RC-Net: Row&Column Net with Text Feature for Deep Parsing Floor Plan Images

Teng Wang[1,2],Weiliang Meng[1,2], Zhengda Lu[2], Jianwei Guo[1,2]*, Jun Xiao[2]*, Xiaopeng Zhang[1,2]
[1]State Key Laboratory of Multimodal Artificial Intelligence Systems,
Institute of Automation, Chinese Academy of Sciences
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences

Abstract

We present a novel learning framework for automatically parsing floor plan images. Our key insight is that the room type text is very common and crucial in floor plan images as it identifies the important semantic information of the corresponding room. However, this clue is never considered in previous learning-based methods. In contrast, we propose the *Row&Column Net* (RC-Net), a novel neural network for recognizing floor plan elements by carefully integrating the text feature. Specifically, we add the text feature branch in the network to extract text features corresponding to the room type for the guidance of room type predictions. More importantly, we formulate an *RC constraint module* to share and constrain features across the entire row and column to ensure that only one type is predicted in each room as much as possible, making the segmentation boundaries between different rooms more regular and cleaner. Extensive experiments on three benchmark datasets validate that our framework substantially outperforms other state-of-the-art approaches.

***Keywords:** floor plan understanding, text feature, RC constraint module, RC-Net*

1. Introduction

As the layout of architectural rooms is composed of architectural elements and text information, floor plans can be used for room designing, understanding, and 3D modeling, while several downstream applications are highly relevant to floor plan analysis, such as raster to vector conversion [11, 17], indoor scene modeling [3, 16], building plans retrieval [25, 26], and augmented reality [28, 31, 33], etc. Although deep learning has been rapidly developed in image processing, automatically parsing floor plan images is still an extremely challenging task due to the diversity of floor details.

*Corresponding Author: jianwei.guo@nlpr.ia.ac.cn;xiaojun@ucas.ac.cn

Figure 1. Parsing Floor Plan Images by our RC-Net. Left: the input floor plans that are composed of regular elements and room type texts. Right: our parsing results by elements recognition and room segmentation.

The goal of parsing floor plan images is to extract the geometric structural information of architectural elements and semantic information such as room types. In the early stages, the hierarchical heuristic pattern [1, 2, 7, 8, 23] is widely used to solve this problem. This pattern firstly divides the floor plan image into the graphic image and the text image, then detects the architectural elements and text information by the hand-crafted features and OCR technology respectively. Obviously, as its core strategy is based on hand-crafted features, the hierarchical heuristic pattern cannot adapt to various conditions of floor plans.

Recently, some deep-learning approaches have been proposed to address this problem. Liu et al. [17] detect junction points in the floor plan image by a convolutional neural network (CNN) and then use an integer programming algorithm to connect the junctions and find the walls. However, this method can only find walls between multiple junctions, making it impossible to detect curved and isolated walls.

Other methods [29, 32] employ segmentation networks (FCN [18] and Unet [24]) to identify the classes of pixels in floor plan images, including architectural elements and

room types. However, their segmentation results contain a lot of noise, even if the boundaries of the walls cannot be preserved as regular straight lines or arcs.

In this paper, we propose the Row&Column Net (RC-Net), a novel network with text features to parse floor plan images into floor plan elements (including walls, doors, and windows) and semantic information containing various types of rooms. Our method is based on two observations of the floor plan images as shown in Fig. 1: (i) Most rooms are marked with the room type texts; (ii) The architectural elements and room boundaries in floor plan images are almost regular straight lines and arcs. Unlike previous methods [1, 23, 32] that use OCR technology in the floor plan images, we add a text branch to extract text features and use a merge module to integrate them with the room features, which can significantly improve the accuracy of room type prediction. Meanwhile, we design an *RC constraint module* for the regularity of the floor plan to share and constrain the features in the entire row and column, which can make the room boundaries clear and reduce the noise predictions in each room. Our main contributions can be summarized as follows:

- We propose a novel Row&Column Net (RC-Net) with text features to learn the geometric structural information and semantic information in the floor plan images, which achieves state-of-the-art segmentation results in parsing floor plan images.
- We combine the text feature with the room features to guide the room segmentation of floor plan images, significantly improving the segmentation results.
- We design an RC Constraint Module to share and constrain features in the entire row and column to promote the room type prediction accuracy and boundary regularity.

2. Related work

Hand-crafted floor plan parser. Early works on parsing floor plan images rely on a bunch of low-level image processing with strong heuristics. Generally, traditional methods [1, 2, 6–8, 10, 21–23] firstly divide the floor plan image into a text image and a graphic image, then extract key graphic elements to generate 3D models. Among them, [10, 22] analyze the architectural drawings by extracting features to detect basic architectural entities and convert the floor plan image into the corresponding 3D description, while Macé et al. [21] segment the architectural floor plans into rooms based on Hough transform. Later, Ahmed et al. [1] separate texts from graphics and extract walls according to the line thickness. Subsequently, Heras et al. [7] assume that the wall is usually modeled by straight parallel lines and can obtain good performance on different graphical styles. Besides, they use the patch-based segmentation approach to detect walls [8], and employ a statistical approach of the structural pattern recognition techniques [6] to detect the basic building blocks. Meanwhile, some methods utilize text information to guide room type detection. For example, Ahmed et al. [2] retrieve the meaningful room labeling by OCR and split the floor plan imaged into rooms according to the distribution of labels, while Ravagli et al. [23] improve the performance of text extraction, classification, and recognition in floor plan images by some pre- and post-processing steps. In general, these hand-crafted floor plan parsers are highly dependent on specifically defined features that impair the parsing speed and versatility.

