Semantic Segmentation of Indoor-Scene RGB-D Images Based on Iterative Contraction and Merging

Jia-Hao Syu[1(✉)], Shih-Hsuan Cho[2], Sheng-Jyh Wang[2], and Li-Chun Wang[1]

[1] Department of Communications Engineering, National Chiao Tung University, Hsinchu, Taiwan
jiahaushiu4@gmail.com
[2] Department of Electronics Engineering, National Chiao Tung University, Hsinchu, Taiwan

Abstract. In this paper, we propose an iterative contraction and merging framework (ICM) for semantic segmentation in indoor scenes. Given an input image and a raw depth image, we first derive the dense prediction map from a convolutional neural network (CNN) and a normal vector map from the depth image. By combining the RGB-D image with these two maps, we can guide the ICM process to produce a more accurate hierarchical segmentation tree in a bottom-up manner. After that, based on the hierarchical segmentation tree, we design a decision process which uses the dense prediction map as a reference to make the final decision of semantic segmentation. Experimental results show that the proposed method can generate much more accurate object boundaries if compared to the state-of-the-art methods.

Keywords: Convolutional neural network · Iterative contraction and merging RGB-D image · Semantic segmentation

1 Introduction

Semantic segmentation, an important topic in computer vision, aims to assign each pixel a semantic label in an input image and to generate a dense semantic prediction for the given image. Up to now, many semantic segmentation algorithms have been proposed to improve the quality of dense semantic prediction. However, semantic segmentation is still a challenging work because of the complex and diverse contents in an indoor scene.

Today, RGB-D cameras are getting more popular and cheaper, such as Microsoft Kinect and Intel RealSense cameras. With RGB-D cameras, semantic segmentation algorithms [1–3] take into account both color and depth data to improve the quality of semantic labeling. On the other hand, deep learning techniques are getting popular due to the availability of large-scale datasets and powerful hardware. In [4], Long et al. proposed a deep learning model, called fully convolutional network (FCN), with both color and depth data to perform impressive semantic segmentation. However, since the

A. Mansouri et al. (Eds.): ICISP 2018, LNCS 10884, pp. 252–261, 2018.
https://doi.org/10.1007/978-3-319-94211-7_28

FCN model derives the dense semantic prediction by combining the up-sampling of multilayer information, the obtained semantic prediction results are usually not very accurate around the boundary area.

On the other hand, some hierarchical segmentation algorithms [5, 6] have adopted the bottom-up graph-based approach to generate a hierarchical segmentation tree for image segmentation. These algorithms can properly partition an image into image segments with very accurate region boundaries. However, due to the lack of semantic information during the bottom-up process, these hierarchical segmentation algorithms have difficulties in obtaining reasonable semantic segmentation results.

In this paper, we propose a semantic segmentation based on an iterative contraction and merging process to achieve semantic segmentation with much improved boundary extraction. In the proposed approach, we first cascade two bilateral filters to improve the quality of the raw depth data. Second, we integrate the RGB-D image with a dense semantic predictor, which extracts high-level information, and a normal estimation map, which extracts mid-level information, to guide the ICM process for the generation of a more accurate hierarchical segmentation tree. Finally, we make decisions over the hierarchical segmentation tree to obtain the final semantic segmentation result.

