A Novel OCR-RCNN for Elevator Button Recognition

Delong Zhu, Tingguang Li, Danny Ho, Tong Zhou, Max Q.-H. Meng*

***Abstract*— Autonomous elevator operation is considered an intelligent solution in handling the inter-floor navigation problem of service robots. As one of the most fundamental steps, elevator button recognition starts to receive more and more attention. However, due to the challenging image conditions and severe class imbalance problem, the performance of existing results is unsatisfying. In this paper, we propose to combine an optical character recognition (OCR) network and the Faster RCNN architecture into a single neural network, called OCR-RCNN to facilitate an end-to-end training and elevator button recognition procedure. To verify our method, we collect a large dataset of elevator panels and carry out extensive comparative experiments. The experiment results show that our method can greatly outperform the traditional recognition pipelines, yielding an accurate and robust performance on recognizing untrained elevator buttons.**

I. Introduction

Autonomous elevator operation plays an indispensable role in the inter-floor navigation of service robots. It can greatly increase robots' working space and boost their automation level. In contrast to the prosperity of artificial intelligence in the robotics field, current inter-floor navigation systems mainly rely on human assistance or elevator retrofitting, which obviously violates the original ideas that robots are designed to release human beings from tedious work. In order to make the inter-floor navigation fully autonomous, we investigate one of the most fundamental problems, elevator button recognition.

A complete pipeline of elevator operation involves button recognition, button tracking, and visual control. Button recognition refers to the task of detection, localization, and labeling, which is the very first step and also the most challenging one. Specifically, a large variety of button shapes and sizes, panel designs, label types, light conditions and specular reflections make it a very challenging problem. Perspective distortions, lack of low-level image features and unexpected blurs of the images further increase the difficulty of the recognition. To tackle these problems, traditional methods usually cascade a location regression step after button detector, then utilize an OCR algorithm to label each detected button [1]. Since such methods are mostly based on hand-crafted features and follow a separated processing pipeline, they are vulnerable to environmental changes and severely rely on the parameter tuning. The recognition accuracy and robustness are also limited when testing in an untrained elevator panel set.

* The corresponding author

The authors are with the Department of Electronic Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong SAR, China. *email: {dlzhu, tgli, dho, tzhou, qhmeng}@ee.cuhk.edu.hk*

Fig. 1: A demonstration of button recognition result. Black and green bounding boxes indicate the ground truth and predicted buttons, respectively. The number within the box indicates the predicted label.

In this work, we focus on improving the Faster RCNN architecture [2], one of the most advanced convolutional neural networks (CNN) for object detection, to tackle the elevator button recognition problem. A typical Faster RCNN architecture is composed of a region proposal network (RPN), a classification branch and a location refinement branch, which can give the location and class label for each detected object. Different from the typical object detection problem, it is difficult to define all the possible classes for the elevator buttons, because there usually exist a large number of combinations of different numbers and letters, e.g. *LL*, *G1*, *101*. Besides, the class imbalance problem is extremely severe in elevator button dataset. It seems that the number of small numbers is always more than that of the large ones, which can badly decrease the recognition accuracy. Hence, it is not a good practice to regard the labeling task as a classification problem for elevator button recognition. Based on this fact, we propose to graft an OCR branch on the Faster RCNN architecture, integrating the detection, localization and labeling task into a single network. Taking advantage of the OCR branch, big floor numbers are decomposed into several single characters, e.g. *101* becomes (*1, 0, 1*), thus significantly decreasing the class numbers and suppressing the class imbalance problem.

To validate our method, we collected a dataset of 2,100 different elevator panels with 21,767 button labels. After 80,000 training steps, the network achieves a detection precision of 94.6% and a character recognition accuracy of 62.0% in the testing set, which outperforms the result reported in work [1] and [3]. A demonstration is shown in Fig. 1. Based on this work, we also publish a ROS package[1],

[1]The code and demonstrations are available at `https://github.com/zhudelong/elevator_button_recognition.git`

which can output the recognition results and corresponding probabilities. Combining with some prior knowledge of the target elevators, e.g. the panel layout, users will be able to design a robust button recognition system of their own.

