Transformation Equivariant Boltzmann Machines

Jyri J. Kivinen and Christopher K. I. Williams

Institute for Adaptive and Neural Computation
School of Informatics, University of Edinburgh, UK
j.j.kivinen@sms.ed.ac.uk, ckiw@inf.ed.ac.uk

Abstract. We develop a novel modeling framework for Boltzmann machines, augmenting each hidden unit with a latent transformation assignment variable which describes the selection of the transformed view of the canonical connection weights associated with the unit. This enables the inferences of the model to transform in response to transformed input data in a *stable and predictable* way, and avoids learning multiple features differing only with respect to the set of transformations. Extending prior work on translation equivariant (convolutional) models, we develop translation *and rotation* equivariant restricted Boltzmann machines (RBMs) and deep belief nets (DBNs), and demonstrate their effectiveness in learning frequently occurring statistical structure from artificial and natural images.

Keywords: Boltzmann machines, transformation equivariant representations, convolutional structures, transformation invariance, steerable filters, image modeling

1 Introduction

We consider the problem of using DBN architectures to model the structure occurring in natural images. One of the desiderata for a computer vision system is that if the input image is transformed (e.g. by a translation of two pixels left), then the inferences made by the system should co-transform in a stable, and predictable way; this is termed *equivariance.* This behavior has been motivational in the development of steerable filters [1], and we argue that obtaining such transformation equivariant representations is important for the architectures that we are considering as well. *Translational* equivariance is readily built in by a convolutional architecture as found in neural networks [2, 3], and more recently for RBMs see e.g. [4]. However, there are additional transformations that should be taken into account: in this paper we focus on equivariance with respect to in-plane *rotations.* Building in such property is important to avoid the system having to learn rotated versions of the same patterns at all levels in the network. For example in Fig. 2 of [4] many of the learned filters/ filter combinations shown are rotated versions of each other. The goal of this paper is to build a DBN architecture that is translation and rotation equivariant. To do this we introduce a novel kind of rotational/steerable unit for Boltzmann machines, as described in section 2.

One of the inspirations for this paper is the work of Fidler and Leonardis [5], in which conjunctions of edge and bar (sine and cosine) Gabor features are built up into more complex patterns that occur frequently in the input image ensemble. Their architecture is translation and rotation invariant. However, their method does not define a generative model of images, but rather performs a layerwise grouping of features from layer $\ell - 1$ to create features at layer ℓ. This means that it is heavily dependent on various thresholds used in the learning algorithm, and also that it is unable to carry out bottom-up/top-down inference in the face of ambiguous input or missing data. We show how such translation and rotation invariant groupings arise naturally in a fully-specified multi-layer generative model.

2 Building in Transformation Equivariance

We first discuss the rotation-equivariant restricted Boltzmann machine (STEER-RBM) model which has one hidden layer; this hidden layer contains the 'steerable' units which are a particular feature of our architecture. Next in section 2.2 we describe a translation equivariant version of the model, and finally in section 2.3 generalize this to a deep belief net, which is the multi-hidden-layer generalization of the translation and rotation equivariant model.

2.1 Rotation Equivariant RBMs

The key feature of the STEER-RBM is the construction of the stochastic steerable hidden units, each of which combines a binary-valued activation variable h_j turning the unit on/off with an associated discrete-valued rotation variable r_j taking on possible states $k = 1, \ldots K$, whose effect is to in-plane rotate the weights of the unit by $360(k-1)/K$ degrees. Let $W_j(\cdot, 1)$ be the canonical pattern of weights connecting hidden unit h_j to visible units $\mathbf{v}$ under no rotation. The transformed weights $W_j(\cdot, k)$ for rotation k are derived from the canonical view using geometrical knowledge of in-plane rotations, so that

$$W_j(\cdot, k) = \mathbf{R}^{(k)} W_j(\cdot, 1) \quad \Rightarrow \quad W_j(i, k) = \sum_{\ell=1}^{D} R^{(k)}(i, \ell) W_j(\ell, 1), \tag{1}$$

where $\mathbf{R}^{(k)}$ is a *fixed* $D \times D$ transformation matrix applying an in-plane rotation of $360(k-1)/K$ degrees, and D denotes the number of pixels/visible units in $\mathbf{v}$. Note that by choosing K large we can approximate rotations to any desired accuracy. An example of this rotation in action is shown in the top row of Figure 1. In our implementation, we bilinearly interpolate the weights into their new locations, such that each of the elements in the rotated view is computed as a convex combination of (maximally) four neighboring rotated canonical weights, each of which have been rotated about the center of the canonical weights plane[1]. Thus

[1] To avoid boundary artifacts with non-circular receptive fields, one can zero pad the canonical weights plane such that each of the rotated canonical weights are within the boundaries defined by the extended plane for any rotation angle.

each row of the rotation matrices contains maximally four non-zero elements which sum to one.

