from low to high. The water will reach a point where only the top of each dam is visible at the waterline. The boundaries of these dams correspond to the dividing line of the watershed. Therefore, they are the (continuous) boundary lines extracted by the watershed algorithm [18–23].

As shown in Fig. 1, the first stage of the water inundation in Fig. (b), the flood is represented in light gray, covering the area corresponding to the dark background in the figure. In Fig. (c) and Fig. (d), we see water rising in the first and second catchment basins, respectively. As the water continues to rise, eventually the water will spill out of one basin into the other. As can be seen in Fig. 1, the water does overflow from the left basin to the right basin, and there is a "short dam" (made up of single pixels) between the two basins that prevents the water from coalescing at this level. As the water level rises, a longer dam is shown between the two converging basins, with another dam in the upper right corner, as shown in Fig. 1. This dam prevents water in the basins from coalescing with water corresponding to the background. This process continues until the maximum value of the water level is reached (corresponding to the maximum value of the gray level in the image). The last remaining part of the dam corresponds to the water splitting line, and this line is the result of the segmentation to be obtained.

The bottom-up simulation flooding process is a recursive process.

$$X_h = T_h(I) \tag{1}$$

$$\begin{aligned} &\forall h \in \left[h_{\min}, h_{\max} - 1\right] \\ &X_{h+1} = \min_{h+1} \cup C_{x_k}\left(X_h \cap X_{h+1}\right) \end{aligned} \tag{2}$$

where, Eq. (1) is the initial condition of the recursive process, which is the pixel with the minimum gray value in the image I; Eq. (2) is a recursive process. h denotes the range of gray value, $h_{\min}$ is the minimum value of gray value range, and $h_{\max}$ is the maximum value of gray value range. X_{h+1} is all the pixel points on the gray value $h+1$ i.e. elevation. The point min_{h+1} is the minimum value of the newly created basin; C_{x_k} is the basin where point x_k is located;$C_{x_k}\left(X_h \cap X_{h+1}\right)$ means the point set X_h intersects the point set X_{h+1} and is in the basin C_{x_h} where the point is located. Through this recursive process, all the pixels in the image are divided into basins, and finally, if a pixel point belongs to more than 2 basins at the same time, it is a point in the watershed.

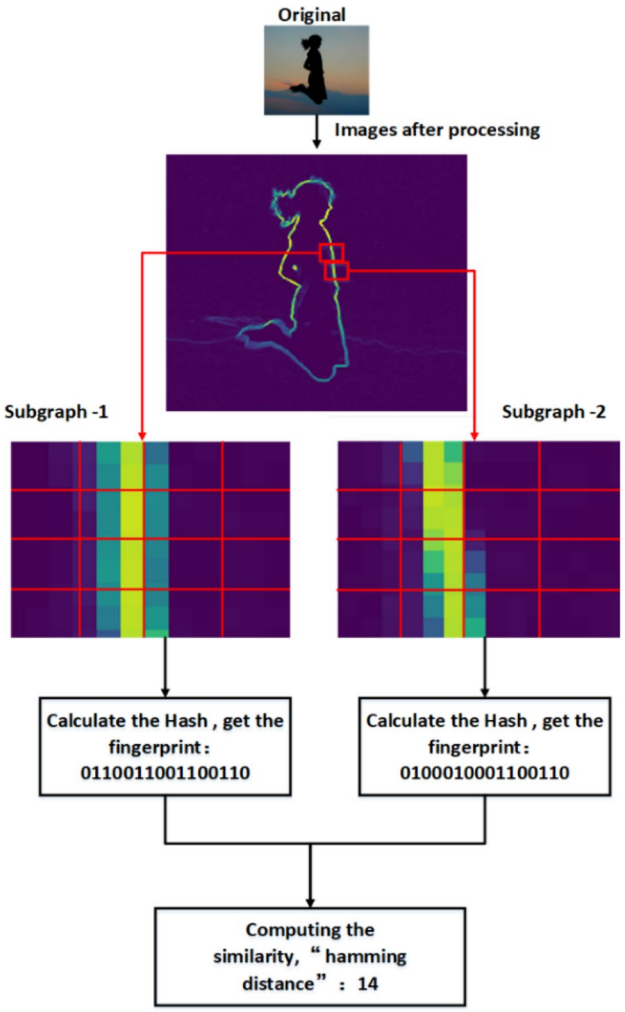

Fig. 2 Schematic diagram of outline similarity calculation

This process is equivalent to expanding the point set belonging to their own basin, circle by circle from the lowest point of each basin until some points belong to different basins at the same time. By linking all the points that belong to more than 2 basins at the same time, the segmentation line is found, and the segmentation of the image is completed.

4.2 Calculation of outline similarity

In this paper, we use the perceptual hashing algorithm to calculate the outline similarity of adjacent subgraphs.

The perceptual hash is not calculated in a strict way, but in a more relative way, because the "similarity" is a relative metric, so the calculation is fast. In this case, to compare the similarity of two images, we first calculate the

hash fingerprint of the two images, which is the 64-bit 0 or 1 value, and then calculate the number of different bits (Hamming distance). The larger the Hamming distance is, the bigger the difference between the two images. Generally speaking, a Hamming distance greater than ten indicates a completely different image [24–27].