Deep floor plan analysis. With the rapid development of deep learning technology, some methods based on deep convolutional neural networks achieve state-of-the-art results in automatic floor plan analysis. Typically, Dodge et al. [9] use a fully convolutional network (FCN) for wall segmentation, the faster R-CNN for object detection, and an optical character recognition API for estimating room sizes. Further, Liu et al. [17] extract a set of junctions by a learning-based approach and apply integer programming to encode high-level constraints such as walls, windows, and doors, which are used to recover the floor plan vector data from the raster. However, this method cannot handle irregular layouts due to the assumption of the Manhattan world and uniform wall thickness. Later, Huang and Zheng [12] apply pix2pixHD in recognizing and generating architectural drawings to detect and generate apartment plans through two convolutional neural networks. Yamasaki et al. [29] also utilize FCN to segment floor plan images and form a graph model to measure the structural similarity. Zeng et al. [32] predict room-boundary elements and rooms with types by a deep multi-task neural network. However, the pixel labels of the rooms and other elements are noisy, thus the performance is unsatisfactory. A joint network proposed by Lu et al. [19] uses U-Net to segment boundaries and detect text by the SSD to determine the room type, and a post-processing method is followed to ensure the accuracy of the area and type of room. Lv et al. [20] apply YOLOv4 to extract the critical region from floor plan images, then YOLOv4 is reused for detecting texts, symbols, and scale numbers, while DeepLabv3+ [4] is used simultaneously for segmenting the area of rooms. Yang et al. [30] predict boundaries and room types using two branches based on the same VGG feature extraction. Their direction-aware kernels and boundary features are also used to enhance semantic information of room-type features.

In general, these methods do not utilize the text information in the floor plan to extract the semantic information, and their prediction results contain a lot of noise, which is inconsistent with the regularity of the floor plan.

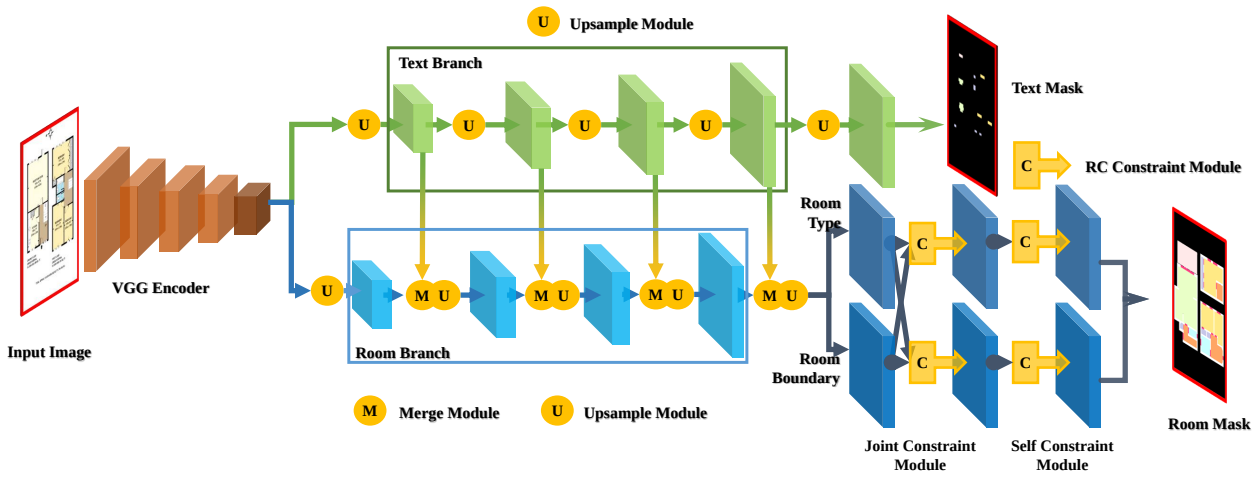

Figure 2. Overview of our RC-Net architecture. Our RC-Net has the text branch and the room branch, and can generate text mask and room mask respectively.