Given this architecture, the joint probability density of a STEER-RBM model consisting of visible units $\mathbf{v}$ and binary hidden units $(\mathbf{h}, \mathbf{r})$ is given by the Boltzmann-distribution $p(\mathbf{v}, \mathbf{h}, \mathbf{r} \mid \theta) \propto \exp\{-\mathrm{E}(\mathbf{v}, \mathbf{h}, \mathbf{r} \mid \theta)\}$ with the following energy, assuming continuous, conditionally Gaussian units:[2]

$$\mathrm{E}(\mathbf{v}, \mathbf{h}, \mathbf{r} \mid \theta) = \sum_i \frac{v_i^2 - 2av_i}{2\sigma^2} - \sum_j h_j b_j - \frac{1}{\sigma} \sum_j h_j \sum_i v_i W_j(i, r_j) \tag{2}$$

where $\theta = \{\mathbf{a}, \mathbf{b}, \mathbf{W}, \sigma\}$ consist of hidden unit biases $\mathbf{b}$, visible unit biases $\mathbf{a}$, connection weights $\mathbf{W}$, and the standard deviation of the Gaussian conditional distributions of the visible units σ. The energy function for binary visible units can be obtained by removing the quadratic term v_i^2, and setting σ to unity. As $W_j(i, r_j) = \sum_{k=1}^{K} \delta(k, r_j) W_j(i, k)$, the model defines a mixture of RBMs, but in contrast with the implicit mixture RBM model of [6], there is parameter sharing between the mixture components due to rotation equivariance. Although we have described the RBMs of above, extensions of other energy-based models to use rotational units could be also considered, such as conditionally full-covariance Gaussian models [7].

2.2 Rotation and Translation Equivariant RBMs

To learn models for whole images, a translation equivariant extension of the STEER-RBM is used, assuming a reduced connectivity structure so that a hidden unit h_j is connected to a subset of visible units specified by a receptive field system, and parameter sharing is used so that the responses of units to a stimulus are translation equivariant. We call the hidden units sharing these parameters a *feature plane*. To extend convolutional RBMs, the STEER-RBM also adds input rotation equivariance to hidden unit activation. Thus we consider a weight kernel ω_α for feature plane α, which is sufficient to define the connection weights between the hidden units in feature layer α and the visible units. The energy function for the convolutional STEER-RBM is then

$$\mathrm{E}(\mathbf{v}, \mathbf{h}, \mathbf{r} \mid \theta) = \sum_i \frac{v_i^2 - 2av_i}{2\sigma^2} - \sum_{\alpha, j} h_{\alpha j} \left(b_\alpha + \frac{1}{\sigma} \sum_{\ell \in \mathrm{N}_{\alpha j}} v_\ell \, \omega_\alpha(d(j, \ell), r_{\alpha j}) \right) \tag{3}$$

where a is visible unit layer bias, b_α is the bias for hidden unit feature plane α, $\mathrm{N}_{\alpha j}$ indexes the visible units within the receptive field of hidden unit $h_{\alpha j}$, σ defines the standard deviation of the univariate Gaussian conditional distribution $p(v_i \mid \mathbf{h}, \mathbf{r}, \theta)$, and $d(j, i)$ computes the spatial-offset dependent index of the weight kernel weight that is used to connect hidden unit h_j to v_i.

[2] The joint probability density of the visible units conditional on the hidden units and model parameters factorizes as a multivariate spherical-covariance Gaussian.

