On spatio-temporal sparse coding: Analysis and an algorithm

Roland Memisevic
Department of Computer Science
University of Frankfurt
ro@cs.uni-frankfurt.de

Abstract

Sparse coding is a common approach to learning local features for object recognition. Recently, there has been an increasing interest in learning features from spatio-temporal, binocular, or other multi-observation data, where the goal is to encode the relationship between images rather than the content of a single image. We discuss the role of multiplicative interactions and of squaring non-linearities in learning such relations. In particular, we show that training a sparse coding model whose filter responses are squared amounts to jointly diagonalizing a set of image transformations. Inference amounts to detecting rotations in the shared eigenspaces. Our analysis helps explain recent experimental results showing that Fourier features and circular Fourier features emerge when training complex cell models on translating or rotating images. And it suggests that it will be crucial to include either squaring or cross-products into deep learning architectures if we want to extend their applicability beyond simple tasks like recognizing objects in a single image.

1 Introduction

Feature learning (AKA dictionary learning, or sparse coding) has gained considerable attention recently, mainly because it can yield image representations that are useful for recognition. Although recognition is an important task in computer vision, many vision problems go beyond single images and involve the relationship between images, examples being tracking, stereopsis, motion and action understanding, or recognition in the presence of large transformations. This raises the question under what conditions sparse coding may be used to encode relationships between images rather than the content of images.

A variety of sparse coding extensions to model motion have been suggested recently (for example, [7, 17, 15, 11, 21]). A common feature of all these methods is that they deploy either (a) multiplicative interactions between features, or (b) a square pooling architecture in which codes are computed from data by squaring, and then summing over, filter responses. Computing the sum over squared filter responses is equivalent to the well-known energy model of *complex cells* [1, 16]. In this paper, we analyse the role of multiplicative interactions and of square pooling in representing the relationship between images. In particular, we show that hidden variables in these models learn to encode image transformations by detecting rotation angles in the eigenspaces of the transformations. Our analysis shows that *learning complex cell models amounts to performing multiple simultaneous diagonalizations of a set of transformations.* A central aspect of our analysis, which is in contrast to previous work on energy models and of complex cells [18, 5], is that it is based on representing transformations as linear transformations in "*pixel space*", in other words as *linear warps*, rather than as a spatial (geometrical) transformation in image coordinates. This allows us to capture both linear and non-linear spatial transformations, such as permutations or local shifts, using the same analytical framework [13].

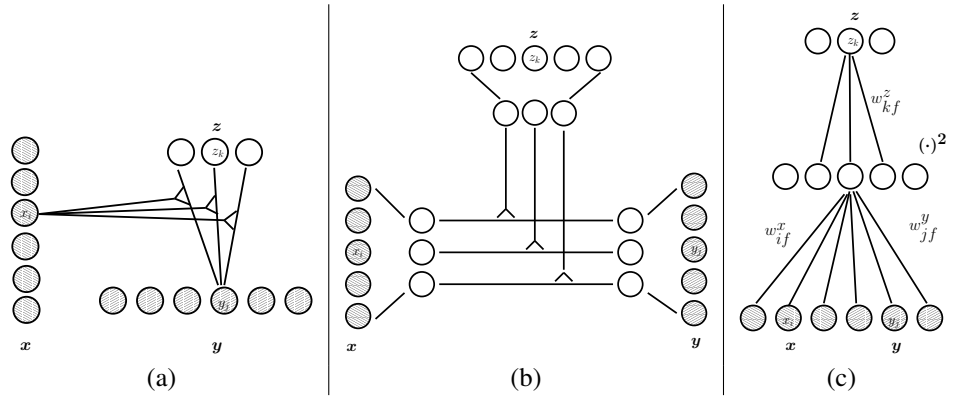


Figure 1: Three ways to model an image pair $(\boldsymbol{x}, \boldsymbol{y})$: Using (a) a cross-correlation model, (b) a factorized cross-correlation model, (c) an energy model applied to the *concatenation* of the images.

2 Background on gated sparse coding and square pooling models

A variety of methods have been suggested to learn transformations, or relationships between images. These can be roughly categorized into three groups, which we shall refer to as *gated sparse coding*, *factored gated sparse coding*, and *energy models*. We shall describe each group in turn in the following.

[7, 17, 14] suggest learning relationships between images by letting a vector of hidden variables $\boldsymbol{z}$ encode the *pair-wise products* $x_i y_j$ between pixels x_i and y_j of two images $\boldsymbol{x}$ and $\boldsymbol{y}$. As this amounts to correlating the images, hidden variables can be thought of as encoding transformations independently of the content of the images [14]. Squared reconstruction error is obviously not the right error function for the products $x_i y_j$, so [7, 17, 14] suggest training the model as a predictive model of $\boldsymbol{y}$, given $\boldsymbol{x}$, which amounts to minimizing an approximation of the reconstruction error:

$$\sum_j \big(y_j^\alpha - \sum_{ik} w_{ijk} x_i^\alpha z_k^\alpha\big)^2 + \lambda \sum_\alpha S(\boldsymbol{z}^\alpha) \tag{1}$$

where $S(\cdot)$ is some sparsifying penalty. One can think of Eq. 1 as a standard-sparse coding model whose parameters $w_{ij} := \sum_k w_{ijk} x_i$ are input-modulated and case-dependent [14]. In principle, any sparse coding method can be turned into a relational sparse coding method, a recent example being sparse auto-encoders [12]. Due to the similarity between the reconstruction cost and that of a standard sparse coding model training is similar [14, 17]. An illustration of a gated sparse coding model is shown in Figure 1 (a). The model is closely related to early cross-correlation models of binocular disparity [2].