II. Related Work

A. Elevator Button Recognition

In contrast to the vast literature on traditional robot navigation [4], only countable publications studied the inter-floor navigation strategies. In spite of human assistance and elevator retrofitting strategy, we will mainly focus on the intelligent elevator operation solutions. Kim *et al.* [5] proposed a button recognition system based on template matching method, which was able to recognize buttons in a particular elevator. Since the system relied much on low-level image features and basic image processing methods, it was vulnerable to light condition changes, and the practicability also remained to be improved. Dong *et al.* [3] and Islam *et al.* [6] presented a similar idea that they formalized the recognition task as a classification problem. Both methods first took advantage of edge information to select some region proposals, then fed some fixed-length features of these proposals into a classification CNN. This idea has much similarity with the RCNN architecture [7], of which the performance is directly restricted by the outcomes of region proposal step.

Another systematic work was reported in [1], where the authors utilized a grid fitting method to regress button locations after a sliding window based object detector. Based on the regression result, a modified LeNet-5 [8] was trained to help carry out OCR task. Using this complicated pipeline, this work reported a recognition accuracy of 88.4% in 50 test images. Besides the small size of test set, we also notice that all the images were in good light conditions without any perspective distortion, which was hard to meet the requirement of real applications. Some other methods related to elevator's external button recognition [9] and visual marker based recognition [10] were also reported, but their abilities to handle untrained elevator panels were very limited.

B. Visual Object Detection

Elevator button recognition can be roughly categorized into the visual object detection task in the computer vision field. Along with the widely application of deep learning technique, a number of detection networks successively reported state-of-the-art performance in several standard datasets. RCNN family [2][7][11][12] inherited the traditional idea of cascading region proposal and classification for object detection and built up an efficient detection architecture with neural networks. Different with RCNN family, YOLO architecture [13] regarded object detection task as a regression problem, which implicitly unified the region proposal step and location regression to one single network, achieving real-time detection performance. SSD [14] integrated several feature maps of different layers to region proposal step, which was capable of detecting more objects of different sizes. Inspired by these deep methods, as well as the particularity of button recognition task, we improve the Faster RCNN architecture by adding an OCR labeling function into the detection network. Compared with traditional pipeline [1], our CNN based method is more accurate and robust after an end-to-end training process.

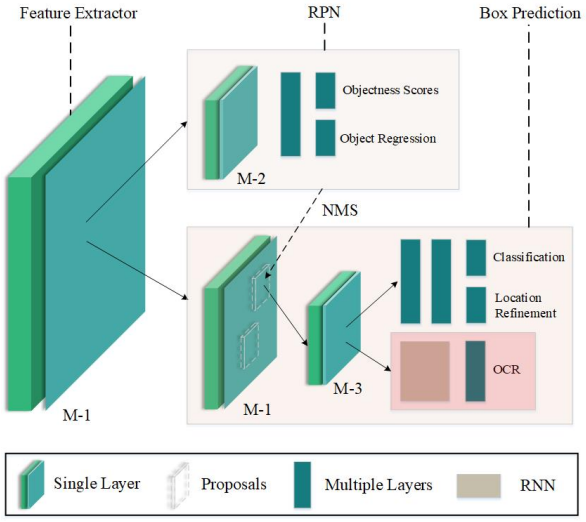

Fig. 2: Network structure. The Feature Extractor first outputs a shared feature map (M-1), then the RPN part takes this map as input and outputs some region proposals. According to these proposals, the Box Prediction part crops corresponding feature patches from M-1 and resizes them to fixed-size ones (M-3). After that, these feature patches are shared by the location refinement, classification and OCR branch for further processing.

III. Methodologies

In this section, we firstly introduce the basic Faster RCNN architecture and the OCR branch, then present our loss function and detailed training strategies.