2.3 Rotation and Translation Equivariant Deep Belief Nets

To learn higher-level patterns from images, we follow the DBN approach of [8], stacking multiple layers of convolutional STEER-RBMs on top of each other. In this model, each of the hidden units in a higher level STEER-RBM is connected to a subset of the hidden units in each of the feature planes in the hidden layer below, again via by a receptive field system. As both the higher and lower level units are rotational, we now need a *triply* indexed weight parameter $\omega^{\ell_\alpha}_{\ell-1_\beta}(j,m,k)$ which connects a unit in feature plane α in layer ℓ to feature plane β in layer $\ell-1$ below. Here j denotes the spatial offset, while m and k index the rotational states in the lower and higher layers respectively. Thus the energy function between layers ℓ and $\ell-1$ is of the following form:

$$\begin{aligned}\mathrm{E}(\mathbf{h}^{\ell-1},\mathbf{r}^{\ell-1},\mathbf{h}^{\ell},\mathbf{r}^{\ell}\mid\theta^{\ell}) &= -\sum_{\beta} b^{\ell-1}_{\beta}\sum_{i} h^{\ell-1}_{\beta i} - \sum_{\alpha} b^{\ell}_{\alpha}\sum_{j} h^{\ell}_{\alpha j}\\ &-\sum_{\alpha}\sum_{j} h^{\ell}_{\alpha j}\sum_{\beta}\sum_{i\in\mathrm{N}^{\ell}_{\alpha j}} h^{\ell-1}_{\beta i}\,\omega^{\ell_\alpha}_{\ell-1_\beta}(d(j,i),r^{\ell-1}_{\beta i},r^{\ell}_{\alpha j}). \qquad (4)\end{aligned}$$

The computation of the transformed weights for these higher hidden layers has to be different from that of the first layer, since changing the rotational state of a higher level pattern needs to rotate the lower level rotational states/patterns accordingly. The transformations for each feature can be again done by knowledge using fixed transformation operators, by first in-plane rotating the lower-level rotation-specific canonical weight matrix slices, and then circularly shifting the dimensions of the resulting matrix. The non-canonical view of a level ℓ weight kernel can be thus written as follows:

$$\omega^{\ell_\alpha}_{\ell-1_\beta}(j,m,k) = \sum_{\rho=1}^{K}\mathbf{S}^{(k)}(\rho,m)\sum_{\delta} R^{(k)}(j,\delta)\,\omega^{\ell_\alpha}_{\ell-1_\beta}(\delta,\rho,1), \qquad (5)$$

where $\mathbf{S}^{(k)}$ is a *fixed* $K\times K$ binary matrix applying a circular shift (of $k-1$ shifts) forward to the columns of $\mathbf{R}^{(k)}\omega^{\ell_\alpha}_{\ell-1_\beta}(\cdot,\cdot,1)$.

3 Inference and Learning in the Models

As with standard RBMs, the conditional distributions of the hidden units are independent given $\mathbf{v}$ for the convolutional STEER-RBM. Thus we have for (3) that $p(\mathbf{h},\mathbf{r}\mid\mathbf{v},\theta)=\prod_\alpha\prod_j p(h_{\alpha j}\mid\mathbf{v},\theta)\,p(r_{\alpha j}\mid h_{\alpha j},\mathbf{v},\theta)$, where

$$p(h_{\alpha j}\mid\mathbf{v},\theta) = \frac{\sum_{k=1}^{K}\exp\left\{h_{\alpha j}\left(b_\alpha+\frac{1}{\sigma}\sum_{\ell\in\mathrm{N}_{\alpha j}} v_\ell\,\omega_\alpha(d(j,\ell),k)\right)\right\}}{K+\sum_{k=1}^{K}\exp\left\{b_\alpha+\frac{1}{\sigma}\sum_{\ell\in\mathrm{N}_{\alpha j}} v_\ell\,\omega_\alpha(d(j,\ell),k)\right\}}, \qquad (6)$$

$$p(r_{\alpha j}\mid h_{\alpha j}=1,\mathbf{v},\theta) = \frac{\exp\left\{b_\alpha+\frac{1}{\sigma}\sum_{\ell\in\mathrm{N}_{\alpha j}} v_\ell\,\omega_\alpha(d(j,\ell),r_{\alpha j})\right\}}{\sum_{k=1}^{K}\exp\left\{b_\alpha+\frac{1}{\sigma}\sum_{\ell\in\mathrm{N}_{\alpha j}} v_\ell\,\omega_\alpha(d(j,\ell),k)\right\}}. \qquad (7)$$