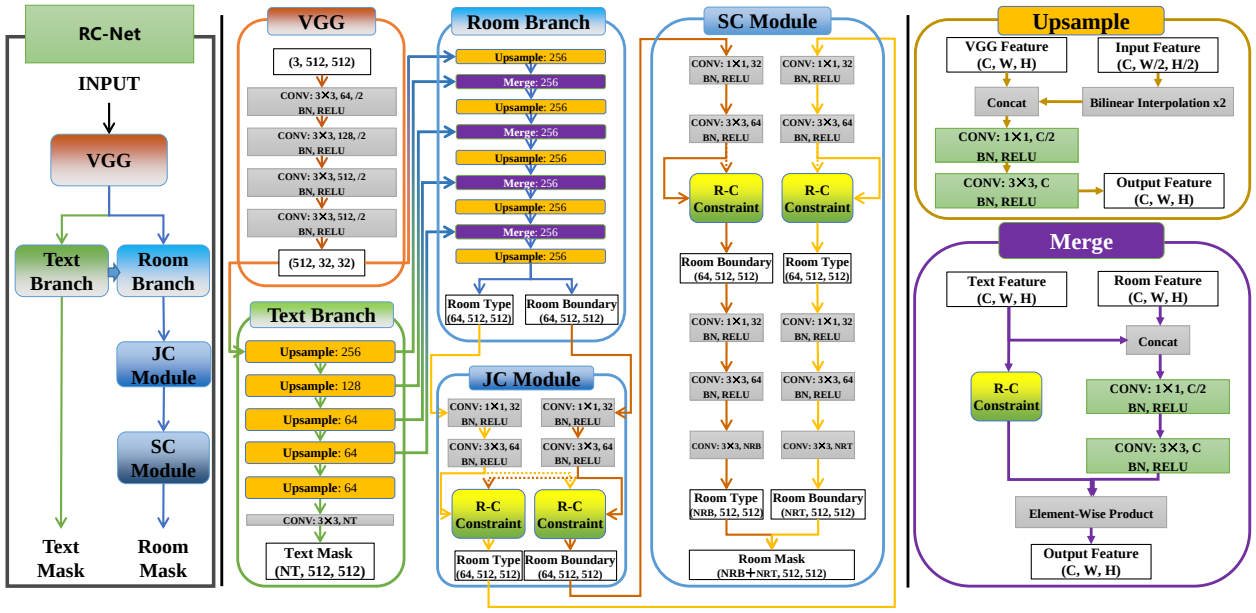

Figure 3. The details of our RC-Net architecture, in which the R-C Constraint is given in Fig. 6.

3. Our RC-Net

The goal of parsing floor plan images is to extract architectural elements (such as walls, doors, windows, *etc*let@tokeneonedot) and semantic information of various rooms. Obviously, the room in the floor plan consists of a set of regular architectural elements and a text that directly indicates the room type. Therefore, we add a text branch to extract the text features and expand it to the entire feature information through our RC constraint module to ensure a more accurate prediction of the room type. Meanwhile, our RC constraint module can obtain smooth wall boundaries due to the constraints on the features of the entire row and column.

In the following, we first present the architecture of our Row&Column Net (RC-Net), then we show the text branch and the implementation of our RC constraint module which shares and constrains the features on the entire row and column. Finally, we describe the implementation details of the training of our RC-Net.

3.1. Network Architecture

The architecture of our RC-Net is illustrated in Fig. 2, and the parameter details of each sub-module are given in Fig. 3. We first use a VGG encoder [27] on the input floor plan image to extract basic features that are used in two branches: a) The room branch obtains features that

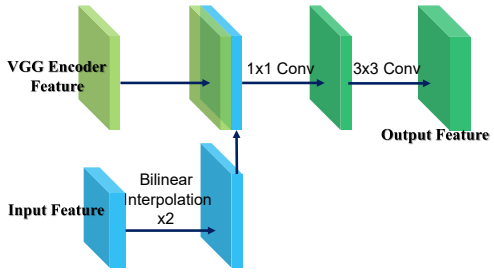

Figure 4. Upsample Module. Bilinear interpolation is used to expand the input feature at first, then the expanded feature is mixed with the feature from the VGG encoder. Finally we use convolution operations to extract features.

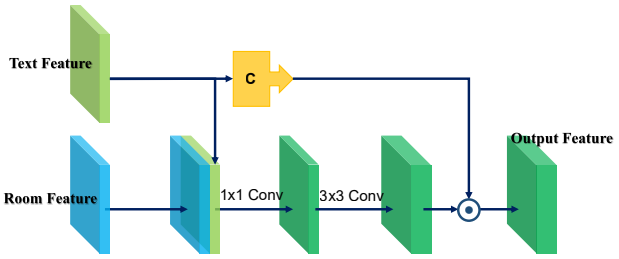

Figure 5. Merge Module. The text feature obtained from the text branch and the room feature obtained from the room branch are merged together in this module, and $\odot$ represents the element-wise product operation.

represent the room boundary (i.e. wall, window, door, *etc*let@tokeneonedot) and the room type (i.e. dining room, washroom, bedroom, *etc*let@tokeneonedot). This branch is divided into two sub-branches further at the end for maximizing the learning of features, one for predicting the room boundary and the other for predicting the room type. Eventually, we merge the output of the two sub-branches to get the final room mask; b) The text branch focuses on the features of the text in the floor plan image, and the output is a text mask which is only used for the loss function in training steps. This text branch obtains text features and then mixes them into the room features to get a more accurate room type.

The *Upsample Module* (Fig. 4) in our RC-Net employs the bilinear interpolation to expand the input feature to its double size at first, then the feature is concatenated with the corresponding feature from the VGG encoder. The following 1×1 convolution is used to share the channel information, while the 3×3 convolution is used to share the spatial information. The *Merge Module* is employed for concatenating the text features with room features together, and the details are given in section 3.2.

3.2. Text Branch

Since the room type text indicates the room type where it is located, we use the text branch to extract text features and concatenate them with room features of the corresponding area in the merge module, in order to obtain richer semantic features for subsequent prediction branches. Note that if there is no room type text feature in the area, we can also use the room feature to predict the room type of the area. In addition, we can get more accurate predictions without text using connection information between rooms learned from the training phase guided by the text branch.

The structure of the merge module is illustrated in Fig. 5. Firstly, we concatenate the given text features with room features, use a 1×1 convolution to share the channel information, and a 3×3 convolution to share the spatial information. Simultaneously, we employ the Row&Column(RC)-constraint module to calculate the constrain mask from the text features and expand the text feature into the entire room feature as it only occupies a part of this room area. Finally, the element-wise product is used for the integration of features and the constraint mask to obtain the merged room features containing ample text context.

We mainly obtain the features of the text in the images in the text branch, rather than the final text result. We can use our RC RC-Constraint module to learn the relationship between rooms using features extracted from the text branch while the object detection for text can only provide what and where those texts are. Furthermore, when the wording of specific room types is flexible, it is hard to build a dictionary to map the word into the type rooms, which makes the program more complex and lacks lots of flexibility. Finally, the room branch can extract auxiliary signs to enhance the correspondence between text and room types in the training phase.