A. Network Architecture

As shown in Fig. 2, Faster RCNN architecture includes a first-stage *RPN* and a second-stage *Box Prediction* sub-networks. These two sub-networks share the same convolutional feature map (M-1), which is usually generated by a certain *Feature Extractor*, e.g. ResNet-101 [15]. The *RPN* first performs an convolutional operation on the shared feature map and output a *RPN* feature map (M-2), where each element is a feature vector corresponding to a specific region of interest (ROI) in the raw image. This *RPN* feature map is then fed into two branches, the *Objectness Scores* and *Object Regression*, in which the objectness score and object location are inferred for each ROI based on its feature vector. According to the objectness scores, a non-max suppression (NMS) operation is conducted and the object locations with high scores are proposed and passed to the *Box Prediction* stage. In *Box Prediction* stage, the coordinate information of these proposals is used to find out their corresponding feature patches in the shared feature maps (M-1). These patches are then cropped and resized to a batch of fixed-length *Abstract Feature Proposal* (M-3) for further classification and location refinement.

Based on this basic architecture, we make the *Location Refinement* and *Classification* branch share the *Abstract Feature Proposal* with an OCR branch for the labeling task, which is indicated by the red rectangular in Fig. 2. This OCR branch is a simplified version of [16], where an attention mechanism is used to handle the syntactic constituency parsing problem in the NLP field. In this paper, we mainly utilize the attention mechanism to help segment optical characters implicitly. A similar idea is also reported in [17].

Here, we denote the *Abstract Feature Proposal* by $\mathbf{d}$. For each such proposal, the initial input of recurrent neural network (RNN), indicated by the gray block in Fig.2, is set to $x^0 = \mathbf{0}$; when $t > 0$, the RNN is decoded according to the following procedure (visualized in Fig. 3):

$$x^t = W_c c^{t-1} + W_{d1} \underbrace{\sum_i \alpha_i^{t-1} \mathbf{d}_{i,:}}_{\text{previous attention}}, \tag{1}$$

$$(h^t, s^t) = \text{RNNstep}(x^t, s^{t-1}), \tag{2}$$

$$o^t = \text{softmax}(W_h h^t + W_{d3} \underbrace{\sum_i \alpha_i^t \mathbf{d}_{i,:}}_{\text{current attention}}), \tag{3}$$

where W is the parameter matrix, c^{t-1} is the previous predicted character with an one-hot encoding, o^t denotes the prediction over characters at step t and α_i^t is the attention mask obtained by

$$\alpha_i^t = \text{softmax}(V_m^T \tanh(W_s s^t + W_{d2} \mathbf{d}_{i,:})), \tag{4}$$

V^T denotes a parameter vector here. In training phase, c^t is set to the ground-truth character; while in testing phase, we adopt the most likely character according to o^t

$$c^t = \arg\max_{c \in S} \ o^t(c), \tag{5}$$

where S is an encoded charset (Table II) based on the statistics over the entire dataset.

As we can see from this decoding process, the hidden state s^t for $t > 0$ additionally integrates the information of previous attention and previous character by Eq.(1) and Eq.(2). Based on this hidden state s^t, the current attention mask is then predicted (Eq.(4)) and utilized to help predict the current c^t (Eq.(4) and Eq.(5)). Here, the attention mask α tells the network where to look at each step, which can be equivalently regarded as a filter for character segmentation when compared with the workflow of traditional OCR methods. With the help of such an attention mechanism, we successfully integrate the segmentation step and classification step of OCR methods into a single neural branch, thus further enabling the whole architecture to be trained in an end-to-end style.