The key quantity in this computation is $a_{\alpha j}(k) = \sum_{\ell \in \mathrm{N}_{\alpha j}} v_\ell \, \omega_\alpha(d(j,\ell),k)$ which computes the dot product of the visible variables in $\mathrm{N}_{\alpha j}$ with weight kernel w_α at rotation k. (6) evaluates a nonlinear combination of these quantities summed over k compared to K in order to compute $p(h_{\alpha j} \mid \mathbf{v}, \theta)$. Similarly in (7) $p(r_{\alpha j} \mid h_{\alpha j} = 1, \mathbf{v}, \theta)$ is computed based on the relative strengths of the $a_{\alpha j}(k)$ terms. For a multi-layer network a crude approximation to full inference is to sample from the learned STEER-RBMs layerwise from bottom to top. More sophisticated alternatives are possible, such as the up-down algorithm described in [8], or e.g. some other Markov chain Monte Carlo sampling methods.

As usual with DBNs we learn the parameters of the models layer-wise. We have used stochastic gradient-descent based methods to train the models in the experiments, optimizing an objective function consisting of a data fit term, plus a term that encourages sparsity[3]. For a datafit term based on the log likelihood L, the gradient wrt a parameter θ is given by $\frac{\partial L}{\partial \theta} = \langle \frac{\partial E}{\partial \theta} \rangle^+ - \langle \frac{\partial E}{\partial \theta} \rangle^-$, where $\langle \cdot \rangle^+$ denotes expectation with the training data clamped, and $\langle \cdot \rangle^-$ the unclamped phase. In fact we generally use the contrastive divergence CD-1 approximation to the negative phase. To understand how the model learns under optimization, it is instructive to consider the partial derivatives of the energy function with respect to the canonical features. Assuming the model of (3), we have that

$$\frac{\partial \mathrm{E}(\mathbf{v}, \mathbf{h}, \mathbf{r} \mid \theta)}{\partial \omega_\alpha(\delta, 1)} = -\frac{1}{\sigma} \sum_j h_{\alpha j} \sum_{\ell \in \mathrm{N}_{\alpha j}} v_\ell \, R^{(r_{\alpha j})}(d(j,\ell), \delta). \tag{8}$$

This has the effect of multiplying the visible pattern in $\mathrm{N}_{\alpha j}$ by $(\mathbf{R}^{(r_{\alpha j})})^T$. As this is a close approximation to applying a reverse rotation, patterns which are detected to be present in a non-canonical orientation, are rotated 'back' into the canonical view, in which the feature-specific canonical statistics are then updated. The learning is similar for the higher layer models, where the alignment also takes into account the lower unit's rotation assignment. Partial derivatives with respect to the biases take the standard forms.

4 Experiments

We first learnt RBM models (3) from a set of whitened natural images [9] using CD-1 learning[4]. Fig. 1 (left, top) shows the type of feature consistently learnt as the most significant, at various rotations. This is an "edge detector", similar to the features found e.g. in [4] at various orientations. The bottom row shows a natural image patch, and most likely states colour-coded according to orientation, at each location. The responses occur at edge-like structures; notice the steady rotational response change, e.g. while tracing the outline of the central object. We have also trained this model with several feature planes; results using three are shown (right). To validate the higher-layer learning we first consid-

[3] Non-sparsity is penalized proportional to a sum of feature-plane specific cross-entropies, each between a Bernoulli target distribution, and the distribution recording the average probability of a unit being off or on at the plane, similar to [6].

[4] σ was set close to the data standard deviation. The total target activation for sparsity encouragement was 0.1.

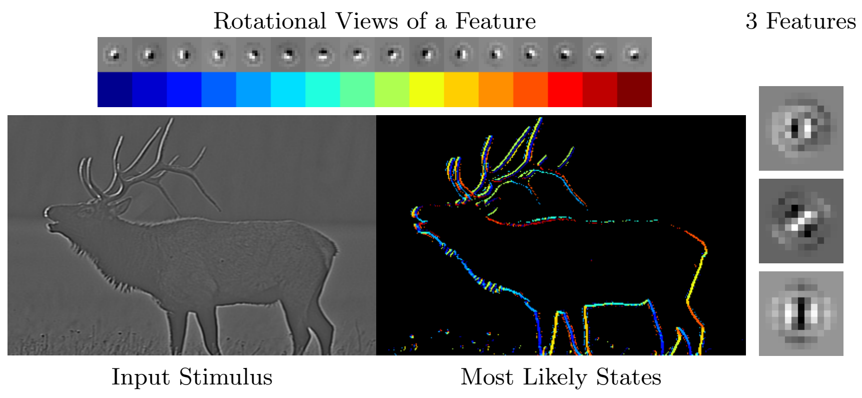

Fig. 1. Left-top: Learned feature at various orientiations (with receptive field diameter = 9). Left-bottom: Whitened natural image region, and most likely unit states (colour-coded according to rotation). Right: Weights of the learned set of 3 features.

