Recurrent Pixel Embedding for Instance Grouping
— *Supplementary Material*

Shu Kong, Charless Fowlkes
Dept. of Computer Science, University of California, Irvine
{skong2, fowlkes}@ics.uci.edu
project page with code and demo

In this supplementary material, we provide proofs of the propositions introduced in the main paper for understanding our objective function and grouping mechanism. Then, we provide the details of the mean-shift algorithm, computation of gradients and how it is adapted for recurrent grouping. We illustrate how the gradients are back-propagated to the input embedding using a toy example. Finally, we include more qualitative results on boundary detection and instance segmentation.

1. Analysis of Pairwise Loss for Spherical Embedding

In this section, we provide proofs for the propositions presented in the paper which provide some analytical understanding of our proposed objective function, and the mechanism for subsequent pixel grouping mechanism.

Proposition 1 *For n vectors $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, the total intra-pixel similarity is bounded as $\sum_{i\neq j} \mathbf{x}_i^T \mathbf{x}_j \geq -\sum_{i=1}^n \|\mathbf{x}_i\|_2^2$. In particular, for n vectors on the hyper-sphere where $\|\mathbf{x}_i\|_2 = 1$, we have $\sum_{i\neq j} \mathbf{x}_i^T \mathbf{x}_j \geq -n$.*

Proof 1 *First note that $\|\mathbf{x}_1 + \cdots + \mathbf{x}_n\|_2^2 \geq 0$. We expand the square and collect all the cross terms so we have $\sum_i \mathbf{x}_i^T \mathbf{x}_i + \sum_{i\neq j} \mathbf{x}_i^T \mathbf{x}_j \geq 0$. Therefore, $\sum_{i\neq j} \mathbf{x}_i^T \mathbf{x}_j \geq -\sum_{i=1}^n \|\mathbf{x}_i\|_2^2$. When all the vectors are on the hyper-sphere, i.e. $\|\mathbf{x}_i\|_2 = 1$, then $\sum_{i\neq j} \mathbf{x}_i^T \mathbf{x}_j \geq -\sum_{i=1}^n \|\mathbf{x}_i\|_2^2 = -n$.* ■

Proposition 2 *Given N vectors $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ on a 2-sphere, i.e. $\mathbf{x}_i \in \mathbb{R}^3$, $\|\mathbf{x}_i\|_2 = 1, \forall i = 1 \dots n$, choosing $\alpha \leq 1 - \left(\frac{2\pi}{\sqrt{3}N}\right)$, guarantees that $[s_{ij} - \alpha]_+ \geq 0$ for some pair $i \neq j$. Choosing $\alpha > 1 - \frac{1}{4}\left(\left(\frac{8\pi}{\sqrt{3}N}\right)^{\frac{1}{2}} - CN^{-\frac{2}{3}}\right)^2$, guarantees the existence of an embedding with $[s_{ij} - \alpha]_+ = 0$ for all pairs $i \neq j$.*

We treat all the n vectors as representatives of n different instances in the image and seek to minimize pairwise similarity, or equivalently maximize pairwise distance (referred to as Tammes's problem, or the hard-spheres problem [14]).

Proof 2 *Let $d = \max_{\{\mathbf{x}_i\}} \min_{i\neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_2$ be the distance between the closest point pair of the optimally distributed points. Asymptotic results in [10] show that, for some constant $C > 0$,*

$$\left(\frac{8\pi}{\sqrt{3}n}\right)^{\frac{1}{2}} - Cn^{-\frac{2}{3}} \leq d \leq \left(\frac{8\pi}{\sqrt{3}n}\right)^{\frac{1}{2}} \tag{1}$$

Since $\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 = 2 - 2\mathbf{x}_i^T \mathbf{x}_j$, we can rewrite this bound in terms of the similarity $s_{ij} = \frac{1}{2}\left(1 + \frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x}_i\|_2 \|\mathbf{x}_j\|_2}\right)$, so that for any $i \neq j$:

$$1 - \left(\frac{2\pi}{\sqrt{3}N}\right) \leq s_{ij} \leq 1 - \frac{1}{4}\left(\left(\frac{8\pi}{\sqrt{3}N}\right)^{\frac{1}{2}} - CN^{-\frac{2}{3}}\right)^2 \tag{2}$$

