Learning to Discriminate in the Wild: Representation-Learning Network for Nuisance-Invariant Image Comparison

Nikolaos Karianakis
University of California
Los Angeles, CA 90095
USA
nikos.karianakis@gmail.com

Yizhou Wang
Peking University
Beijing, 100871
China
Yizhou.Wang@pku.edu.cn

Stefano Soatto
University of California
Los Angeles, CA 90095
USA
soatto@cs.ucla.edu

Abstract

We test the hypothesis that a representation-learning architecture can train away the nuisance variability present in images, owing to noise and changes of viewpoint and illumination. First, we establish the simplest possible classification task, a binary classification with no intrinsic variability, which amounts to the determination of co-visibility from different images of the same underlying scene. This is the Occlusion Detection problem and the data are typically two sequential, but not necessarily consecutive or in order, video frames. Our network, based on the Gated Restricted Boltzmann machine (Gated RBM), learns away the nuisance variability appearing on the background scene and the occluder, which are irrelevant with occlusions, and in turn is capable of discriminating between co-visible and occluded areas by thresholding a one-dimensional semi-metric. Our method, combined with Superpixels, outperforms algorithms using features specifically engineered for occlusion detection, such as optical flow, appearance, texture and boundaries. We further challenge our framework with another Computer Vision problem, Image Segmentation from a single frame. We cast it as binary classification too, but here we also have to deal with the intrinsic variability of the scene objects. We perform boundary detection according to a similarity map for all pairs of patches and finally provide a semantic image segmentation by leveraging Normalized Cuts.

1. Introduction

Representation-learning architectures have shown the ability to learn class-specific variability despite significant nuisance variability [9, 17, 19, 27, 37, 38]. The problem of *nuisance variability* is particularly acute in Computer Vision, where even the same object or scene can yield a large variety of images depending on vantage point, illumination and partial occlusion, which can be nuisance factors for certain tasks [33]. This point has been recently emphasized by Poggio [26], who set forth the hypothesis that much of the ventral stream is tasked with managing the infinite amount of nuisance variability, and by Sundaramoorthi et al. [35], who showed that the intrinsic variability of objects in images is infinitesimal compared to nuisance variability. These theses would seem to challenge the possibility that nuisance variability in images can be *learned away* by even powerful learning architectures. In this manuscript, we put this challenge to the test by establishing two visual classification tasks, and deploying a fairly simple *representation-learning* architecture to tackle them.

The first task we selected is the determination of *co-visibility*. This is a binary decision where, given two video frames, we wish to determine for each pixel whether or not back-projects onto the same point in physical space. This completely eliminates intrinsic variability, because the underlying scene is known to be the same and the diversity between images of the same scene is entirely attributable to nuisance factors such as different vantage points and illumination. We then deploy a scheme based on factorized *Gated Restricted Boltzmann machine* [23] and *Superpixels* [24] of different scales to learn away such nuisance variability, given training samples consisting of multiple pairs of images related by different transformations. Violation of co-visibility occurs in regions of an image that are not recognized as sufficiently similar according to the model.

The second problem we deal with is segmentation in a single image, which is also cast as binary classification. Class variability makes nuisance elimination more challenging, but we show that the Gated RBM in combination with *Normalized Cuts* [32] yield a semantic final segmentation. We present our framework in Sec. 2, Sec. 3 has the experiments and Sec. 4 consists of our conclusions. The upshot is that, even though in theory nuisances account for almost all the variability in the data [35], in practice the finite cardinality of data space acts as a regularizer, and since the classification occurs in data space, nuisance variability can be learned away.

1.1. Related work

The determination of co-visibility is related to the general problem of *correspondence*, that underlies a significant portion of Computer Vision research [31]. When correspondence is trivial, for instance when multiple images of the same scene are taken from a stationary camera at different time instants, this problem is known as *background subtraction* [25] and violations of co-visibility are due to the presence of moving foreground objects. In the more general setting, the determination of co-visibility is entangled with correspondence, so this problem relates to *optical flow*, another broad concern in the Computer Vision literature [1, 3, 4, 11, 15, 29, 34, 36]). Occlusion detection is often formulated as classification problem, where motion estimation is performed in a discrete setting ([18, 20]), which is a well-known difficult problem. Occlusion detection is closely related with occlusion boundary detection, where estimations are performed in video sequences [2, 11, 15, 21, 29, 34, 36] or single images [13, 30]. Martin et al. [22] fuse multiple cues from local image measurements to precisely infer the object boundaries in natural scenes. We compare with the occlusion learning work of Humayun et al. [14], who use various hand-crafted visual features, a subset of which is selected for each testing pair within a Random Forest-based framework.

1.2. Contributions

The flexibility of our method lies in the fact that by selecting an appropriate training set, the network becomes insensitive to nuisance variability due to certain predetermined factors (e.g., viewpoint and illumination changes), so the residual is informative for other factors, such as co-visibility in our occlusion detection setting or interclass variability for segmentation. We use a large training set, which has been generated by applying various transformations on random binary images. In our occlusion detection method there is no constraint regarding the order of the frames or the baseline range, as it is just a powerful image comparison mechanism which proves to be robust with local deformations and non-rigid motions. Additionally, there is no assumption for the orientation of the occlusion boundaries or the shape of the occluded regions. However, discriminating between occlusions and disocclusions has a small post-processing overhead. It is noticeable that our occlusion detection algorithm, when considering the superpixel maps as well, often outperforms state-of-the-art algorithms based on optical flow and miscellaneous visual features. Moreover, although the training needs hours depending on the size of the training set (~ 5 hours for $30,000$ image pairs with size 13×13 on a standard laptop), it is performed once and offline, and then the testing (e.g., 640×480 images) takes only a few seconds. Finally, we propose applying our network on image segmentation and demonstrate how invariance and patchwise comparisons contribute to a semantically meaningful segmentation.

2. Framework for Nuisance-Invariant Image Comparison

Boltzmann machines are probabilistic bidirectionally connected networks that capture important information of an unknown distribution based on samples from this distribution. However, their learning is computationally consuming. *Restricted Boltzmann machines* impose the probabilistic restriction of statistical independence between variables of same layer given the state of variables of all other layers and simplifies the learning process. The 2-layer architecture can be modeled as bipartite undirected graph.

2.1. Gated Restricted Boltzmann machine

A *Gated Restricted Boltzmann machine* is a parameterized generative model representing a probability distribution. Given some observations (i.e., the training data), learning means adjusting the parameters so that the represented distribution fits the training data as well as possible. The Gated RBM consists of 3 layers of binary variables: two layers of visible units that correspond to the observations and one hidden layer, which encodes dependences between two observable layers. Therefore, this model can capture the relationship (modulo a set of factors that it is trained to be invariant to) and in turn "similarity" between two images.