ered modeling artificial rotational pattern data simulating first-layer responses. Fig. 2 shows input patterns; the four colours denote four different orientation responses. There are two patterns in the noisy data, one consisting of 3 active inputs, and the other of 2. These are successfully learned, see panels 2 and 4, and the caption for more details. We have also applied the learning to the natural images, using the single edge-like feature in the first layer. The results (see Fig. 3) show this yields higher-order conjunctions of this feature, such as extended and curved edges, and intersections. Note that these features are similar to SIFT-descriptors [10], but in a generative framework.

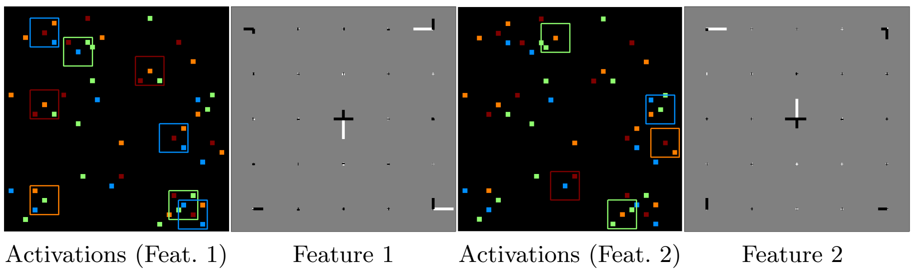

Fig. 2. Features learned from rotation colour-coded (→,↑,←,↓) artificial data containing rotated, randomly placed instances of two rotational shapes, in clutter. Panels 1 and 3 show the data, with the respective higher level features denoted by bounding boxes of the units' receptive field size centered on the unit location, and coloured according to the rotation assignment. In panels 2 and 4 the 5×5 canonical weight kernels are visualized using oriented black/white line segments, placed to start from an evenly spaced grid. The grid locations denote the spatial offsets for the weight kernels weights, the different orientations index the lower-level rotational states, the segment lengths denote the weight magnitude, and colour denoting the sign with black denoting a negative, and white denoting positive a weight. (Essentially this extends the Hinton plot to deal with (multi-way-)oriented weights.)

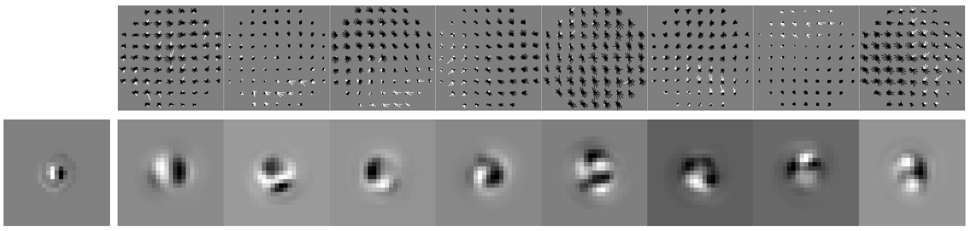

Fig. 3. (Top) The second layer weights (displayed as in Fig. 2) for 8 second-layer features. (Bottom) The first panel shows the first-layer basis feature; the other panels show the (linearly combined) first layer basis projections of the 8 features.

5 Related Work

We have discussed above the work of Fidler and Leonardis [5]. The work of Zhu *et. al.* [11] is similar to [5], except that there is a top-down stage to the learning process (but not in the given the inference algorithm) to fill in missing parts of the hierarchy. Both papers use hand-crafted algorithms for detecting groupings of lower-level features, involving various thresholds. In contrast we formulate the problem as a standard DBN learning algorithm, but build in transformation equivariance. One advantage of the DBN is that it is naturally set up for bottom-up/top-down inference in the face of ambiguity or missing data.