In this paper, we mainly use the more robust pHash algorithm. The working process of pHash is as follows.

(1) Size reduction: pHash starts with small images, but images larger than 8×8, and 32×32 is the best. The purpose of this is to simplify the computation of the discrete cosine transform (DCT) rather than reducing the frequency.

(2) Simplify the color: Transform the picture into a grayscale image to further simplify the computation.

(3) Calculating DCT: The DCT transform of the image is calculated to obtain a 32×32 DCT coefficient matrix.

(4) Reduce DCT: Although the result of DCT is a matrix of 32×32 size, we just need to keep the 8×8 matrix in the upper left corner, which presents the lowest frequency in the picture.

(5) Calculate the mean value: Just like the mean hash, the mean value of the DCT is calculated.

(6) Calculate the hash value: This is the main step, according to the 8×8 DCT matrix, set the 64-bit hash value of 0 or 1, greater than or equal to the DCT average value is set to "1", less than the DCT average value is set to "0".

Combined together, they form a 64-bit integer, which is the fingerprint of this image. As long as the overall structure of the image remains unchanged, the hash result value remains the same, while avoiding the effects of gamma correction or color histogram adjustment. As with the mean hash, the pHash can also be compared using the Hamming distance, and Fig. 2 shows the comparison process of two subgraph fingerprints.

5 Color similarity of images

Color space is the theoretical basis for conducting color information research, which quantifies color from people's subjective feelings into concrete mathematical expressions and provides a strong basis for recording and representing color by computer.

Fig. 3 Multi granularity image

5.1 Multi-grain color space

The purpose of choosing color space is to make the color dissimilarity in the original picture increase, not to use as much color space as possible [13]. RGB, luv, lab, YUV and HSV color space are widely used at present,as shown in Fig. 3. By using a variety of color space, based on the original image, this paper obtains a multi granularity image.

Some of these color spaces divide colors into finer units (fine-grained) such as RGB and HSV, while others use a more general division (coarse-grained) such as Lab and YUV. Lab color spaces are based on the LMS space. The Lab color model consists of three elements.One element is luminance (L), and "a" and "b" are two color channels.

"a" includes colors ranging from dark green(low luminance value) to gray (medium luminance value) to bright pink (high luminance value); "b" is from bright blue (low luminance value) to gray (medium luminance value) to yellow (high luminance value). YUV is divided into three components, with"Y" indicating brightness (Luminance or Luma), which is the gray value;"U" and"V" are the

Table 1 Color space

Color space		Elements		
Fine-grained	RGB	Red	Green	Blue
	HSV	Hue	Saturation	Value
Coarse-grained	Lab	Luminance (L)	Color channel a	Color channel b
	YUV	Brightness (Luminance or Luma)	Color	Saturation

chrominance(or Chroma), which describes the color and saturation of an image. It is used to specify the color of a pixel. The color space in Table 1 is not all the color space used in this paper. Table 1 is only an example.

5.2 Calculation of color similarity

Through the five color spaces in Fig. 3, using the picture of"color type","color composition","color saturation" and the way to enrich the basic color, for the picture of Therefore, this paper uses a combination of the above color spaces, rather than using any of them. Finally, the histogram of 25 bins for each color channel of the same image in different color spaces is obtained using L1-norm normalization. In practice, a different number of bins can be used. Because each color space for the interpretation of the color dimension, there may be 2, 3, or 4 dimensions, so that each region can get a vector of 50, 75, or 100 dimensions , the color similarity between regions is calculated by the following formula.

$$S_c(r_i, r_j) = \sum_{k=1}^{n} \min\left(c_i^k, c_j^k\right) \tag{3}$$

Since $\{c_i^l, \ldots\ldots, c_i^n\}$ is the normalized value, the cumulative sum of the histogram of each color channel is 1.0, for example, the cumulative sum of three channels is 3.0, if the area c_i and the area c_j histogram are exactly the same, the maximum color similarity is 3.0 at this time, if not, as the cumulative sum takes the minimum value of the two area bin for the cumulative sum, when the histogram gap is bigger, the smaller the cumulative sum will be, that is the smaller the color similarity. The new region needs to be used in the region merging process to calculate its histogram, which is calculated by

$$C_t = \frac{\text{size}(r_i) \times C_i + \text{size}(r_i) \times C_j}{\text{size}(r_i) + \text{size}(r_i)} \tag{4}$$

0	1	0
1	-4	1
0	1	0

(a)

1	1	1
1	-8	1
1	1	1

(b)

0	-1	0
-1	4	-1
0	-1	0

(c)

-1	-1	-1
-1	8	-1
-1	-1	-1

(d)