Inference amounts to computing the hidden variable activities, given both images $\boldsymbol{x}$ and $\boldsymbol{y}$ which, by utilizing the case-dependent parameter-gating ($w_{ij} := \sum_k w_{ijk} x_i$), amounts to evaluating a quadratic form in $\boldsymbol{x}$ and $\boldsymbol{y}$ [14, 12, 13]:

$$z_k = \sum_j w_{jk} y_j = \sum_j \big(\sum_i w_{ijk} x_i\big) y_j = \sum_{ij} w_{ijk} x_i y_j \tag{2}$$

It is common to furthermore apply an elementwise sigmoid non-linearity [14, 12]. Note that Eq. 2 is equal to inference in a standard sparse coding model with inputs being all products $x_i y_j$ of pixels across the two images. In practice, it is common to include bias terms in Eqs. 1 and 2 but we shall refrain from doing so to avoid cluttering up the derivations. We shall instead think of data and hidden variables as being in "homogeneous notation" with an extra, constant 1-dimension.

2.1 Factored gated sparse coding

In a predictive model, the number of parameters is equal to the product of the dimensionalities of inputs, outputs and hiddens. [15] suggest reducing that number by factorizing the parameter tensor

W into three matrices, such that each component w_{ijk} is given by the "three-way inner product"

$$w_{ijk} = \sum_{ijk} \sum_{f=1}^{F} w_{if}^x w_{jf}^y w_{kf}^z \tag{3}$$

Here, F is a number of hidden "factors", which, like the number K of hidden units, has to be chosen by hand or by cross-validation. The matrices w^x, w^y and w^z are $I \times F$, $J \times F$ and $K \times F$, respectively. Under this factorization, inference can be written (cf. Eq. 2 and Figure 1):

$$z_k = \sum_{ij} w_{ijk} x_i y_j = \sum_{ij} (\sum_f w_{if}^x w_{jf}^y w_{kf}^z) x_i y_j = \sum_f w_{kf}^z \cdot \boldsymbol{w}_{\cdot f}^{x\,\mathrm{T}} \boldsymbol{x} \cdot \boldsymbol{w}_{\cdot f}^{y\,\mathrm{T}} \boldsymbol{y} \tag{4}$$

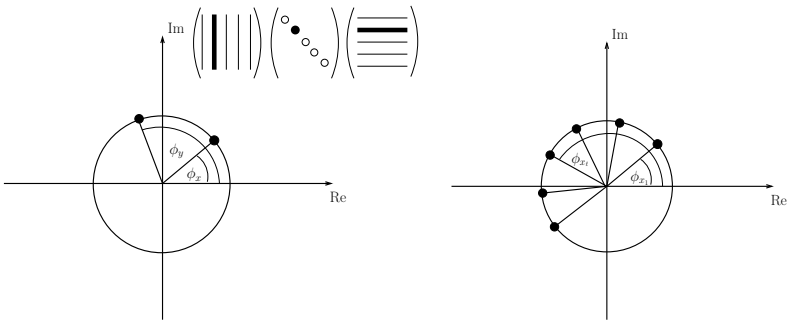
possibly followed by an elementwise non-linearity. One obtains similar expressions for y_j and for the energy in a gated Boltzmann machine [15]. Eq. 4 shows that factorization can be viewed as *filter matching*: During inference, $\boldsymbol{x}$ and $\boldsymbol{y}$ are projected onto linear basis functions which are subsequently multiplied and pooled over to compute the activity of a hidden unit z_k. It is important to note that the way factorization reduces parameters is *not* by projecting data onto a lower-dimensional space before computing the multiplicative interactions — a claim that can be found frequently in the literature. In fact, it is common to choose F to be *larger* than I or J. The way that factorization reduces the number of parameters is by restricting the *three-way connectivity*. Eq. 3, for example, amounts to allowing each image projection to engage only in *a single* multiplicative interaction. Learning amounts to finding basis functions that can deal with this restriction optimally. In principle, all gated sparse coding models can be subjected to this factorization. Training is similar to training an unfactored model by using the chain rule and differentiating Eq. 3. Figure 1 (b) shows an illustration of the model.