A Gated RBM consists of K hidden units $\mathbf{H} = (H_1, \ldots, H_K)$ that capture the dependencies between two layers of observed variables with units $\mathbf{X} = (X_1, \ldots, X_I)$ and $\mathbf{Y} = (Y_1, \ldots, Y_J)$. Adopting binary random variables, ($\mathbf{X}$, $\mathbf{Y}$, $\mathbf{H}$) takes values ($\mathbf{x}$, $\mathbf{y}$, $\mathbf{h}$) $\in \{0,1\}^{I+J+K}$. Given that the image transformations do not include arbitrary motions of individual pixels, this gives us the intuition to model the three-way interactions among the layers as the product of all possible two-way interactions with F factors [23]. Thus, the joint probability distribution is $p(\mathbf{x}, \mathbf{y}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{x},\mathbf{y},\mathbf{h};\theta)}$ with energy function

$$E(\mathbf{x}, \mathbf{y}, \mathbf{h}; \theta) = -\sum_{f=1}^{F}(\sum_{i=1}^{I} u_{if} x_i)(\sum_{j=1}^{J} v_{jf} y_j)(\sum_{k=1}^{K} w_{kf} h_k) - \sum_{i=1}^{I} a_i x_i - \sum_{j=1}^{J} b_j y_j - \sum_{k=1}^{K} c_k h_k, \quad (1)$$

where $\theta = \{\mathbf{U}, \mathbf{V}, \mathbf{W}, \mathbf{a}, \mathbf{b}, \mathbf{c}\}$ are the model parameters and $Z(\theta) = \sum_{\mathbf{x},\mathbf{y},\mathbf{h}} e^{-E(\mathbf{x},\mathbf{y},\mathbf{h};\theta)}$ is the *partition* function. Instead

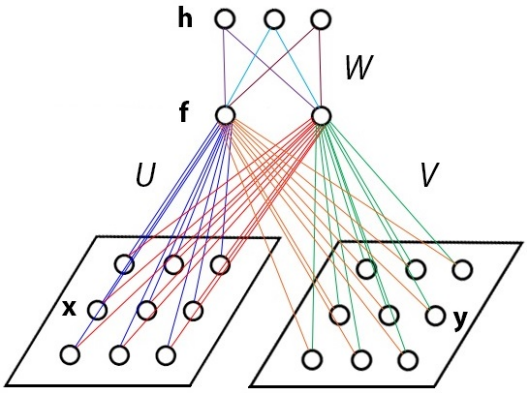

Figure 1: Graphical representation of a Gated Restricted Boltzmann machine (Gated RBM).

of $I \times J \times K$ interaction tensor, three matrices with sizes $I \times F$, $J \times F$ and $K \times F$ are factorized in a common product. Hence, the order of parameter complexity decreases from cubic to square. For all $i \in \{1, \ldots, I\}$, $j \in \{1, \ldots, J\}$, $k \in \{1, \ldots, K\}$ and for all $f \in \{1, \ldots, F\}$, u_{if}, v_{jf} and w_{kf} are real-valued weights associated with the f factor and i, j or k unit, correspondingly. Weight matrices $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{W}$ consist of "filters" $\{\mathbf{u}_f, \mathbf{v}_f, \mathbf{w}_f, f = 1 \ldots F\}$. Additionally, a_i, b_j and c_k are real-valued bias terms associated with the ith and jth visible units and kth hidden unit, respectively. The model is illustrated in Fig. 1.

Intuitively the first energy term represents a similarity score, as its high value coincides with co-occurrence of high projection scores of images $\mathbf{x}$, $\mathbf{y}$ and some subset of hidden variables $\mathbf{h}$ on F factors. The filters' shape and the semantics of similarity inferred by the model depend on the training set. For example, after training with pairs of images which are related by affine transformations, the hidden variables capture "elementary" dependencies between the observed variables like translational shifts, planar rotations and other small-dimensional (local) group transformations. In that case, two testing images will be considered as "similar" by the model when they are almost identical or parts of them are related by affine transformations of magnitude similar with these ones appearing on the training data.

The *symmetric* model is a special case where the weights of both visible layers are equal, that is $\{u_{if} = v_{if}, i = 1 \ldots I, I = J\}$. It operates a complex transformation that is determined by the hidden layer and maps a set of representations of one visible layer to the other. The representations are projections on a common space, which topologically "compensates" the transformations that appear on the training set. More generally, the non-symmetric model is essentially a mapping induced by the hidden layer between different, but related (according to the training set) representations of the observable layers. In Fig. 2, we show the observed layers' filters $\{\mathbf{u}_f, \mathbf{v}_f, f = 1 \ldots F, F = 100\}$ when the model is trained exclusively with shifted and scaled image pairs, respectively. The non-symmetric Gated RBM can be applied on image pairs of different size ($I \neq J$), while the numbers of hidden variables K and factors F can be selected by the user. We mainly experimented with values: $I = J = 13 \times 13 = 169$ and $I = J = 26 \times 26 = 676$, $K = 50$–200, $F = 100$–200.

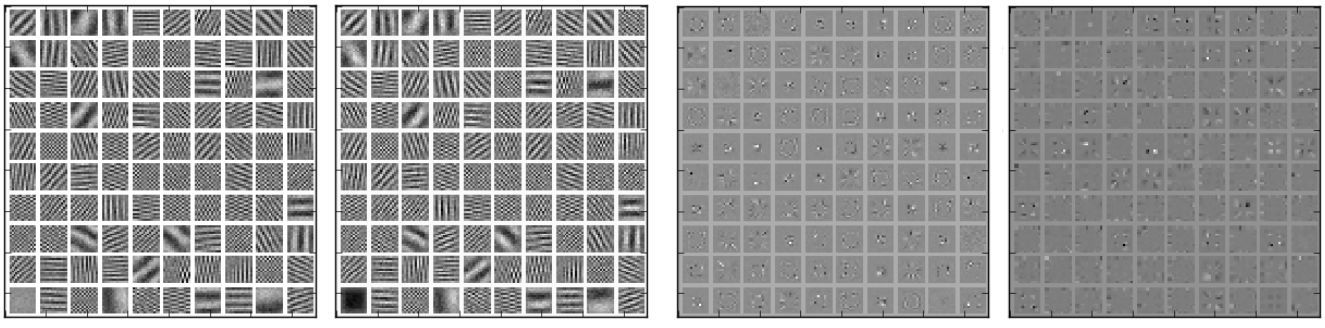

Figure 2: Filters generated when training exclusively with shifted (left) and scaled (right) random binary images, respectively.