Fig. 4 Schematic diagram of Laplace operator

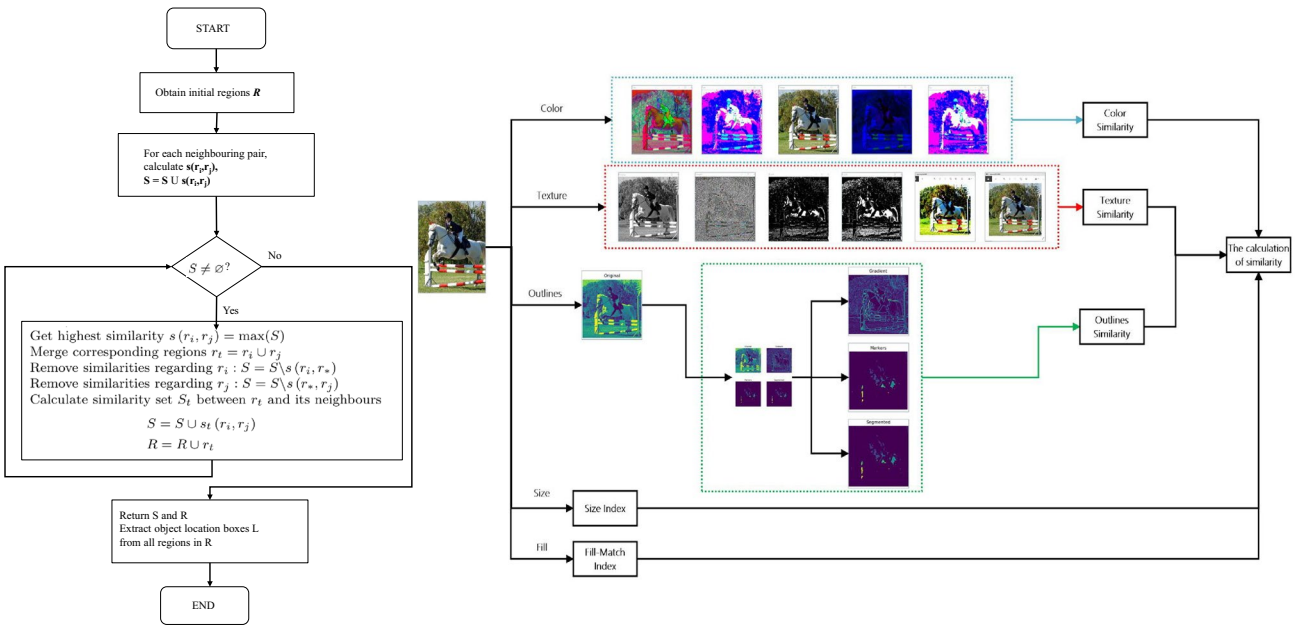

Fig. 5 Schematic diagram of similarity calculation (right) and the flowchart of full method (left)

6 Texture similarity of images

In a picture, the same color ("color type","color composition","color saturation") does not completely indicate that the two sub-pictures belong to the same object, but also from the texture of the object to analyze the judgment. For example, the texture of a green lawn is different from that of a green leaf, and the texture of a cat lying on a yellow sofa is different from that of the sofa. So you can't just use "color algorithms" for image processing. But it is far more difficult to capture the difference in texture than to find the difference in color, so this paper uses the "texture algorithm" with a lot of processing of the original image. In this paper, we use the Laplace operator to extract the texture features of the image.

6.1 Laplace operator

(1) For coarse-grained texture differences, the "gray map" is used directly to capture the texture differences.

(2) For fine-grained (subtle) texture differences, the Laplacian operator is first used to "enlarge".

The Laplace operator is an integral transform commonly used in engineering mathematics [28–36] in image processing and is also a common method for edge detection [37]. First, the Laplace operator is the simplest isotropic differential operator with rotational invariance. The Laplace transform of a two-dimensional image function is an isotropic second-order derivative defined as:

$$\text{Laplace}\,(f) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{5}$$

To be more suitable for digital image processing, the equation is expressed in discrete form as:

$$\begin{aligned}\nabla^2 f = &[f(x+1,y) + f(x-1,y) \\ &+ f(x,y+1) + f(x,y-1)] \\ &- 4f(x,y)]\end{aligned} \tag{6}$$

However, this paper uses a form of template for more convenient programming use, as shown in Fig. 4.

Like the gradient operator, the Laplace operator also enhances the noise in the image, and sometimes when edge detection is performed with the Laplace operator, the image can be smoothed first. From the template form, it is easy to see that if a bright spot appears in a darker region of the image, then using the Laplace operator will make that bright spot brighter. Since the edges in an image are those areas where the grayscale jumps, the Laplace sharpening template is useful in edge detection. The general enhancement technique is difficult for steep edges and slowly changing edges to determine the location of their edge lines. This operator, however, can be used to determine the over-zero point between the positive and negative peaks of the quadratic differential, and is more sensitive to isolated points or endpoints, so it is especially suitable for situations where the purpose is to highlight isolated points, isolated lines or line endpoints in an image. Schematic diagram of similarity calculation and the flowchart of full method are showed in Fig. 5.

6.2 Pre-processing for image textures

The function of image sharpening is to make the gray contrast enhanced, thus making the blurred image clearer. The essence of image blurring is that the image is subjected to averaging or integration operations, so inverse operations can be performed on the image, such as differential operations can highlight image details and make the image sharper. Since Laplace is a differential operator, its application can enhance the areas of sudden gray changes in the image and diminish the areas of slow gray changes. Therefore, the sharpening process can select the Laplace operator to process the original image to produce an image describing the abrupt gray scale changes, and then superimpose the Laplace image with the original image to produce a sharpened image. The basic method of Laplace sharpening can be expressed by the following equation. MCF means mask center factor.