2.2 Energy models

Energy models [1, 16] are different, but closely related, approaches to modeling image motion and disparities, and they have been deployed monocularly, too. A main application of energy models has been the detection of small translational motion in image pairs. This makes them suitable as biologically plausible mechanisms of both local motion estimation and binocular disparity estimation. Energy models detect motion by projecting two images onto two phase-shifted Gabor functions each (for a total of four basis function responses). The two responses *across* the images are added and squared. The sum of these two squared, spatio-temporal responses then yields the response of the energy model. The rationale behind the energy model is that, since each within-image Gabor filter pair can be thought of as a localized spatio-temporal Fourier component, the sum of the squared components yields an estimate of spectral energy, which is not dependent of the *phase* — and thus to a large degree not dependent on the *content* — of the input images. The two filters within each image need to be sine/cosine pairs, which is commonly referred to as being "in quadrature". A local shift detector can be built by using a set of energy models tuned to different frequencies. To turn the set of energy responses into an estimate of local translation, one can, for example, pick the model with the strongest response [19, 18], or use pooling to get a more stable estimate [5]. Both energy models and cross-correlation models are standard models for the simple–complex cell dichotomy in visual cortex (for example, [5]).

[10, 9, 22, 3] suggest *learning* energy models from data by extending a sparse coding model with an element-wise squaring operation, followed by a linear pooling layer. In contrast to the original energy model, one may use more than exactly two filters to pool over, and pooling weights may be learned along with basis functions, instead of being fixed to be 1. For training, [9] suggest adopting ICA, and to force the responses of latent variables, which are now sums of squared basis function responses to be sparse, while keeping the filters orthogonal to avoid degenerate solutions. This approach is known as "Independent Subspace Analysis" (ISA) [9]. Since they compute the sum over squares of features, energy models are also referred to as "square pooling" models [20].

Both ISA and factored gated Boltzmann machines were recently shown to yield state-of-the-art performance in various motion recognition tasks [11, 21]. We shall refer to the hidden layer nodes as "factors" in analogy to the hidden layer of a factored gated Boltzmann machine [15]. When applied to the concatenation of *two* images, ISA is closely related to gated sparse coding. This becomes obvious when considering inputs given by the *concatenation of two images* $\boldsymbol{x}$ and $\boldsymbol{y}$ (cf.



(a) (b)

Figure 2: (a) Rotation in an invariant subspace: In a multiplicative sparse coding model, a hidden unit can detect the rotation angle $\phi_y - \phi_x$ between projections of two images $\boldsymbol{x}$ and $\boldsymbol{y}$ if eigenvalues are absorbed into the eigenvectors. (b) A hidden unit that is connected to more than two eigenvectors will detect the repeated application of the same transformation.

Figure 1 (c)): In this case, the weights that connect a first-layer hidden unit f with the inputs can be separated into those that affect $\boldsymbol{x}$ and those that affect $\boldsymbol{y}$. If we denote the first by $w^x_{\cdot f}$ and the latter by $w^y_{\cdot f}$, we can write the activity of top-level hidden unit z_k in the energy model as

$$z_k = \sum_f w^z_{kf} \left({\boldsymbol{w}^x_{\cdot f}}^{\mathrm{T}} \boldsymbol{x} + {\boldsymbol{w}^y_{\cdot f}}^{\mathrm{T}} \boldsymbol{y} \right)^2 \tag{5}$$

$$= \sum_f w^z_{kf} \left(2({\boldsymbol{w}^x_{\cdot f}}^{\mathrm{T}} \boldsymbol{x})({\boldsymbol{w}^y_{\cdot f}}^{\mathrm{T}} \boldsymbol{y}) + ({\boldsymbol{w}^x_{\cdot f}}^{\mathrm{T}} \boldsymbol{x})^2 + ({\boldsymbol{w}^y_{\cdot f}}^{\mathrm{T}} \boldsymbol{y})^2 \right) \tag{6}$$

This shows that up to the quadratic terms $({\boldsymbol{w}^x_{\cdot f}}^{\mathrm{T}} \boldsymbol{x})^2$ and $({\boldsymbol{w}^y_{\cdot f}}^{\mathrm{T}} \boldsymbol{y})^2$, inference in an energy model that is applied to concatenation of inputs $\boldsymbol{x}$ and $\boldsymbol{y}$ is exactly the same as in a gated sparse coding model. We shall discuss the role of the squared terms in the next section. Figure 1 shows an illustration of an energy model applied to an image pair.

3 Complex cells and joint eigenspaces

Many interesting transformations, like translation, rotation or any permutation ("shuffling pixels"), can be expressed approximately as an orthogonal transformation L in pixel space, in other words, as an *orthogonal warp*. A matrix L is orthogonal, per definition, if $L^{\mathrm{T}}L = LL^{\mathrm{T}} = I$, where I is the identity matrix, in other words if $L^{-1} = L^{\mathrm{T}}$.