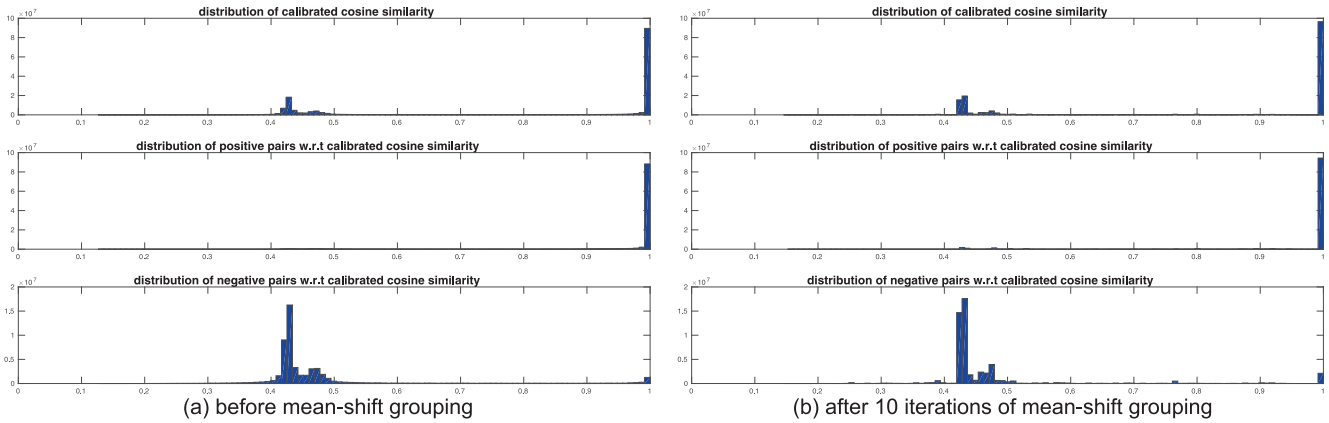
Therefore, choosing $\alpha \leq 1 - \left(\frac{2\pi}{\sqrt{3}N}\right)$, guarantees that $[s_{ij} - \alpha]_+ \geq 0$ for some pair $i \neq j$. Choosing $\alpha > 1 - \frac{1}{4}\left(\left(\frac{8\pi}{\sqrt{3}N}\right)^{\frac{1}{2}} - CN^{-\frac{2}{3}}\right)^2$, guarantees the existence of an embedding with $[s_{ij} - \alpha]_+ = 0$. ■

2. Details of Recurrent Mean Shift Grouping

There are two commonly used multivariate kernels in mean shift algorithm. The first, Epanechnikov kernel [7, 5], has the following profile

$$K_E(\mathbf{x}) = \begin{cases} \frac{1}{2}c_d^{-1}(d+2)(1 - \|\mathbf{x}\|_2^2), & \text{if } \|\mathbf{x}\|_2 \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where c_d is the volume of the unit d-dimensional sphere. The standard mean-shift algorithm computes the gradient



(a) before mean-shift grouping (b) after 10 iterations of mean-shift grouping

Figure 1: Distribution of calibrated cosine similarity between pairs of pixels. After 10 iterations of mean-shift grouping. Margin is 0.5 for negative pairs. From the figures, we believe that the mean shift grouping mechanism forces learning to focus on those pixel pairs that will not be corrected by mean shift grouping itself if running offline, and thus pushing down to parameters in the the deep neural network to learn how to correct them during training.

of the kernel density estimate given by

$$p(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} K_E(\frac{\mathbf{x}-\mathbf{x}_i}{b})$$

and identifies modes (local maxima) where $\nabla p(\mathbf{x}) = 0$. The scale parameter b is known as the kernel bandwidth and determines the smoothness of the estimator. The gradient of $p(\mathbf{x})$ can be elegantly computed as the difference between $\mathbf{x}$ and the mean of all data points with $\|\mathbf{x}-\mathbf{x}_i\| \leq b$, hence the name "mean-shift" for performing gradient ascent.

Since the Epanechnikov profile is not differentiable at the boundary, we use the squared exponential kernel adapted to vectors on the sphere:

$$K(\mathbf{x}, \mathbf{x}_i) \propto \exp(\delta^2 \mathbf{x}^T \mathbf{x}_i) \tag{4}$$

which can be viewed as a natural extension of the Gaussian to spherical data (known as the von Mises Fisher (vMF) distribution [9, 3, 13, 12]). In our experiments we set the bandwidth δ based on the margin α so that $\frac{1}{\delta} = \frac{1-\alpha}{3}$.

Our proposed algorithm also differs from the standard mean-shift clustering (i.e., [6]) in that rather than performing gradient ascent on a fixed kernel density estimate $p(\mathbf{x})$, at every iteration we alternate between updating the embedding vectors $\{\mathbf{x}_i\}$ using gradient ascent on $p(\mathbf{x})$ and re-estimating the density $p(\mathbf{x})$ for the updated vectors. This approach is termed Gaussian Blurring Mean Shift (GBMS) in [4] and has converge rate guarantees for data which starts in compact clusters.

In the paper we visualized embedding vectors after GBMS for specific examples. Figure 1 shows aggregate statistics over a collection of images (in the experiment of instance segmentation). We plot the distribution of pairwise similarities for positive and negative pairs during forward propagation through 10 iterations. We can observe that the mean shift module produces sharper distributions, driving the similarity between positive pairs to 1 making it trivial to identify instances.