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y), & \text{MCF} < 0 \\ f(x,y) + \nabla^2 f(x,y), & \text{MCF} > 0 \end{cases} \tag{7}$$

This simple sharpening method produces the effect of Laplace sharpening processing while preserving the background information. The original image is superimposed on the processing result of Laplace transform, and the final result is to highlight the small detail information in the image while preserving the image background. For the extraction of image texture, we also need to modify the contrast and sharpness of the image. In order to obtain the image texture more accurately, this paper turns on the parameters of contrast and sharpness to the maximum by default. In general, it is not possible to modify the hue and color saturation of the image. Contrast and sharpness modifications are modifications of a relative nature, and these operations can be the increase of the phase anisotropy of the individual colors in the picture. The modification of hue and color saturation

Table 2 Experimental environment and platform

Kernel version	Linux 4.15.0;gcc version 5.4.0
Language	Python 3.7.7
Main package	Pytorch 1.4.0, torchvision 0.5.0
CPU	40Intel(R) Xeon(R)
GPU	Tesla V100 PCIe 32GB

Table 3 Comparison of various methods MABO

Method	MABO	#Windows
Alexe [38, 39]	0.694 ± 0.111	1,853
Endress [40]	0.791 ± 0.082	790
Felzenszwalb [41]	0.829 ± 0.052	100,352 per class
SS-fast [13]	0.804 ± 0.046	2,134
SS-quality [13]	0.879 ± 0.039	10,097
Ours (VOC 2012)	0.909 ± 0.059	250 per class
Ours (VOC 2007)	0.890 ± 0.080	180 per class

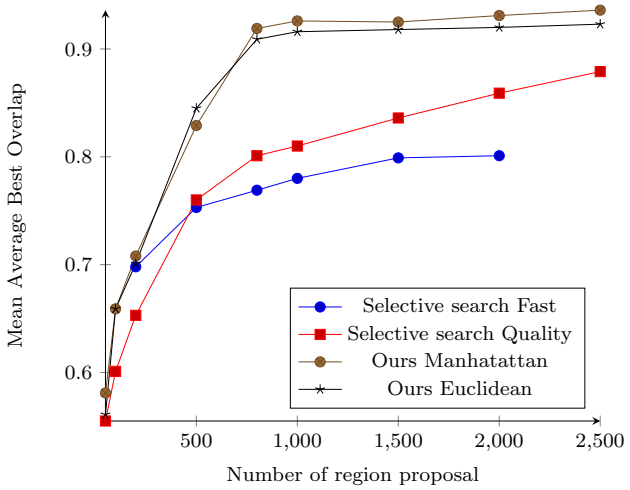

Fig. 6 MABO changes of various algorithms

is an absolute modification, which makes the parameters of each different but exhaustive color converge, but is not conducive to the extraction of picture textures.

6.3 Calculation of texture similarity

A similar processing scheme was used in the above regarding the color and texture of the images. Here, the textures are treated with SIFT-Like features. The Gaussian Derivative of the variance ($\delta = 1$) is calculated for eight different directions of each color channel, and the histogram of ten bins of each direction of each color channel of the image is obtained using L1-norm normalization, which is also analyzed on a case-by-case basis, so that a 240 ($10 \times 8 \times 3$) dimensional vector can be obtained. The texture similarity between regions is calculated in a similar way as the color similarity, and the texture features of the new region after merging are calculated in the same way as the color features:

$$S_t(r_i, r_j) = \sum_{k=1}^{n} \min\left(t_i^k, t_j^k\right) \tag{8}$$

7 Experiments and analysis

Because of the similarity calculation process, there are many dimensions and different ways to measure them, so the results in the final similarity are normalized. In this paper, we mainly use Minkowski Distance as the measure, and see **Algorithm 1** for more details. For experimental environment and platform, please refer to Table 2.

7.1 Evaluation and comparison and evaluation of experimental results

In this paper, we use the overlap ratio average best overlap (ABO) between bounding boxes and ground truth to evaluate the experimental effect. For each fixed category C, each true case (ground truth) is denoted as $g_i^c \in G^c$, such that each value l_j in the computed location hypothesis L , then the formula of ABO is expressed as:

$$\text{ABO} = \frac{1}{|G^c|} \sum_{g_i \in G} \max_{l_j \le L} Overlap\left(g_i^c, L_j\right) \tag{9}$$

Table 4 Difference of ABO values in different categories

Category	Manhattan	Euclidean	Chebyshev	Category	Manhattan	Euclidean	Chebyshev
Aeroplane	**0.830**	**0.870**	**0.899**	Diningtable	0.852	0.852	0.852
Bicycle	0.740	0.790	0.740	Dog	0.891	0.843	0.901
Bird	**0.916**	**0.856**	**0.936**	Horse	0.752	0.768	0.852
Boat	0.763	0.826	0.763	Motorbike	0.842	0.842	0.833
Bottle	0.884	0.784	0.764	**Person**	**0.985**	**0.965**	**0.935**
Bus	0.874	0.874	0.874	Pottedplant	0.772	0.802	0.792
Car	0.785	0.749	0.785	Sheep	0.876	0.749	0.880
Cat	0.788	0.766	0.729	Sofa	0.753	0.753	0.833
Chair	0.667	0.767	0.567	Train	0.840	0.699	0.640
Cow	0.873	0.873	0.873	**tv/monitor**	**0.954**	**0.909**	**0.912**

