A Segmentation-aware Object Detection Model with Occlusion Handling Supplementary Material

Tianshi Gao[1] Benjamin Packer [2] Daphne Koller[2]
[1] Department of Electrical Engineering, Stanford University
[2] Department of Computer Science, Stanford University
{tianshig,bpacker,koller}@cs.stanford.edu

1. Introduction

In this supplementary material, we provide more detailed derivations of the bound used in our efficient inference with branch-and-bound introduced in section 4.2 of the submitted paper. Furthermore, for readers who are not familiar with the branch-and-bound algorithm provide the detailed branch-and-bound procedure in a pseudo-code level as described in section 4.2 of the paper.

2. Derivation of the bound

The objective function for our inference is as follows:

$$\begin{aligned} &F(\mathbf{x}_k,\ \mathbf{y};\mathbf{w}) + \Delta(\mathbf{y}_k,\mathbf{y}) \\ &= F^h(\mathbf{x}_k,\mathbf{p},\mathbf{v}) + F^{\hat{h}}(\mathbf{x}_k,\mathbf{p},\mathbf{v}) + F^{prior}(\mathbf{v}) \\ &\quad + F^c(\mathbf{x}_k,\mathbf{p}) + F^{pair}(\mathbf{v}) + F^{loss}(\mathbf{x}_k,\mathbf{p}_k,\mathbf{p}) \end{aligned} \tag{1}$$

where the definitions of different terms are described in section 4.1 in the paper. The inference task given a candidate space $\tilde{\mathcal{P}} \times \mathcal{V}$ is:

$$\underset{(\mathbf{p},\mathbf{v})\in\tilde{\mathcal{P}}\times\mathcal{V}}{\text{argmax}}\ F(\mathbf{x}_k,\ \mathbf{y};\mathbf{w}) + \Delta(\mathbf{y}_k,\mathbf{y}) \tag{2}$$

We group the objective 1 into three parts: (i) terms that involve the visibility variables $\mathbf{v}$; (ii) the term for empty cell count, *i.e.*, $F^c(\mathbf{x}_k,\mathbf{p})$; (iii) the term for the loss, *i.e.*, $F^{loss}(\mathbf{x}_k,\mathbf{p}_k,\mathbf{p})$. Specifically, the first part is:

$$\begin{aligned} F^{\mathbf{v}}(\mathbf{x}_k,\mathbf{p},\mathbf{v}) &= F^h(\mathbf{x}_k,\mathbf{p},\mathbf{v}) + F^{\hat{h}}(\mathbf{x}_k,\mathbf{p},\mathbf{v}) \\ &\quad + F^{prior}(\mathbf{v}) + F^{pair}(\mathbf{v}) \end{aligned} \tag{3}$$

In the branch-and-bound algorithm, we need an upper bound for the scores of all candidates $(\mathbf{p},\mathbf{v})$ in a candidate space $\tilde{\mathcal{P}} \times \mathcal{V}$, *i.e.*, we want to find a $U : \tilde{\mathcal{P}} \times \mathcal{V} \rightarrow \mathbb{R}$ such that

$$\max_{(\mathbf{p},\mathbf{v})\in\tilde{\mathcal{P}}\times\mathcal{V}} F(\mathbf{x}_k,\ \mathbf{y};\mathbf{w}) + \Delta(\mathbf{y}_k,\mathbf{y}) \leq U(\tilde{\mathcal{P}}\times\mathcal{V}) \tag{4}$$

As described in the paper, we derive the bound by pushing the maximization into the three components of Equation 1:

$$\begin{aligned} &\max_{(\mathbf{p},\mathbf{v})\in\tilde{\mathcal{P}}\times\mathcal{V}} F(\mathbf{x}_k,\ \mathbf{y};\mathbf{w}) + \Delta(\mathbf{y}_k,\mathbf{y}) \\ &= \max_{(\mathbf{p},\mathbf{v})} \Big(F^{\mathbf{v}}(\mathbf{x}_k,\mathbf{p},\mathbf{v}) + F^c(\mathbf{x}_k,\mathbf{p}) + F^{loss}(\mathbf{x}_k,\mathbf{p}_k,\mathbf{p}) \Big) \\ &\leq \max_{(\mathbf{p},\mathbf{v})} F^{\mathbf{v}}(\mathbf{x}_k,\mathbf{p},\mathbf{v}) + \max_{\mathbf{p}} F^c(\mathbf{x}_k,\mathbf{p}) \\ &\quad + \max_{\mathbf{p}} F^{loss}(\mathbf{x}_k,\mathbf{p}_k,\mathbf{p}) \end{aligned} \tag{5}$$

