Learning invariant features by harnessing the aperture problem

Roland Memisevic MEMISEVR@IRO.UMONTREAL.CA
University of Montreal

Georgios Exarchakis EXARCHAKIS@BERKELEY.EDU
University of California, Berkeley

Abstract

The energy model is a simple, biologically inspired approach to extracting relationships between images in tasks like stereopsis and motion analysis. We discuss how adding an extra pooling layer to the energy model makes it possible to learn encodings of transformations that are mostly invariant with respect to image content, and to learn encodings of images that are mostly invariant with respect to the observed transformations. We show how this makes it possible to learn 3D pose-invariant features of objects by watching videos of the objects. We test our approach on a dataset of videos derived from the NORB dataset.

1. Introduction

The learning of invariant representations of objects has been a long-standing research goal in machine learning and vision over many years. Among the multitude of methods that have been proposed, the most common approach is to use transformation sequences (e.g. Foldiak, 1991; Wiskott & Sejnowski, 2002) in order to learn to *discount* variations in nearby frames. Since object class typically does not change on a fast time scale in natural videos, this leads to representations that retain object identity and that discard within-class variations (see, e.g., Wiskott, 2006, for a discussion). Learning of complex cell models, such as independent subspace analysis (Hyvärinen & Hoyer, 2000) and related models, can be viewed as an alternative approach to achieving invariance. Complex cell models typically learn invariances with respect to *small*, localized translations by utilizing the invariance of subspace energies, computed from quadrature Gabor features, to such translations (e.g. Hyvärinen et al., 2009).

Common to practically all approaches to learning invariance is that they utilize multiple views, typically videos, of objects at training time, and still images at test time. In this work, we explore an alternative approach to learning invariant representations by learning features that encode videos not images to represent objects. Our approach thus utilizes videos for training *and* for testing. We show how this makes it possible to derive a very simple, biologically plausible, deep learning approach to extracting invariant features based on the learning of subspace energies from videos.

In particular, we show that learning a 3-layer network containing subspace energies at the second layer will naturally learn *both* the transformations inherent in the training data and a representation that is invariant to those transformations. In contrast to classical complex cell models, our approach is not invariant to small, but also to large transformations, and it is invariant to transformations other than translation, such as rotation, or changes in 3-D orientation of objects.

To explore the learning of invariant 3-D features of objects from data, we introduce a derivative of the NORB data set (LeCun et al., 2004) consisting of videos of objects transforming in 3-D.

1.1. Related work

Our work is related to the approach described in (Hoyer & Hyvärinen, 2002) which is also based on adding a third feature learning layer on top of a layer of complex cell energy units. In contrast to that work, our approach involves learning to encode transformations not still images. Furthermore, we show how the *second* layer changes its semantics as a result of adding the third layer, and how this allows us to learn invariant features. Because of this, learning of all layers has to be performed jointly in our model, whereas related

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

complex cell models, such as (Hoyer & Hyvärinen, 2002), are trained greedily, layer-by-layer.

A prerequisite to a representation which is invariant to 3-D transformations is some degree of *understanding* of the 3-D structure of objects. Our approach may therefore be viewed also as a way to learn mid-level features of objects. Learning mid-level features using complex cell like models is also discussed in (Cadieu & Olshausen, 2011; Zou et al., 2012). In contrast to that work, our method is somewhat simpler, because it does not utilize an explicit polar decomposition of features. On a more technical level, (Cadieu & Olshausen, 2011) and (Zou et al., 2012) learn mid-level features by utilizing the transformation invariance of *subspace norms*, whereas our approach makes use of the invariance of *phase-derivatives* instead. We therefore utilize videos not still images to extract representations, even at test-time.

Our model makes use of the close relationship between the *cross-correlation* (or "gating") view (e.g., Arndt et al., 1995; Fleet et al., 1996) and the *oriented energy* view (Adelson & Bergen, 1985) of motion analysis. In this work, we show that the gating perspective can be viewed as a special case of the energy model, when using a denoising autoencoder (Vincent et al., 2008) with a particular noise-process for learning.

2. Complex cells and subspace pooling

Consider a short video sequence $\vec{X} \in \mathbb{R}^N$ consisting of T frames, $\vec{x}_1, \ldots, \vec{x}_T$, ($\vec{x}_i \in \mathbb{R}^M$, $i = 1, \ldots, T$, with $M = N/T$) which are related by the repeated application of a transformation $\boldsymbol{L}$, so that $\vec{x}_t = \boldsymbol{L}^t \vec{x}_1$. Most transformations between frames in a natural video can be represented by such a linear transformation in pixel space. $\boldsymbol{L}$ could, for example, take the form of a permutation matrix which moves any pixel x_i in $\boldsymbol{x}_t$ to any other position x_j in $\boldsymbol{x}_{t+1}$.