Bold values mean the results of these categories are significantly higher than others

$$\text{Overlap}\left(g_i^c, L_j\right) = \frac{\text{area}\left(g_i^c\right) \cap \text{area}\left(l_j\right)}{\text{area}\left(g_i^c\right) \cup \text{area}\left(l_j\right)} \tag{10}$$

The above results are given for one category of ABO, and for the performance evaluation under all categories, it is natural to use (mean average best overlap MABO, namely average value of all categories' ABO) to evaluate. According to Uijlings et al. [13], the performance metrics of MABO for diversity strategy combinations are better than those evaluated for singularity strategies, so this paper uses diversity strategy combinations.

In this paper, we mainly compare the performance of previous algorithms on Pascal VOC 2012 and Pascal VOC 2007 datasets. As shown in Table 3, this paper compares MABO and window numbers of our own method with several mainstream methods, and we can see that the IoU index of the method used in this paper remains around 0.900 in MABO, which is 10% and 3% better than the SS "Fast" and SS "Quality" results, respectively. However, the method in this paper has a fluctuation of 5.9% and 8.0% in MABO value, which is about 1.3% and 4.4% higher than SS "Fast" and SS "Quality" results, respectively.

Our method reduces the number of windows significantly after adding constraints. And RPN method generates a large number of alternative candidate regions randomly and then performs screening. Other methods did not perform a large intensity screening, so RPN does not participate in the comparison in Table 3.

Figure 6 shows the results of MABO with SS "Fast" and SS"Quality" and MGSS under Euclidean Distance and Manhattan Distance calculation methods used in this paper. It is clear that MABO values of all four methods increase as the number of candidate regions increases [41]. By analyzing the graphs, we can find that MABO values of each algorithm are increasing as the number of candidate regions increases, but MABO of MGSS in this paper reaches stability faster and stays around 0.900 when the number of candidate regions is 1000, while MABO value of the SS "Quality" method increases at around 0.900 when the number of candidate regions is 2000. The SS"Quality" method is stable at 0.880 when the number of candidate regions is 2000. The convergence of the SS method is slower than that of the MGSS method. It can be seen that this method not only improves the value of MABO, but also reduces the computation time cost. We can clearly get that MGSS improves FPS and saves time cost from Table 7.

As shown in Table 4, ABO values are counted by category, and it is obvious that there are great differences in ABO, because of different categories. Combining the three charts below, it is clear that ABO and AP values for bird, person, aerolpane, and TV/monter categories are significantly higher than those of other categories.

7.2 Comparison of object detection effects

In this paper, we also designed a comparison test in which the SS method, RPN method and MGSS method were placed in the corresponding object detection algorithm to compare the final result of mAP (mean Average Precision) values. As shown in Table 5, the number of candidate regions generated by our method decreases significantly compared to that of SS and RPN methods. RPN method and MGSS use different candidate region screening criteria in the training and validation time, and use more stringent criteria in testing phase, which is significantly different from SS method. In the final result comparison, mAP values of MGSS method achieved a 0.6 % improvement compared to RPN+ZF method. The MGSS is more adept at identifying classes with complex backgrounds or hierarchical complexity compared to the RPN method in Fig. 7.

7.3 Experimental analysis

As can be seen from Table 6, the final results mAP and classification are compared between the two algorithms R-CNN and Fast R-CNN using the traditional SS algorithm and MGSS of this paper, respectively. In terms of mAP values, R-CNN and Fast R-CNN algorithms improve 1.5 and 0.8 %, respectively, over the traditional SS algorithm using MGSS algorithm. From the categorical AP values, most of the categories have different degrees of improvement in detection, especially the background categories such as chair, table, and sofa have obvious and gratifying improvement. This indicates that the outlines calculation method added to the method in this paper plays a significant role. The outlines calculation method can help the candidate region acquisition algorithm to capture the external outlines of object and prevent the potential error of object being cut by the candidate boxes.

In Table 7, we can see running time for SS, MGSS and RPN used by R-CNN , Fast R-CNN algorithms during training and validation time. The MGSS method can significantly reduce the running time of R-CNN and Fast R-CNN algorithms, and also has a more significant reduction effect relative to RPN. MGSS can also significantly improve the FPS relative to SS and RPN.