The importance of orthogonal matrices for learning transformations follows from the fact that the eigen-decomposition $L = UDU^{\mathrm{T}}$ of an orthogonal matrix L is complex, with eigenvalues (diagonal of D) that have absolute value 1 [8]. Multiplying by a complex number with absolute value 1 amounts to performing a rotation in the complex plane [8], as illustrated in Figure 2 (a). Applying an orthogonal warp to an image is therefore equivalent to (i) projecting the image onto *filter pairs* (the real and imaginary parts of each eigenvector), (ii) performing a rotation within each eigenspace, known also as an *invariant subspace* of the transformation, and (iii) projecting back into the image-space. In other words, we can decompose an orthogonal transformation into a set of independent, 2-dimensional rotations. The most well-known examples are translations: A translation matrix contains ones along one of its secondary diagonals, and it is zero elsewhere[1]. The eigenvectors of this matrix are Fourier-components [6], and the rotation in each invariant subspace amounts to a phase-shift of the corresponding Fourier-feature. This leaves the norm of the projections onto the Fourier-components (the power spectrum of the signal) constant, which is a well known property of translation.

It is interesting to note that the imaginary and real parts of the eigenvectors of a translation matrix correspond to sine and cosine features, respectively, reflecting the fact that Fourier components

[1]To be exactly orthogonal it has to contain an additional one in another place, so that it performs a rotation with wrap-around.

naturally come in *pairs*. These are commonly referred to as *quadrature pairs* in the literature. In the special case of Gabor features, the importance of quadrature pairs is that they allow us to detect translations independently of the local content of images [18, 5]. Importantly, however, the property that eigenvectors come in *pairs* is not specific to translations. It is shared by all transformations that can be represented by an orthogonal matrix, so they can be composed from 2-dimensional rotations. [4] therefore introduced the term *generalized quadrature pair* to refer to the eigen-features of these transformations[2].

3.1 Commuting warps share eigenspaces

An observation that is central to the following analysis is that eigenspaces can be *shared* among transformations. When eigenspaces are shared, then the only way in which the translations differ, is in the angles of rotation within the eigenspaces. So shared eigenspaces allow us to represent *multiple transformations with a single set of features*. An example of a shared eigenspace is the Fourier-basis, which is shared among translations. This well-known observation follows from the fact that the set of all circulant matrices (which are 1-D translation-matrices) of the same size share the Fourier-basis as the eigen-basis [6]. Eigenspaces can be shared between many more transformation not just translation. An obvious generalization are local translations (cf. Figure 3), which may be considered the constituting transformations of natural videos. Another, less obvious generalization is spatial rotation. Formally, a set of matrices share eigenvectors if they *commute* [8]. This can be seen by considering any two matrices A and B with $AB = BA$ and with λ, v an eigenvalue/eigenvector pair of B with multiplicity one. It holds that $BAv = ABv = \lambda Av$ Therefore, Av is also an eigenvector of B with the same eigenvalue.

The importance of commuting transformations for our analysis is that, since these transformations share an eigen-basis, they differ only in the angle of rotation in the joint eigenspace. As a result, one may *extract* a particular transformation from a given image pair $(\boldsymbol{x}, \boldsymbol{y})$ by recovering the angles of rotation between the projections of $\boldsymbol{x}$ and $\boldsymbol{y}$ onto the eigenspaces. For this end, consider the real and complex parts $\boldsymbol{v}_{\mathrm{R}}$ and $\boldsymbol{v}_{\mathrm{I}}$ of some eigen-feature $\boldsymbol{v}$. That is, $\boldsymbol{v} = \boldsymbol{v}_{\mathrm{R}} + i\boldsymbol{v}_{\mathrm{I}}$, where $i = \sqrt{-1}$. The real and imaginary coordinates of the projection of $\boldsymbol{x}$ onto the invariant subspace associated with $\boldsymbol{v}$ are given by $\boldsymbol{v}_{\mathrm{R}}^{\mathrm{T}}\boldsymbol{x}$ and $\boldsymbol{v}_{\mathrm{I}}^{\mathrm{T}}\boldsymbol{x}$, respectively. For the output image, they are $\boldsymbol{v}_{\mathrm{R}}^{\mathrm{T}}\boldsymbol{y}$ and $\boldsymbol{v}_{\mathrm{I}}^{\mathrm{T}}\boldsymbol{y}$.

Call ϕ_x and ϕ_y the angles of the projections of $\boldsymbol{x}$ and $\boldsymbol{y}$ with the real axis in the complex plane. If we normalize the projections to have unit norm, then the cosine of the angle between the projections, $\phi_y - \phi_x$, may be written $\cos(\phi_y - \phi_x) = \cos\phi_y \cos\phi_x + \sin\phi_y \sin\phi_x$ by a trigonometric identity. This is equivalent to computing the inner product between the two normalized projections (cf. Figure 2 (a)). In other words, to estimate the (cosine of) the angle of rotation between the projections of $\boldsymbol{x}$ and $\boldsymbol{y}$, we need to *sum over the product of two filter responses*.

However, normalizing each projection to 1 amounts to dividing by the sum of squared filter responses, an operation that is highly unstable if a projection is close to zero. Unfortunately, this will be the case, whenever one of the images is almost orthogonal to the invariant subspace. This, however, means that the rotation angle *cannot be recovered from the given image*, because the image is too close to the axis of rotation. One may view this as a subspace-generalization of the well-known aperture problem beyond translation, to the set of commuting transformations. Normalization would ignore this problem and provide the illusion of a recovered angle even when the aperture problem makes the detection of the transformation component impossible. In the next section we discuss how one may overcome this problem by rephrasing the problem as a *detection* task.