3.3. Row&Column Constraint Module

As rooms in the floor plan are usually regular polygons due to most of the composed architectural elements being straight lines, we design the **Row&Column (RC) Constraint Module** (see Fig. 6) to obtain the constraint mask for the row and column features, as well as more regular boundaries between the wall and rooms with fewer noises. At the same time, the feature receptive field is expanded because we share text features to the entire row and column through RC constraints.

As illustrated in Fig. 6, given a local feature X as the input, we firstly extract the column feature X_c and the row feature X_r using Eq (1):

$$\begin{aligned} X_r &= \frac{\sum_r W_r X}{C}, \\ X_c &= \frac{\sum_c W_c X}{R}, \end{aligned} \tag{1}$$

where $X \in \mathbb{R}^{R \times C \times k}$ is the given feature, $W_r \in \mathbb{R}^{R \times 1 \times k}$ and $W_c \in \mathbb{R}^{1 \times C \times k}$ are the learnable weights. The size of X_r and X_c are $1 \times C \times k$ and $R \times 1 \times k$ individually,

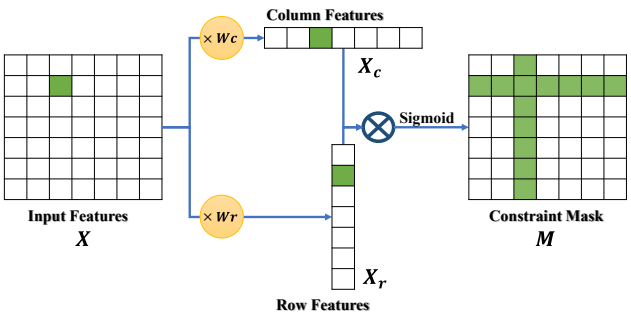

Figure 6. RC Constraint Module. For input features, we get the row and column features by multiplying learnable weights, then calculate the raw constraint mask by Kronecker Product represented by $\otimes$. The sigmoid function is used to limit the value of the final constraint mask.

while the R, C, and k are the number of rows, columns and channels of the given feature, with $\sum_r$ and $\sum_c$ denote the sum in rows and columns respectively.

Then we use matrix multiplication to multiply X_c and X_r to get the raw constraint mask and propagate the features to the entire image. The sigmoid function is utilized to generate the constraint mask distributions over the features, and the final constraint mask can be expressed as follows:

$$\begin{aligned} M &= X_r \otimes X_c \\ &= \frac{\sum_r W_r X}{C} \otimes \frac{\sum_c W_c X}{R} \\ &= \frac{\sum_r W_r X \otimes \sum_c W_c X}{RC}. \end{aligned} \tag{2}$$

Since there are no text features on some rows and columns in the input feature map, we give two types of RC Constraint Modules in our RC-Net: joint-constraint module and self-constraint module, which are shown in Fig. 7 (a) and (b). Different from the self-constraint module, the joint constraint module has one more text feature input as the constraint, so using it in the merge module can extend the text feature to the entire row and column of the room feature. In the subsequent room branch, the joint constraint module is used again to mix the room type features and room boundary features. The self-constraint module is only used in the room branch, which can constrain the row and column features to get a clearer segmentation boundary.

3.4. Implementation Details

Loss function. Our RC-Net has three output masks in training steps: text, room type, and room boundary. Since the number of pixels in different categories is extremely unbalanced in every output mask, we employ the focal loss [14] to define our final loss function as:

$$L = W_t L_t + W_{rt} L_{rt} + W_{rb} L_{rb}, \tag{3}$$

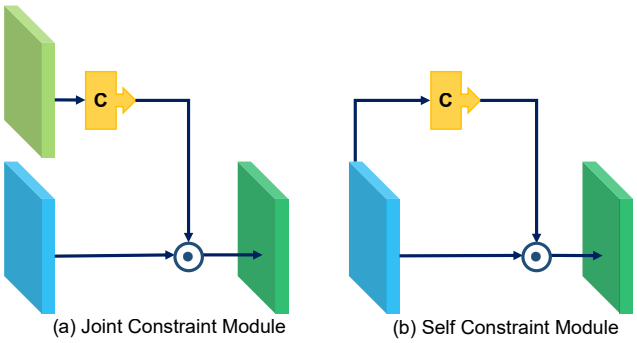

Figure 7. Two types of the RC Constraint Modules that we used in our RC-Net. (a) The Joint Constraint is used for sharing information from two inputs features. (b) The Self Constraint is used to constrain the features on the entire row and column to obtain a smoother segmentation boundary. $\odot$ represents the element-wise product operation.

where L_t, L_{rt}, and L_{rb} are focal losses for measuring the output mask of the text, room type, and room boundary respectively, while W_t, W_{rt}, and W_{rb} are the corresponding weights for these three loss functions. We set higher weights on L_t and L_{rb} in order to improve the accuracy of effective categories because the mask of text and room boundary has more background pixels. In practice, we set $W_t = 10.0$, $W_{rt} = 5.0$, and $W_{rb} = 10.0$ which work well in all of our experiments.

Training details. We train our network with a total of 100k steps on R2V dataset, 60k steps on R3D dataset, and 120k steps on the Cubicasa dataset, with the details of these datasets are given in section 4.1. We use a fixed learning rate of $1e-4$ and employ Adam as the optimizer to train our network with a weight decay of $1e-5$ to avoid overfitting. The resolution of the input floor plan image is 512512 with good details. During the training, we randomly resize and select a part from the whole image in the initial 30k steps, and use the whole image in the remaining steps. We also randomly distort the color in our training steps for data augmentation.