B. Loss Function

Corresponding to the network structure, we formulate a multi-task loss, which includes *RPN* loss L_p, box prediction loss L_b and OCR loss L_r:

$$L = \alpha_p L_p + \alpha_b L_b + \alpha_r L_r \tag{6}$$

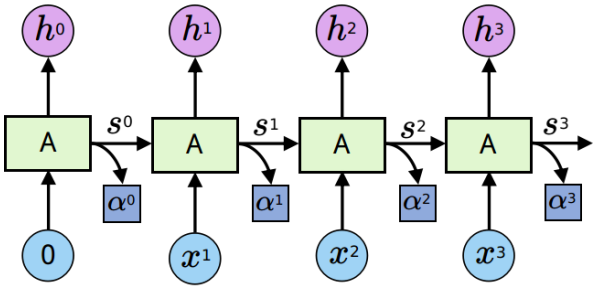

Fig. 3: RNN decoding procedure. The biggest difference from traditional RNN is that an attention mask, denoted by α, is additionally predicted at each step. Each block A is composed of 256 parallel cells for detecting different features.

where α_p, α_b and α_r are balance coefficients. For the *RPN* loss, we adopt the standard objective function presented in Faster RCNN [2]:

$$L_p = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \frac{\lambda}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*), \tag{7}$$

where N_{cls} and N_{reg} are both equal to the size of a randomly sampled mini-batch during *RPN* training process; L_{cls} and L_{reg} denote the binary classification loss and total coordinates regression loss, respectively. (p_i, p_i^*) and (t_i, t_i^*) are the (prediction, ground-truth) pairs of the class labels and object locations, respectively. $p_i^* \in \{0, 1\}$ is a binary variable and each location t includes four parametered coordinates as follows:

$$\begin{aligned} t_x &= (x - x_a)/w_a, \ \ t_y = (y - y_a)/h_a, \\ t_x^* &= (x^* - x_a)/w_a, \ t_y^* = (y^* - y_a)/h_a, \\ t_w &= \log(w/w_a), \ \ t_h = \log(h/h_a), \\ t_w^* &= \log(w^*/w_a), \ t_h^* = \log(h^*/h_a), \end{aligned} \tag{8}$$

variables x, x_a and x^* denote predicted box, proposals and ground-truth box, respectively (likewise for y, w and h).

For box prediction loss, it has a similar formula to Eq.(7), but N_{cls} and N_{reg} both become the number of valid proposals and L_{reg} is replaced by the following L_1 loss:

$$L_{reg} = \begin{cases} 0.5(t_i - t_i^*)^2 & |t_i - t_i^*| < 1 \\ |t_i - t_i^*| - 0.5 & \text{otherwise} \end{cases} \tag{9}$$

Here, since we use OCR branch to discriminate different buttons, the original multi-classification problem is degenerated to a binary classification problem as well.

For OCR loss, we adopt a weighted cross entropy L_{cross} to eliminate the influence of background class:

$$L_r = \frac{1}{N_{ocr}} \sum_i p_i^* \sum_t L_{cross}(o_i^t, r_i^{t*}) \tag{10}$$

where $p_i^* \in \{0, 1\}$ denotes the weight, N_{ocr} equals to the sum of p_i^*, t indicates the RNN decoding step, o_i^t is the RNN output in Eq.(3) for a particular sample, and r_i^{t*} is the corresponding ground truth.