The MGSS in this paper is an improvement over the RPN method in mAP values, but there are significant differences in AP values of different categories. The MGSS is able to identify categories with complex background or hierarchical complexity more accurately than RPN method. As shown in Table 6, in the categories of cat, boat, dog and sofa, this MGSS method performs better than RPN method. The MGSS is based on the processing mechanism of hierarchical merging, and RPN method is the selection of different sizes and proportions of anchors on the feature map. The

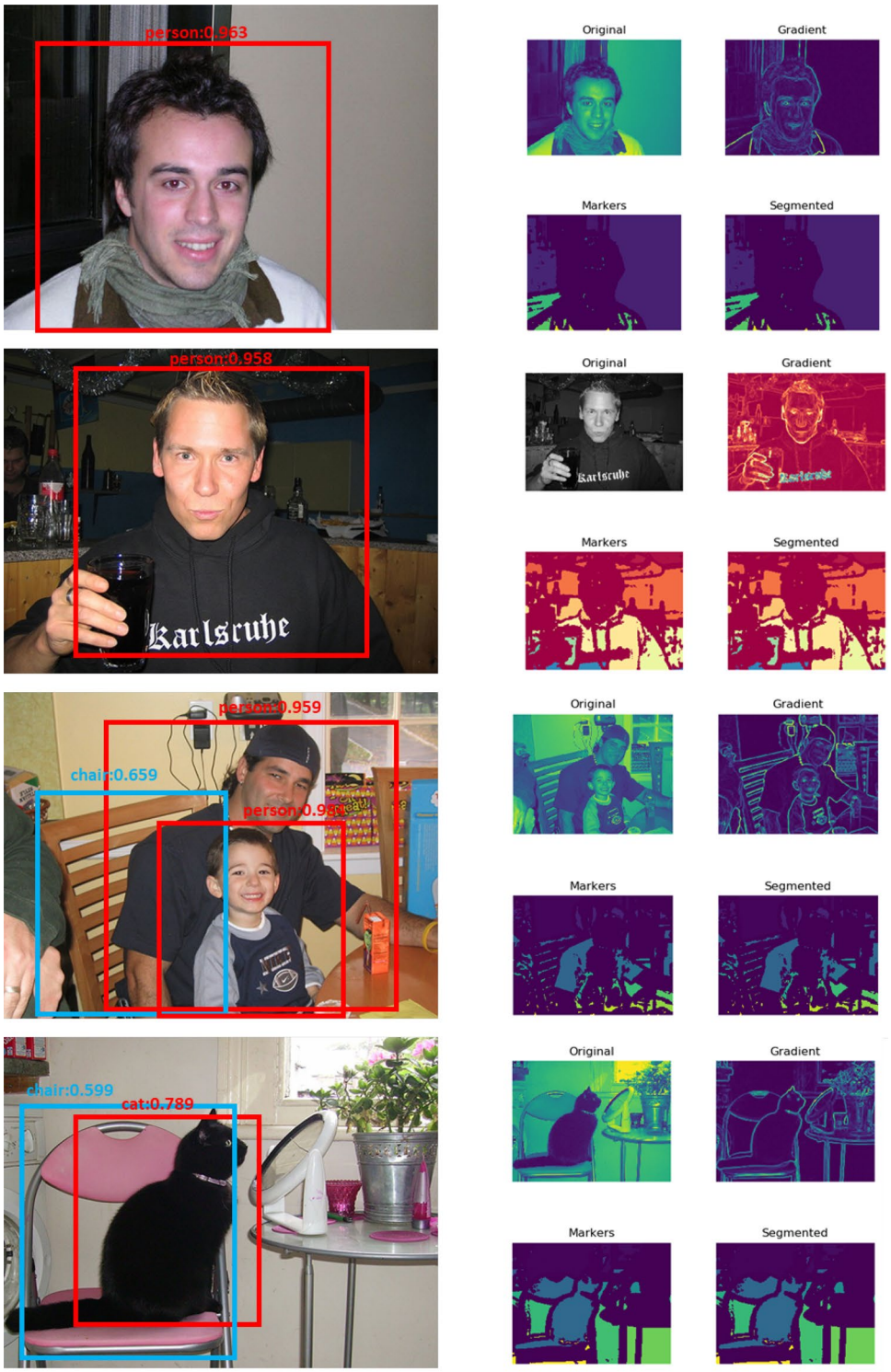

Fig. 7 Difference of ABO scores of different types of images. It is used to show the outline with different granularity and the final effect of MGSS method

MGSS in this paper is more adept at acquiring categories with overlapping object regions compared to RPN.

Through Tables 4, 5, 6, it can be seen that the MGSS in this paper can more accurately find the region proposal that is closer to ground truth, thus providing a better basic data for subsequent image recognition, image classification or object detection. However, Tables 4, 5, 6 show that MGSS in this paper has obvious differences in the ABO values of different categories. We can see that ABO values of aeroplane,

Table 5 Comparison of candidate region generation results of SS, RPN and MGSS method

Train-time		Test-time		mAP(%)
Method	Boxes	Method	Boxes	
SS [13]	~ 2000	SS	~ 2000	58.7
RPN+ZF [8]	2000	RPN+ZF	~ 300	59.9
Ours	~ 300	Ours	~ 180	**60.5**
Ours	~ 300	Ours	~ 150	59.9

[1]The datasets are PASCAL VOC2012 and PASCAL VOC2007; the setting of RPN in the comparison experiment is "3 scales, 3 ratios"; "~" means approximately in Table 5

[2]In the comparative experiment, this paper only uses MGSS method to replace the original candidate region acquisition algorithm

Table 7 Comparison 2 of SS, MGSS and RPN used by R-CNN , Fast R-CNN and Faster R-CNN algorithm

Method	Time for training (hours)	Time for validation (hours)	FPS
R-CNN + SS	49.6	0.56	5.6
FR-CNN + SS	48.2	0.50	5.9
FR-CNN + RPN	36.4	0.44	7.5
R-CNN + ours	42.1	0.49	6.5
FR-CNN + ours	32.0	0.40	8.2