2.2. Conditionals and Marginals

The complexity advantage of Restricted Boltzmann machines is that all variables of one layer are independent given the state of all other layers' variables. Thus, the joint conditional distribution is the product of all conditional distributions and calculations can be done in parallel. After some mathematical manipulations[1], the conditional distributions are:

$$p(\mathbf{x}|\mathbf{y},\mathbf{h}) = \prod_{i=1}^{I} \mathcal{B}(x_i; \sigma[\sum_{f=1}^{F} u_{if}(\sum_{j=1}^{J} v_{jf} y_j)(\sum_{k=1}^{K} w_{kf} h_k) + a_i]) \tag{2a}$$

$$p(\mathbf{y}|\mathbf{x},\mathbf{h}) = \prod_{j=1}^{J} \mathcal{B}(y_j; \sigma[\sum_{f=1}^{F} v_{jf}(\sum_{i=1}^{I} u_{if} x_i)(\sum_{k=1}^{K} w_{kf} h_k) + b_j]) \tag{2b}$$

$$p(\mathbf{h}|\mathbf{x},\mathbf{y}) = \prod_{k=1}^{K} \mathcal{B}(h_k; \sigma[\sum_{f=1}^{F} w_{kf}(\sum_{i=1}^{I} u_{if} x_i)(\sum_{j=1}^{J} v_{jf} y_j) + c_k]) \tag{2c}$$

where $\mathcal{B}(x;p)$ is the pdf of a Bernoulli random variable x with parameter p and $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid activation function. The distribution over an image pair $(\mathbf{x}, \mathbf{y})$ is taken by marginalizing the joint distribution over $\mathbf{h}$:

$$p(\mathbf{x},\mathbf{y}) = \sum_{\mathbf{h}\in\{0,1\}^K} p(\mathbf{x},\mathbf{y},\mathbf{h}). \tag{3}$$

The number of possible $\mathbf{h}$ increases exponentially with the number K of hidden variables, making the computation intractable for reasonable values. However, approximating the unknown distribution with Gibbs sampling allows us to work only with the conditionals. This fact, along with the conditional independence among variables in each layer of Gated RBM given the other two layers, make computational cost reasonable. Additionally, a GPU-based implementation[2] of the model speeds up the training process by an order of magnitude.

2.3. Maximum Likelihood Learning

Given a set of *i.i.d.* training examples $D = \{(\mathbf{x}^{(1)},\mathbf{y}^{(1)}),\dots(\mathbf{x}^{(N)},\mathbf{y}^{(N)})\}$, the model parameters θ are learned via an unsupervised learning framework. The log-likelihood given D observed training pairs is maximized:

$$\max \ \log \mathcal{L}(\theta|\mathbf{x},\mathbf{y}) = \frac{1}{N}\sum_{n=1}^{N} \log \ p(\mathbf{x}^{(n)},\mathbf{y}^{(n)};\theta). \tag{4}$$

For a single training pair $(\mathbf{x},\mathbf{y})$ the log-likelihood gradient w.r.t. a single model parameter θ is:

$$\frac{\partial \log \mathcal{L}(\theta|\mathbf{x},\mathbf{y})}{\partial\theta} = \frac{\partial \log(\frac{1}{Z}\sum_{\mathbf{h}} e^{-E(\mathbf{x},\mathbf{y},\mathbf{h};\theta)})}{\partial\theta} \tag{5a}$$

$$= \frac{\partial(\log\sum_{\mathbf{h}} e^{-E(\mathbf{x},\mathbf{y},\mathbf{h})} - \log\sum_{\mathbf{x},\mathbf{y},\mathbf{h}} e^{-E(\mathbf{x},\mathbf{y},\mathbf{h})})}{\partial\theta} \tag{5b}$$

$$= -\sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{x},\mathbf{y})\frac{\partial E(\mathbf{x},\mathbf{y},\mathbf{h})}{\partial\theta} + \sum_{\mathbf{x},\mathbf{y},\mathbf{h}} p(\mathbf{x},\mathbf{y},\mathbf{h})\frac{\partial E(\mathbf{x},\mathbf{y},\mathbf{h})}{\partial\theta}. \tag{5c}$$

By combining Eqs. 4 and 5c, the mean of this derivative over the training set can be expressed as:

$$\frac{1}{N}\sum_{n=1}^{N}\frac{\partial \log \mathcal{L}(\theta|\mathbf{x},\mathbf{y})}{\partial\theta} = \frac{1}{N}\sum_{n=1}^{N}\left(-\mathbb{E}_{p(\mathbf{h}|\mathbf{x},\mathbf{y})}\left[\frac{\partial E(\mathbf{x},\mathbf{y},\mathbf{h})}{\partial\theta}\right] + \mathbb{E}_{p(\mathbf{x},\mathbf{y},\mathbf{h})}\left[\frac{\partial E(\mathbf{x},\mathbf{y},\mathbf{h})}{\partial\theta}\right]\right) \tag{6a}$$

$$= \left\langle \frac{\partial E(\mathbf{x},\mathbf{y},\mathbf{h})}{\partial\theta} \right\rangle_{p(\mathbf{h}|\mathbf{x},\mathbf{y})q(\mathbf{x},\mathbf{y})} - \left\langle \frac{\partial E(\mathbf{x},\mathbf{y},\mathbf{h})}{\partial\theta} \right\rangle_{p(\mathbf{x},\mathbf{y},\mathbf{h})} \tag{6b}$$

$$\propto \left\langle \frac{\partial E(\mathbf{x},\mathbf{y},\mathbf{h})}{\partial\theta} \right\rangle_{data} - \left\langle \frac{\partial E(\mathbf{x},\mathbf{y},\mathbf{h})}{\partial\theta} \right\rangle_{model}. \tag{6c}$$

[1] Please see the Appendix for detailed mathematical derivations/proofs.

[2] Publicly available online from Roland Memisevic and Josh Susskind at *http://learning.cs.toronto.edu/ rfm/code/gbmcuda.py*. Additionally, the code that generates the training set along with the Occlusion Detection and Image Segmentation code, which build on the model, will be uploaded on my website after paper's publication. The entire source code is written in Python.

Algorithm 1 Training with 3-way Contrastive Divergence

Input: Gated RBM $(\mathbf{X}, \mathbf{Y}, \mathbf{H})$, training batch D
Output: Weights (model parameters) update $\Delta\theta$

Initialize all weights $\Delta\theta = 0$
for all $(\mathbf{x}, \mathbf{y}) \in D$ **do**
  $(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}) \leftarrow (\mathbf{x}, \mathbf{y})$
  for $t = 0, \ldots, k-1$ **do** in random order
    $\forall k = 1, \ldots, K$ sample $h_k^{(t)} \sim p(h_k|\mathbf{x}^{(t)}, \mathbf{y}^{(t)})$
    $\forall i = 1, \ldots, I$ sample $x_i^{(t+1)} \sim p(x_i|\mathbf{h}^{(t)}, \mathbf{y}^{(t)})$
    $\forall j = 1, \ldots, J$ sample $y_j^{(t+1)} \sim p(y_j|\mathbf{h}^{(t)}, \mathbf{x}^{(t)})$
  end for
  for all weights **do**
    $\Delta\theta = \Delta\theta - \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{v}^{(0)}) \frac{\partial E(\mathbf{v}^{(0)}, \mathbf{h})}{\partial \theta} + \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{v}^{(k)}) \frac{\partial E(\mathbf{v}^{(k)}, \mathbf{h})}{\partial \theta}$
  end for
end for

The second term in Eq. 6c is intractable, as it is computed over all configurations $(\mathbf{x}, \mathbf{y}, \mathbf{h})$ (increases exponentially with $I + J + K$ number of units). However, Gibbs sampling of the unknown distribution gives a tractable approximation. Samples are drawn alternatingly from the conditional distributions $p(\mathbf{h}|\mathbf{x}, \mathbf{y})$, $p(\mathbf{x}|\mathbf{h}, \mathbf{y})$ and $p(\mathbf{y}|\mathbf{h}, \mathbf{x})$ in random order and the sampling process is terminated in k steps (usually $k = 1$ works well [12]). Given the tri-partite structure of the model, the learning process has been characterized as *3-way Contrastive Divergence* [37]. The training process is summarized in Alg. 1.