3.2 Detecting subspace rotations

It is possible to deal with the subspace aperture problem if we *absorb* the rotations into the eigenvectors: For each eigenvector $\boldsymbol{v}$, consider a set of rotation angles θ, and define the complex output image filters

$$\boldsymbol{v}^{\theta} = \exp(i\theta)\boldsymbol{v} \tag{7}$$

Each output filter $\boldsymbol{v}^{\theta}$ represents a projection and simultaneous rotation by θ. Furthermore, for each $\boldsymbol{v}^{\theta}$, we use as the corresponding input filter a copy of the unchanged eigenvector $\boldsymbol{v}$. (Alternatively,

[2] Strictly speaking, [4] considered the eigen-features of the anti-symmetric part of an orthogonal matrix.

one may absorb the rotation into the input filters instead of into the output filters, or parts of the rotation into both).

A subspace rotation-detector with a preferred angle θ can now be defined as

$$r^\theta = (\boldsymbol{v}_R{}^{\mathrm{T}}\boldsymbol{y})(\boldsymbol{v}_R^\theta{}^{\mathrm{T}}\boldsymbol{x}) + (\boldsymbol{v}_I{}^{\mathrm{T}}\boldsymbol{y})(\boldsymbol{v}_I^\theta{}^{\mathrm{T}}\boldsymbol{x}), \tag{8}$$

where subscripts R and I denote the real and imaginary part of the filters like before. As before, if projections would be normalized to length 1, we would have

$$r^\theta = \cos\phi_y \cos(\phi_x - \theta) + \sin\phi_y \sin(\phi_x - \theta) = \cos(\phi_y - \phi_x - \theta), \tag{9}$$

which would be maximal whenever $\phi_y - \phi_x = \theta$, thus when the observed angle of rotation, $\phi_y - \phi_x$, is equal to the preferred angle of rotation, θ. However, like before, normalizing projections is not a good idea in practice because of the subspace aperture problem.

If features and data are contrast normalized *before* projecting onto the invariant subspace, then projections will depend only on how well the image pair represents a given subspace rotation. The value r^θ, in turn, will then depend on (a) the transformation (via the subspace angle) and (b) the content of the images (via the angles between each image and the invariant subspace). As a result the following holds: **The peak response of r^θ will be attained for image pairs that are transformed according to the detector's preferred angle, *and* that reside exactly within the invariant subspace.** Thus the output of the detector factors in both, the presence of a transformation and the ability to discern it.

Learning filter-pairs amounts to extracting the eigenwarps of a class of orthogonal image transformations. One can apply a sigmoid non-linearity after computing r^θ, in which case the output of a hidden variable can be interpreted as a probability. For learning, it has been common to project into subspaces of dimension larger than two (for example, [15], [12]). Since learning amounts to adapting both filters and pooling weights, this may make it possible to share filters among hidden variables and to facilitate a trade-off between model capacity and angular resolution when using a small number of filters or of hidden variables. Furthermore, it may help deal with edge effects and noise, as well as with the fact that in real data, the matrix $\boldsymbol{x}\boldsymbol{y}^{\mathrm{T}}$ encoding the transformation will never be exactly orthogonal.

According to the aforegoing analysis, one has to contrast normalize images in order recognize transformations. Contrast normalization also allows us to replace a cross-correlation by an energy model. By concatenating images $\boldsymbol{x}$ and $\boldsymbol{y}$, as well as filters $\boldsymbol{v}$ and $\boldsymbol{v}^\theta$, we can approximate the subspace rotation detector (Eq. 8) with the response of an energy detector as follows;

$$\begin{aligned} r^\theta &= \big((\boldsymbol{v}_R{}^{\mathrm{T}}\boldsymbol{y}) + (\boldsymbol{v}_R^\theta{}^{\mathrm{T}}\boldsymbol{x})\big)^2 + \big((\boldsymbol{v}_I{}^{\mathrm{T}}\boldsymbol{y}) + (\boldsymbol{v}_I^\theta{}^{\mathrm{T}}\boldsymbol{x})\big)^2 \\ &= 2\big((\boldsymbol{v}_R{}^{\mathrm{T}}\boldsymbol{y})(\boldsymbol{v}_R^\theta{}^{\mathrm{T}}\boldsymbol{x}) + (\boldsymbol{v}_I{}^{\mathrm{T}}\boldsymbol{y})(\boldsymbol{v}_I^\theta{}^{\mathrm{T}}\boldsymbol{x})\big) + (\boldsymbol{v}_R{}^{\mathrm{T}}\boldsymbol{y})^2 + (\boldsymbol{v}_R^\theta{}^{\mathrm{T}}\boldsymbol{x})^2 + (\boldsymbol{v}_I{}^{\mathrm{T}}\boldsymbol{y})^2 + (\boldsymbol{v}_I^\theta{}^{\mathrm{T}}\boldsymbol{x})^2 \end{aligned} \tag{10}$$

Eq. 10 is equivalent to Eq. 8 up to the four quadratic terms. The four quadratic terms are equal to the sum of the squared norms of the projections of $\boldsymbol{x}$ and $\boldsymbol{y}$ onto the invariant subspace. Thus, like the norm of the projections, they contribute information about the discernibility of transformations. This makes the energy response less conservative than the cross-correlation response (Eq. 8). But the peak response is still attained only when both images reside within the detector's invariant subspace and when their projections are rotated by the detectors preferred angle θ.