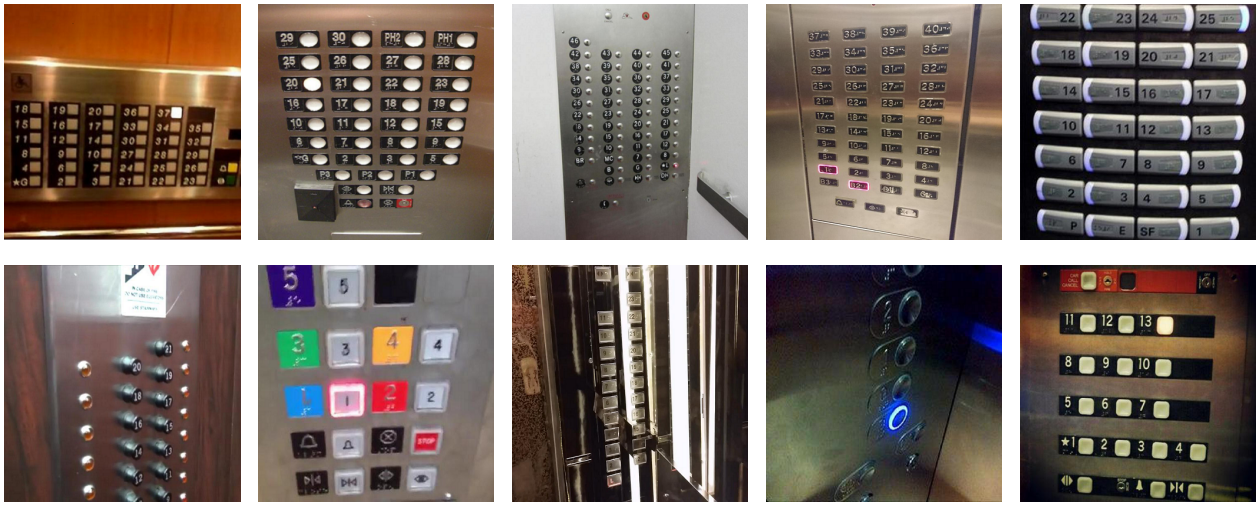

Fig. 4: Some elevator panel designs in the dataset. The difficulties of elevator button recognition is actually beyond our imagination. Different light conditions, perspective distortion and various button contents make it an extremely difficult problem.

TABLE I: Experiment results. The average recall (AR) and average precision (AP) are calculated according to COCO metric [18]. The number suffix denotes button classes that the model can recognize. The sign '-' represents *not reported*.

No.	Model	Testing Set Size			Detection			Character Recognition	
		Panels	Buttons	Distortions	AR^{100}	$AP^{0.5IoU}$	F-score	Classes	Accuracy
1	Traditional Pipeline [1]	50	686	N	0.862	0.881	0.875	15	**0.884**
2	CNN based Classification [3]	4	60	Y	**0.900**	0.876	0.888	15	-
3	SSD-2 [14]	630	6869	Y	0.801	0.856	0.828	-	-
4	RFCN-2 [12]	630	6869	Y	0.719	0.926	0.809	-	-
5	Faster RCNN-2 [2]	630	6869	Y	0.742	0.941	0.830	-	-
6	Faster RCNN-15 [2]	630	6869	Y	0.883	0.901	**0.891**	15	0.837
7	OCR-RCNN	**630**	**6869**	Y	0.726	**0.946**	0.822	**365**	0.620

C. Training Strategies

The proposed network is trained in an end-to-end style by back-propagation and a Momentum Optimizer. To train the *RPN*, we specify a mini-batch sampling size of 128 for each training step, and add an additional constraint that each ground-truth target is at least assigned with a proposal. For localization task, considering the small size of elevator buttons, we utilize a smaller basic anchor, i.e. 200x200, with a scale list of [0.5, 1.0, 2.0, 3.0] and an aspect ration list of [0.8, 1.0, 1.3]. For recognition task, we assume a maximum sequence length of 3 for OCR branch. If the length is less than 3, we pad the ground-truth sequence label with *null*. To balance the three loss functions in Sec.III-B, we set α_p=α_b=2.0, α_r=10 and λ=1.0.

In this paper, the final inception block *Conv2d_7b_1x1* of Inception Resnet V2 [19] is adopted as the feature extractor, which is initialized by a pre-trained model on ImageNet classification task [20]. We also initialize the RNN layers in OCR branch with an orthogonal initializer [21] and initialize all other layers using a zero-mean Gaussian distribution with a variance of 0.001. The learning rate is set to 0.002 in the first 20k steps, then 0.0002 in the next 20k steps, finally 0.00002 for the remaining 40k steps. We implement our method based on work [17] and [22] using a GeForce TITAN-X GPU card. A more detailed setting can be found in the code repository.