[1] *FPS* represents the number of images that can be processed per second

Table 6 Comparison 1 of SS, MGSS and RPN used by R-CNN , Fast R-CNN and TSPOA+Caps algorithm

Method	Aerop	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow
R-CNN+SS	73.9	72.3	62.5	51.5	44.4	74.4	73.0	74.4	42.3	73.6
FR-CNN+SS	74.5	78.3	69.3	53.2	36.6	77.3	78.2	82.0	40.7	72.7
FR-CNN+RPN	74.1	77.2	67.7	53.9	**51.0**	75.1	79.2	78.9	**50.7**	78.0
TSPOA + caps [42]	79.5	84.0	**75.3**	61.9	48.7	82.7	88.5	89.7	47.9	78.4
R-CNN + ours	75.6	71.8	65.9	53.4	48.0	78.4	73.9	75.6	44.9	73.6
FR-CNN+ours	75.5	79.2	71.8	55.4	44.5	77.3	78.4	82.9	41.0	73.6
TSPOA + caps + ours	**80.9**	**87.1**	**75.3**	**62.1**	49.4	**87.0**	**89.9**	**91.4**	47.9	**79.1**

Method	Table	Dog	Horse	Mbike	Person	Plant	Sheep	Sofa	Train	Tv	mAP
R-CNN + SS	57.7	70.3	74.6	74.3	54.2	34.0	56.4	56.4	67.9	73.5	63.1
FR-CNN + SS	67.9	79.6	79.2	73	69	30.1	65.4	70.2	75.8	65.8	66.9
FR-CNN + RPN	61.1	79.1	**81.9**	72.2	**75.9**	37.2	71.4	62.5	77.4	66.4	68.5
TSPOA + caps	70.0	75.5	76.1	73.5	69.6	36.5	64.9	72.0	76.7	66.5	75.0
R-CNN +ours	57.8	75.3	75	**76.2**	58.2	35.1	54.4	55.9	69	73.6	64.6
FR-CNN + ours	68.2	79.5	70.2	73.5	69.6	36.5	64.9	72.0	76.7	66.5	68.7
TSPOA + caps + ours	**72.7**	**79.6**	76.8	75.1	72.9	**46.9**	**71.5**	**78.0**	**78.9**	**78.5**	**78.4**

Bold values mean the best performance

[1] Notes: FR-CNN + ours = fast R-CNN + ours, FR-CNN + RPN = faster R-CNN; TSPOA + Caps = TSPOANet + CapsNet; aerop = aeroplane; table = diningtable; mbike = motorbike; plant = pottedplane; tv = tv/monitor, test datasets are PASCAL VOC2012 and PASCAL VOC2007; mAP = mAP(%)in Table 6.

[2]In the comparative experiment, this paper only uses MGSS method to replace the original candidate region acquisition algorithm in Table 6.

bird, person, and TV/moniter are significantly higher than those of all categories of MABO and other categories. In particular, ABO score for person is significantly higher than MABO, which is mainly due to the variation in the object profile metric quoted in this paper.

8 Conclusion

This paper focuses on the acquisition method of candidate regions at the basic level, and creatively enriches the acquisition method of candidate region for object detection, MGSS, which improves 9.55% and 2.05% compared with the results of SS "Fast" and SS "Quality", respectively. We also verify that mAP values of R-CNN and Fast R-CNN algorithms improve by 1.5 and 0.8 %, respectively, compared with the traditional SS algorithm. The MGSS is more adept at identifying classes with complex backgrounds or hierarchical complexity compared to the RPN method.

Acknowledgements This research was supported in part by the National Natural Science Foundation of China Grant Nos. 61976158 and 62006172.

