Supplementary Material of Deep Spectral Clustering Learning

Marc T. Law [1] **Raquel Urtasun** [1] **Richard S. Zemel** [1 2]

This is the supplementary material of (Law et al., 2017). The indices of the different sections and equations correspond to those in (Law et al., 2017).

A. Proof of Section 3.1

Let us note $\hat{C}F = [\hat{\mathbf{f}}_1, \cdots, \hat{\mathbf{f}}_n]^\top \in \mathbb{R}^{n \times d}$, *i.e.* $\hat{\mathbf{f}}_i^\top$ is the i-th row of $\hat{C}F$, Eq. (4) can then be written:

$$f(F) = \arg\max_{\hat{C} \in \mathcal{C}^{n,k}} \mathbf{1}^\top \Phi(\hat{C}F) + \langle F - \hat{C}F, \nabla\Phi(\hat{C}F) \rangle \tag{12}$$

$$= \arg\min_{\hat{C} \in \mathcal{C}^{n,k}} \underbrace{\mathbf{1}^\top \Phi(F) - \mathbf{1}^\top \Phi(\hat{C}F) - \langle F - \hat{C}F, \nabla\Phi(\hat{C}F) \rangle}_{= \sum_{i=1}^n \mathsf{d}_\phi(\mathbf{f}_i, \hat{\mathbf{f}}_i)} \tag{13}$$

Eq. (13) is nonnegative as it can be written as a sum of Bregman divergences, and Bregman divergences are nonnegative. We note that if $s = \mathrm{rank}(F) \leq k$, then we can formulate $\hat{C} \in \mathcal{C}^{n,k}$ such that $\hat{C}F = F$ and the objective value in Eq. (13) is then 0 (*i.e.* the global minimum of Eq. (13) is reached). By using the identity property of Bregman divergences (*i.e.* $\forall \mathbf{a}, \mathbf{b}, \mathsf{d}_\phi(\mathbf{a}, \mathbf{b}) = 0$ if and only if $\mathbf{a} = \mathbf{b}$), the set of solutions of Eq. (13) is the following set of matrices:

$$\{\hat{C} \in \mathcal{C}^{n,k} : \hat{C}F = F\} \tag{14}$$

which can be rewritten:

$$f(F) = \{\hat{C} \in \mathcal{C}^{n,k} : \hat{C} = FF^\dagger + VV^\top, VV^\top \in \mathcal{C}^{n,(k-s)}, VV^\top F = 0\} \tag{15}$$

by using for instance the properties in (Fan, 1949). Indeed, any matrix $A \in \mathcal{C}^{n,k}$ that does not belong to the set in Eq. (15) does not satisfy $AF = F$. One can verify that any matrix $\hat{C} \in \mathcal{C}^{n,k}$ that is in the set in Eq. (15) satisfies $\hat{C}F = F$. $\square$

B. Lower bound of Eq. (6)

By using the definition of $f(F)$ in Eq. (15), we can write Eq. (6):

$$\max_{F \in \mathcal{F}^n} \min_{\hat{C} \in f(F)} \mathrm{tr}(C\hat{C}) = \max_{F \in \mathcal{F}^n} \min_{\{VV^\top \in \mathcal{C}^{n,(k-s)} : VV^\top F = 0\}} \overbrace{\mathrm{tr}(CFF^\dagger)}^{\geq 0} + \overbrace{\mathrm{tr}(CVV^\top)}^{\geq 0} \tag{16}$$

Both terms $\mathrm{tr}(CFF^\dagger)$ and $\mathrm{tr}(CVV^\top)$ are nonnegative as the matrices C, $FF^\dagger$ and $VV^\top$ are all symmetric positive semidefinite. Eq. (7) corresponds to selecting and maximizing the left term.

[1] Department of Computer Science, University of Toronto, Toronto, Canada [2] CIFAR Senior Fellow. Correspondence to: Marc T. Law <law@cs.toronto.edu>.

C. About the gradient in Eq. (8)

We assume in the following that $\text{rank}(F) > 0$. Otherwise, $F = 0$ is the optimal solution.