The orientation-adapted Gaussian scale mixture (OAGSM) model [12] describes how a Gaussian model for a wavelet coefficient responses corresponding to an image patch can be augmented with latent variables to handle signal amplitude and dominant orientation. This allows e.g. modelling of oriented texture at arbitrary rotations. The learned edge filters at the first level of our model are analogous to the wavelet responses, while our second level units model the correlations between the coefficients. However, note (i) that the OAGSM model is only a model for patches not entire images, and (ii) that it does not provide a mixture model over the types of higher-level regularity, e.g. lines, corners, T-junctions etc. On the other hand the real-valued modelling of wavelet coefficients by OAGSM is more powerful than the binary activations of units in the STEER-DBN.

Our goal is to build in equivariance to known (translational and rotational) transformations. In contrast Memisevic and Hinton [13] describe how to *learn* transformations based on pairs of training images using factored 3-way Boltzmann machines. Such a network could be used in to identify rotated versions of a given pattern, e.g. by fixing a reference version of the pattern and inferring the transformation. However, it seems rather excessive to learn the machinery for this when it can be built in. Our work should not be confused with the directional unit Boltzmann machine (DUBM) network of Zemel *et al* [14]. Although DUBM units contain a rotational variable, this is not used to model relative rotations of subcomponents. For example in [15] the authors present a convolutional architecture where the rotational variable denotes the phase of an oscillator, relating to the theory of binding-by-synchrony.

6 Discussion

As we have shown, the STEER-DBN architecture handles translation and rotation invariances. The other natural transformation to consider is image scaling. However, this can be relatively easily handled by the standard computer vision method of downsampling the input image by various factors, and applying the similar processing to each scale. Higher layers at a given scale can also take inputs from various scales. Alternatively one could introduce scaling assignment variables for each unit similar to the ones for rotation, scaling the features.

Other future work includes learning more hidden layers, and using more expressive bottom-layer models, such as those allowing dependent Gaussian distributions for the visibles conditional on the hidden units [7].

Acknowledgements JJK is partially funded by a SICSA studentship. This work is supported in part by the IST Programme of the European Community, under the PASCAL2 Network of Excellence, IST-2007- 216886. This publication only reflects the authors' views.

References

1. Simoncelli, E.P., Freeman, W.T., Adelson, E.H., Heeger, D.J.: Shiftable multi-scale transforms. IEEE Trans. IT **38** (1992) 857–607
2. Waibel, A., Hanazawa, T., Hinton, G.E., Shikano, K., Lang, K.: Phoneme recognition using Time Delay Neural Networks. IEEE Trans. ASSP **37** (1989) 328–339
3. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86** (1998) 2278–2324
4. Lee, H., Grosse, R., Ranganath, R., Ng, A.Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: ICML. (2009)
5. Fidler, S., Leonardis, A.: Towards scalable representations of visual categories: Learning a hierarchy of parts. In: CVPR. (2007)
6. Nair, V., Hinton, G.E.: 3D object recognition using deep belief nets. In: NIPS 22. (2009)
7. Ranzato, M., Mnih, V., Hinton, G.E.: Generating more realistic images using gated MRF's. In: NIPS 23. (2010)
8. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Comp. **18** (2006) 1527–1554
9. Olshausen, B.A., Field, D.J.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature **381** (1996) 607–609
10. Lowe, D.G.: Distinctive image features from scale–invariant keypoints. IJCV **60** (2004) 91–110
11. Zhu, L., Lin, C., Huang, H., Chen, Y., Yuille, A.: Unsupervised structure learning: Hierarchical recursive composition, suspicious coincidence and competitive exclusion. In: ECCV. (2008)
12. Hammond, D., Simoncelli, E.P.: Image modelling and denoising with orientation-adapted Gaussian scale mixtures. IEEE Trans. IP **17** (2008) 2089–2101
13. Memisevic, R., Hinton, G.E.: Learning to represent spatial transformations with factored higher-order Boltzmann machines. Neural Comp. **22** (2010) 1473–1492
14. Zemel, R.S., Williams, C.K.I., Mozer, M.C.: Lending direction to neural networks. Neural Networks **8** (1995) 503–512
15. Mozer, M.C., Zemel, R.S., Behrmann, M., Williams, C.K.I.: Learning to segment images using dynamic feature binding. Neural Comp. **4** (1992) 650–665