4. Experimental Results

In this section, we first introduce the benchmark datasets as well as the metrics we used for the evaluation. Then, we provide a complete qualitative and quantitative comparison with state-of-the-art approaches. Finally, we perform a series of ablation experiments to validate the effectiveness of our RC-Net. All experiments are conducted on a server equipped with an Intel Xeon Gold 6226R processor 2.9GHz with 64 cores, 256 GB of RAM, and an NVIDIA GeForce RTX 2080Ti (11GB memory) graphics card.

Figure 8. Visual comparison of parsing floor plan images on R2V and R3D datasets. From left to right are (a) input floor plan image, (b) ground truth, (c) Raster-to Vector [17], (d) DeepFloorPlan [32], (e) our method, and (f) the text feature mask. The examples in the top four rows are from the R2V dataset while the other two are from the R3D dataset.

4.1. Experimental Setup

Datasets. For the performance evaluation and comparison, we carry out experiments on three commonly used benchmarks:

- **R2V** [17]: It contains 870 floor plan images, where 770 images are used as the training data and the remaining 100 images are served as the testing data;
- **R3D** [15]: It originally contains 214 floor plan images. We follow the modified pattern of [32] which adds 18 floor plan images and splits the dataset into 179 training images and 53 testing images;
- **Cubicasa** [13]: It consists of 5000 ground-truth Finnish floor plan images. Since we need to label the text mask, we randomly collect 1,000 images from the training images and 156 images from the testing images.

Evaluation metrics. For quantitative evaluations, we adopt several widely-used metrics in [9, 32], including *mean Accuracy* (mAcc) and *mean Intersect over Union* (mIoU). Besides, we use the *class Intersect over Union* (class_IoU) and *Frequency Weighted Intersection over Union* (FWIoU) which can reflect the correlation between the predictions and the ground truth more realistically. These metrics are formally defined as follows:

$$mAcc = \frac{1}{C}\sum_i \frac{N_{ii}}{\sum_j N_{ij}}, \tag{4}$$

$$mIoU = \frac{1}{C}\sum_i \frac{N_{ii}}{\sum_j N_{ij} + \sum_j N_{ji} - N_{ii}}, \tag{5}$$

$$class\_IoU_i = \frac{N_{ii}}{\sum_j N_{ij} + \sum_j N_{ji} - N_{ii}}, \tag{6}$$

$$FWIoU = \frac{1}{\sum_i \sum_j N_{ij}}\sum_i \frac{N_{ii}}{\sum_j N_{ij} + \sum_j N_{ji} - N_{ii}}. \tag{7}$$

Here N_{ij} is the number of pixels whose ground-truth class is i while the prediction class is j, and C is the number of classes.

4.2. Evaluation and Comparison

For comparison, we select two representative learning approaches to evaluate the floor plan parsing performance of our RC-Net: Raster-to-Vector [17] and DeepFloorPlan [32]. Both methods provide a plethora of comparisons to other techniques including traditional approach [1] and general semantic image segmentation network [5, 34] and establish themselves as state-of-the-art methods, thus we omit comparisons with other works that have already been compared.

Comparison on R2V and R3D dataset. We first evaluate the performance of our method on the R2V and R3D datasets with rectangular and various shapes. For a fair comparison, we train our network separately on the corresponding training data to compare to Raster-to-Vector [17] and DeepFloorPlan [32]. Since the network of Raster-to-Vector only detects the junctions of the floor plan, we use their released weights to obtain their network output. Finally, Raster-to-Vector [17] outputs a vector-graphics representation by an integer programming, thus we follow its procedures and instructions to convert its output to pixel images for comparison. For DeepFloorPlan [32], we use its released weights to get the results of the R3D dataset and retrain its network for the R2V dataset based on the official open-source code released by the authors.

Fig. 8 shows the visual comparisons, where the top four rows are comparisons on R2V, and the bottom two rows are comparisons on R3D. The Raster-to-Vector network obtains the basic elements such as walls, doors, and windows by connecting the junctions, and finds the room area by the integer programming with various geometric and semantic constraints. Since its results of basic elements are related to the predicted junctions, the top two rows in Fig. 8 (c) lack some isolated wall lines or create some extra wall lines compared with the ground-truth in Fig. 8 (b). Meanwhile, it produces some wrong room topology and room type prediction results in the second, third, and bottom rows with the limitation of the integer programming and semantic constraints. In particular, this method can only generate some non-existent regular rectangles even if some joint points can be detected for the circular floor plan shown in the fifth row. The reason is that Raster-to-Vector is based on the assumption of the Manhattan world, thus it cannot process such non-rectangle images. Due to the per-pixel prediction, the results of DeepFloorPlan always contain noises at the boundary of different categories, and its predicted results of the room type may be wrong (see Fig. 8 (d)). In contrast, our room prediction results are more accurate than other two algorithms with the guide of the text branch, as shown in Fig. 8 (e). Obviously, the room types prediction of our text branch is more accurate as seen in Fig. 8 (f). Besides, due to the impact of our RC constraint module, our room prediction results are less noisy and boundaries are much smoother.

The numerical statistics about the predicted IoU and mean accuracy for each method are reported in Table 1. There are many non-rectangular room shapes in the R3D dataset, so the performance of Raster-to-Vector is worse compared on the R2V dataset. For the average, our method has a higher mACC with 3%$\sim$24% improvement, and a higher mIoU with 4%$\sim$31% improvement, which means that the predicted results of our network are more accurate.