We note $P = FF^\dagger$ the orthogonal projector onto the column space of F, we have by definition $PF = F$. Let us note $\nabla_F = (I - FF^\dagger)S(F^\dagger)^\top$ where S is a symmetric matrix. We observe that the gradient ∇_P of Eq. (7) *w.r.t.* P satisfies the properties in (Golub & Pereyra, 1973)[Lemma 4.1] with this form of ∇_F. Indeed, (Golub & Pereyra, 1973)[Eq. (4.2)] can be written:

$$\nabla_P = (I - FF^\dagger)\nabla_F F^\dagger + (F^\dagger)^\top(\nabla_F)^\top(I - FF^\dagger) = (I - FF^\dagger)S(F^\dagger)^\top F^\dagger + [(I - FF^\dagger)S(F^\dagger)^\top F^\dagger]^\top$$

One can verify that $(\nabla_P)F = (I - FF^\dagger)S\overbrace{(F^\dagger)^\top F^\dagger F}^{(F^\dagger)^\top} + (F^\dagger)^\top(\nabla_F)^\top \overbrace{(I - FF^\dagger)F}^{0} = \nabla_F$.

As $PF = F$, one can also verify that $\nabla_{(PF)} = (\nabla_P)F + P\nabla_F = (\nabla_P)F + \overbrace{FF^\dagger(I - FF^\dagger)}^{0}S(F^\dagger)^\top = \nabla_F$.

From (Fan, 1949), Eq. (7) reaches its maximum value (*i.e.* $\nabla_P = 0$) iff $\text{tr}(FF^\dagger C) = \text{tr}(FF^\dagger)$ since $\text{rank}(C) \geq \text{rank}(F)$. Multiple optimum solutions of F may exist. However, the gradient must satisfy: $\forall F \in \mathcal{F}^n, P = FF^\dagger, \nabla_P = 0 \Leftrightarrow FF^\dagger C = FF^\dagger$, which is achieved iff there exists $\gamma > 0$ such that $C = \gamma S$. $\square$

We just proved that the gradient can be written $\nabla_F = \gamma(I - FF^\dagger)C(F^\dagger)^\top$ for some $\gamma > 0$, which is sufficient to use gradient-based methods since a step size plays a scaling factor role.

We verify experimentally that $\gamma = 2$ by using the tutorial presented in http://ufldl.stanford.edu/wiki/index.php/Gradient_checking_and_advanced_optimization

The tutorial uses the definition of derivatives. To follow the notations of the tutorial, let us note the objective function of Eq. (7):

$$J(F) = \text{tr}(FF^\dagger C) \tag{17}$$

where $F \in \mathbb{R}^{n \times d}$ is our variable. We note $E_{ij} \in \mathbb{R}^{n \times d}$ the matrix that contains only 0 except for its (i, j)-th element whose value is 1. We also note $\varepsilon \leq 10^{-4}$ a small real value.

The tutorial explains that:

$$\forall i \in \{1, \cdots, n\}, \forall j \in \{1, \cdots, d\}, (\nabla_F)_{ij} \simeq \frac{J(F + \varepsilon E_{ij}) - J(F - \varepsilon E_{ij})}{2\varepsilon} \tag{18}$$

where $(\nabla_F)_{ij}$ is the (i, j)-th element of ∇_F. By using the following Matlab code, we verify that $\gamma = 2$.

```
clear all;

n = 10;
d = 4;
rank_of_C = 7;

Y = rand(n,rank_of_C)*10;
C = Y * pinv(Y);

F = (rand(n,d)*10) - 5;
Real_gradient = 2*(eye(n) - F*pinv(F)) * C * pinv(F)';
epsilon = 10^-7;

Approximate_gradient = zeros(n,d);

for i=1:n
  for j=1:d
    E = zeros(n,d);
    E(i,j) = epsilon;

    X1 = F+E;
```

```
    X2 = F-E;

    Approximate_gradient(i,j) = (trace(X1 * pinv(X1) * C) - trace(X2 * pinv(X2) * ...
        C))/(2*epsilon);
  end
end


Real_gradient
Approximate_gradient

disp('Frobenius distance')
norm(Real_gradient-Approximate_gradient,'fro')
```

D. t-SNE plots

The following figures illustrate the 2-dimensional t-SNE (Van Der Maaten, 2014) embedding vectors produced by our method and different baselines on the test categories (*i.e.* which are unseen) on the Birds, Cars196 and Stanford Online products datasets, respectively. Each image is represented by a point and its ground truth label is represented by a color. One can observe that our learned model is able to generalize and group examples from new categories that were unseen during training.

Note that results are difficult to visualize for the Products dataset as there are thousands of categories whereas there are about 100 categories in the other datasets.

t-SNE figures of some baselines for the same test sets are also provided in (Song et al., 2016) and (Song et al., 2017).

References

Fan, Ky. On a theorem of weyl concerning eigenvalues of linear transformations i. *Proceedings of the National Academy of Sciences of the United States of America*, 35(11):652, 1949.