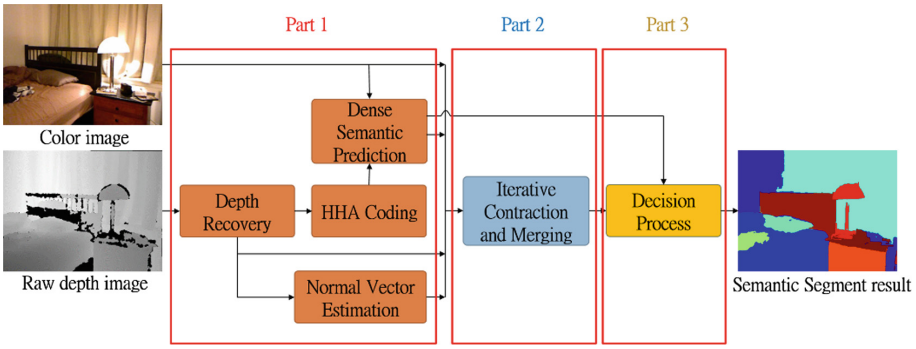

Fig. 1. Block diagrams of the proposed method

2 Proposed Method

In this paper, we propose an architecture that includes three parts to solve the semantic segmentation problem, as illustrated in Fig. 1. In the first part, we capture the mid-level information from the depth image, such as the refined depth image and the normal vector map, and the high-level information from the deep learning model, such as the dense semantic prediction map. To get the refined depth image, we cascade two bilateral filters to fill in the holes in the original depth image. Based on the refined depth image, we estimate the normal vector map based on cross-product computation. Moreover, the depth image is transformed into an HHA image, which has been defined in [3]. The FCN model combines both the RGB image and the HHA image to derive a semantic prediction map. In the second part, the iterative contraction and merging (ICM) process, which was originally proposed in [5], is used for unsupervised image

segmentation. In this paper, we maintain the original ICM features in [5] and add a few more features captured from the first part to derive a more robust hierarchical segmentation tree. In the final part, we design a decision process to decide the semantic segmentation result based on the hierarchical segmentation tree.

3 First Part: Feature Extraction

3.1 Depth Recovery

The raw depth data as shown in Fig. 2 (b) includes holes with no depth values. In order to fill in the estimated depth values over the holes, we assume adjacent pixels with similar RGB color values should have similar depth values. Besides, it is reasonable to trust nearby depth values than far-away depth values in the spatial domain. Hence, we design a spatial kernel which refers to the depth value. On the other hand, we design a range kernel which refers to the RGB values. Based on the spatial kernel and the range kernel, we define a bilateral filter $D^h(\mathrm{x})$ to fill in the depth values over the hole regions, as expressed below:

$$D^h(\mathrm{x}) = \mathrm{W}_1^{-1}(\mathrm{x}) \int_{-\mathrm{k}_1}^{\mathrm{k}_1} \int_{-\mathrm{k}_1}^{\mathrm{k}_1} D(\xi) e^{-\frac{1}{2}\left(\frac{\|x-\xi\|}{\sigma_{s1}}\right)^2} e^{-\frac{1}{2}\left(\frac{\|I(\xi)-I(\mathrm{x})\|}{\sigma_{r1}}\right)^2} \mathrm{d}\xi, \text{ with} \tag{1}$$

$$\mathrm{W}_1(\mathrm{x}) = \int_{-\mathrm{k}1}^{\mathrm{k}1} \int_{-\mathrm{k}1}^{\mathrm{k}1} e^{-\frac{1}{2}\left(\frac{\|x-\xi\|}{\sigma_{s1}}\right)^2} e^{-\frac{1}{2}\left(\frac{\|I(\xi)-I(\mathrm{x})\|}{\sigma_{r1}}\right)^2} \mathrm{d}\xi, \tag{2}$$

where I and D denote the RGB and depth value, k_1 denotes the half-width of the filter, σ_{s1} denote the sigma of the spatial kernel, and σ_{r1} denotes the sigma of the range kernel. In our experiments, we choose $\mathrm{k}_1 = 20$, $\sigma_{s1} = 10$ and $\sigma_{r1} = 0.1$. In some case, the hole area is too large in the original depth images so we need to iteratively fill the holes based on (1) until there is no hole in the depth image. In order to reduce the noise effect on the depth image, we apply the following bilateral filter to smooth the depth image:

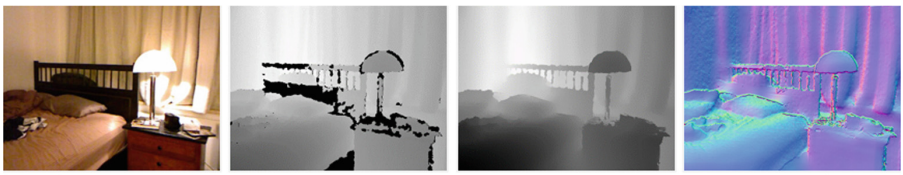

Fig. 2. (a) Original RGB image. (b) Original depth image. (c) Refined depth image. (d) Normal vector map.