2.4. Distance function

The model can be trained with pairs of images related by many transformations. This process makes it invariant over all these transformations, so an appropriate similarity score given by the model can potentially discriminate between similar/non-similar images modulo these factors. The log-likelihood that is assigned to a testing image pair $(\mathbf{x}, \mathbf{y})$ is:

$$log\ p(\mathbf{x}, \mathbf{y}) = -logZ + \sum_{i=1}^{I} a_i x_i + \sum_{j=1}^{J} b_j y_j + \sum_{k=1}^{K} log(1 + e^{c_k + \sum_{f=1}^{F} w_{kf} (\sum_{i=1}^{I} u_{if} x_i)(\sum_{j=1}^{J} v_{jf} y_j)}). \quad (7)$$

The normalizing term $logZ$ is very demanding to compute, as it is marginalized over $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$. Fortunately, when we compare many pairs of images, this term is common and can be eliminated. However, to use the *unnormalized* likelihood as distance of two images is problematic, as the likelihood of a single pair $(\mathbf{x}, \mathbf{y})$ could be made arbitrarily small by rescaling both images with some constant. To deal with that the following distance function is used:

$$d(\mathbf{x}, \mathbf{y}) = -log\ p(\mathbf{x}, \mathbf{y}) - log\ p(\mathbf{y}, \mathbf{x}) + log\ p(\mathbf{x}, \mathbf{x}) + log\ p(\mathbf{y}, \mathbf{y}), \quad (8)$$

as was first proposed in [39] for a 2-layer Restricted Boltzmann machine. The normalizing terms are eliminated, and the likelihood of any single image is normalized for both observable layers. Strictly speaking, d is a *semi-metric*, because the triangle inequality is not guaranteed to hold among three testing images.

3. Experiments

After training with a large dataset of image pairs which are related by a specific set of transformations, the model is invariant with respect to them and the distribution of distances calculated by Eq. 8 for a set of testing image pairs can be informative for determining other factors, such as *occlusions* and *interclass variability*.

3.1. Occlusion Detection

The model is trained using Alg. 1 with training image pairs related by affine transformations, shifts and rotations, scale and illumination variation. The first factors intend to deal with different vantage points where these images are captured from, while the latter one with different lightning conditions. In our experiments the Gated RBM's observable layers have

Figure 3: Occlusion Detection between frames 7 and 8 of the "Cars8" sequence of the Berkeley Motion Segmentation dataset. The occlusion (and disocclusion) areas are displayed on both frames. The left image pair is obtained with the baseline algorithm, which gives many false alarms on turbulent and with variant lightning scene areas, such as the road and the car's front surface. The right image pair displays our detection having used the aggregate superpixel distance from Eq. 9 and $m = 8$ superpixel maps.

size either 13×13 or 26×26 and the range of pixel-wise transformations on the training set varies between $3 - 6$ pixels. In order to obtain the results of this section, $F = 200$ filters and $K = 100$ hidden variables were used. The model was trained over $10,000$ epochs, where the training set included $10,000$ purely shifted, $5,000$ purely rotated images, $5,000$ general affine transformations, $5,000$ illumination variant and $5,000$ scaled pairs. All these transformations were applied to random, binary training images, which empirically proved to give equally effective model compared to when training with patches cropped from natural images. Applied transformations is all that counts instead of specific information of any single image/visible layer. Batch size $D = 100$–$1,000$ and $5,000$–$10,000$ epochs were used in these experiments.

During testing, two frames were partitioned into $d \times d$ densely overlapping patches ($d = 13$ or $d = 26$) and Eq. 8 was used to estimate the "distance" of corresponding patches according to the model. After training over all these factors, which are nuisances in our setting, $d(\mathbf{x}, \mathbf{y})$ provides a score to quantify co-visibility in a testing pair $(\mathbf{x}, \mathbf{y})$ because occlusions are the main cause of disagreement. Thresholding the distance map yields the binary occlusion map. Comparisons are made at the patch level, but the resulting distance is applied only to the central pixel of each patch. All comparisons can be performed in parallel. Our framework was tested on sequential video frames taken from Berkeley Motion Segmentation [6], Middlebury [4] and UCL Optical Flow [1] datasets.

A *baseline* algorithm can be built where simple differences of average intensities over 13×13 patches are extracted and thresholded. This procedure yields a large number of false alarms, because any movement or lightning change in the background or the occluder affects the "naive" patch distance. On the other hand, our network's invariance over all these transformations provides background/foreground subtraction and the residual is mostly occlusions. Fig. 3 demonstrates this concept.

Toward superpixels: Training the model with bigger visible layers offers invariance over larger transformations, but it typically gives less accurate predictions close to the boundaries of the occlusion regions, as the model needs to examine a bigger patch and deal with more nuisances. It can drive occlusion detection though by providing a mask that offers subtraction of the larger nuisances and then testing with smaller visible layers refines the occluded regions. Moreover, a deeper architecture would not be especially helpful, because the primary purpose of this network is to perform image comparison modulo small deformations. It does not intend to simulate complicated transformations like facial expressions or body poses. However, empirical work suggests that the network *per se* can give decent, but not competitive results, mainly because of computational resources limitations. Training a very large Gated RBM with thousands of hidden variables and filters over all possible transformations in theory could give an oracle that could eliminate all possible nuisances and in turn discriminate occlusions with infinitesimal classification error. In practice, though, for a computationally tractable solution that yields competitive occlusion detection, we turn to *superpixels*.

Superpixels are basically uniform intensity areas on the image domain and our conjecture is that with high probability their pixels back-project to points in the scene with the same motion. Therefore, it is natural to resolve for entire superpixels whether they are co-visible or not. Averaging the model's distances over the whole superpixel is a meaningful grouping mechanism, which is robust against outliers and follows faithfully the occlusion boundaries. However, superpixel partitions of any single image are not useful when comparing image pairs. Therefore, we design a mechanism of *jointly* extracting superpixels in two images (one common superpixel partition) in order to have pixel groups that "follow" the boundaries on both frames and share common appearance/texture. The superpixel code [24] is based on the Boundary Detector from [22],