We've derived the bounds for the second (the term corresponding to the empty cell count) and third (the term corresponding to the loss) components, *i.e.*, $U^c(\tilde{\mathcal{P}})$ and $U^{loss}(\tilde{\mathcal{P}})$ as shown in equation (9) and equation (10) respectively in the paper. Here, we provide the deriviation of the bound for the first part (terms involving $\mathbf{v}$), *i.e.*, $U^v(\tilde{\mathcal{P}} \times \mathcal{V})$ as shown in equation (8) in the paper. Specifically,

$$\begin{aligned} &\max_{(\mathbf{p},\mathbf{v})\in\tilde{\mathcal{P}}\times\mathcal{V}} F^{\mathbf{v}}(\mathbf{x}_k,\mathbf{p},\mathbf{v}) \\ &= \max_{(\mathbf{p},\mathbf{v})\in\tilde{\mathcal{P}}\times\mathcal{V}} \Big(F^h(\mathbf{x}_k,\mathbf{p},\mathbf{v}) + F^{\hat{h}}(\mathbf{x}_k,\mathbf{p},\mathbf{v}) \\ &\quad + F^{prior}(\mathbf{v}) + F^{pair}(\mathbf{v}) \Big) \\ &\leq \max_{\mathbf{v}} \Big(\max_{\mathbf{p}} F^h(\mathbf{x}_k,\mathbf{p},\mathbf{v}) + \max_{\mathbf{p}} F^{\hat{h}}(\mathbf{x}_k,\mathbf{p},\mathbf{v}) \\ &\quad + F^{prior}(\mathbf{v}) + F^{pair}(\mathbf{v}) \Big) \end{aligned} \tag{6}$$

Furthermore,

$$\begin{aligned} \max_{\mathbf{p}\in\tilde{\mathcal{P}}} F^h(\mathbf{x}_k,\mathbf{p},\mathbf{v}) &= \max_{\mathbf{p}\in\tilde{\mathcal{P}}} \sum_{i=1}^{wh} F_i^h(\mathbf{x}_k,\mathbf{p},v_i) \\ &\leq \sum_{i=1}^{wh} \max_{\mathbf{p}\in\tilde{\mathcal{P}}} \Big(\mathbf{w}_{h,i}^T\, \phi_i(\mathbf{x}_k,\mathbf{p}) \Big) \cdot v_i \end{aligned} \tag{7}$$

Similarly,

$$\max_{\mathbf{p}\in\tilde{\mathcal{P}}} F^{\hat{h}}(\mathbf{x}_k,\mathbf{p},\mathbf{v}) = \max_{\mathbf{p}\in\tilde{\mathcal{P}}} \sum_{i=1}^{wh} F_i^{\hat{h}}(\mathbf{x}_k,\mathbf{p},v_i)$$
$$\leq \sum_{i=1}^{wh} \max_{\mathbf{p}\in\tilde{\mathcal{P}}} \left(\mathbf{w}_{\hat{h},i}^T\, \phi_i(\mathbf{x}_k,\mathbf{p})\right)\cdot(1-v_i) \tag{8}$$

These two inequalities mean that for each cell, we use the best HOG response among all bounding boxes $\mathbf{p}\in\tilde{\mathcal{P}}$ with respect to the model. This maximization can be done very efficiently by caching the cell response. Combining Equation 7 and Equation 8 with $F^{prior}(\mathbf{v})$ and $F^{pair}(\mathbf{v})$, the right hand side of the inequality of Equation 6 becomes:

$$U^v(\tilde{\mathcal{P}}\times\mathcal{V}) = \max_{\mathbf{v}}\Big(\sum_{i=1}^{wh}\max_{\mathbf{p}}\left(\mathbf{w}_{h,i}^T\,\phi_i(\mathbf{x}_k,\mathbf{p})\right)\cdot v_i$$
$$+\sum_{i=1}^{wh}\max_{\mathbf{p}}(\mathbf{w}_{\hat{h},i}^T\,\phi_i(\mathbf{x}_k,\mathbf{p}))\cdot(1-v_i)$$
$$+F^{prior}(\mathbf{v})+F^{pair}(\mathbf{v})\Big) \tag{9}$$

