Learning Rotation-Aware Features: From Invariant Priors to Equivariant Descriptors

Supplemental Material

Uwe Schmidt Stefan Roth
Department of Computer Science, TU Darmstadt

In this supplemental material we provide additional details that are helpful to reproduce the results in the main paper [10].

1. Parameter Learning in Transformation-Invariant Product Models

In this section we briefly demonstrate that parameter learning in product models with integrated transformation invariance (*cf*. Sec. 2.1 from the main paper) involves only minor modifications to existing gradient-based learning algorithms. We begin by considering a generic product model

$$p(\mathbf{x};\Theta) = \frac{1}{Z(\Theta)} \prod_{i=1}^{|\mathcal{F}|} \phi_i\big(\mathbf{F}_{(i)}\mathbf{x};\theta_i\big) \tag{1}$$

as defined in Eq. (1) of the main paper. Now the task is to learn all parameters $\Theta = \{\mathbf{F}_{(i)}, \theta_i | i = 1, \dots\}$ by using gradients of the log-probability (density) $\log p(\mathbf{x};\Theta)$. It is quite straightforward to see that the partial derivatives of the log-probability (density) w.r.t. the factor parameters θ_i and the feature transformations $\mathbf{F}_{(i)}$ are given as

$$\frac{\partial \log p(\mathbf{x};\Theta)}{\partial \theta_i} = \frac{\partial \log \phi_i\big(\mathbf{F}_{(i)}\mathbf{x};\theta_i\big)}{\partial \theta_i} - \frac{\partial \log Z(\Theta)}{\partial \theta_i} \tag{2}$$

$$\frac{\partial \log p(\mathbf{x};\Theta)}{\partial \mathbf{F}_{(i)}} = \frac{\partial \log \phi_i\big(\mathbf{F}_{(i)}\mathbf{x};\theta_i\big)}{\partial \mathbf{F}_{(i)}} - \frac{\partial \log Z(\Theta)}{\partial \mathbf{F}_{(i)}}. \tag{3}$$

The gradient of the log-partition function $\log Z(\Theta)$ (second term of Eqs. (2,3)) is usually approximated by evaluating the first term of the respective equation on a set of samples from the product model (*cf*. [6]).

According to Eq. (4) of the main paper, we define a transformation-invariant product model w.r.t. a set of linear image transformations $\mathcal{T} = \{\mathbf{T}_{(j)} | j = 1, \dots\}$ as

$$p_{\mathcal{T}}(\mathbf{x};\Theta) = \frac{1}{Z(\Theta)} \prod_{j=1}^{|\mathcal{T}|} \prod_{i=1}^{|\mathcal{F}|} \phi_i\big(\mathbf{F}_{(i)}\mathbf{T}_{(j)}\mathbf{x};\theta_i\big). \tag{4}$$

In analogy to Eq. (1) the partial derivatives of the log-probability (density) thus follow as

$$\frac{\partial \log p_{\mathcal{T}}(\mathbf{x};\Theta)}{\partial \theta_i} = \left[\sum_{j=1}^{|\mathcal{T}|} \frac{\partial \log \phi_i\big(\mathbf{F}_{(i)}\mathbf{T}_{(j)}\mathbf{x};\theta_i\big)}{\partial \theta_i}\right] - \frac{\partial \log Z(\Theta)}{\partial \theta_i} \tag{5}$$

$$\frac{\partial \log p_{\mathcal{T}}(\mathbf{x};\Theta)}{\partial \mathbf{F}_{(i)}} = \left[\sum_{j=1}^{|\mathcal{T}|} \frac{\partial \log \phi_i\big(\mathbf{F}_{(i)}\mathbf{T}_{(j)}\mathbf{x};\theta_i\big)}{\partial \mathbf{F}_{(i)}}\right] - \frac{\partial \log Z(\Theta)}{\partial \mathbf{F}_{(i)}}. \tag{6}$$

Please note that Eqs. (5,6) involve quite similar derivative calculations as Eqs. (2,3). In essence, the derivatives of the log-factors $\log \phi_i$ need to be summed over $|\mathcal{T}|$ transformed inputs. Hence, integrating transformation-invariance into existing product model implementations generally requires little implementation effort.