$$D^s(\mathrm{x}) = \mathrm{W}_2^{-1}(\mathrm{x}) \int_{-\mathrm{k}_2}^{\mathrm{k}_2} \int_{-\mathrm{k}_2}^{\mathrm{k}_2} D(\xi) e^{-\frac{1}{2}\left(\frac{\|x-\xi\|}{\sigma_{s2}}\right)^2} e^{-\frac{1}{2}\left(\frac{\|D(\xi)-D(\mathrm{x})\|}{\sigma_{r2}}\right)^2} \mathrm{d}\xi, \text{with} \tag{3}$$

$$\mathrm{W}_2(\mathrm{x}) = \int_{-\mathrm{k}_2}^{\mathrm{k}_2} \int_{-\mathrm{k}_2}^{\mathrm{k}_2} e^{-\frac{1}{2}\left(\frac{\|x-\xi\|}{\sigma_{s2}}\right)^2} e^{-\frac{1}{2}\left(\frac{\|D(\xi)-D(\mathrm{x})\|}{\sigma_{r2}}\right)^2} \mathrm{d}\xi. \tag{4}$$

In our experiments, we choose $\mathrm{k}_2 = 20$, $\sigma_{s2} = 12$ and $\sigma_{r2} = 0.05$. In Fig. 2(c), we show an example of the refined depth image.

3.2 Normal Vector Map

On the surface of the objects, each point is described as $(\mathrm{x}, \mathrm{y}, \mathrm{d}(\mathrm{x}, \mathrm{y}))$ in the 3D coordinate where $\mathrm{d}(\mathrm{x}, \mathrm{y})$ denotes the depth value at (x, y). We then estimate the gradient maps by computing the derivatives along the x-direction and y-direction of the depth values. On the surface S, the normal vector at a point i is derived by computing the cross product $\frac{\partial \mathrm{S_i}}{\partial \mathrm{x}} \times \frac{\partial \mathrm{S_i}}{\partial \mathrm{y}}$ which is expressed as

$$\mathrm{N_i} = \frac{\partial \mathrm{S_i}}{\partial \mathrm{x}} \times \frac{\partial \mathrm{S_i}}{\partial \mathrm{y}} = \frac{\partial(\mathrm{x}, \mathrm{y}, \mathrm{d}(\mathrm{x}, \mathrm{y}))}{\partial \mathrm{x}} \times \frac{\partial(\mathrm{x}, \mathrm{y}, \mathrm{d}(\mathrm{x}, \mathrm{y}))}{\partial \mathrm{y}} = \left(-\frac{\partial \mathrm{d}(\mathrm{x}, \mathrm{y})}{\partial \mathrm{x}}, -\frac{\partial \mathrm{d}(\mathrm{x}, \mathrm{y})}{\partial \mathrm{y}}, 1\right), \tag{5}$$

where $\frac{\partial \mathrm{d}(\mathrm{x},\mathrm{y})}{\partial \mathrm{x}} \approx \frac{\mathrm{d}(\mathrm{x}+1,\mathrm{y})-\mathrm{d}(\mathrm{x}-1,\mathrm{y})}{2}$ and $\frac{\partial \mathrm{d}(\mathrm{x},\mathrm{y})}{\partial \mathrm{y}} \approx \frac{\mathrm{d}(\mathrm{x},\mathrm{y}+1)-\mathrm{d}(\mathrm{x},\mathrm{y}-1)}{2}$. Based on the above computations, we can get the normal vector map as shown in Fig. 2(d).

3.3 Dense Semantic Prediction

In our approach, we generate a dense prediction map by using the fully convolutional network (FCN) [4]. The FCN model was originally designed for RGB images. In order to fit the 3-dimensional input format of the FCN model, we follow Gupta's approach in [3] and transform the refined depth image to the HHA image format. Besides, we fine-tune the FCN model for color images and learn another FCN model for depth images based on the NYUD-V2 dataset. Since the NYU-Depth V2 dataset [2] contains 40-class labels, the learned FCN model generates 40 layers of score maps. We combine each end of the feature map with the weight 0.5 and perform up-sampling to derive the score maps with each layer of the score maps representing one class. The dense prediction map is assigned by finding the maximal value at each pixel among 40 layers of score maps:

$$Class_i = arg\ \min_K(S_{iK}), \tag{6}$$

where S_{iK} is the score of pixel i correspond to Class K.