2.1. Gradient Calculation for Recurrent Mean Shift

To backpropagate gradients through an iteration of GBMS, we break the calculation into a sequence of steps below where we assume the vectors in the data matrix $\mathbf{X}$ have already been normalized to unit length for its columns.

$$\begin{aligned}
\mathbf{S} &= \mathbf{X}^T\mathbf{X} \\
\mathbf{K} &= \exp(\delta^2\mathbf{S}), \\
\mathbf{d} &= \mathbf{K}^T\mathbf{1} \\
\mathbf{q} &= \mathbf{d}^{-1} \\
\mathbf{P} &= (1-\eta)\mathbf{I} + \eta\mathbf{K}\mathrm{diag}(\mathbf{q}) \\
\mathbf{Y} &= \mathbf{X}\mathbf{P}
\end{aligned} \tag{5}$$

where $\mathbf{Y}$ is the updated data after one iteration which is subsequently renormalized to project back onto the sphere. Let ℓ denote the loss and $\odot$ denote element-wise product.

Backpropagation gradients are then given by:

$$
\begin{aligned}
&\frac{\partial \ell}{\partial \mathbf{X}} = 2\mathbf{X}\frac{\partial \ell}{\partial \mathbf{S}} \\
&\frac{\partial \ell}{\partial \mathbf{S}} = \delta^2 \exp(\delta^2 \mathbf{S}) \odot \frac{\partial \ell}{\partial \mathbf{K}} \\
&\frac{\partial \ell}{\partial \delta} = 2\delta \sum_{ij} \Big((s_{ij}) \odot \exp(\delta^2 s_{ij}) \odot \frac{\partial \ell}{\partial k_{ij}} \Big) \\
&\frac{\partial \ell}{\partial \mathbf{K}} = \mathbf{1}\Big(\frac{\partial \ell}{\partial \mathbf{d}}\Big)^T \\
&\frac{\partial \ell}{\partial \mathbf{d}} = \frac{\partial \ell}{\partial \mathbf{q}} \odot (-\mathbf{d}^{-2}) \\
&\frac{\partial \ell}{\partial \mathbf{K}} = \eta\Big(\frac{\partial \ell}{\partial \mathbf{P}}\Big)(\mathbf{q}\mathbf{1}^T) \\
&\frac{\partial \ell}{\partial \mathbf{q}} = \eta\Big(\frac{\partial \ell}{\partial \mathbf{P}}\Big)^T \mathbf{K}\mathbf{1} \\
&\frac{\partial \ell}{\partial \mathbf{X}} = \frac{\partial \ell}{\partial \mathbf{Y}}\mathbf{P}^T \\
&\frac{\partial \ell}{\partial \mathbf{P}} = \mathbf{X}^T\frac{\partial \ell}{\partial \mathbf{Y}}
\end{aligned}
\tag{6}
$$

2.2. Toy Example of Mean Shift Backpropagation

In the paper we show examples of the gradient vectors backpropagated through recurrent mean shift to the initial embedding space. Backpropagation through this fixed model modulates the loss on the learned embedding, increasing the gradient for initial embedding vectors whose instance membership is ambiguous and decreasing the gradient for embedding vectors that will be correctly resolved by the recurrent grouping phase.

Figure 2 shows a toy example highlighting the difference between supervised and unsupervised clustering. We generate a set of 1-D data points drawn from three Gaussian distributions with mean and standard deviation as $(\mu = 3, \sigma = 0.2)$, $(\mu = 4, \sigma = 0.3)$ and $(\mu = 5, \sigma = 0.1)$, respectively, as shown in Figure 2 (a). We use mean squared error for the loss with a fixed linear regressor $y_i = 0.5 * x_i - 0.5$ and fixed target labels. The optimal embedding would set $x_i = 3$ if $y_i = 1$, and $x_i = 5$ if $y_i = 2$. We perform 30 gradient updates of the embedding vectors $x_i \leftarrow x_i - \alpha \nabla_{x_i} \ell$ with a step size α as 0.1. We analyze the behavior of Gaussian Blurring Mean Shift (GBMS) with bandwidth as 0.2.

If updating the data using gradient descent without GBMS inserted, as shown in Figure 2 (b), we can see the data move towards the ideal embedding in terms of classification and they are squeezed in shape yet still falling into three visible clusters. Figure 2 (c) depicts the trajectories of 100 random data points during the 30 updates. However, if running GBMS for unsupervised clustering on these data with the default setting (bandwidth is 0.2), we can see they are grouped into three piles, as shown in Figure 2 (d).

Now we insert the GBMS module to update these data with different loops, and compare how this effects the performance. We show the updated data distributions and those after five loops of GBMS grouping in column (e) and (f) of Figure 2, respectively. We notice that, with GBMS, all the data are grouped into two clusters; while with GBMS grouping they become more compact and are located exactly on the "ideal spot" for mapping into label space (i.e. 3 and 5) and achieving zero loss. On the other hand, we also observe that, even though these settings incorporates different number of GBMS loops, they achieve similar visual results in terms of clustering the data. To dive into the subtle difference, we randomly select 100 data and depict their trajectories in column (g) and (h) of Figure 2, using a single loss on top of the last GBMS loop or multiple losses over every GBMS loops, respectively. We have the following observations:

1. By comparing with Figure 2 (c), which depicts update trajectories without GBMS, GBMS module provides larger gradient to update those data further from their "ideal spot" under both scenarios.