Comparison on CubiCasa dataset. Since the Cubicasa dataset divides the category of *Opening* into *Window* and *Door*, while the number of room types is more than R2V

Table 1. Quantitative comparison on R2V and R3D datasets. The best results of each measurement are marked in **bold** font.

Dataset	Methods	$FWIoU$	$class\_IoU$								$mACC$	$mIoU$
			Wall	Opening	Closet	Bathroom	Living room	Bedroom	Hall	Balcony		
R2V	Raster-to-Vector	0.75	0.60	0.57	0.68	0.79	0.74	0.84	0.53	0.72	0.83	0.69
	DeepFloorPlan	0.72	**0.62**	0.53	0.62	0.73	0.73	0.80	0.37	0.75	0.84	0.64
	Our RC-Net	**0.77**	**0.62**	**0.58**	**0.77**	**0.82**	**0.77**	**0.82**	**0.65**	**0.83**	**0.87**	**0.73**
R3D	Raster-to-Vector	0.49	0.65	0.41	0.34	0.31	0.58	0.46	0.51	0.27	0.60	0.44
	DeepFloorPlan	0.69	**0.81**	**0.61**	0.37	0.77	0.71	0.66	0.50	0.50	0.80	0.65
	Our RC-Net	**0.84**	0.77	0.53	**0.59**	**0.84**	**0.89**	**0.86**	**0.79**	**0.72**	**0.84**	**0.75**

Table 2. Comparison on CubiCasa dataset.

Methods	$FWIoU$	$class\_IoU$													$mACC$	$mIoU$
		Wall	Window	Door	Outdoor	Kitchen & *etc*let@tokeneonedot	Living Room	Bedroom	Bath	Hall	Railing	Storage	Garage	Others		
Raster-to-Vector	0.38	0.60	0.37	0.14	0.40	0.36	0.42	0.46	0.21	0.38	0.10	0.18	0.12	0.20	0.46	0.30
DeepFloorPlan	0.46	0.64	0.43	0.36	0.38	0.47	0.52	0.48	0.53	0.46	0.07	0.39	0.25	0.31	0.63	0.41
Our RC-Net	**0.70**	**0.70**	**0.66**	**0.53**	**0.61**	**0.77**	**0.81**	**0.86**	**0.61**	**0.67**	**0.43**	**0.54**	**0.57**	**0.53**	**0.76**	**0.64**

(a) (b) (c) (d) (e) (f)

Figure 9. Visual comparison on CubiCasa dataset. From left to right: (a) input floor plan, (b) ground truth, (c) Raster-to-Vector [17], (d) DeepFloorPlan [32], (e) our method, and (f) the text feature mask.

and R3D dataset, we modify the output categories of all algorithms and retrain them on this dataset. We follow the method introduced in [13] to convert their SVG format annotations into pixel labels, where the floor plan elements include wall, door and window, with the room types have 10 categories as shown in Table 2.

Fig. 9 shows the visual comparisons and Table 2 reports the numerical statistics about the performance on the Cubicasa dataset. Compared with the R2V and R3D datasets, the Cubicasa dataset is more extensive, and the floor plan

Figure 10. Prediction results of floor plan images without text, fewer text and all texts by our RC-Net. From left to right: (a) Images without texts and predictions of our RC-Net, (b) Images with part of texts and predictions of our RC-Net, (c) Images with all texts and predictions of our RC-Net, and (d) Ground truth.

styles and structures vary greatly in color and quality. At the same time, its annotations divide areas containing multiple texts (such as living room, dining room, and kitchen) into different categories. Thus the numerical statistics in Table 2 show that the overall classification results are worse. As shown in Fig. 9 (c), it's difficult for the Raster-to-Vector method to find all of the junctions in complex scenes, leading to lots of missing walls and rooms in its prediction results. Meanwhile, its converted wall widths are the same and not matching the actual image as the wall width may be different in the floor plan image. Finally, some slightly inclined lines are converted into straight lines as shown in the bottom row of Fig. 9 (c). Therefore, the prediction results of the Raster-to-Vector method on the Cubicasa dataset are extremely unsatisfactory. Similarly, the floor plan recognition results produced by DeepFloorPlan have too much noise due to too many room categories and the similarity of room structures in the Cubicasa dataset shown in Fig. 9 (d).

In contrast, our RC-Net can still get accurate room prediction results with the powerful guide by our text branch shown in Fig. 9 (f). Our RC constraint module greatly reduces the noise pixels in the room and the boundary to ensure that each room is predicted correctly with regular boundaries. It is worth noting that for a room without text information, our RC-Net can still accurately infer the type of the room based on its internal information. For example, in most floor plan images, the living room and balcony are adjacent, our method can detect the relationship and infer the balcony room without any text when the living room is confirmed using the text. There are bathtub, toilet and other signs in the bathroom, while the room branch can extract these features to determine whether the room is a bathroom or not when removing the text in the floorplan. Fig. 10 shows the prediction results by our RC-Net on two floorplan images with no texts, fewer texts, and full texts, respectively. We can see that the regions with text changing can be segmented, but may be classified into another type that is different from the ground truth when the texts are lost. After the texts are added, they will be classified correctly. In general, the predictions of our RC-Net are better when the texts exist, which reflects our advantage of using the text branch.

4.3. Ablation Study of Our Text Branch

We first study the effectiveness of the added text branch in our RC-Net and conduct ablation experiments for the text branch. Note that the merge module is also removed while not using the text branch. Specifically, we use the previous weights of the whole RC-Net for initialization, which can significantly reduce the training time.