4 Second Part: Iterative Contraction and Merging

The iterative contraction and merging process (ICM) in [5] can construct a hierarchical segmentation tree in two phases. In this paper, we aim to derive a more robust hierarchical segmentation tree for indoor-scene images. Hence, we maintain the original features captured in [5] and add additional features extracted in the first part of our system. In Phase 1 of the ICM process, the pixel-wise contraction and merging process quickly merges pixels with similar features into regions. In this phase, the definition of the affinity value $\mathrm{A}(\mathrm{i},\mathrm{j})$ between pixels i and j is based on pixel-wise features, such as color, depth, normal vector, and dense semantic prediction value. After that, a remnant removal process is used to remove small remnant regions around the boundary. In Phase 2 of the ICM process, since similar image pixels have already been merged into regions, several kinds of regional information, including features in [5] and other additional features, such as depth, normal vector, and dense semantic prediction, are taken into account to define a more informative affinity value $\mathrm{A}(\mathrm{R}_m, \mathrm{R}_n)$ to describe the similarity between the region pair $\mathrm{R_m}$ and $\mathrm{R_n}$. Based on the region-wise affinity matrix, we iteratively apply the contraction and merging process to merge image regions into larger ones and progressively build a hierarchical segmentation tree. In the following subsections, we will describe the details of each module in the ICM process.

4.1 Phase-1: Pixel-Wise Contraction and Merging

Phase-1 of the ICM process aims to quickly merge pixels with similar features into regions. Here, we use a mixed feature space that consists of five subspaces: color space, spatial location space, depth space, normal vector space and dense prediction score space. The features at an image pixel i is mapped into a vector $(L_\mathrm{i}, a_\mathrm{i}, b_\mathrm{i}, x_\mathrm{i}, y_\mathrm{i}, d_\mathrm{i}, u_\mathrm{i}, v_\mathrm{i}, z_\mathrm{i}, S_{\mathrm{i}1}\, S_{\mathrm{i}2} \ldots S_{\mathrm{iK}})$ in the feature space where $(L_\mathrm{i}, a_\mathrm{i}, b_\mathrm{i})$, $(x_\mathrm{i}, y_\mathrm{i})$, (d_i), $(u_\mathrm{i}, v_\mathrm{i}, z_\mathrm{i})$ and $(S_{\mathrm{i}1}\, S_{\mathrm{i}2} \ldots S_{\mathrm{iK}})$ denote the color values, spatial coordinates, depth value, normal vector values and dense prediction values, respectively.

The contraction process aims to pull pixel pairs with similar features closer in the feature space than pixel pairs with less similar features. Here, the contraction process is formulated as the problem of finding the twisted coordinates $\left(\tilde{L}_\mathrm{i}, \tilde{a}_\mathrm{i}, \tilde{b}_\mathrm{i}, \tilde{x}_\mathrm{i}, \tilde{y}_\mathrm{i}, \tilde{d}_\mathrm{i}, \tilde{u}_\mathrm{i}, \tilde{v}_\mathrm{i}, \tilde{z}_\mathrm{i}, \tilde{S}_{\mathrm{i}1}\tilde{S}_{\mathrm{i}2} \ldots \tilde{S}_{\mathrm{iK}}\right)$ and is defined as

$$\mathrm{E}\left(\tilde{\boldsymbol{\theta}}\right) = \sum\nolimits_{\mathrm{i}=1}^{\mathrm{N}} \sum\nolimits_{\mathrm{j}\in \mathrm{w_i}, \mathrm{i}\neq \mathrm{j}} \mathrm{A}(i,j)\left(\tilde{\theta}_\mathrm{i} - \tilde{\theta}_\mathrm{j}\right)^2 + \lambda_\theta \sum\nolimits_{\mathrm{i}=1}^{\mathrm{N}} \left(\tilde{\theta}_\mathrm{i} - \theta_\mathrm{i}\right)^2, \tag{7}$$