3.3 Learning as multiple simultaneous diagonalization

Since computing the sum of squared filter responses amounts to detecting rotation angles in invariant subspaces, *learning* an energy or cross-correlation model amounts to *identifying the subspaces*. Training a sparse coding model with complex cells can therefore be thought of as performing **multiple simultaneous diagonalizations of a set of related cross-correlation matrices** L. When a dataset contains more than one transformation class, learning involves partitioning the set of orthogonal warps into commutative subsets and simultaneously diagonalizing each subset. For exactly diagonal matrices, the subsets are commutative conjugacy classes of the group of orthogonal matrices.

Diagonalizing a single matrix is like performing canonical correlations analysis (CCA) (for example, [4]), so learning a gated sparse coding model may be thought of as performing multiple, related

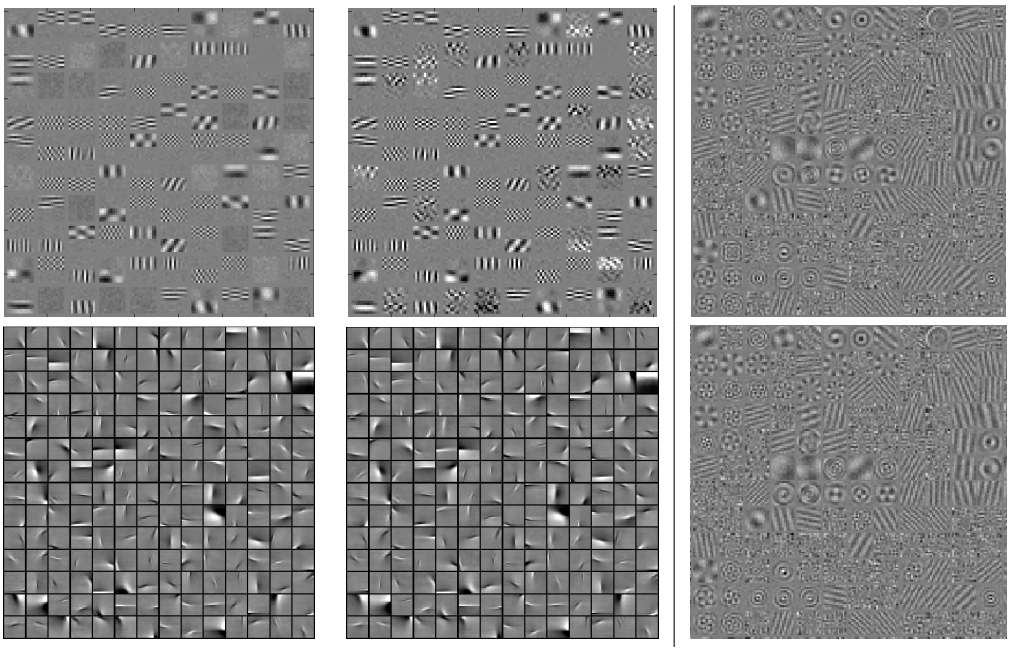


Figure 3: Input/output-filters learned from transforming image pairs (split-screen shifts top-left; natural movies bottom-left; mix of rotations and translations: right). In each pair, filters that are in the same position have their responses multiplied.

canonical correlation analyses. In the same way that auto-encoder networks perform PCA, and thus an eigen-decomposition only up to a linear transformation, the result of training an energy or cross-correlation model is also a set of diagonalizations up to linear transformations.

The close relation between energy models and gated sparse coding makes it possible to implement one via the other. In particular, we can implement an energy model using cross-correlation, by simply applying the latter to an *identical* pair $(\boldsymbol{x}, \boldsymbol{x})$. Likewise, we can implement a cross-correlation model via an energy model by passing it the *concatenation* $(\boldsymbol{x}, \boldsymbol{y})$ as input. We shall discuss an example of this approach in Section 4.

Figure 3 shows sets of input/output filter pairs learned from "split-screen shifts" of random dots where the top half of the image translates independently of the bottom half of the image (top left), natural movies, (bottom-left; same data-set as in [12]) a mixed dataset, consisting of random *rotations* (50%) and random *translations* (50%) of random dot images. Note, that for the mixed data-set, both, the set of rotations and the set of translations are sets of commuting warps (up to edge effects), but rotations do *not* commute with translations and vice versa. The figure shows that the model has learned to separate out the two types of transformation by devoting a subset of filters to encoding rotations and a subset to modeling translations. Training was performed using a gated autoencoder [12] (bottom left and right) and a factored gated Boltzmann machine [15] (top left).