Golub, Gene H and Pereyra, Victor. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM Journal on numerical analysis*, 10(2):413–432, 1973.

Law, Marc Teva, Urtasun, Raquel, and Zemel, Richard. Deep spectral clustering learning. In *ICML*, 2017.

Song, Hyun Oh, Xiang, Yu, Jegelka, Stefanie, and Savarese, Silvio. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4004–4012, 2016.

Song, Hyun Oh, Jegelka, Stefanie, Rathod, Vivek, and Murphy, Kevin. Deep metric learning via facility location. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

Van Der Maaten, Laurens. Accelerating t-sne using tree-based algorithms. *Journal of machine learning research*, 15(1): 3221–3245, 2014.

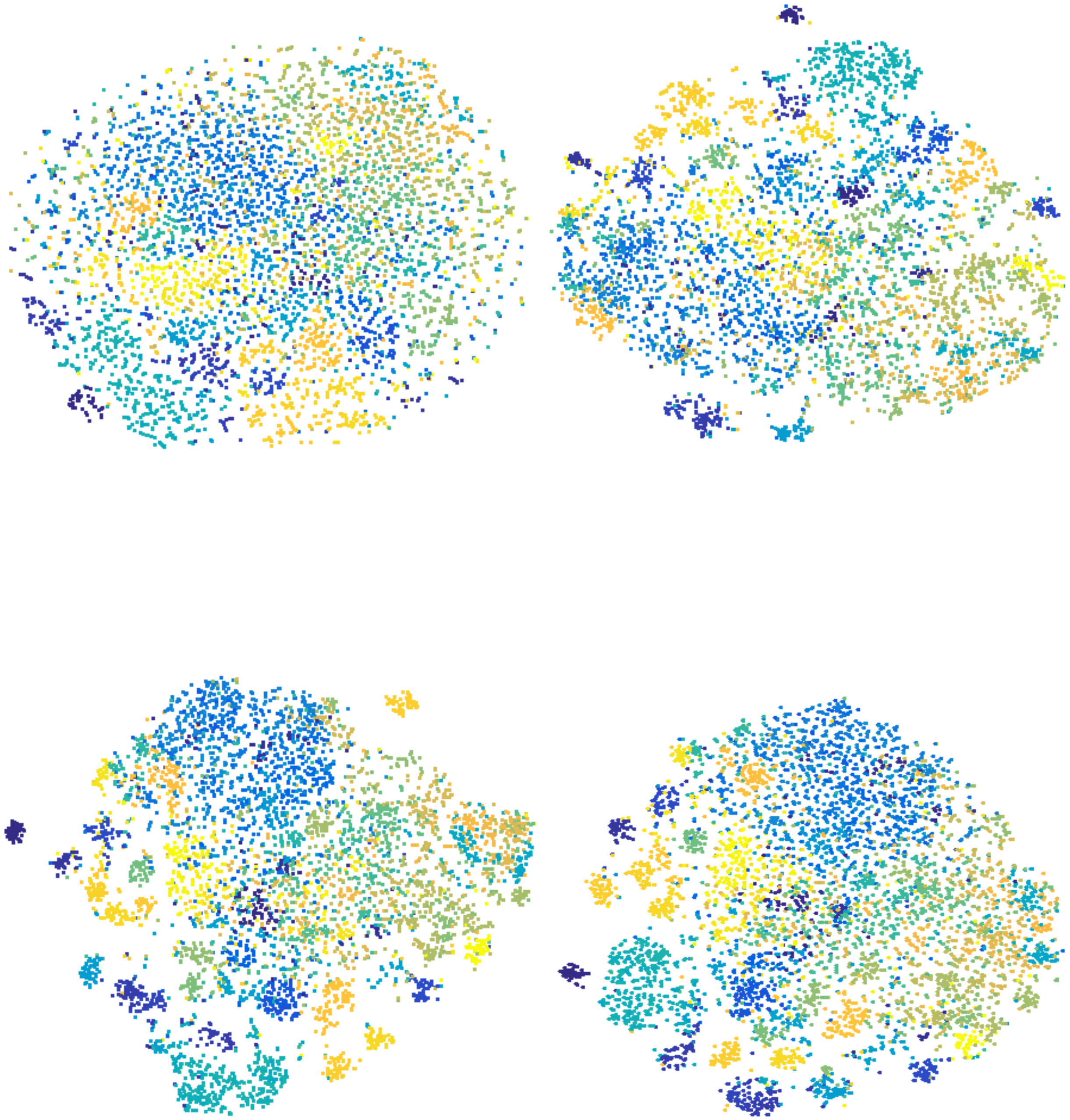

Figure 2. Barnes-Hut t-SNE visualization (Van Der Maaten, 2014) of the embedding on the test/unseen categories of the Birds dataset produced by the vanilla GoogLeNet pretrained on Image (top left), logistic regression for classification (top right), our method that updates only the parameters in the last layer during fine-tuning (bottom left), our end-to-end method (bottom right). Examples in the same category have the same color.