Based on Equation 6, we have $\max_{(\mathbf{p},\mathbf{v})\in\tilde{\mathcal{P}}\times\mathcal{V}} F^{\mathbf{v}}(\mathbf{x}_k,\mathbf{p},\mathbf{v}) \leq U^v(\tilde{\mathcal{P}}\times\mathcal{V})$. Therefore, $U^v(\tilde{\mathcal{P}}\times\mathcal{V})$ is the bound for the first component of the objective. Furthermore, to compute the bound,*i.e.*, solving the MAP problem over $\mathbf{v}$, a single graph cut is sufficient. The cost of this computation is independent of $|\tilde{\mathcal{P}}|$ (number of bounding boxes in $\tilde{\mathcal{P}}$).

Finally, combining the bounds for three parts of the objective, we get the bound for the scores of a candidate space $\tilde{\mathcal{P}}\times\mathcal{V}$:

$$\max_{(\mathbf{p},\mathbf{v})\in\tilde{\mathcal{P}}\times\mathcal{V}} F(\mathbf{x}_k,\,\mathbf{y};\mathbf{w})+\Delta(\mathbf{y}_k,\mathbf{y})$$
$$\leq U^v(\tilde{\mathcal{P}}\times\mathcal{V})+U^c(\tilde{\mathcal{P}})+U^{loss}(\tilde{\mathcal{P}})$$
$$\stackrel{\mathbf{def}}{=} U(\tilde{\mathcal{P}}\times\mathcal{V}) \tag{10}$$

3. Efficient inference with branch-and-bound

For readers who are not familiar with the branch-and-bound procedure applied to window search [1], we elaborate the description of the algorithm (section 4.2 in the paper) by providing a high-level pseudo code description as shown Algorithm 1. For the parameterization of the space, a bounding box space $\tilde{\mathcal{P}}$ is specified by a subspace of the image pyramids $[s_1,s_2]\times[w_1,w_2]\times[h_1,h_2]$ where s, w and h are the scale, width and height (normalized to $[0,1]$). The full image pyramid space is then defined as $[0,1]\times[0,1]\times[0,1]$. In our implementation, when we split the bounding box space $\tilde{\mathcal{P}}$ as line 4 in Algorithm 1, we split the dimension with the largest interval into half. For example, given a bounding box subspace $\tilde{\mathcal{P}} = [0,0.5]\times[0,0.25]\times[0.25,0.5]$, we split it into two disjoint subspaces: $\tilde{\mathcal{P}}_1 = [0,0.25]\times[0,0.25]\times[0.25,0.5]$ and $\tilde{\mathcal{P}}_2 = (0.25,0.5]\times[0,0.25]\times[0.25,0.5]$.

Algorithm 1 Efficient Inference with Branch-and-Bound

Require: Image I
Require: Score bounding function U (Equation 10)
1: Initialize P as an empty priority queue
2: Initialize $\tilde{\mathcal{P}}$ as the whole image pyramid space, i.e., $\tilde{\mathcal{P}} = [s_1,s_2]\times[w_1,w_2]\times[h_1,h_2] = [0,1]\times[0,1]\times[0,1]$
3: **repeat**
4: split $\tilde{\mathcal{P}} \rightarrow \tilde{\mathcal{P}}_1 \cup \tilde{\mathcal{P}}_2$
5: push $(U(\tilde{\mathcal{P}}_1\times\mathcal{V}),\tilde{\mathcal{P}}_1)$ and $(U(\tilde{\mathcal{P}}_2\times\mathcal{V}),\tilde{\mathcal{P}}_2)$ into P
6: retrieve the top state $\tilde{\mathcal{P}}$, *i.e.*, $U(\tilde{\mathcal{P}}\times\mathcal{V})$ is the largest, from P
7: **until** $\tilde{\mathcal{P}}$ consists of only one element, *i.e.*, $\tilde{\mathcal{P}} = \{\mathbf{p}^*\}$
Return: p^* and it's corresponding $\mathbf{v}^*$

References

[1] C. Lampert, M. Blaschko, and T. Hofmann. Beyond sliding windows:object localization by efficient subwindow search. In *CVPR*, 2008. 2