4 Learning "eigenmovies"

Both energy models and cross-correlation models can be applied to more than two images. For gated sparse coding, Eq. 8 could be modified to contain all cross-terms, or all the ones that are deemed relevant (for example, adjacent frames, which would amount to a "Markov"-type gating model of a video). Alternatively, for the energy mechanism, one can compute the square of the concatenation of more than two images in plane of Eq. 10. A multi-frame rotation extractor (cf. Section 3.1) will yield the response

$$r = \big(\sum_s \boldsymbol{v}_R^{s\,\mathrm{T}} \boldsymbol{x}_s\big)^2 + \big(\sum_s \boldsymbol{v}_I^{s\,\mathrm{T}} \boldsymbol{x}_s\big)^2 = \Omega + \sum_{st} \boldsymbol{v}_R^{s\,\mathrm{T}} \boldsymbol{x}_s \cdot \boldsymbol{v}_R^{t\,\mathrm{T}} \boldsymbol{x}_t + \sum_{st} \boldsymbol{v}_I^{s\,\mathrm{T}} \boldsymbol{x}_s \cdot \boldsymbol{v}_I^{t\,\mathrm{T}} \boldsymbol{x}_t \qquad (11)$$

where Ω contains the quadratic terms of the energy model. In analogy to Section 3.1, a detector can be implemented by setting $v_R^s = \exp(i\theta s)\boldsymbol{v}_R$ and $v_I^s = \exp(i\theta s)\boldsymbol{v}_I$ for appropriate filters $\boldsymbol{v}_R$ and $\boldsymbol{v}_I$

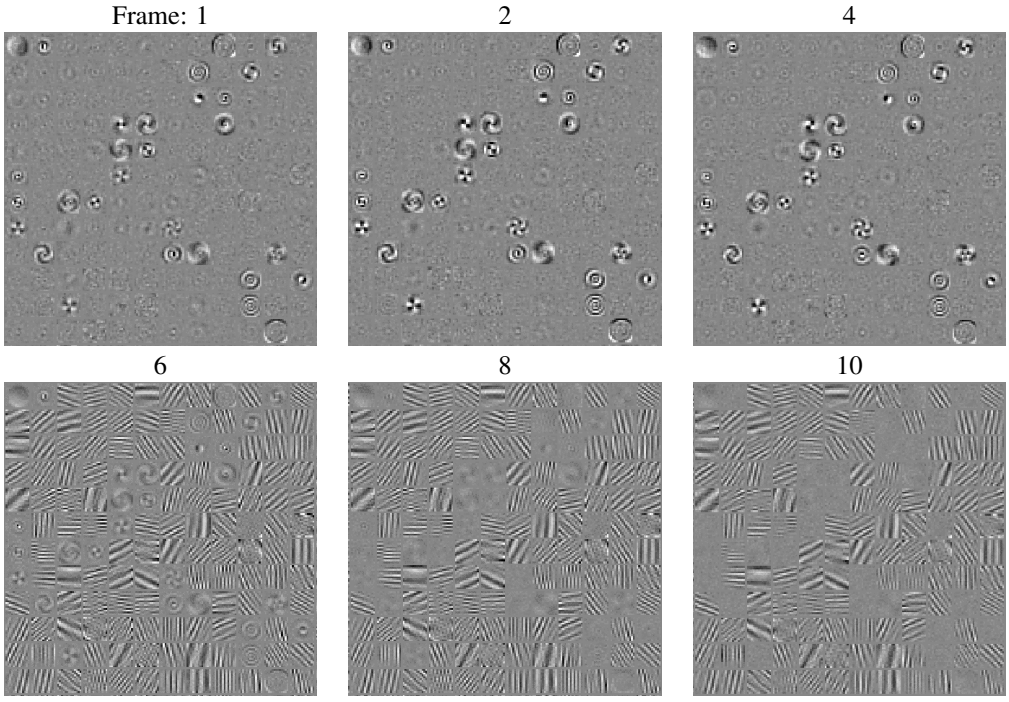


Figure 4: Six frames from the "eigenmovies" of transforming random dots, which rotate for 5 frames, then translate for 5 frames. The figure shows that each filter specializes, such that it encodes either only the first half of a movie or only the second half of a movie.

(cf. Figure 2 (b)). The sequences of filters are the "eigenmovies" of a given class of transformation sequences.

An important observation is that this will require consistency between the filters across time. Each factor — corresponding to a sequence of T filters — can only model the *repeated application of the same transformation*. We verify that this is what happens when training an energy model (where we use a gated autoencoder, as suggested in the aforegoing section) using the following experimental setup: We trained an energy model on concatenated frames from videos showing moving random dots. We used $F = 256$ factors and $K = 128$ mapping units, where $\boldsymbol{x} = \boldsymbol{y}$ is given by the concatenation of 10 frames. Filters are constrained, such that $w^x_{if} = w^y_{if}$. Each 10-frame input shows random dots that first rotate at a constant speed for 5 frames, then translate at a constant speed for the remaining 5 frames. Orientation, speed and direction vary across movies.