2. From (g), we can see the final data are not updated into tight groups. This is because that the updating mechanism only sees data after (some loops of) GBMS, and knows that these data will be clustered into tight groups through GBMS.

3. A single loss with more loops of GBMS provides greater gradient than that with fewer loops to update data, as seen in (g).

4. With more losses over every loops of GBMS, the gradients become even larger that the data are grouped more tightly and more quickly. This is because that the updating mechanism also incorporates the gradients from the loss over the original data, along with those through these loops of GBMS.

To summarize, our GBMS based recurrent grouping module indeed provides meaningful gradient during training with back-propagation. With the convergent dynamics of GBMS, our grouping module becomes especially more powerful in learning to group data with suitable supervision.

3. Additional Boundary Detection Results

We show additional boundary detection results on BSDS500 dataset [1] based on our model in Figure 4, 5, 6, 7 and 8. Specifically, besides showing the boundary detection result, we also show 3-dimensional pixel embeddings as RGB images before and after fine-tuning using logistic loss. From the consistent colors, we can see (1) our model essentially carries out binary classification even using the

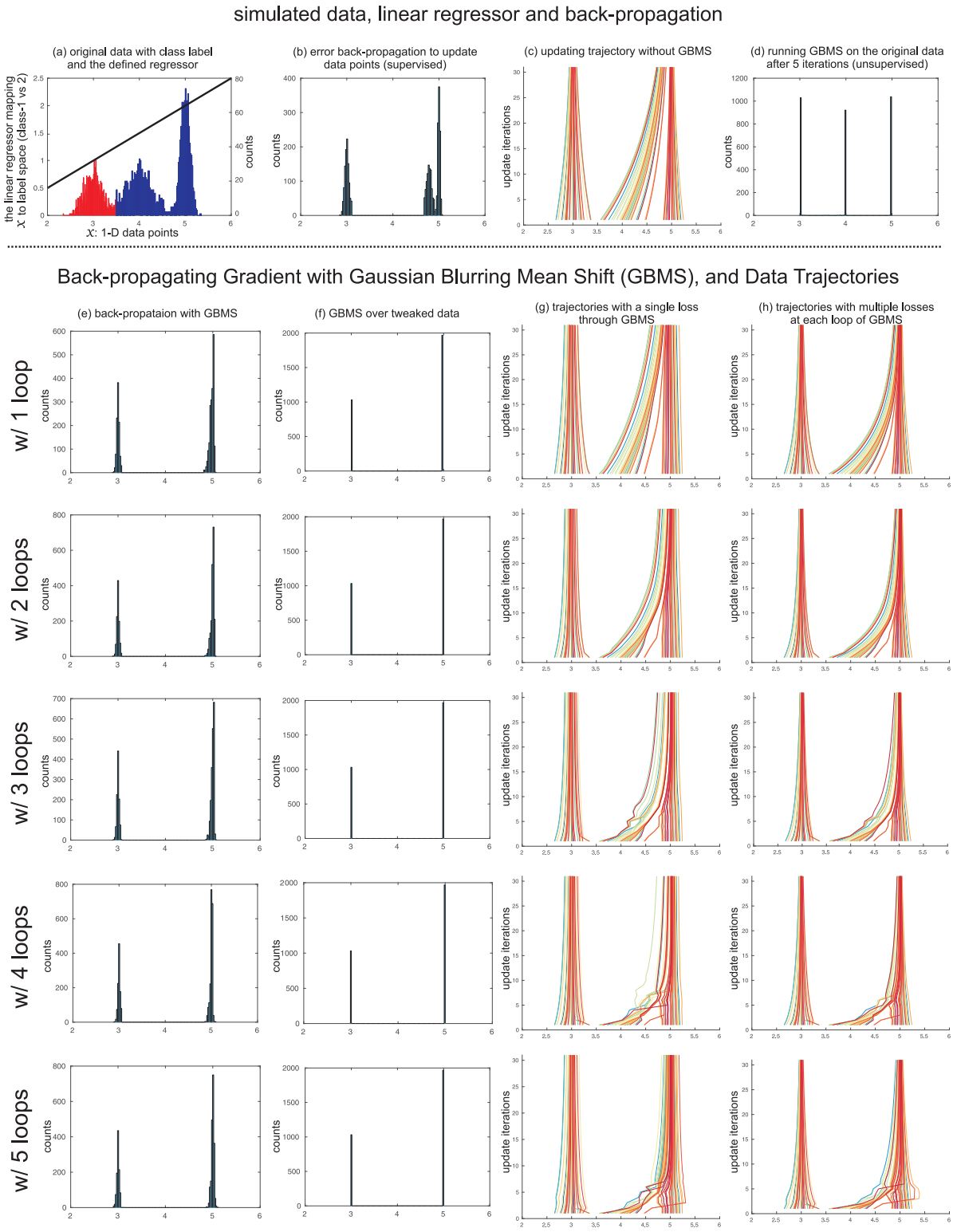


Figure 2: Trajectory of updating data using back-propagation without mean shift module (top row), and with the Gaussian Blurring Mean Shift (GBMS). To compare the results, we vary the number of GBMS loops in the grouping module, and use either a single loss at the final GBMS loop or multiple losses on all GBMS loops. All the configurations can shift data towards the "ideal spots" (3 or 5 depending on the label) in terms of the fixed regressor.