where N denotes the total number of pixels, w_i denotes the neighborhood around the pixel i, $\theta_\mathrm{i} \in \{L_\mathrm{i}, a_\mathrm{i}, b_\mathrm{i}, x_\mathrm{i}, y_\mathrm{i}, d_\mathrm{i}, u_\mathrm{i}, v_\mathrm{i}, z_\mathrm{i}, S_{\mathrm{i}1}S_{\mathrm{i}2} \ldots S_{\mathrm{iK}}\}$ denotes the original image feature values, $\tilde{\theta}_1 \in \left\{\tilde{L}_\mathrm{i}, \tilde{a}_\mathrm{i}, \tilde{b}_\mathrm{i}, \tilde{x}_\mathrm{i}, \tilde{y}_\mathrm{i}, \tilde{d}_\mathrm{i}, \tilde{u}_\mathrm{i}, \tilde{v}_\mathrm{i}, \tilde{z}_\mathrm{i}, \tilde{S}_{\mathrm{i}1}\tilde{S}_{\mathrm{i}2} \ldots \tilde{S}_{\mathrm{iK}}\right\}$ denotes the twisted image feature values, and $\tilde{\boldsymbol{\theta}} = \left[\tilde{\theta}_1, \tilde{\theta}_2, \ldots, \tilde{\theta}_\mathrm{N}\right]^\mathrm{T}$. In our simulation, we empirically choose $\lambda_\mathrm{x} = \lambda_\mathrm{y} = 0.001$ and $\lambda_\theta = 0.01$ otherwise. More details of contraction process can be found in [5]. In this paper, we define the affinity value $\mathrm{A}(i,j)$ of pairwise pixels as

$$\mathrm{A}(i,j) = \exp\left(-\mathrm{D}(i,j)/\rho\right) \tag{8}$$

where $\mathrm{D}(i,j)$ denotes the feature difference between i and j and is defined as

$$\begin{aligned}\mathrm{D}(i,j) = {} & \kappa_1 \left\| [L_\mathrm{i}\, a_\mathrm{i}\, b_\mathrm{i}]^\mathrm{T} - [L_\mathrm{j}\, a_\mathrm{j}\, b_\mathrm{j}]^\mathrm{T} \right\|_2 + \kappa_2 \left\| [d_i]^\mathrm{T} - [d_\mathrm{j}]^\mathrm{T} \right\|_2 + \kappa_3 \left\| [u_\mathrm{i}\, v_\mathrm{i}\, z_\mathrm{i}]^\mathrm{T} - [u_\mathrm{j}\, v_\mathrm{j}\, z_\mathrm{j}]^\mathrm{T} \right\|_2 \\ & + \kappa_4 \left\| [\mathrm{S_{i1}\, S_{i2} \ldots S_{iK}}]^\mathrm{T} - [\mathrm{S_{j1}\, S_{j2} \ldots S_{jK}}]^\mathrm{T} \right\|_2 .\end{aligned} \tag{9}$$

Different from [5], here we add depth, normal vector, and dense prediction information into the affinity function. The score weight κ_1, κ_2, κ_3 and κ_4 controls the strength of the impact from the color, depth, normal vector and dense prediction, respectively. Similar to [5], the parameter ρ is adjusted to satisfy the condition that 70% of the $\mathrm{A}(i,j)$ values is larger than 0.01.

After applying the contraction process, we use the same grid-based merging process in [5] to group nearby pixels into regions. In order to efficiently perform the merging process, we only consider the $(L_\mathrm{i}, a_\mathrm{i}, b_\mathrm{i}, x_\mathrm{i}, y_\mathrm{i})$ features during merging. Here, the feature space $\mathrm{S}^\theta = \max(\theta) - \min(\theta)$ denotes the dynamic range in each feature value. To achieve grid-based merging, we divided the feature space from $\mathrm{S^L} \times \mathrm{S^a} \times \mathrm{S^b} \times \mathrm{S^x} \times \mathrm{S^y}$ into $[\mathrm{S^L}/15][\mathrm{S^a}/15][\mathrm{S^b}/15][\mathrm{S^x}/25][\mathrm{S^y}/25]$ regions.