IV. Experiments

A. Elevator Button Dataset

In order to evaluate the performance of our method, we collect an elevator button dataset that includes both external and internal panel images. This dataset totally contains 2,100 different elevator panels with 21,767 button labels and 365 different classes. Among these classes, 150 classes have only one sample and 84 classes have 2-4 samples, which means nearly 3/4 classes don't have enough training samples (shown in Fig. 5). Such distributions are actually determined by the properties of elevator panels rather than data collection methods. For instance, almost every elevator has floor 1-5, but few elevators have floor 101. Apparently, we cannot directly apply the existing object detection algorithms to our

TABLE II: Charset encoding schema. We use some special characters to represent different function buttons, i.e. ><for *close*, $ for *alarm*, # for *stop*, & for *call*, (for *up*, 's' for *star*, * for *textureless button*, % for *special function button*, ? for *hazy button*, ! for *unknown text button*, ' ' for *null*.

'1': 1	'2': 2	'<': 3	'>': 4	'3': 5	'!': 6
'4': 7	'*': 8	'5': 9	'?': 10	'6': 11	'7': 12
'8': 13	'%': 14	'0': 15	'$': 16	'9': 17	'G': 18
'B': 19	'(': 20	')': 21	'&': 22	'L': 23	's': 24
'P': 25	'-': 26	'M': 27	'#': 28	'U': 29	'D': 30
'R': 31	'A': 32	'C': 33	'S': 34	'E': 35	'F': 36
'O': 37	'K': 38	'H': 39	'N': 40	'T': 41	'V': 42
'I': 43	'Z': 44	'J': 45	'X': 46	' ': 47	

TABLE III: Button size distribution. The width and height are measured in 100-pixels, e.g. 2-3 represents (200,300] pixels.

Item	0-1	1-2	2-3	3-4	4-5	5-6	>6
Width	17,485	3,478	573	194	28	4	5
Height	15,903	4,517	854	353	99	21	20

dataset due to the variety of label classes. To handle this problem, we decompose the content of text buttons into a sequence of characters, e.g. -L1 becomes (-, L, 1) and encode function buttons with punctuations, e.g. *open* is denoted by<>. With this approach, the floor number 101 can be sequentially recognized by a 0-1 OCR reader. Accordingly, the class scale will be significantly decreased and the class imbalance problem will also be suppressed. The detailed encoding schema is shown in Table II and its corresponding number distribution is visualized in Fig. 6. It can be seen that we only need to deal with 47 classes and nearly half of these classes can be effectively trained.

In spite of the diversity of label classes, the panel images are also quite challenging. There exist a variety of different panel designs, button shapes, and perspective distortions in the dataset. Restricted by these challenge conditions, it is indeed a difficult task to correctly recognize target buttons. As a demonstration, we show some different type of panels in Fig. 4 and a statistic about button size is also reported in Table III. Compared with work [5][6] that only deal with a particular elevator, and work [1][3] that test their algorithms in a limited number of new panels, this dataset can provide a more reliable and fairer evaluation for our method.

B. Experiment Results

Before training, the dataset is randomly divided into a training set and a testing set with a ratio of 7:3. The training set includes 1,470 images and 14,898 buttons, and the testing set includes 630 images and 6,869 buttons. Using this dataset, we conduct several comparative experiments to evaluate our method. Firstly, we directly present the detection and character recognition performance reported in work [1] (traditional pipeline) and [3] (CNN based classification), which serve as button recognition baselines[2]. Secondly, we

[2]Since their code and dataset are not available, we are not able to test our method in their dataset or test their methods in our dataset.

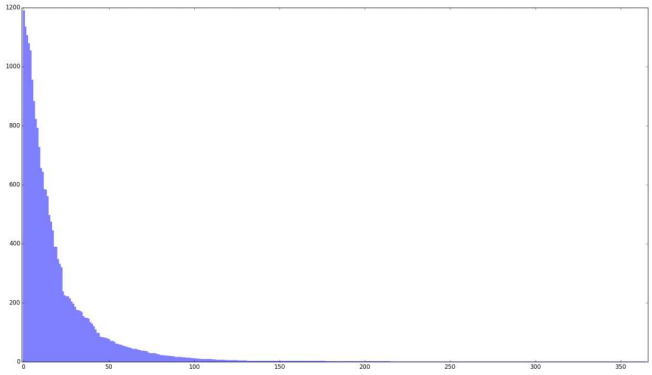

Fig. 5: Class distribution among 365 classes. x-axis indicates the class numbers and y-axis indicates the corresponding numbers. There exist a severe class imbalance problem, and nearly 3/4 classes do not have enough training samples.