pixel pair embedding loss; (2) after fine-tuning with logistic loss, our model captures also boundary orientation and signed distance to the boundary. Figure 3 highlights this observation for an example image containing round objects. By zooming in one plate, we can observe a "colorful Mobius ring", indicating the embedding features for the boundary also capture boundary orientation and the signed distance to the boundary.

4. Additional Results on Instance-Level Semantic Segmentation

We show more instance-level semantic segmentation results on PASCAL VOC 2012 dataset [8] based on our model in Figure 9, 10 and 11. As we learn 64-dimensional embedding (hyper-sphere) space, to visualize the results, we randomly generate three matrices to project the embeddings to 3-dimension vectors to be treated as RGB images. Besides showing the randomly projected embedding results, we also visualize the semantic segmentation results used to product instance-level segmentation. From these figures, we observe the embedding for background pixels are consistent, as the backgrounds have almost the same color. Moreover, we can see the embeddings (e.g. in Figure 9, the horses in row-7 and row-13, and the motorbike in row-14) are able to connect the disconnected regions belonging to the same instance. Dealing with disconnected regions of one instance is an unsolved problem for many methods, e.g. [2, 11], yet our approach has no problem with this situation.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33(5):898–916, 2011. 3

[2] M. Bai and R. Urtasun. Deep watershed transform for instance segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2858–2866. IEEE, 2017. 5

[3] A. Banerjee, I. S. Dhillon, J. Ghosh, and S. Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *Journal of Machine Learning Research*, 6(Sep):1345–1382, 2005. 2

[4] M. A. Carreira-Perpinán. Generalised blurring mean-shift algorithms for nonparametric clustering. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008. 2

[5] Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 17(8):790–799, 1995. 1

[6] D. Comaniciu and P. Meer. Mean shift analysis and applications. In *The Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1197–1203. IEEE, 1999. 2

[7] V. A. Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications*, 14(1):153–158, 1969. 1

[8] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision (IJCV)*, 88(2):303–338, 2010. 5

[9] R. Fisher. Dispersion on a sphere. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 217, pages 295–305. The Royal Society, 1953. 2

[10] W. Habicht and B. Van der Waerden. Lagerung von punkten auf der kugel. *Mathematische Annalen*, 123(1):223–234, 1951. 1

[11] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother. Instancecut: from edges to instances with multicut. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5

[12] T. Kobayashi and N. Otsu. Von mises-fisher mean shift for clustering on a hypersphere. In *International Conference on Pattern Recognition (ICPR)*, pages 2130–2133. IEEE, 2010. 2

[13] K. V. Mardia and P. E. Jupp. *Directional statistics*, volume 494. John Wiley & Sons, 2009. 2

[14] E. B. Saff and A. B. Kuijlaars. Distributing many points on a sphere. *The mathematical intelligencer*, 19(1):5–11, 1997. 1