After the pixel-wise contraction and merging process, pixels are merged into regions. However, there may exist a few pixels around the boundary of objects looking like noisy data. To deal with this problem, the remnant removal process in [5] is used to merge regions whose size is smaller than the predefined threshold into one of their adjacent regions with the most similar color appearance.

4.2 Phase-2: Region-Wise Contraction and Merging

After Phase-1, image pixels have been merged into a set of regions. Similar to Phase-1, the averaged feature values in each region R_m can be mapped into a feature vector $(L_\mathrm{m}, a_m, b_m, x_m, y_m, d_m, u_m, v_m, z_m, S_\mathrm{m1} S_\mathrm{m2} \ldots S_\mathrm{mK})$ in the feature space. Likewise, we can derive the twisted coordinates and define the energy function as

$$\mathrm{E}(\tilde{\boldsymbol{\theta}}) = \sum\nolimits_{\mathrm{m}=1}^{\mathrm{N_R}} \sum\nolimits_{\mathrm{R_n} \in \varphi} \mathrm{A}(R_\mathrm{m}, R_\mathrm{n}) \left(\tilde{\theta}_{\mathrm{R_m}} - \tilde{\theta}_{\mathrm{R_n}} \right)^2 + \lambda_\theta \sum\nolimits_{\mathrm{m}=1}^{\mathrm{N_R}} \left(\tilde{\theta}_{\mathrm{R}_m} - \theta_{\mathrm{R_m}} \right)^2 \tag{10}$$

where $\mathrm{N_R}$ denotes the total number of regions, φ denotes the neighboring regions of R_m, $\theta_{\mathrm{R}_m} \in \{L_\mathrm{m}, a_m, b_m, x_m, y_m, d_m, u_m, v_m, z_m, S_\mathrm{m1} S_\mathrm{m2} \ldots S_\mathrm{mK}\}$ denotes the original feature values, $\tilde{\theta}_{\mathrm{R_m}} \in \{\tilde{L}_\mathrm{m}, \tilde{a}_\mathrm{m}, \tilde{b}_\mathrm{m}, \tilde{x}_\mathrm{m}, \tilde{y}_\mathrm{m}, \tilde{d}_\mathrm{m}, \tilde{u}_\mathrm{m}, \tilde{v}_\mathrm{m}, \tilde{z}_\mathrm{m}, \tilde{S}_\mathrm{m1} \tilde{S}_\mathrm{m2} \ldots \tilde{S}_\mathrm{mK}\}$ denotes the twisted feature values, and $\tilde{\boldsymbol{\theta}} = \left[\tilde{\theta}_1, \tilde{\theta}_2, \ldots, \tilde{\theta}_{\mathrm{N_R}}\right]^\mathrm{T}$. The affinity function of pairwise region is defined as

$$\mathrm{A}(R_{\mathrm{m}}, R_{\mathrm{n}}) = \exp\left(-\mathrm{D}(\mathrm{R}_m,\mathrm{R}_n)/\rho\right), \tag{11}$$

where the parameter ρ is set so that 10% of the affinity values are larger than 0.01 if $\mathrm{N_R} > 200$ and ρ is set so that 1% of the affinity values are larger than 0.01 if $\mathrm{N_R} \leq 200$. The difference function between two regions R_{m} and R_n is defined as

$$\mathrm{D}(\mathrm{R}_m, \mathrm{R}_{\mathrm{n}}) = \mathrm{D}_N(\mathrm{R}_m, \mathrm{R}_n)\left[\mathrm{M}_1\left(\frac{\alpha \mathrm{D_C}(\mathrm{R}_m, \mathrm{R}_{\mathrm{n}}) + \beta \mathrm{D_T}(\mathrm{R}_{\mathrm{m}}, \mathrm{R}_{\mathrm{n}}) + \gamma \tilde{\mathrm{D}}_{\mathrm{C}}^{\mathrm{B}}(\mathrm{R}_{\mathrm{m}}, \mathrm{R}_n)}{\sqrt{(80 + \mathrm{SI}(\mathrm{R}_{\mathrm{m}}, \mathrm{R}_n))}}\right)\right.$$