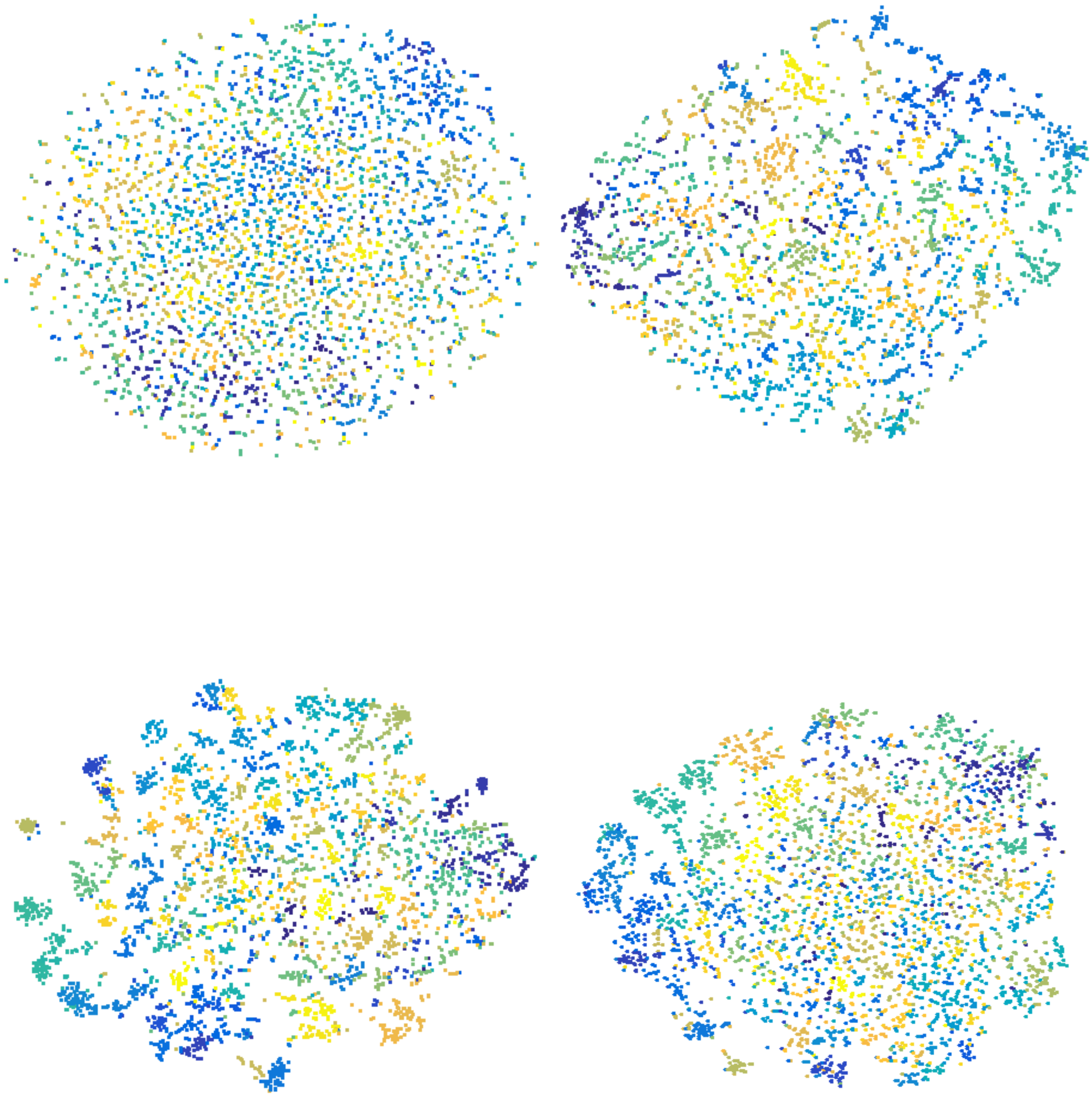

Figure 3. Barnes-Hut t-SNE visualization (Van Der Maaten, 2014) of the embedding on the test/unseen categories of the Cars dataset produced by the vanilla GoogLeNet pretrained on Image (top left), logistic regression for classification (top right), our method that updates only the parameters in the last layer during fine-tuning (bottom left), our end-to-end method (bottom right). Examples in the same category have the same color.