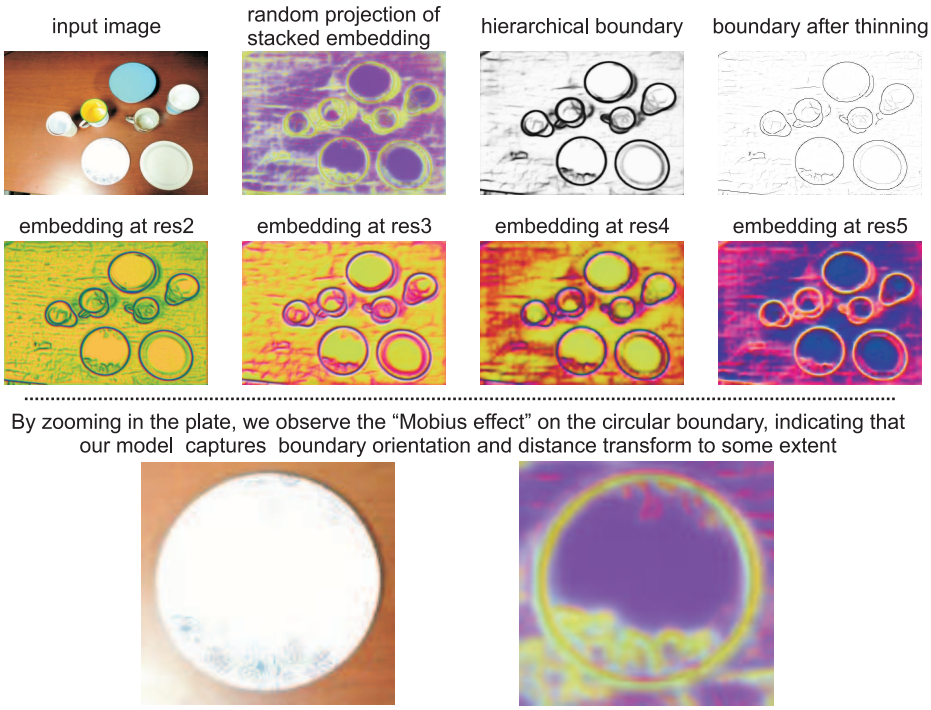


Figure 3: An image highlighting the structure of the embedding for an image with circular boundaries. We observe a “Mobius effect” where the embedding encodes both the orientation and distance to the boundary.

Figure 4: Visualization for boundary detection (part-1/5). Images are randomly selected from BSDS500 test set. For each image, we show the embedding vectors at different layers from the model before and after fine-tuning using logistic loss. We can see that the boundary embedding vectors after fine-tuning not only highlights the boundary pixels, but also captures to some extent the edge orientation and distance from the colors conveyed.

Figure 5: Visualization for boundary detection ($2/5$). Images are randomly selected from BSDS500 test set. For each image, we show the embedding vectors at different layers from the model before and after fine-tuning using logistic loss. We can see that the boundary embedding vectors after fine-tuning not only highlights the boundary pixels, but also captures to some extent the edge orientation and distance from the colors conveyed.