References

1. Girshick R, Donahue J, Darrell T, Malik J (2014) 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
2. Zhang N, Donahue J, Girshick R, Darrell T (2014) Part-based r-cnns for fine-grained category detection. In: European conference on computer vision. Springer, pp 834–849
3. He K, Zhang X, Ren S, Sun J (2015) Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans Pattern Anal Mach Intell 37(9):1904–1916
4. He K, Sun J, Zhang X, Ren S (2016) Spatial pyramid pooling networks for image processing
5. Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 1440–1448
6. Li J, Liang X, Shen S, Xu T, Feng J, Yan S (2017) Scale-aware fast r-cnn for pedestrian detection. IEEE Trans Multimed 20(4):985–996
7. Zhao ZQ, Bian H, Hu D, Cheng W, Glotin H (2017) Pedestrian detection based on fast r-cnn and batch normalization. In: International conference on intelligent computing. Springer, pp 735–746
8. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: Towards real-time object detection with region proposal networks. arXiv preprint arXiv:1506.01497
9. Dai J, Li Y, He K, Sun J (2016) R-fcn: Object detection via region-based fully convolutional networks. arXiv preprint arXiv:1605.06409
10. Hosang J, Benenson R, Dollár P, Schiele B (2015) What makes for effective detection proposals? IEEE Trans Pattern Anal Mach Intell 38(4):814–830
11. Pang Y, Cao J, Li X (2016) Learning sampling distributions for efficient object detection. IEEE Trans Cybernet 47(1):117–129
12. Wang X, Liang C, Chen J (2016) Multi-pedestrian detection from effective proposal in crowd scene. In: Proceedings of the international conference on internet multimedia computing and service, pp 156–159
13. Uijlings JR, Van De Sande KE, Gevers T, Smeulders AW (2013) Selective search for object recognition. Int J Comput Visi 104(2):154–171
14. Felzenszwalb PF, Huttenlocher DP (2004) Efficient graph-based image segmentation. Int J Comput Vis 59(2):167–181
15. Lin T-Y, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2117–2125
16. Yao Y (2013) Granular computing and sequential three-way decisions. In: International conference on rough sets and knowledge technology. Springer, pp 16–27
17. Xiaodong Y, Duoqian M, Caiming Z (2010) Color image segmentation method based on roughness metric. Automa J 36(6):807–816
18. Chahine C, Vachier-Lagorre C, Chenoune Y, El Berbari R, El Fawal Z, Petit E (2017) Information fusion for unsupervised image segmentation using stochastic watershed and hessian matrix. IET Image Process 12(4):525–531
19. Noyel G, Angulo J, Jeulin D (2007) Random germs and stochastic watershed for unsupervised multispectral image segmentation. In: International conference on knowledge-based and intelligent information and engineering systems. Springer, pp 17–24
20. Chahine C, El Berbari R, Lagorre C, Nakib A, Petit E (2015) Evidence theory for image segmentation using information from stochastic watershed and hessian filtering. In: 2015 International conference on systems, signals and image processing (IWSSIP).1em plus 0.5em minus 0.4emIEEE, pp 141–144
21. Nouri M, Khezeli A, Ramezani A, Ebrahimi A (2012) A dynamic chaotic hash function based upon circle chord methods. In: 6th International symposium on telecommunications (IST). IEEE, pp 1044–1049
22. Liu Y, Yan P, Xia R et al (2016) Fp-cnnh: a fast image hashing algorithm based on deep convolutional neural network. Comput Sci 43(9):39–51
23. Qu W, Wang D, Feng S, Zhang Y, Yu G (2017) A novel cross-modal hashing algorithm based on multimodal deep learning. Sci China Inf Sci 60(9):092104
24. Liu X, Zhang Q, Luan R, Yu F, (2013) Applications of perceptual hash algorithm in agriculture images. In: 2013 6th International congress on image and signal processing (CISP), vol 2.1. IEEE, pp 698–702
25. Ruchay A, Kober V, Yavtushenko E (2017) Fast perceptual image hash based on cascade algorithm. In: Applications of digital image processing XL, vol 10396. International Society for Optics and Photonics, p 1039625
26. Yuan J, Xu D, Xiong H-C, Li Z-Y (2016) A novel object tracking algorithm based on enhanced perception hash and online template matching. In: 2016 12th International conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD). IEEE, pp 494–499
27. Imasaki K, Dandamudi S (2002) An adaptive hash join algorithm on a network of workstations. In: Proceedings 16th international parallel and distributed processing symposium. IEEE, p 8
28. Raposo CA, Ribeiro J, Cattai A (2018) Global solution for a thermoelastic system with p-laplacian. Appl Math Lett 86:119–125
29. Alzaid A, Kim JS, Proschan F (1991) Laplace ordering and its applications. J Appl Prob 28:116–130
30. Alves CO, de Lima RN, Nóbrega AB (2018) Bifurcation properties for a class of fractional laplacian equations in. Math Nachr 291(14–15):2125–2144

31. Merris R (1994) Laplacian matrices of graphs: a survey. Linear Algebra Its Appl 197:143–176
32. Zhang X-D (2011) The laplacian eigenvalues of graphs: a survey. arXiv preprint arXiv:1111.2897
33. Li J-S, Zhang X-D (1998) On the laplacian eigenvalues of a graph. Linear Algebra Its Appl 285(1–3):305–307
34. Pirzada S, Ganie HA (2015) On the laplacian eigenvalues of a graph and laplacian energy. Linear Algebra Its Appl 486:454–468
35. Gutman I, Zhou B (2006) Laplacian energy of a graph. Linear Algebra Its Appl 414(1):29–37
36. Liu Y, Wu B (2010) Some results on the laplace energy of graphs (English). J East China Normal Univ (Natural Sciences Edition) 1:161–171
37. Bandeira AS (2018) Random laplacian matrices and convex relaxations. Found Comput Math 18(2):345–379
38. Pan H, Wang B, Jiang H (2015) "Deep learning for object saliency detection and image segmentation," arXiv preprint arXiv:1505.01173
39. Alexe B, Deselaers T, Ferrari V (2012) Measuring the objectness of image windows. IEEE Trans Pattern Anal Mach Intell 34(11):2189–2202
40. Endres I, Hoiem D (2010) Category independent object proposals. In European conference on computer vision. Springer, pp 575–588
41. Felzenszwalb PF, Girshick RB, McAllester D, Ramanan D (2009) Object detection with discriminatively trained part-based models. IEEE Trans Pattern Anal Mach Intell 32(9):1627–1645
42. Liu Y, Zhang Q, Zhang D, Han J (2019) Employing deep part-object relationships for salient object detection. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 1232–1241