Figure 4 depicts a subset of 144 filters as they evolve throughout the succession of the 10 frames. It shows that the model learns to decompose the movies into (a) Fourier filters which are quiet during the first half of a movie and (b) rotation features which are quiet in the second half of the movie. There are no filters which model both types of transformation, as predicted by the analysis.

5 Discussion

We analyzed gated sparse coding models and energy models of complex cells and we showed that training these models can be viewed as performing a joint diagonalization of a set of orthogonal matrices. Our analysis suggests that computing cross-correlations (genuine multiplication) may be advantageous over squaring, because it allows for a more flexible connectivity of filter products. More importantly, our analysis suggests that either squares or cross-correlations will be crucial for learning *relations*, which in turn is crucial for extending feature learning and deep learning applications beyond the object recognition in static, single images, towards tasks that involve the fusion of motion, stereo or other types of multi-modal data.

Acknowledgments

This work was supported by the German Federal Ministry of Education and Research (BMBF) in the project 01GQ0841 (BFNT Frankfurt).

References

[1] E.H. Adelson and J.R. Bergen. Spatiotemporal energy models for the perception of motion. *J. Opt. Soc. Am. A*, 2(2):284–299, 1985.

[2] P.A. Arndt, H.A. Mallot, and H.H. Bülthoff. Human stereovision without localized image features. *Biological cybernetics*, 72(4):279–293, 1995.

[3] James Bergstra, Yoshua Bengio, and Jérôme Louradour. Suitability of V1 energy models for object classification. *Neural Computation*, pages 1–17, 2010.

[4] M Bethge, S Gerwinn, and JH Macke. Unsupervised learning of a steerable basis for invariant image representations. In B. E. Rogowitz, editor, *Human Vision and Electronic Imaging XII*, pages 1–12, Bellingham, WA, USA, February 2007. SPIE.

[5] D. Fleet, H. Wagner, and D. Heeger. Neural encoding of binocular disparity: Energy models, position shifts and phase shifts. *Vision Research*, 36(12):1839–1857, June 1996.

[6] Robert M. Gray. Toeplitz and circulant matrices: a review. *Commun. Inf. Theory*, 2:155–239, August 2005.

[7] David Grimes and Rajesh Rao. Bilinear sparse coding for invariant vision. *Neural Computation*, 17(1):47–73, 2005.

[8] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

[9] Aapo Hyvärinen and Patrik Hoyer. Emergence of phase- and shift-invariant features by decomposition of natural images into independent feature subspaces. *Neural Comput.*, 12:1705–1720, July 2000.

[10] Teuvo Kohonen. The Adaptive-Subspace SOM (ASSOM) and its use for the implementation of invariant feature detection. In *Proc. ICANN'95, Int. Conf. on Artificial Neural Networks*, volume I, pages 3–10. EC2, 1995.

[11] Q.V. Le, W.Y. Zou, S.Y. Yeung, and A.Y. Ng. Learning hierarchical spatio-temporal features for action recognition with independent subspace analysis. In *Proc. CVPR, 2011*. IEEE, 2011.

[12] Roland Memisevic. Gradient-based learning of higher-order image features. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.

[13] Roland Memisevic. Learning to relate images: Mapping units, complex cells and simultaneous eigenspaces. *ArXiv e-prints*, 2011.

[14] Roland Memisevic and Geoffrey Hinton. Unsupervised learning of image transformations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[15] Roland Memisevic and Geoffrey E Hinton. Learning to represent spatial transformations with factored higher-order Boltzmann machines. *Neural Computation*, 22(6):1473–92, 2010.

[16] Izumi Ohzawa, Gregory C. Deangelis, and Ralph D. Freeman. Stereoscopic Depth Discrimination in the Visual Cortex: Neurons Ideally Suited as Disparity Detectors. *Science (New York, N.Y.)*, 249(4972):1037–1041, August 1990.

[17] Bruno Olshausen, Charles Cadieu, Jack Culpepper, and David Warland. Bilinear models of natural images. In *SPIE Proceedings: Human Vision Electronic Imaging XII*, San Jose, 2007.

[18] Ning Qian. Computing stereo disparity and motion with known binocular cell properties. *Neural Comput.*, 6:390–404, May 1994.

[19] T. Sanger. Stereo disparity computation using Gabor filters. *Biological Cybernetics*, 59:405–418, 1988. 10.1007/BF00336114.

[20] A. Saxe, P. Koh, Z. Chen, M. Bhand, B. Suresh, and A. Y. Ng. On random weights and unsupervised feature learning. In *International Conference in Machine Learning*, 2011.

[21] Graham, W. Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *Proc. European Conference on Computer Vision (ECCV'10)*, 2010.

[22] Max Welling, Geoffrey E. Hinton, and Simon Osindero. Learning Sparse Topographic Representations with Products of Student-t Distributions. In *NIPS 2002*, 2002.