Bi-level Alignment for Cross-Domain Crowd Counting
Supplementary Material

Shenjian Gong[1], Shanshan Zhang[1,*], Jian Yang[1], Dengxin Dai[2], and Bernt Schiele[2]
[1]PCA Lab, Key Lab of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, and Jiangsu Key Lab of Image and Video Understanding for Social Security, School of Computer Science and Engineering, Nanjing University of Science and Technology
[2]MPI Informatics
{shenjiangong, shanshan.zhang, csjyang}@njust.edu.cn
{ddai, schiele}@mpi-inf.mpg.de

In this supplementary material, we first provide additional ablation studies, which are all implemented on SHA; and then we illustrate the detailed architecture of our network.

S1. Impact of Segmentation Threshold

For each image, we obtain its ground truth density map as follows:

$$y(n) = \sum_{i=1}^{N} \delta(n - a_i) * G_\sigma(n), \tag{1}$$

where n denotes the pixel index; a_i represents the i-th annotated head point (total N points); δ is the delta function; G_σ is the Gaussian kernel with a size of 15×15 and a variance of $\sigma = 4$.

Our fine-grained feature alignment strategy does feature alignment on foreground and background separately. To separate foreground and background regions from the density maps, we apply a threshold close to zero on each local patch. We analyze how the segmentation threshold th affects the performance. As shown in Tab. S1, the value of th has a very small impact on the counting performance, indicating that our method is not sensitive to it. Since 0.005 gives the best performance, we choose $th = 0.005$ for all our experiments.

Segmentation threshold th	MAE	MSE
0.0005	102.9	155.7
0.001	99.4	147.9
0.005	99.3	145.0

Table S1. Effect of different segmentation threshold th.

S2. Style Transfer from S+ to Target

Our task-driven data alignment tries to select the best combination of augmentation to reduce the domain gap between the source and target domains. This is not in conflict with style transfer. Therefore we train a style transfer model between the augmented-source (S+) and target domains. From Tab. S2, we can see that a style transfer model between the S^+ and target domain can bring extra improvement.

Method	MAE	MSE
S^+ to Target	97.8	145.1
BLA	99.3	145.0

Table S2. Effect of Style Transfer from S+ to Target.

S3. Effect of More Transformations

For crowd counting, following [2], we choose RGB2Gray, scaling and perspective transform as our data-augmentation methods. Here, we consider to add other augmentation methods such as brightness, contrast and saturation randomization to our BLA. From Tab. S3, we find more transformations can not bring more improvements. This enlightened us that, for crowd counting, inter-domain differences are mainly reflected in saturation, scale and perspective.

*Corresponding author

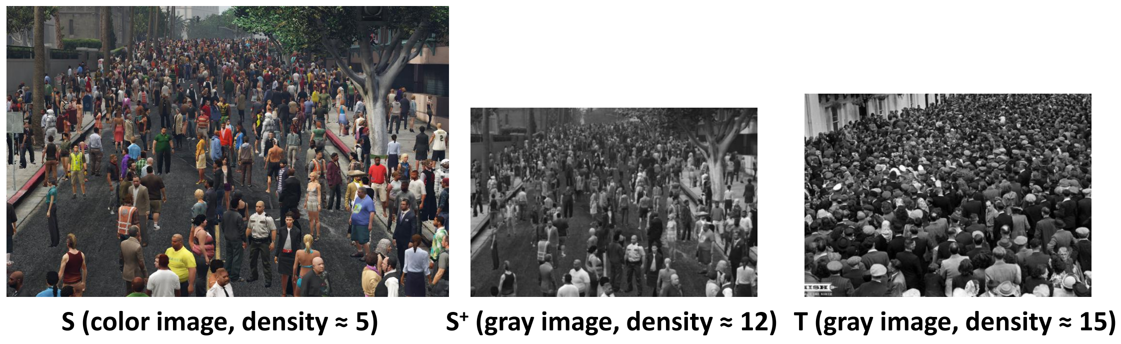
S (color image, density ≈ 5)
S⁺ (gray image, density ≈ 12)
T (gray image, density ≈ 15)

Figure S1. The transformed image (S^+) is closer to the target image (T) w.r.t. saturation and density (#heads per 100×100 patch).

Method	MAE	MSE
BLA(More Transformations)	101.1	151.6
BLA	99.3	145.0

Table S3. Effect of More Augmentation Methods.

Method	MAE	MSE
Grid Search	111.9	173.4
Ours	99.3	145.0

Table S4. Effect of our AutoML based search method.

S4. How Does Task-Driven Data Alignment Reduce the Domain Gap?