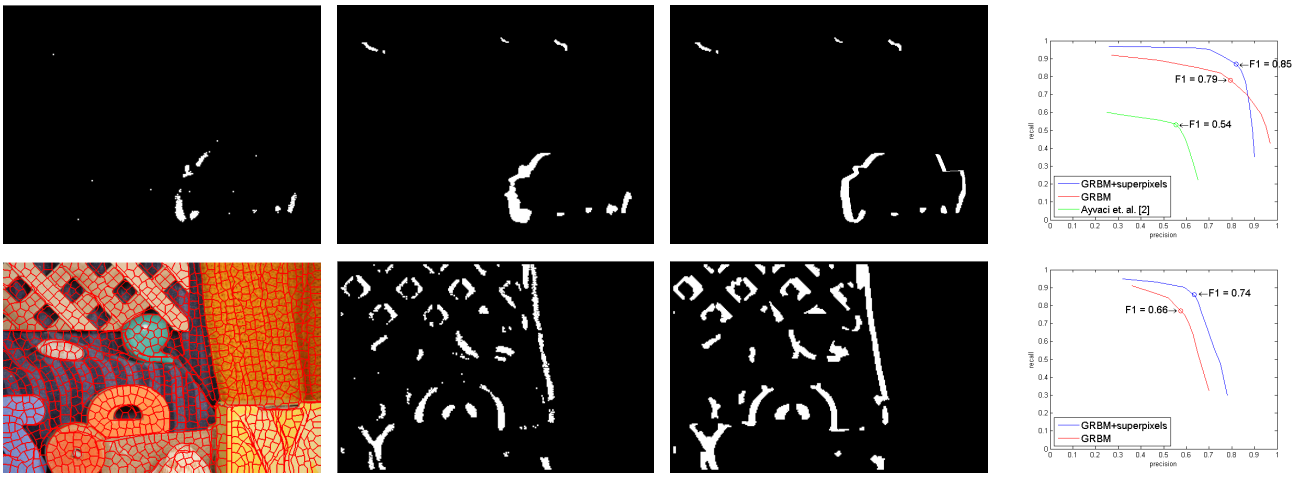

Figure 4: This set of images demonstrates the influence of superpixel information in the occlusion detection task. The first row pertains to the "Cars8" sequence from Fig. 3, while the second row shows results from the "RubberWhale" sequence of Middlebury dataset (frames 7 and 14). At the first two images of first row we see the occlusion detection without and with superpixel information, respectively. The third image is the ground truth. The PR curve shows an improved detection with superpixels (maximum $F1$-score along the PR curve 85% instead of 79%). The corresponding curve from Ayvaci et al. [3] is also displayed. In the second row, the first figure displays the joint superpixel partition, while the next two figures represent the occlusion detection without and with superpixels considered. The occlusion regions consist of fewer, more compact connected components, have fewer outliers, and fit better on the occluders' boundaries. The PR curves verify the visual impression.

and in order to extract joint superpixels we modified it adopting as edge probability map the maximum of the two images' probability maps and choosing as angle θ for every pixel (i, j) the corresponding angle $\theta_1(i, j)$ or $\theta_2(i, j)$ of the image with dominant gradient there. Furthermore, in order to obtain a mechanism that is less dependent on the algorithm's randomness, the maximum number of pixels per superpixel, the number of eigenvectors and other parameters, we extracted m superpixel partitions for each testing image pair according to various values for the above-mentioned parameters and averaged the distance scores over all of them. The *aggregate* superpixel distance is defined as:

$$d_{aggregate}(i) = \frac{1}{m} \sum_{m\ maps} dist_m(i) \quad \forall i \in I, \tag{9}$$

where $dist_m(i)$ is the average distance score over the superpixel in map m that contains pixel i. Fig. 4 demonstrates how occlusion detection becomes more accurate when superpixels are considered via PR curves.

In Fig. 5 we plot the performance of our occlusion detection algorithm against the number of transformations in our training set. In Table 1 we present a quantitative comparison of our occlusion detection results with [18] and [14] at sequences from Middlebury and UCL Optical Flow datasets in terms of precision and recall statistics. Kolmogorov and Zabih [18] designed an algorithm to detect occlusions in stereo image pairs, and therefore, unsurprisingly, their method can not effectively deal with transformations more complex than the horizontal translations, which commonly appear in these sequences. We

	Venus	RubberWhale	Text1	BrickBox1t1
Recall [18]	0.60	0.23	0.82	0.51
Precision [18]	0.63	0.31	0.68	0.49
Precision [14]	0.69	0.47	0.88	**0.96**
Precision (ours)	**0.75**	**0.81**	**0.91**	0.92

Table 1: Comparison of our occlusion detection algorithm with [18] and [14] on Middlebury and UCL Optical Flow sequences. The comparison is in terms of precision for the *same* recall values.

Baseline algorithm based on differences of patchwise intensity averages

Gated RBM trained on shifts

Gated RBM trained on shifts and rotations

Gated RBM trained on shifts, rotations, affine, scale and illumination variation

Gated RBM trained on shifts, rotations, affine, scale and illumination variation, plus considering superpixel maps

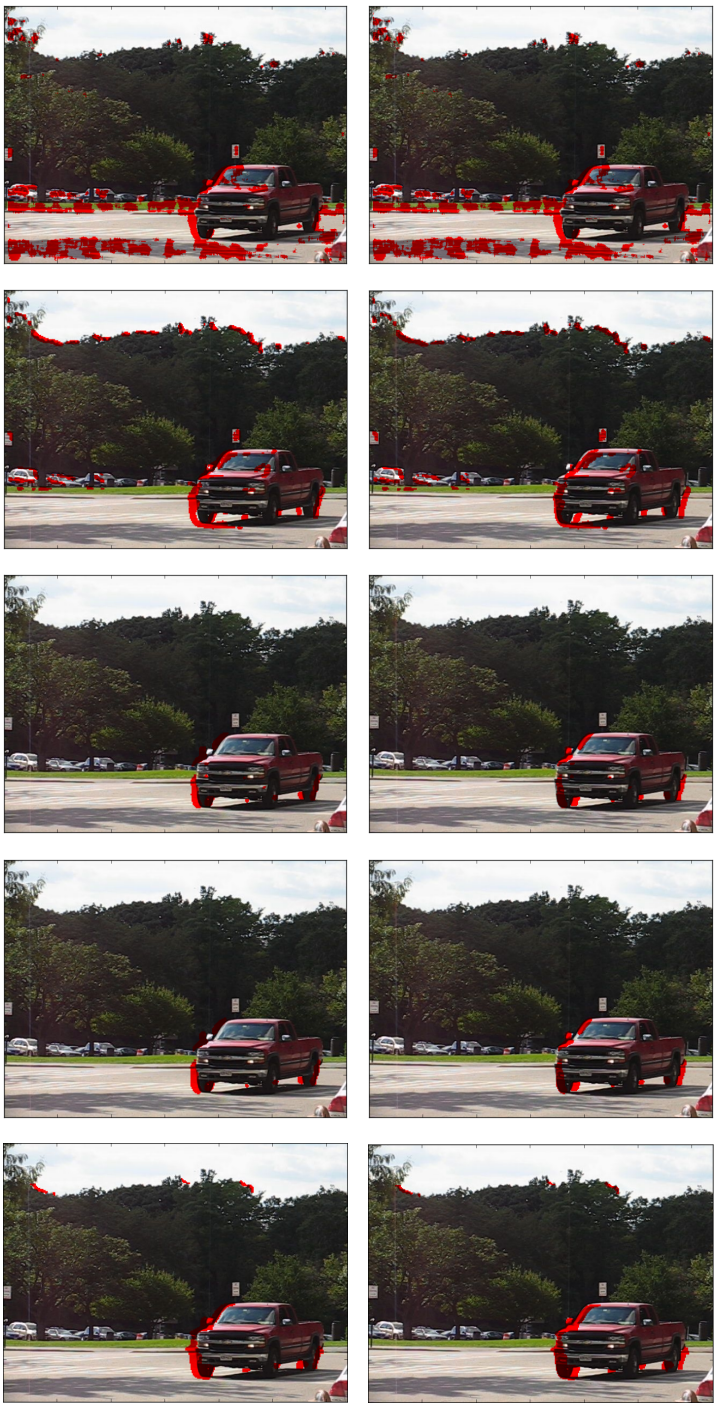

Figure 5: Performance of our occlusion detection algorithm for different training set synthesis and against the baseline algorithm (between frames 7 and 8 of "Cars8" sequence of Berkeley Motion Segmentation dataset).