$$\left. + \left(\mathrm{M}_2 \mathrm{D_D}(\mathrm{R}_m, \mathrm{R}_n) + \mathrm{M}_3 \tilde{\mathrm{D}}_{\mathrm{D}}^{\mathrm{B}}(\mathrm{R}_m, \mathrm{R}_n)\right) + \mathrm{M}_4 \mathrm{D_N}(\mathrm{R}_m, \mathrm{R}_{\mathrm{n}}) + \mathrm{M}_5 \mathrm{D_S}(\mathrm{R}_m, \mathrm{R}_n)\right]. \tag{12}$$

where the terms D_N, $\mathrm{D_C}$, $\mathrm{D_T}$, $\tilde{\mathrm{D}}_{\mathrm{C}}^{\mathrm{B}}$ and SI denote the difference of region-size, color, texture, color-of-border and spatial-intertwining. Some of these terms have been introduced in [5]. In this paper, we add four additional difference terms, including depth $\mathrm{D_D}$, depth-of-border $\tilde{\mathrm{D}}_{\mathrm{D}}^{\mathrm{B}}$, normal-vector $\mathrm{D_N}$ and dense prediction score $\mathrm{D_S}$. In our simulation, we empirically choose $\alpha = 1$, $\beta = 3$, $\gamma = 6$, $\mathrm{M}_1 = 3$, $\mathrm{M}_2 = 1$, $\mathrm{M}_3 = 6$, $\mathrm{M}_4 = 1$ and $\mathrm{M}_5 = 2$. After the ICM, we can construct a hierarchical segmentation tree. Please refer to [5] for the details of hierarchical segmentation tree construction.

5 Decision Process

Based on the hierarchical segmentation tree, we propose a decision process to find a semantic segmentation by referring to the dense prediction map. In the dense prediction map, we merge pixels with the same class into regions. For each region $\mathrm{S_K}$ in the dense semantic prediction map, we check the corresponding node in the hierarchical segmentation tree that has the largest overlapping with $\mathrm{S_K}$. We call this region the candidate region of $\mathrm{S_K}$. In other words, given a class region $\mathrm{S_K}$ from the dense semantic prediction map and a node region $\mathrm{T_n}$, the candidate region C_K is defined as

$$\mathrm{C}_K = \arg\ \min_n\left(\frac{|\mathrm{S}_K \cap \mathrm{T}_n|}{|\mathrm{T}_n|}, \frac{|\mathrm{T}_n|}{|\mathrm{S}_K \cap \mathrm{T}_n|}\right), \tag{13}$$

where n denotes the node in the hierarchical segmentation tree and $|\cdot|$ denotes number of pixels in the region. After computing candidate regions, we calculate the covering in the image with these candidate regions. Here, three cases may occur at each pixel:

(1) one candidate region: the pixel is only covered by one candidate region,
(2) more than one candidate region: the pixel is covered by more than two candidate regions, and
(3) no candidate region: the pixel is not covered by any candidate region.

For the first case, we can immediately assign the semantic label based on the corresponding class label. In the second case, we tend to trust the candidate region with a smaller size and assign the semantic label accordingly. In the third case, we have to

assign these no-candidate regions with some semantic labels. Here, we reverse the search from the no-candidate region into the dense prediction map. We can search multiple nodes in the no-candidate region and find the larger overlap with the class of the dense prediction map.

6 Experimental Results

Our propose model is evaluated on NYU-Depth V2 dataset [2] which includes 1449 RGBD images captured by Microsoft Kinect V1. The dataset contains dense per-pixel labeling and are classified into 40 class for semantic segmentation task by Gupta et al. [3]. The quantitative evaluation is measured by four common metrics: pixel accuracy, mean accuracy, mean IU, and frequency weighted IU. In order to know the most important features in our proposed method, we add depth, normal vector, and dense prediction score. In Table 1, we list the quantitative evaluation over the 100 random sampling images in NYUD-V2 dataset by using different combinations of the RGB, depth, normal vector, and dense prediction score. It can be observed that the high-level dense prediction score provides more important information than other features. It turns out the combination of all features does provide the most preferred results. In Table 2, we also compare our proposed method with other semantic segmentation methods. We can observe that the quantitative evaluation of our proposed method is only close to the original FCN model. This is because most boundary regions of the objects are not taken into account in the quantitative evaluation and the improvement of our method mainly occur in those boundary areas.