From Table 3, we can see that in most cases, the text branch makes predictions more accurate. This is because the types of rooms are closely dependent on the corresponding text of the room. The accuracy improvements on the R2V and R3D datasets are extremely obvious because the text information in these floor plans has a large proportion of the room, and other icon information is less.

Besides, due to a large amount of iconic information in the room for the Cubicasa dataset, the room branch can learn enough features, making prediction accuracy on the Cubicasa is only slightly improved by adding the text branch. In general, the results with our text branch are better with 3%∼20% improvement.

4.4. Ablation Study of Our RC Constraint Module

We also investigate whether our RC Constraint Module can share and constraint features effectively. We conduct ablation experiments on our RC Constraint Module used in

Table 3. The performance of our RC-Net without and with the text branch.

Dataset	Text Branch?	mAcc	mIoU	FWIoU
R2V	✗	0.80	0.63	0.66
	✓	**0.87**	**0.73**	**0.77**
R3D	✗	0.62	0.50	0.65
	✓	**0.84**	**0.75**	**0.85**
CubiCasa	✗	0.73	0.60	0.64
	✓	**0.76**	**0.64**	**0.70**

Table 4. The performance of different combination for **JC***, **JC** and **SC**. **JC*** is the joint constraint module used in merge module. **JC** and **SC** are the joint constraint module and the self constraint module of the bottom layers of our network.

Dataset	Method			mAcc	mIoU	FWIoU
	JC*	**JC**	**SC**			
R2V	✗	✗	✗	0.81	0.62	0.68
	✓	✗	✗	0.83	0.68	0.73
	✓	✓	✗	0.84	0.71	0.75
	✓	✗	✓	0.84	0.70	0.75
	✓	✓	✓	**0.87**	**0.73**	**0.77**
R3D	✗	✗	✗	0.65	0.55	0.69
	✓	✗	✗	0.73	0.62	0.79
	✓	✓	✗	0.81	0.74	0.84
	✓	✗	✓	0.82	0.74	0.82
	✓	✓	✓	**0.84**	**0.75**	**0.85**
CubiCasa	✗	✗	✗	0.68	0.58	0.66
	✓	✗	✗	0.71	0.61	0.69
	✓	✓	✗	0.74	0.61	0.67
	✓	✗	✓	0.71	0.63	0.69
	✓	✓	✓	**0.76**	**0.64**	**0.70**

the self constraint module (**SC**), the joint constraint module (**JC**) of the bottom layers, as well as the joint constraint module (**JC***) in the merge module. To this end, we have designed five different experiments: i) without them all; ii) without **SC** and **JC**;iii) without **SC**; iv) without **JC**; v) using all of them. Specifically, we initialize the weights of each experiment with the previously trained weights of the entire RC-Net.

From Table 4, we have the following observations: (i)The **JC*** can provide more precise segmentation on the R2V and R3D datasets since it can maximize the sharing of textual context to the room features. However, it has less impact on CubiCasa dataset as the floor plan images of CubiCasa have huge iconic information. (ii)The **JC** and **SC** can improve the accuracy slightly since it can provide better constraints on the boundaries of rooms to make the segmentation of boundaries more regular;(iii)The combination of **JC** and **SC** has better performance due to its ability to share features on the room boundary features and the room type features;(iv)With the features sharing on the entire row and column for using our RC Constraint model completely, each pixel can be classified more accurately.

In sum, all the evaluation metrics are improved by using our RC constraint module partly or completely.

5. Conclusion and Future Work

We propose a novel RC-Net to extract semantic information from floor plan images for segmentation to parse residential buildings. By introducing the text branch for guiding room type predictions and designing the RC constraint module to share and constraint features on the entire row and column, the text features are fully integrated with the room features, while the noise points are greatly reduced in the prediction. Experiments demonstrate the advantages of our method by comparing to other state-of-the-art methods on three benchmark datasets.

Limitations and future work. When the floor plan images lack text labels, the segmentation accuracy is not improved obviously based on our method. Besides, there are still a few noise points (as shown in Fig. 8 & Fig. 9) in some segmentation results where the boundaries are not obvious. In addition, our method may not achieve good performance in extremely complicated settings. For example, when the rooms are very small and crowded together, our method is difficult to segment and identify these rooms properly.

In the future, we will improve the prediction of room types by using more iconic information and more constraints, thereby promoting the prediction of room types. In addition, we will explore the applications of the RC constraint module to image detection and other tasks and verify the effectiveness of this module on other public benchmarks.

Acknowledgments

This work is being supported in part by the National Natural Science Foundation of China (U21A20515, 62172416, 52175493, U2003109, 61972459, and 62102414), and the Youth Innovation Promotion Association of the Chinese Academy of Sciences (2022131).