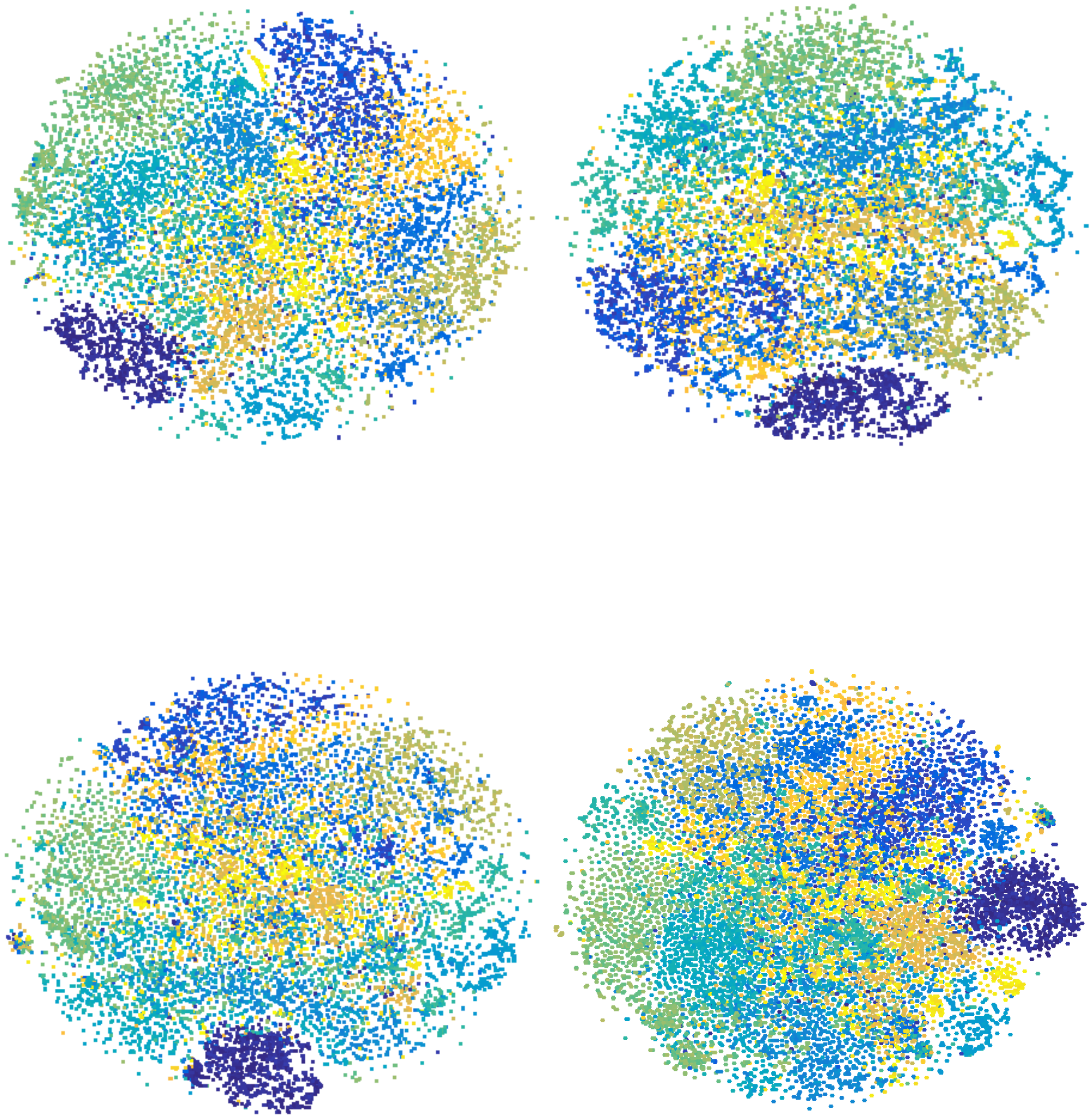

Figure 4. Barnes-Hut t-SNE visualization (Van Der Maaten, 2014) of the embedding on the test/unseen categories of the Products dataset produced by the vanilla GoogLeNet pretrained on Image (top left), logistic regression for classification (top right), our method that updates only the parameters in the last layer during fine-tuning (bottom left), our end-to-end method (bottom right). Examples in the same category have the same color.