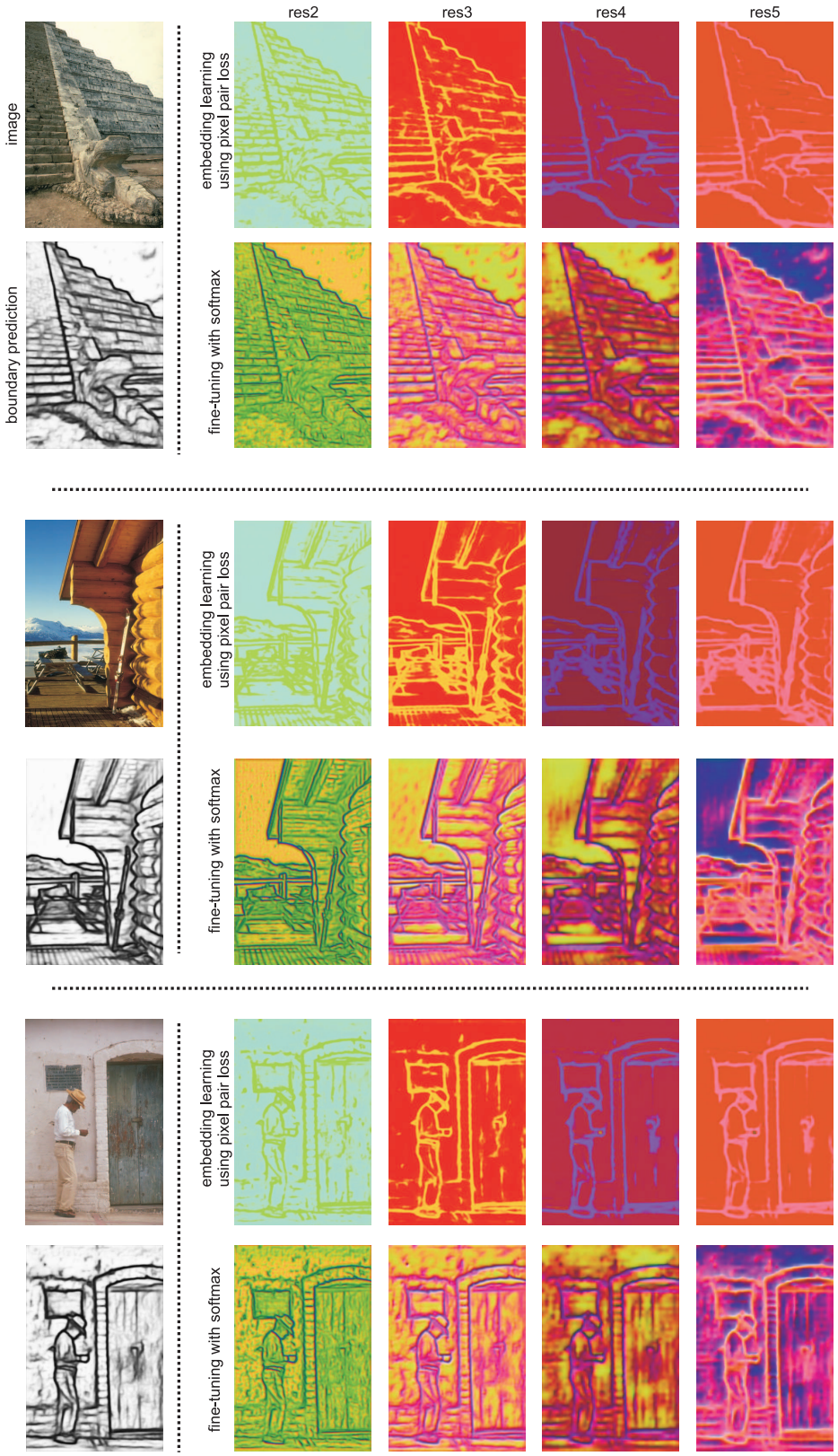


Figure 6: Visualization for boundary detection ($3/5$). Images are randomly selected from BSDS500 test set. For each image, we show the embedding vectors at different layers from the model before and after fine-tuning using logistic loss. We can see that the boundary embedding vectors after fine-tuning not only highlights the boundary pixels, but also captures to some extent the edge orientation and distance from the colors conveyed.

Figure 7: Visualization for boundary detection (4/5). Images are randomly selected from BSDS500 test set. For each image, we show the embedding vectors at different layers from the model before and after fine-tuning using logistic loss. We can see that the boundary embedding vectors after fine-tuning not only highlights the boundary pixels, but also captures to some extent the edge orientation and distance from the colors conveyed.

Figure 8: Visualization for boundary detection $(5/5)$. Images are randomly selected from BSDS500 test set. For each image, we show the embedding vectors at different layers from the model before and after fine-tuning using logistic loss. We can see that the boundary embedding vectors after fine-tuning not only highlights the boundary pixels, but also captures to some extent the edge orientation and distance from the colors conveyed.