References

[1] S. Ahmed, M. Liwicki, M. Weber, and A. Dengel. Improved automatic analysis of architectural floor plans. In *2011 International Conference on Document Analysis and Recognition*, pages 864–869. IEEE, 2011. 1, 2, 7

[2] S. Ahmed, M. Liwicki, M. Weber, and A. Dengel. Automatic room detection and room labeling from architectural floor plans. In *2012 10th IAPR International Workshop on Document Analysis Systems*, pages 339–343. IEEE, 2012. 1, 2

[3] K. Chen, Y. Lai, Y.-X. Wu, R. R. Martin, and S.-M. Hu. Automatic semantic modeling of indoor scenes from low-quality rgb-d data using contextual information. *ACM Trans. Graph.*, 33(6), 2014. 1

[4] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018. 2

[5] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Eur. Conf. Comput. Vis.*, pages 801–818, 2018. 7

[6] L.-P. de las Heras, S. Ahmed, M. Liwicki, E. Valveny, and G. Sánchez. Statistical segmentation and structural recognition for floor plan interpretation. *International Journal on Document Analysis and Recognition (IJDAR)*, 17(3):221–237, 2014. 2

[7] L.-P. de las Heras, D. Fernández, E. Valveny, J. Lladós, and G. Sánchez. Unsupervised wall detector in architectural floor plans. In *12th International Conference on Document Analysis and Recognition*, pages 1245–1249. IEEE, 2013. 1, 2

[8] L.-P. de las Heras, J. Mas, E. Valveny, et al. Wall patch-based segmentation in architectural floorplans. In *2011 International Conference on Document Analysis and Recognition*, pages 1270–1274. IEEE, 2011. 1, 2

[9] S. Dodge, J. Xu, and B. Stenger. Parsing floor plan images. In *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*, pages 358–361. IEEE, 2017. 2, 7

[10] P. Dosch, K. Tombre, C. Ah-Soon, and G. Masini. A complete system for the analysis of architectural drawings. *International Journal on Document Analysis and Recognition*, 3(2):102–116, 2000. 2

[11] O. Hori and S. Tanigawa. Raster-to-vector conversion by line fitting based on contours and skeletons. In *Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR '93)*, pages 353–358, 1993. 1

[12] W. Huang and H. Zheng. Architectural drawings recognition and generation through machine learning. In *Proceedings of the 38th Annual Conference of the Association for Computer Aided Design in Architecture*, pages 18–20, 2018. 2

[13] A. Kalervo, J. Ylioinas, M. Häikiö, A. Karhu, and J. Kannala. Cubicasa5k: A dataset and an improved multi-task model for floorplan image analysis. In *Scandinavian Conference on Image Analysis*, pages 28–40. Springer, 2019. 7, 8

[14] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5

[15] C. Liu, A. G. Schwing, K. Kundu, R. Urtasun, and S. Fidler. Rent3d: Floor-plan priors for monocular layout estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3413–3421, 2015. 7

[16] C. Liu, J. Wu, and Y. Furukawa. Floornet: A unified framework for floorplan reconstruction from 3d scans. In *Eur. Conf. Comput. Vis.*, pages 201–217, 2018. 1

[17] C. Liu, J. Wu, P. Kohli, and Y. Furukawa. Raster-to-vector: Revisiting floorplan transformation. In *Int. Conf. Comput. Vis.*, pages 2195–2203, 2017. 1, 2, 6, 7, 8

[18] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1

[19] Z. Lu, T. Wang, J. Guo, W. Meng, J. Xiao, W. Zhang, and X. Zhang. Data-driven floor plan understanding in rural residential buildings via deep recognition. *Information Sciences*, 567:58–74, 2021. 2

[20] X. Lv, S. Zhao, X. Yu, and B. Zhao. Residential floor plan recognition and reconstruction. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16712–16721, 2021. 2

[21] S. Macé, H. Locteau, E. Valveny, and S. Tabbone. A system to detect rooms in architectural floor plan images. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, pages 167–174, 2010. 2

[22] S.-h. Or, K.-H. Wong, Y.-k. Yu, M. M.-y. Chang, and H. Kong. Highly automatic approach to architectural floorplan image understanding & model generation. *Pattern Recognition*, pages 25–32, 2005. 2

[23] J. Ravagli, Z. Ziran, and S. Marinai. Text recognition and classification in floor plan images. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 1, pages 1–6. IEEE, 2019. 1, 2

[24] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 1

[25] D. Sharma, N. Gupta, C. Chattopadhyay, and S. Mehta. Daniel: A deep architecture for automatic analysis and retrieval of building floor plans. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 420–425. IEEE, 2017. 1

[26] D. Sharma, N. Gupta, C. Chattopadhyay, and S. Mehta. A novel feature transform framework using deep neural network for multimodal floor plan retrieval. *International Journal on Document Analysis*

and Recognition (IJDAR), 22(4):417–429, 2019. 1

[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015. 3

[28] Z. Xu, Z. Rong, and Y. Wu. A survey: which features are required for dynamic visual simultaneous localization and mapping? *Visual Computing for Industry, Biomedicine, and Art*, 4(1):1–16, 2021. 1

[29] T. Yamasaki, J. Zhang, and Y. Takada. Apartment structure estimation using fully convolutional networks and graph model. In *Proceedings of the 2018 ACM Workshop on Multimedia for Real Estate Tech*, pages 1–6, 2018. 1, 2

[30] B. Yang, T. Jiang, W. Wu, Y. Zhou, and L. Dai. Automated semantics and topology representation of residential-building space using floor-plan raster maps. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 15:7809–7825, 2022. 2

[31] S.-T. Yang, F.-E. Wang, C.-H. Peng, P. Wonka, M. Sun, and H.-K. Chu. Dula-net: A dual-projection network for estimating room layouts from a single rgb panorama. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3363–3372, 2019. 1

[32] Z. Zeng, X. Li, Y. K. Yu, and C.-W. Fu. Deep floor plan recognition using a multi-task network with room-boundary-guided attention. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9096–9104, 2019. 1, 2, 6, 7, 8

[33] Y. Zhang, S. Song, P. Tan, and J. Xiao. Panocontext: A whole-room 3d context model for panoramic scene understanding. In *Eur. Conf. Comput. Vis.*, pages 668–686. Springer, 2014. 1

[34] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2881–2890, 2017. 7