Table 1. Quantitative evaluation over different combinations of distance functions. (1) RGB, (2) Depth, (3) Normal Vector and (4) FCN

ICM + DP	Pixel acc.	Mean acc.	Mean IU	f.w. IU
1	64.9	51.3	37.9	51.7
1, 2	65.9	51.8	38.6	52.5
1, 3	66.0	52.3	39.0	53.0
1, 4	66.9	53.8	40.1	54.7
1, 2, 3, 4	67.5	54.2	41.0	54.9

Table 2. Comparison of semantic segmentation methods over 100 random sampling images on NYUD-V2 dataset (average value of 5 experiments)

Architecture	Pixel acc.	Mean acc.	Mean IU	f.w. IU
Gupta et al. [3]	60.3	28.6	31.3	47.0
FCN [4]	67.9	56.6	42.4	55.0
Ours	67.5	54.2	41.0	54.9

In Fig. 3, we compare the semantic segmentation results of our method with the original FCN [4] method over four image samples. It can be easily seen that our proposed method provides more accurate semantic segmentation results around the object boundaries.

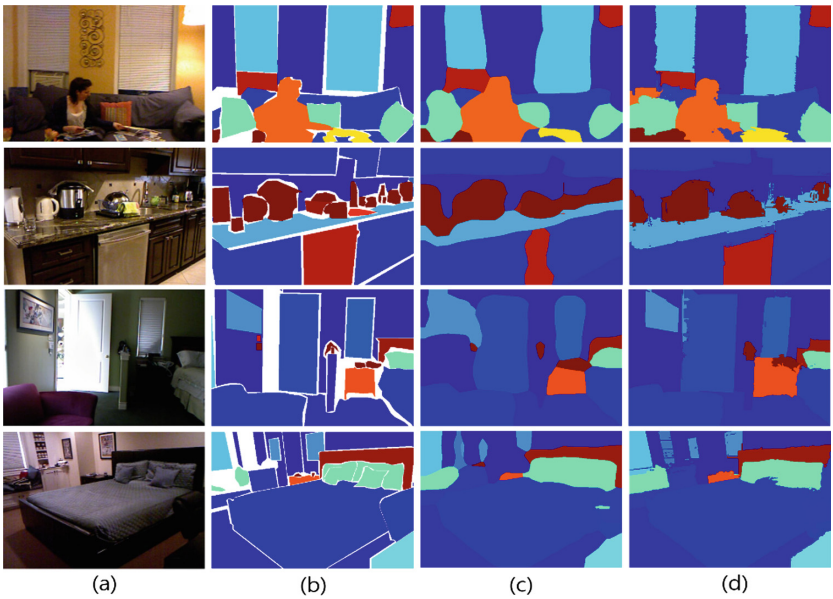

Fig. 3. (a) Original image. (b) Ground truth. (c) FCN [4] method. (d) Our proposed method.

7 Conclusion

In this paper, we propose an iterative contraction and merging framework for semantic segmentation of indoor-scene images. Based on the ICM framework, we improve the quality of the hierarchical segmentation tree by considering more mid-level and high-level features. We also design a decision process to decide the final semantic segmentation result based on the hierarchical segmentation tree. Experimental results show that the proposed method can generate more accurate object boundaries on semantic segmentation results.

References

1. Ren, X., Bo, L., Fox, D.: RGB-(D) scene labeling: features and algorithms. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2759–2766 (2012)
2. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from RGBD images. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012 Part V. LNCS, vol. 7576, pp. 746–760. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33715-4_54

3. Gupta, S., Girshick, R., Arbeláez, P., Malik, J.: Learning rich features from RGB-D images for object detection and segmentation. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014 Part VII. LNCS, vol. 8695, pp. 345–360. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10584-0_23
4. Long, J., Shelhamer, E., Darrell, T: Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
5. Syu, J.H., Wang, S.J., Wang, L.C.: Hierarchical image segmentation based on iterative contraction and merging. IEEE Trans. Image Process. **26**(5), 2246–2260 (2017)
6. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(5), 898–916 (2011)