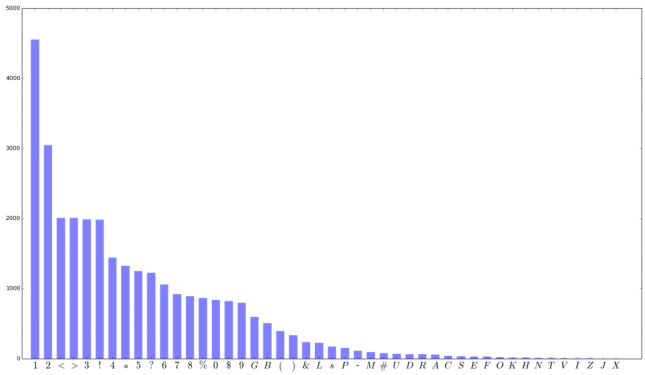

Fig. 6: Char distribution among 47 classes. x-axis is corresponding to the characters in Table II and y-axis indicates the numbers. There still exists a class imbalance problem, but much better than the original label's distribution.

train three start-of-art networks as button detection baselines, i.e. Faster RCNN, RFCN and SSD, which are fine tuned to a comparatively good performance. At last, we train an OCR-RCNN with the char encoding schema, as well as an additional Faster RCNN that can recognize 15 commonly used buttons. After 80k-step training, we start to test our method and compare it with the baselines in three aspects, the testing set size, detection performance and OCR performance. The localization threshold is set to 0.5IoU [18] during the whole experiments. If the IoU between the predicted location and ground-truth location is greater than 0.5, we regard it as a successful detection. The detailed comparative results are reported in Table I, where the best performance of each column is highlighted.

By comparing the Row 1, 2, and 7, we can see that our method outperforms the traditional pipeline and CNN based classification method in terms of testing set size, detection precision and the number of button classes. Since a lot of testing images have severe perspective distortions and specular reflections, the OCR fails to recognize a complete character sequence, thus leading to low accuracy, but compared with the traditional pipeline in Row 1, where the dataset is composed of clear images without any distortion, the accuracy of our OCR is still competitive. We notice that

the performance of CNN based classification method in Row 2 is only based on 4 different elevator panels with 60 buttons. Although it reports a high average recall, the fairness and reliability of this result may need further evaluation. In order to verify the class balancing function of our method, we also visualize the distribution of original classes in Fig. 5 and that of the charset in Fig. 6. It can be seen that the class imbalance problem is much suppressed by char encoding (Table II). Besides, taking advantage of the char encoding schema, our method can recognize a large number of button classes, which is more accessible to real applications.

By comparing Row 3, 4, 5 and 7, we can see that the detection performance of OCR-RCNN is only slightly affected by the OCR branch, and the character recognition accuracy is also acceptable. The decrease of detection performance may be caused by the different training goals of the classification branch and OCR branch in Fig. 2, where the binary classification branch tries to minimize the differences within button classes, while the OCR branch wants to extract discriminative features from the shared *Abstract Feature Proposals*. Besides, according to Eq.(10), the OCR branch only recognizes the positive outputs of the classification branch. Consequently, the false negative outputs cannot be labeled and false positive outputs are mislabeled. which can also cause the lower recognition accuracy. Such a working schema needs to be improved by more intelligent approaches.

Additionally, we also train a Faster RCNN with 15 classes of commonly used buttons, i.e. *(1-9, G, B, L, open, close, alarm)*, which is consistent with work [1]. The testing result is reported in Row 6. Compared with Row 7, the character recognition accuracy of Faster RCNN-15 is quite well. For most low-rise buildings, this model may be more effective and useful.