use it as a baseline algorithm like in Humayun et al. [14]. The latter ones leverage various flow and appearance features and present a competitive accuracy. However, they have many false alarms on edges that are not occlusion boundaries, which originate from Canny edge detector which is one component of their algorithm. This behavior becomes obvious also through the qualitative comparison in Fig. 6.

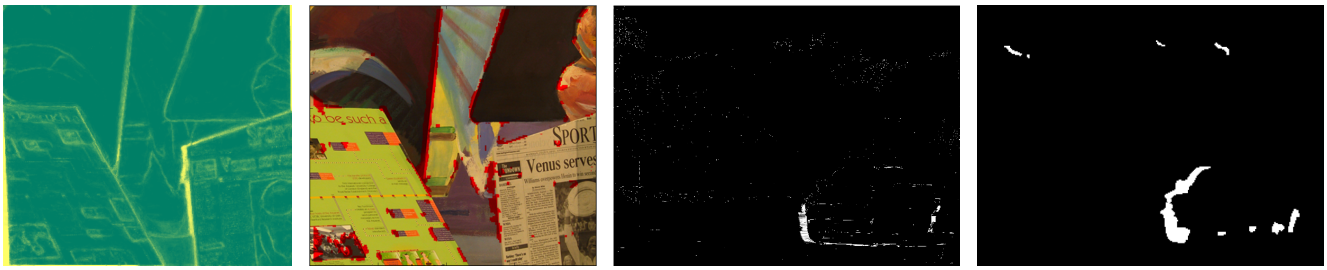

Figure 6: The left image pair compares our method with an algorithm that considers both flow and boundary features [14] on the tough, short-baseline "Venus" sequence from the Middlebury dataset. Our method (right) is able to disregard most edges which are not occlusion boundaries because of affine-invariance; they are visible on different positions of compared patch pairs that include them. However, although superpixels drive the occlusion boundaries, flow features still occasionally display better behavior on boundaries. In the right image pair we compare our algorithm with a state-of-the-art optical flow algorithm [3] on the "Cars8" sequence. Our method gives more accurate occlusion detection, especially on areas with varying illumination, such as the windscreen and the shadow of the car. This should be compared to the ground truth in Fig. 4.

3.2. Image Segmentation

In an effort to further investigate network's capability, we challenge it in a binary classification task with intrinsic class variability, image segmentation from a single frame. The distance function from Eq. 8 is now used as an estimator of dissimilarity between neighboring patches in a single image. The similarity "discontinuities" (i.e., pairs that have a lower similarity score compared to others) is a cue of object boundaries. After thresholding the distance map, the task becomes a binary decision problem. When a pair is dissimilar according to our comparison framework, their common boundary is considered as object boundary. The left image pair in Fig. 7 demonstrates boundary detection examples. A less sensitive threshold results in a finer segmentation. The images are taken from the Make3d Cornell dataset 1 [30].

The model is trained over the same spectrum of transformations that were used in occlusion detection (shifts, rotations, affine, illumination, scale) in the same manner as before. After obtaining a binary map of similar/dissimilar neighboring patch pairs, the Normalized Cuts algorithm [32] is used for the final segmentation, where instead of using boundary, brightness or spatial information in the input matrix W, we use Gated RBM's dissimilarity scores:

$$\forall p(i_1, j_1), p(i_2, j_2) \in \mathcal{P},\ w_{12} = \begin{cases} d_{12}, & \text{if } |i_1 - i_2| = 0, r, 2r, 3r \text{ or } |j_1 - j_2| = 0, r, 2r, 3r \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where $\mathcal{P}$ is an image partition in overlapping patches, r is the patch size ($r = 13$ in these experiments) and $p(i, j)$ is patch centered on pixel (i, j). Our network is nuisance invariant, thereby capable of ignoring shadow and changing illumination effects and detecting similar textons at different scales, positions and angles. The final segmentation is semantically sensible, in view of the fact that mainly object boundaries are detected instead of other edges which are false alarms in this setting.

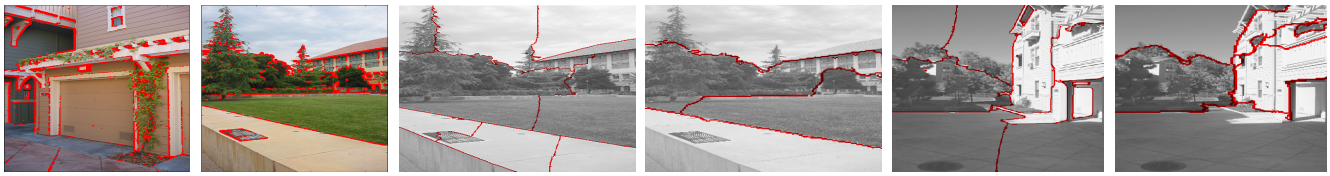

Figure 7: These figures qualitatively demonstrate the concept of "semantic" image segmentation. The left image pair shows boundary detection, based on our patch dissimilarity score. The model ignores tiles or cloud edges, which are not object boundaries. In the middle and right image pairs we compare Normalized Cuts (left) and our method (right). As expected, the final segmentations are similar, but our algorithm successfully disregards any boundary on the front line of the yard wall, as a wall exists on both sides. In the right pair Normalized Cuts give a segmentation that follows the shadows. Our algorithm, being illumination invariant, crosses the shadow while following the building wall.

4. Conclusions

We have empirically tested the hypothesis that a fairly simple learning architecture can satisfactorily manage the nuisance variability in the imaging process. To this end, we have established two binary classification tasks; one with intrinsic variability (in segmentation, patches from the same object present intraclass variability) and one without intrinsic variability (in occlusion detection, the underlying scene is known to be the same). We have shown empirically that our network manages to reduce nuisance variability significantly, thus challenging recent work that suggests that nuisance variability accounts for most of the complexity in imaging data.

