Supplementary material of Submission 784

1. Dynamic programming:

   In our main submission, we claim that the maximization for each structure could be done efficiently with dynamic programming. More details are as follows:

   Let child(i) be the set of children of i in V_k. The messages that part i passes to its parent part j could be computed by the following:

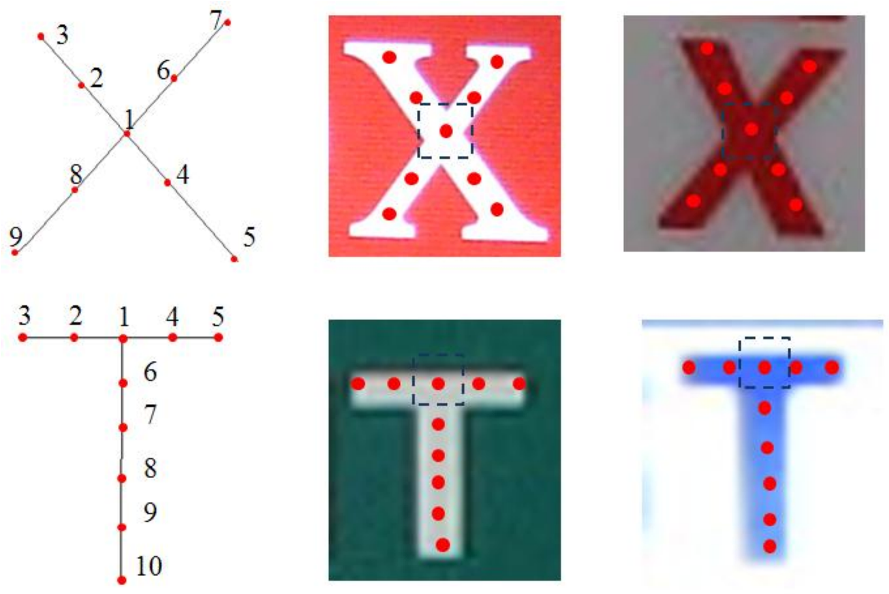
$$score_i(l_i) = w_i\varphi(I, l_i) + \sum_{k\in child(i)} m_{k\to i}(l_i) \quad (1)$$

$$m_{i\to j}(l_j) = \max_{l_i}(score_i(l_i) + w_{i,j}\cdot \emptyset(l_i - l_j)) \quad (2)$$

   Eq. (1) computes the local score of part i, which equals to the appearance score of part i plus the messages collected from the children of i. Eq. (2) computes the best scoring location of its child part i, for every locations of part j. We could first compute the local appearance score of the leaf parts, and then the score of their parent parts could be computed. Once messages are passed to the root part (j = 1), $score_1(l_1)$ represents the best scoring configuration for each root position. We can use these scores to generate multiple detections in character images I by using suitable thresh and applying non-maximum suppression (NMS). By keeping track of the argmax indices in (2), we can backtrack to find the locations of each part in each maximal configuration.

2. Learning for part-based tree-structured model

   In our main submission, we state that we design the tree-structure for each type of character by our experience. Next we will give more details about how we design structures for different types of characters and how we label the parts. After designing the tree-structure for each type of character as shown in Fig. 1 (a), we manually label the center of each part and extract features from the region centered by the node to represent the part as shown in Fig. 1 (b)-(c).



(a) (b) (c)

Figure 1: Illustration of how we design tree structure for different characters and how we label parts for training samples. (a) Tree structure for 'X' and 'T', where red points are the nodes of the tree and '1' refers to the root node. (b)-(c) Examples of how we label the parts. We only label the centroid (red points) of each part and extract features from the regions shown in dashed rectangles to represent the parts.

3. More character detection results on scene text images

We give more character detection results on scene text images as shown in Figure 2. The results are acquired after applying NMS on the raw detection results. The red rectangle labels the position of the root node of the tree while the blue ones label other parts of the character. As we can see, as the tree-structured model makes use of both global structure information and local appearance information, the detection results contain less false positives and are more reliable.