From Fig. S1, by observing the images of the source domain and the target domain, we can find that there are great differences in saturation and crowd density between the two domains. Our task-driven data-alignment mainly aligns the source domain with the target domain in color and content. Data augmentation makes counter more robust to saturation, scale and perspective. On the other hand, smaller inter-domain differences make the model better generalized to the target domain.

Comparison with Grid Search In order to illustrate the effect of our AutoML based search algorithm, we compare it with grid search [1], which is a straight forward way of searching. For fair comparison, we implement the experiments in our framework by only replacing our AutoML based search with grid search and keeping other components the same.

For each parameter, we set the size of the grid to 5 and distribute the parameters evenly, specifically, angle $\in \{0°, 11°, 22°, 33°, 45°\}$ $P_G \in \{0, 0.25, 0.5, 0.75, 1.0\}$, and scale factor $\in \{0, 0.25, 0.5, 0.75, 1.0\}$. We employ candidate transform validation and find the best transform. As shown in Tab. S4, the best results we obtain through grid search is 111.9 w.r.t. MAE and 173.4 w.r.t. MSE. In contrast, our AutoML based search algorithm achieves 99.3 w.r.t. MAE and 145.0 w.r.t. MSE. These results demonstrate the effectiveness of our controller.

S5. Detailed Architecture of Our Network

There are four types of layers in our network, including convolutional layer (Conv), fully connected layer (FC), deconvolution layer (Deconv) and max pooling layer (Maxpool). We use the following notation to indicate the settings of a convolutional/deconvolutional layer: Conv/Deconv: [k(3,3)-c256-s1-BN-Relu] representing a convolutional/deconvolutional operation with a kernel size of 3×3, number of output channels of 256, stride of 1, with Batch Normalization and ReLU applied afterwards. Similarly, Maxpool: [k(2,2)-s2] denotes a max pooling layer with a kernel size of 2×2 and stride of 2; FC: [c16-Relu] represents a fully connected layer with size of 16, activated by Relu. We show the detailed network architecture of our $\mathcal{F}$, $\mathcal{E}$, $\mathcal{D}$ and controller $\mathcal{C}$ in Tab. S5.

Input	Layer	Output
	$\mathcal{F}$	
3×576×768	Conv:[k(3,3)-c64-s1-Relu] × 2	64×576×768
64×576×768	Maxpool:[k(2,2)-s2]	64×288×384
64×288×384	Conv:[k(3,3)-c128-s1-Relu]× 2	128×288×384
128×288×384	Maxpool:[k(2,2)-s2]	128×144×192
128×144×192	Conv:[k(3,3)-c256-s1-Relu]× 3	256×144×192
256×144×192	Maxpool:[k(2,2)-s2]	256×72×96
256×72×96	Conv:[k(3,3)-c512-s1-Relu] × 3	512×72×96
	$\mathcal{E}$	
512×72×96	Conv:[k(3,3)-c128-s1-BN-Relu]	128×72×96
128×72×96	Deconv:[k(2,2)-c128-s2-BN-Relu]	128×144×192
128×144×192	Conv:[k(3,3)-c64-s1-BN-Relu]	64×144×192
64×144×192	Deconv:[k(2,2)-c64-s2-BN-Relu]	64×288×384
64×288×384	Conv:[k(3,3)-c32-s1-BN-Relu]	32×288×384
32×288×384	Deconv:[k(2,2)-c32-s2-BN-Relu]	32×576×768
32×576×768	Conv:[k(3,3)-c1-s1-Relu]	32×576×768
	$\mathcal{D}$	
512×72×96	Conv:[k(3,3)-c512-s1-LeakyReLU]	512×72×96
512×72×96	Conv:[k(3,3)-c256-s1-LeakyReLU]	256×72×96
256×72×96	Conv:[k(3,3)-c2-s1]	2×72×96
2×72×96	Resize	$2\times(H/g_h)\times(W/g_w)$
	$\mathcal{C}$	
3	FC:[c16-Relu]	16
16	FC:[c32-Relu]	32
32	FC:[c64-Relu]	64
64	FC:[c1-Relu]	1 ($\widetilde{p}_k$)
64	FC:[c32-Relu]	32
32	FC:[c16-Relu]	16
16	FC:[c3-Relu]	3 ($\widetilde{d}_k$)

Table S5. The network architecture of $\mathcal{F}$, $\mathcal{E}$, $\mathcal{D}$ and controller $\mathcal{C}$.

References

[1] Petro Liashchynskyi and Pavlo Liashchynskyi. Grid search, random search, genetic algorithm: a big comparison for nas. *arXiv*, 2019. 2

[2] Yongtuo Liu, Qiang Wen, Haoxin Chen, Wenxi Liu, Jing Qin, Guoqiang Han, and Shengfeng He. Crowd counting via cross-stage refinement networks. *IEEE TIP*, 29:6800–6812, 2020. 1