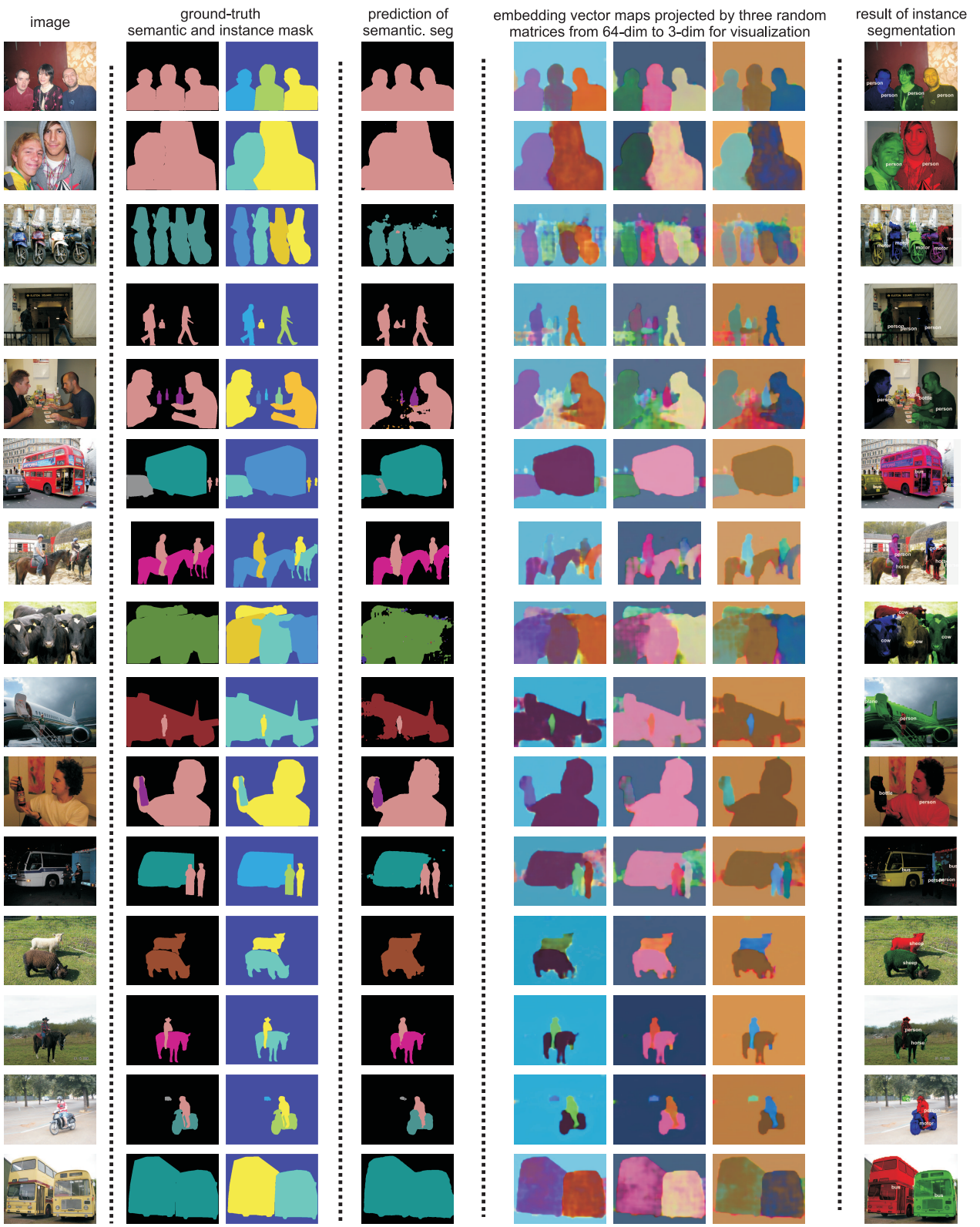


Figure 9: Visualization of generic and instance-level semantic segmentation with random projection of the embedding vectors (part-1/3).

Figure 10: Visualization of generic and instance-level semantic segmentation with random projection of the embedding vectors (part-2/3).

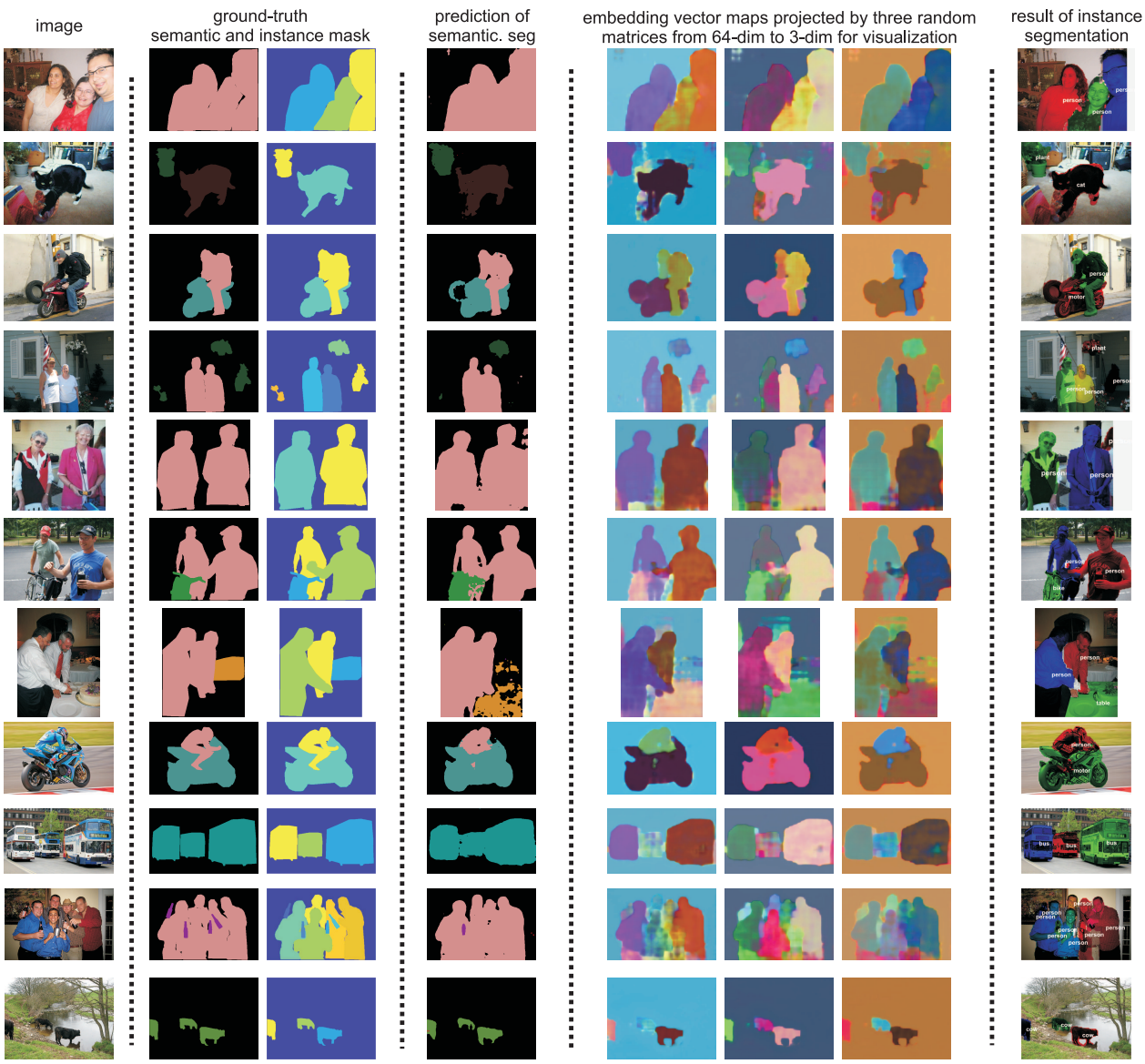


Figure 11: Visualization of generic and instance-level semantic segmentation with random projection of the embedding vectors (part-3/3).