2. Details of Parameter Learning

2.1. R-FoE

The R-FoE model of Sec. 3 of the main paper was trained on a database of 5000 natural images (50×50 pixels) using persistent contrastive divergence [12] (also known as stochastic maximum likelihood). Learning was done with stochastic gradient descent using mini-batches of 100 images (and model samples) for a total of 10000 (exponentially smoothed) gradient steps with an annealed learning rate. We trained the model using conditional sampling to avoid boundary issues [8]. Both learned filters were initialized randomly from a standard normal distribution, and constrained to have mean 0 and norm 1 throughout learning. We initialized the shapes of the potential functions to be very broad (*cf*. [5]).

2.2. RC-RBM

We trained our RC-RBM from Sec. 4 of the main paper akin to the algorithm of [8], in particular using contrastive divergence [6] with one step of Gibbs sampling, although applying Rao-Blackwellization [11] to minimize sample variance. We used two datasets for unsupervised training (always using one visible unit per image pixel): a random subset of 10000 binary images from the training set of the MNIST handwritten digits [1], and the same dataset of natural images as used for the R-FoE, but here ZCA-whitened. In both cases, we performed stochastic gradient descent with mini-batches of 20 images (100 for MNIST), an annealed learning rate, and exponential gradient smoothing. For training on natural images, we also relied on conditional sampling to avoid boundary issues [8].

We initialized all hidden biases to $\mathbf{b} = -\mathbf{3}$ and all visible biases to $\mathbf{c} = \mathbf{0}$; note that training on MNIST relied on individual biases $\mathbf{c}$, and training on natural images used a shared (scalar) bias c for all visible units. Instead of fixing the hidden biases $\mathbf{b}$ to a small value to encourage sparsity [8], we learned them together with the features, which we constrained to have the same norm, updated slowly over time through exponential smoothing. We did not use any additional regularization terms to encourage learning of sparse features (such as [7]).

We only define the filters $\mathbf{w}_i$ inside a circular area by actually using $\hat{\mathbf{R}}_{(\omega)} = \mathbf{B} \cdot \mathbf{R}_{(\omega)}$ instead of $\mathbf{R}_{(\omega)}$ in Eq. (11) of the main paper, where multiplication with $\mathbf{B}$ extracts the circular interior of the image patch as a vector.

3. Details of Feature Extraction

We extract RC-RBM features by computing the hidden activation probabilities $p_{\text{RC-RBM}}(\mathbf{h} = \mathbf{1}|\mathbf{x})$ for each feature i convolutionally at all image locations (k, l) and all specified rotation angles ω. Computation is straightforward since $p_{\text{RC-RBM}}(\mathbf{h} = \mathbf{1}|\mathbf{x})$ decomposes into a product of univariate distributions

$$p_{\text{RC-RBM}}(h_{(\omega),(k,l),i} = 1|\mathbf{x}) = \text{sig}(\mathbf{w}_i^{\mathrm{T}}\mathbf{R}_{(\omega)}\mathbf{C}\mathbf{S}_{(k,l)}\mathbf{x} + b_i) \tag{7}$$

with the logistic function $\text{sig}(x) = 1/(1 + e^{-x})$. We note that we set all hidden biases $\mathbf{b} = \mathbf{0}$ for feature extraction after learning, as this significantly increased performance in recognition and detection tasks. Furthermore, non-maximum suppression (over rotations ω) is used to only retain activations with maximum probability at each location (k, l). The EHOF/IHOF descriptor is computed separately for each (of the four) learned features; the final descriptor is then obtained by concatenation of the individual descriptor vectors.

Oriented gradient features are extracted the same way as in the popular HOG [2] descriptor: Centered image derivatives ($[1, 0, -1]$ and $[1, 0, -1]^{\mathrm{T}}$) are first computed at all image locations to obtain horizontal and vertical derivative images. Each pixel is then assigned to one of B orientation angles (using linear interpolation) according to its gradient angle and represented by its gradient magnitude.

4. Additional Descriptor Details

The aim of this section is to provide further descriptor details that were omitted from the main paper due to lack of space. All details can also be inferred from the MATLAB code, which is available on the authors' webpages.

4.1. Local cell normalization

When using oriented gradient features, we also perform two different normalizations of all cells (except the central one), akin to the block normalization procedure in HOG. We do this because it significantly increases performance.