Using superpixel information, our framework provides competitive occlusion detection that in many cases outperforms current state-of-the-art algorithms based on optical flow and boundary features. This is reasonable if we realize that our method is at least equivalent with a standard optical flow algorithm, when the latter one is applied in a sufficiently large and diverse set of different representations of the image space. However, hand-crafting features is a more complicated and time-consuming process. The Gated RBM is capable of learning similar features automatically, while we can specialize the setting and the nuisances that we need to deal with per application by providing the appropriate training set.

We should underline the fact that there is no preprocessing matching step. Given two images, the purpose of our framework is to detect occluded regions and ignore false alarms arising from local deformations. Actually some inaccuracy in correspondence that might be caused by motion blur or rolling-shutter phenomena is acceptable and can be "absorbed" by our framework given its invariance in properties such as translations, rotations and scale.

5. Appendix - Mathematical Proofs

5.1. Proof of Eq. 2a (conditional distributions)

Let $\mathbf{x}_{-\mathbf{l}}$ denote the state of all units in layer $\mathbf{x}$ except for the lth one and then define the quantities:

$$\alpha_l(\mathbf{y},\mathbf{h}) := -\sum_{f=1}^{F} u_{lf}(\sum_{j=1}^{J} v_{jf}y_j)(\sum_{k=1}^{K} w_{kf}h_k) - a_l,$$

$$\beta(\mathbf{x_{-l}},\mathbf{y},\mathbf{h}) := -\sum_{f=1}^{F}(\sum_{i=1,i\neq l}^{I} u_{if}x_i)(\sum_{j=1}^{J} v_{jf}y_j)(\sum_{k=1}^{K} w_{kf}h_k) - \sum_{i=1,i\neq l}^{I} a_i x_i - \sum_{j=1}^{J} b_j y_j - \sum_{k=1}^{K} c_k h_k.$$

Given the definition of the energy function in Eq. 1, we have $E(\mathbf{x},\mathbf{y},\mathbf{h}) = \beta(\mathbf{x_{-l}},\mathbf{y},\mathbf{h}) + x_l\alpha_l(\mathbf{y},\mathbf{h})$. Thus:

$$p(X_l=1|\mathbf{y},\mathbf{h}) = p(X_l=1|\mathbf{x_{-l}},\mathbf{y},\mathbf{h}) = \frac{p(X_l=1,\mathbf{x_{-l}},\mathbf{y},\mathbf{h})}{p(\mathbf{x_{-l}},\mathbf{y},\mathbf{h})} = \frac{\frac{1}{Z}e^{-E(X_l=1,\mathbf{x_{-l}},\mathbf{y},\mathbf{h})}}{\frac{1}{Z}e^{-E(X_l=1,\mathbf{x_{-l}},\mathbf{y},\mathbf{h})} + \frac{1}{Z}e^{-E(X_l=0,\mathbf{x_{-l}},\mathbf{y},\mathbf{h})}}$$

$$= \frac{e^{-\beta(\mathbf{x_{-l}},\mathbf{y},\mathbf{h})-\alpha_l(\mathbf{y},\mathbf{h})}}{e^{-\beta(\mathbf{x_{-l}},\mathbf{y},\mathbf{h})-\alpha_l(\mathbf{y},\mathbf{h})} + e^{-\beta(\mathbf{x_{-l}},\mathbf{y},\mathbf{h})}} = \frac{e^{-\beta(\mathbf{x_{-l}},\mathbf{y},\mathbf{h})}\cdot e^{-\alpha_l(\mathbf{y},\mathbf{h})}}{e^{-\beta(\mathbf{x_{-l}},\mathbf{y},\mathbf{h})}\cdot(e^{-\alpha_l(\mathbf{y},\mathbf{h})}+1)}$$

$$= \frac{1}{1+e^{\alpha_l(\mathbf{y},\mathbf{h})}} = \sigma(-\alpha_l(\mathbf{y},\mathbf{h})) = \sigma[\sum_{f=1}^{F} u_{lf}(\sum_{j=1}^{J} v_{jf}y_j)(\sum_{k=1}^{K} w_{kf}h_k) + a_l],$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid activation function.

Given the conditional independence of variables $\mathbf{X}$, the joint conditional distribution is written as:

$$p(\mathbf{x}|\mathbf{y},\mathbf{h}) = \prod_{i=1}^{I} \mathcal{B}(x_i;\sigma[\sum_{f=1}^{F} u_{if}(\sum_{j=1}^{J} v_{jf}y_j)(\sum_{k=1}^{K} w_{kf}h_k) + a_i]),$$

where $\mathcal{B}(x;p)$ is the pdf of a Bernoulli random variable x with parameter p:

$$\mathcal{B}(x;p) = \begin{cases} p, & \text{if } x=1. \\ 1-p, & \text{if } x=0. \end{cases}$$

Similar proofs hold for Eqs. 2b and 2c.

5.2. Proof of Eq. 7 (marginal distribution of the visible variables)

The conditional independence of each layer's units given the other two layers simplifies the calculations:

$$p(\mathbf{x},\mathbf{y}) = \frac{1}{Z}\sum_{\mathbf{h}} e^{-E(\mathbf{x},\mathbf{y},\mathbf{h})} = \frac{1}{Z} e^{\sum_{i=1}^{I} a_i x_i + \sum_{j=1}^{J} b_j y_j} \sum_{h_1} \cdots \sum_{h_K} \prod_{k=1}^{K} e^{h_k\left(c_k + \sum_{f=1}^{F} w_{kf}(\sum_{i=1}^{I} u_{if}x_i)(\sum_{j=1}^{J} v_{jf}y_j)\right)}$$

$$= \frac{1}{Z} e^{\sum_{i=1}^{I} a_i x_i + \sum_{j=1}^{J} b_j y_j} \sum_{h_1} e^{h_1\left(c_1 + \sum_{f=1}^{F} w_{1f}(\sum_{i=1}^{I} u_{if}x_i)(\sum_{j=1}^{J} v_{jf}y_j)\right)} \cdots \sum_{h_K} e^{h_K\left(c_K + \sum_{f=1}^{F} w_{Kf}(\sum_{i=1}^{I} u_{if}x_i)(\sum_{j=1}^{J} v_{jf}y_j)\right)}$$

$$= \frac{1}{Z} e^{\sum_{i=1}^{I} a_i x_i} e^{\sum_{j=1}^{J} b_j y_j} \prod_{k=1}^{K} \sum_{h_k} e^{h_k\left(c_k + \sum_{f=1}^{F} w_{kf}(\sum_{i=1}^{I} u_{if}x_i)(\sum_{j=1}^{J} v_{jf}y_j)\right)}$$

$$= \frac{1}{Z} \prod_{i=1}^{I} e^{a_i x_i} \prod_{j=1}^{J} e^{b_j y_j} \prod_{k=1}^{K} \left(1 + e^{c_k + \sum_{f=1}^{F} w_{kf}(\sum_{i=1}^{I} u_{if}x_i)(\sum_{j=1}^{J} v_{jf}y_j)}\right).$$