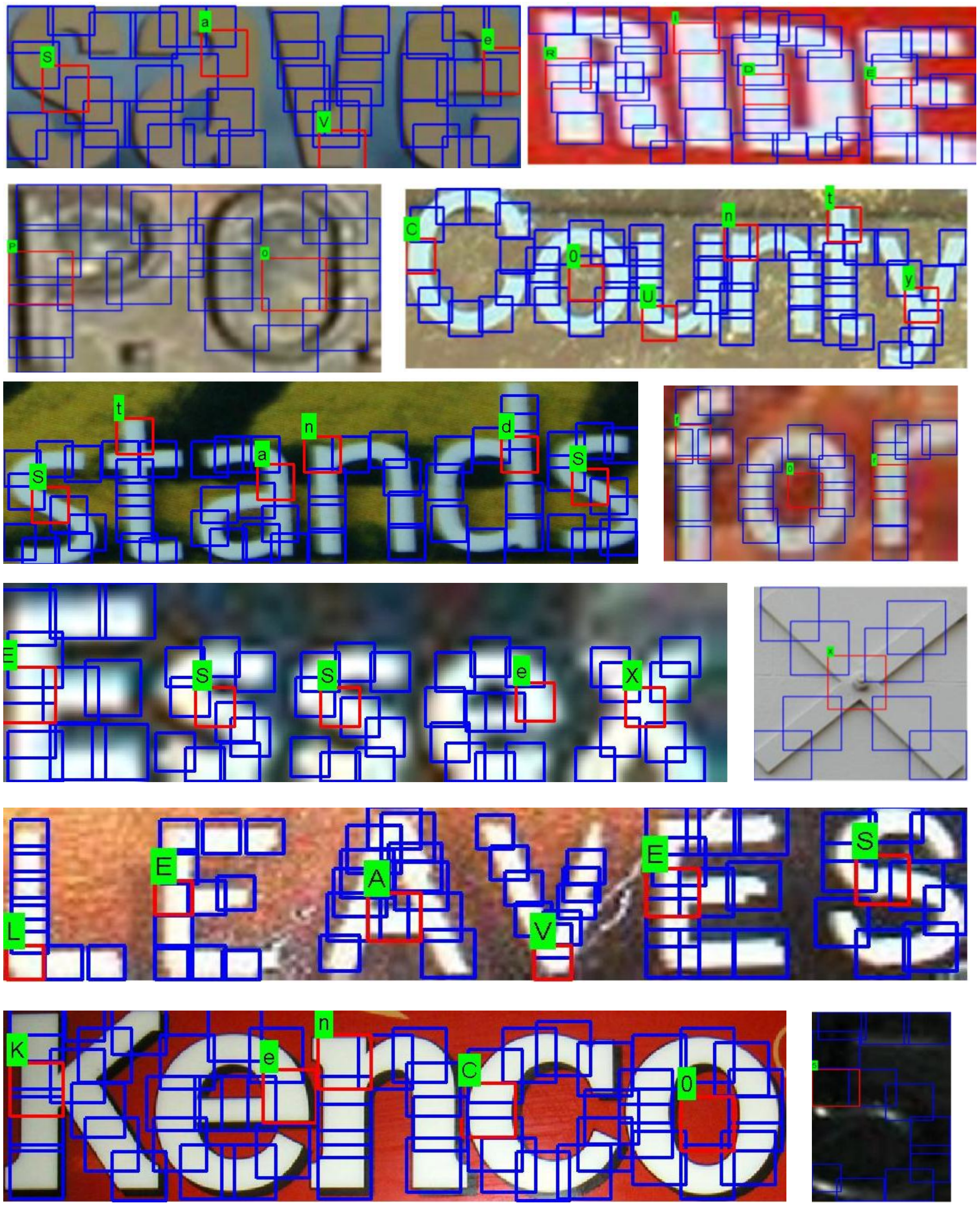


Figure 2: Some character detection results on scene text images.