Here, each block consists of two neighboring cells on a ring, *i.e.* each cell is normalized with its predecessor and successor cell. We use L$_2$-normalization $g(\mathbf{v}, \mathbf{z}) = \mathbf{v}/\sqrt{\|\mathbf{z}\|^2 + \epsilon}$ for cell histogram vector $\mathbf{v}$ with block vector $\mathbf{z}$ and $\epsilon = 10^{-4}$; we do not "clip" values after normalization. The layout in the descriptor matrix is adjusted in order to retain the equivariance property: Different normalizations of the same cell are (deterministically) grouped together in the columns of the matrix, *i.e.* first come all (both) normalizations of the normalized entry for orientation angle 1, then all normalizations for angle 2, *etc.*

Unfortunately, this local cell normalization procedure does not improve results when applied to our learned RC-RBM features, hence we do not use it; however, we can assume that a suitable normalization scheme would also enhance the performance for our learned features, but we have not explored this yet. In this sense, gradient features are at an advantage due to previous research on suitable normalizations, which we leverage here.

4.2. Descriptor dimensionality and scaling

For an EHOF descriptor with R rings, C cells per ring, and features extracted at O orientations, we obtain a 3-dimensional histogram $\mathbf{H}_3 \in \mathbb{R}^{R \times C \times O}$, which is reshaped into the 2-dimensional $\mathbf{H}_2 \in \mathbb{R}^{R \cdot C \times n \cdot O}$. For gradient features, $n = 2$ due to the two normalizations of each cell, and $n = 1$ for our learned RC-RBM features. Hence, we obtain the descriptor dimensionality $D = R{\cdot}C{\cdot}n{\cdot}O + O$, where the last term comes from the histogram vector of central cell $\mathbf{c} \in \mathbb{R}^O$. For IHOF, we additionally compute the 2D-DFT of $\mathbf{H}_2$ and the 1D-DFT of $\mathbf{c}$ and only retain the magnitude in both cases. The DFT magnitude of real inputs exhibits redundancies, which we have only removed in case of 1D-DFT, resulting in the IHOF descriptor dimensionality $D = R{\cdot}C{\cdot}n{\cdot}O + \lceil \frac{O}{2} \rceil$.

The resulting descriptor vectors are scaled to unit infinity norm in case of IHOF (and EHOF for car detection).

5. Experimental Details

5.1. Denoising

For the results in Tab. (1) of the main paper, we used a fixed sampling scheme similar to [9] with four independent samplers, each running for 60 iterations to yield 120 samples overall after discarding 30 burn-in iterations each. We employed the same procedure for the denoising example in Fig. (5) of the paper, but used 240 samples in total to further reduce to variability induced by the sampling process.

5.2. Car detection

RC-RBM features were extracted from whitened grayscale versions of the RGB color input images, whereas gradient features were obtained by taking the maximum gradient response (in terms of magnitude) over the three channels of the color images. For the HOG baseline performance, we use the implementation of [3] with 5×5 pixel-sized cells and 9 (unsigned) orientation angles.

We trained the linear SVM initially by performing cross-validation to find the best regularization parameter C, then used two rounds of bootstrapping to obtain the final model. Detection was carried out with a search stride of 5 pixels over five image scales $[1.0^{-1}, 1.1^{-1}, 1.2^{-1}, 1.3^{-1}, 1.4^{-1}]$, where 1.0 refers to the original image size. We evaluated the performance according to the PASCAL VOC 2010 criteria [4], *i.e.* requiring 50% overlap with a ground-truth annotation and not allowing multiple detections of the same car. Calculation of the average precision also followed [4].

References

[1] http://yann.lecun.com/exdb/mnist/.

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *CVPR 2005*.

[3] P. Dollár. Piotr's Image and Video Matlab Toolbox (PMT). http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html.

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html.

[5] Q. Gao and S. Roth. How well do filter-based MRFs model natural images? *Pattern Recognition (DAGM) 2012*.

[6] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14(8), 2002.

[7] H. Lee, C. Ekanadham, and A. Y. Ng. Sparse deep belief net model for visual area V2. *NIPS 2007*.

[8] M. Norouzi, M. Ranjbar, and G. Mori. Stacks of convolutional restricted Boltzmann machines for shift-invariant feature learning. *CVPR 2009*.

[9] U. Schmidt, Q. Gao, and S. Roth. A generative perspective on MRFs in low-level vision. *CVPR 2010*.

[10] U. Schmidt and S. Roth. Learning rotation-aware features: From invariant priors to equivariant descriptors. *CVPR 2012*.

[11] K. Swersky, B. Chen, B. Marlin, and N. de Freitas. A tutorial on stochastic approximation algorithms for training restricted Boltzmann machines and deep belief nets. *ITA 2010*.

[12] T. Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. *ICML 2008*.