Then the log-likelihood is calculated as:

$$log\ p(\mathbf{x},\mathbf{y}) = -logZ + log \prod_{i=1}^{I} e^{a_i x_i} + log \prod_{j=1}^{J} e^{b_j y_j} + log \prod_{k=1}^{K} \left(1 + e^{c_k + \sum_{f=1}^{F} w_{kf}(\sum_{i=1}^{I} u_{if}x_i)(\sum_{j=1}^{J} v_{jf}y_j)}\right)$$

$$= -logZ + \sum_{i=1}^{I} a_i x_i + \sum_{j=1}^{J} b_j y_j + \sum_{k=1}^{K} log\left(1 + e^{c_k + \sum_{f=1}^{F} w_{kf}(\sum_{i=1}^{I} u_{if}x_i)(\sum_{j=1}^{J} v_{jf}y_j)}\right).$$

References

[1] O. M. Aodha, A. Humayun, M. Pollefeys, and G. Brostow. "Learning a confidence measure for optical flow", *TPAMI*, 2013. 2, 6

[2] N. Apostoloff and A. W. Fitzgibbon. "Learning Spatiotemporal T-Junctions for Occlusion Detection", *CVPR*, 2005. 2

[3] A. Ayvaci, M. Raptis, and S. Soatto. "Sparse Occlusion Detection with Optical Flow", *IJCV*, 2012. 2, 7, 9

[4] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. "A database and evaluation methodology for optical flow", *IJCV*, 2011. 2, 6

[5] Y. Bengio, A. C. Courville, and P. Vincent. "Representation Learning: A Review and New Perspectives", *TPAMI*, 2013.

[6] T. Brox and J. Malik. "Object segmentation by long term analysis of point trajectories", *ECCV*, 2010. 6

[7] A. Fischer and C. Igel. "An Introduction to Restricted Boltzmann Machines", *CIARP*, 2012.

[8] R. Fransens, C. Strecha, and L. J. V. Gool. "A Mean Field EM-algorithm for Coherent Occlusion Handling in MAP-Estimation Prob", *CVPR*, 2006.

[9] B. J. Frey, N. Jojic, and A. Kannan. "Learning appearance and transparency manifolds of occluded objects in layers", *CVPR*, 2003. 1

[10] D. B. Grimes and R. P. N. Rao. "Bilinear Sparse Coding for Invariant Vision", *Journal of Neural Computation*, 2005.

[11] X. He and A. L. Yuille. "Occlusion Boundary Detection Using Pseudo-depth", *ECCV*, 2010. 2

[12] G. E. Hinton. "Training Products of Experts by Minimizing Contrastive Divergence", *Journal of Neural Computation*, 2002. 5

[13] D. Hoiem, A. A. Efros, and M. Hebert. "Recovering Occlusion Boundaries from an Image", *IJCV*, 2010. 2

[14] A. Humayun, O. M. Aodha, and G. J. Brostow. "Learning to find occlusion regions", *CVPR*, 2011. 2, 7, 8, 9

[15] N. Jacobson, Y. Freund, and T. Q. Nguyen. "An Online Learning Approach to Occlusion Boundary Detection", *TIP*, 2012. 2

[16] P. M. Jodoin, C. Rosenberger, and M. Mignotte. "Detecting half-occlusion with a fast region-based fusion procedure", *BMVC*, 2006.

[17] K. Kavukcuoglu, M. A. Ranzato, R. Fergus, and Y. LeCun. "Learning invariant features through topographic filter maps", *CVPR*, 2009. 1

[18] V. Kolmogorov and R. Zabih. "Computing Visual Correspondence with Occlusions via Graph Cuts", *ICCV*, 2001. 2, 7

[19] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. "Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis", *CVPR*, 2011. 1

[20] K. P. Lim, A. Das, and M. N. Chong. "Estimation of Occlusion and Dense Motion Fields in a Bidirectional Bayesian Framework", *TPAMI*, 2002. 2

[21] E. Lobaton, R. Vasudevan, R. Bajcsy, and R. Alterovitz. "Local occlusion detection under deformations using topological invariants", *ECCV*, 2010. 2

[22] D. R. Martin, C. Fowlkes, and J. Malik. "Learning to Detect Natural Image Boundaries Using Local Brightness, Color, and Texture Cues", *TPAMI*, 2004. 2, 6

[23] R. Memisevic and G. E. Hinton. "Learning to Represent Spatial Transformations with Factored Higher-Order Boltzmann Machines", *Neural Computation*, 2010. 1, 2

[24] G. Mori. "Guiding Model Search Using Segmentation", *ICCV*, 2005. 1, 6

[25] M. Piccardi. "Background subtraction techniques: a review", *SMC*, 2004. 2

[26] T. Poggio. "The computational magic of the ventral stream", *Nature Precedings*, 2011. 1

[27] M. A. Ranzato, A. Krizhevsky, and G. E. Hinton. "Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images", *AISTATS*, 2010. 1

[28] R. Salakhutdinov, A. Mnih, and G. E. Hinton. "Restricted Boltzmann machines for collaborative filtering", *ICML*, 2007.

[29] M. E. Sargin, L. Bertelli, B. S. Manjunath, and K. Rose. "Probabilistic occlusion boundary detection on spatio-temporal lattices", *ICCV*, 2009. 2

[30] A. Saxena, M. Sun, and A. Y. Ng. "Make3d: Learning 3d scene structure from a single still image", *PAMI*, 2009. 2, 9

[31] D. Scharstein and R. Szeliski. "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms", *IJCV*, 2002. 2

[32] J. Shi and J. Malik. "Normalized Cuts and Image Segmentation", *TPAMI*, 2000. 1, 9

[33] S. Soatto. "Actionable information in vision", *ICCV*, 2009. 1

[34] A. N. Stein and M. Hebert. "Occlusion Boundaries from Motion: Low-Level Detection and Mid-Level Reasoning", *IJCV*, 2009. 2

[35] G. Sundaramoorthi, P. Petersen, V. S. Varadarajan, and S. Soatto. "On the set of images modulo viewpoint and contrast changes", *CVPR*, 2009. 1

[36] P. Sundberg, T. Brox, M. Maire, P. Arbelaez, and J. Malik. "Occlusion Boundary Detection and Figure/Ground Assignment from Optical Flow", *CVPR*, 2011. 2

[37] J. Susskind, G. E. Hinton, R. Memisevic, and M. Pollefeys. "Modeling the joint density of two images under a variety of transformations", *CVPR*, 2011. 1, 5

[38] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler. "Convolutional Learning of Spatio-temporal Features", *ECCV*, 2010. 1

[39] Y. W. Teh and G. E. Hinton. "Rate-coded Restricted Boltzmann Machines for Face Recognition", *NIPS*, 2000. 5

[40] V. Vonikakis, D. Chrysostomou, R. Kouskouridas, and A. Gasteratos. "A biologically inspired scale-space for illumination invariant feature selection", *Measurement Science and Technology*, 2013.

[41] A. L. Yuille. "The Convergence of Contrastive Divergences", *NIPS*, 2004.

[42] S. C. Zhu, K. Shi, and Z. Si. "Learning explicit and implicit visual manifolds by information projection", *Pattern Recognition Letters*, 2010.