V. Conclusion

In this paper, we present an OCR-RCNN architecture for the elevator button detection. The architecture integrates the detection, localization and recognition tasks into a single neural network, thus enabling an end-to-end training approach. To verify our method, we collected a large button dataset and trained the OCR-RCNN on it, yielding a good performance. The experiments show that our method can effectively handle the data imbalance problem, and generalize well on untrained elevator panel images. For the current version, the implementation is not efficient enough. A main goal for the future work is to increase the system's computational efficiency.

Acknowledgment

This project is partially supported by the Hong Kong RGC GRF grants #14200618 and Shenzhen Science and Technology Innovation projects c.02.17.00601 awarded to Max Q.-H. Meng.

References

[1] E. Klingbeil, B. Carpenter, O. Russakovsky, and A. Y. Ng, "Autonomous operation of novel elevators for robot navigation," in *IEEE International Conference on Robotics and Automation*, 2010, pp. 751–758.

[2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," in *International Conference on Neural Information Processing Systems*, 2015, pp. 91–99.

[3] Z. Dong, D. Zhu, and M. Q.-H. Meng, "An autonomous elevator button recognition system based on convolutional neural networks," in *Robotics and Biomimetics (ROBIO), 2017 IEEE International Conference on*. IEEE, 2017, pp. 2533–2539.

[4] D. Zhu, T. Li, D. Ho, and M. Q.-H. Meng, "Deep reinforcement learning supervised autonomous exploration in office environments," in *Robotics and Automation (ICRA), 2018 IEEE International Conference on*. IEEE, 2018.

[5] H. H. Kim, D. J. Kim, and K. H. Park, "Robust elevator button recognition in the presence of partial occlusion and clutter by specular reflections," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 3, pp. 1597–1611, 2011.

[6] K. T. Islam, G. Mujtaba, R. G. Raj, and H. F. Nweke, "Elevator button and floor number recognition through hybrid image classification approach for navigation of service robot in buildings," in *International Conference on Engineering Technology and Technopreneurship*, 2017, pp. 1–4.

[7] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *CoRR*, vol. abs/1311.2524, 2013. [Online]. Available: http://arxiv.org/abs/1311.2524

[8] Y. Lcun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] N. F. W. Z. Wan, M. R. Daud, S. Razali, and M. F. Abas, "Elevators external button recognition and detection for vision-based system," *Proceeding of the Electrical Engineering Computer Science & Informatics*, 2014.

[10] A. A. Abdulla, H. Liu, N. Stoll, and K. Thurow, "A robust method for elevator operation in semi-outdoor environment for mobile robot transportation system in life science laboratories," in *IEEE Jubilee International Conference on Intelligent Engineering Systems*, 2016.

[11] R. Girshick, "Fast r-cnn," *Computer Science*, 2015.

[12] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," 2016.

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[14] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European Conference on Computer Vision*, 2016, pp. 21–37.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," pp. 770–778, 2015.

[16] O. Vinyals, L. Kaiser, T. Koo, S. Petrov, I. Sutskever, and G. E. Hinton, "Grammar as a foreign language," *CoRR*, vol. abs/1412.7449, 2014. [Online]. Available: http://arxiv.org/abs/1412.7449

[17] Z. Wojna, A. Gorban, D. S. Lee, K. Murphy, Q. Yu, Y. Li, and J. Ibarz, "Attention-based extraction of structured information from street view imagery," 2017.

[18] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[19] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, inception-resnet and the impact of residual connections on learning," *CoRR*, vol. abs/1602.07261, 2016. [Online]. Available: http://arxiv.org/abs/1602.07261

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[21] A. M. Saxe, J. L. McClelland, and S. Ganguli, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks," *CoRR*, vol. abs/1312.6120, 2013. [Online]. Available: http://arxiv.org/abs/1312.6120

[22] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors," *CoRR*, vol. abs/1611.10012, 2016. [Online]. Available: http://arxiv.org/abs/1611.10012