2.1. Translation and Fourier components

Videos showing only *spatial translations* in time can be represented naturally using the phase components of the Fourier spectra of the images across time: the Fourier components will show phase shifts, which together uniquely identify the translation direction and amount. The amplitude spectra, in contrast, will be constant, and so they constitute a representation that is *invariant* with respect to translation. The reason why the temporal evolution of the Fourier spectrum is a natural way to represent translations is that Fourier components (sines and cosines) are the eigenvectors of translations (e.g., Gray, 2005). For the same reason, the Fourier amplitude spectrum is invariant to translation.

The well-known energy model of complex cells (Adelson & Bergen, 1985) is an example of an application of this principle to the extraction of motion from videos: the energy model computes projections onto Fourier components in the form of simple cell responses (or localized Fourier components if one uses Gabor features), and subsequently measures phase changes in the subspaces spanned by these components. The whole set of phase changes across multiple spectral components constitutes a population code of motion. The same line of thought applies to estimation of binocular disparity (e.g. Fleet et al., 1996), because it also amounts to the estimation of local translations.

In real world data, however, not all images are represented equally well by all Fourier frequencies and orientations. If some Fourier components are not present in an image sequence, then it is not possible to extract any motion from that component. The absence of Fourier components of a given orientation makes it impossible to recover motion in that direction. Horizontal shifts, for example, are undetectable in images containing no horizontal Fourier components. The inability to detect motion as a result of missing Fourier components is typically referred to as the *aperture problem* in vision. In this work, we discuss how the aperture problem is intimately related to the task of *learning representations*, and how this relationship allows us learn invariant features from data.

To this end, first note that the partial absence of spectral components typically has an adverse affect on the analysis of motion, because it entails *content-dependence*: a population code of phase deltas encoding some observed motion will be different for any two images that contain different spectral components. That is, the *same* transformation applied to two different images will give rise to *different* code vectors to the degree that the images have different spatial frequency content.

In this work we shall show how using multi-layer pooling allows us to *utilize* this content-dependence, by learning features that are both invariant to learned transformations and selective to other properties of the inputs.

2.2. Complex cell video encodings

To learn about motion from videos, it is common to use complex cell models, which sum over two squared linear ("simple cell") responses (cf., Figure 1 (a)) assumed to be in quadrature (for example, sines and

cosines). The reason is that the transformations may be decomposed into rotations in two-dimensional subspaces (e.g. Memisevic, 2012).

When the input data is a video sequence of T frames ($\vec{X} = [\vec{x}_1, \ldots, \vec{x}_T]$), then the response of a vector $\vec{a}$ of complex cell responses can be written:

$$\vec{a} = \boldsymbol{Q}\left(\boldsymbol{W}\vec{X}\right)^2 \tag{1}$$

where $\boldsymbol{W} \in \mathbb{R}^{F\times N}$ denotes F feature vectors $\vec{W} \in \mathbb{R}^N$stacked side-by-side. Each feature vector spans a whole video, so it is composed of frame features $\vec{w}_s^{(f)}$, each of which spans a single frame. The matrix $\boldsymbol{Q} \in \mathbb{R}^{(F/2)\times F}$ is a band-diagonal pooling matrix with entries $Q_{i,2i} = 1$ and $Q_{i,2i+1} = 1$ and zero elsewhere. It pools over the elementwise products of $(\boldsymbol{W}\vec{x}) \cdot (\boldsymbol{W}\vec{x}) = (\boldsymbol{W}\vec{x})^2$ and remains constant during learning. When trained on natural images, the two features added together by $\boldsymbol{Q}$ tend to turn into Gabor features that are approximately 90 degree out-of-phase, so we refer to $\boldsymbol{Q}$ as *quadrature pooling* matrix.

A single a_q may also be written

$$\begin{aligned} a_q &= \sum_{f=1}^{2}\left(\vec{W}_f^{\mathrm{T}}\vec{X}\right)^2 \\ &= \sum_{f=1}^{2}\Big(\sum_{\substack{s,t=1\\ s\neq t}}^{T}\left(\vec{w}_{fs}^{\mathrm{T}}\vec{x}_s\right)\cdot\left(\vec{w}_{ft}^{\mathrm{T}}\vec{x}_t\right) + \sum_{s=1}^{T}\left(\vec{w}_{fs}^{\mathrm{T}}\vec{x}_s\right)^2\Big) \\ &= \sum_{\substack{s,t=1\\ s\neq t}}^{T}\sum_{f=1}^{2}\left(\vec{w}_{fs}^{\mathrm{T}}\vec{x}_s\right)\cdot\left(\vec{w}_{ft}^{\mathrm{T}}\vec{x}_t\right) + \sum_{s=1}^{T}\sum_{f=1}^{2}\left(\vec{w}_{fs}^{\mathrm{T}}\vec{x}_s\right)^2 \end{aligned} \tag{2}$$

where $\vec{w}_{fs} \in \mathbb{R}^M$, for $s = 1, \ldots, T$, are the components of $\vec{W}_f$, which span the individual frames.

The decomposition in Eq. 2 (rhs) follows from the binomial identity (see, e.g., Fleet et al., 1996; Memisevic, 2012). It shows that the complex cell response consists of two parts: The computation of relative 2-D *phase-angles*, that take the form of 2-D inner products between all pairs of projected frames (first sum); the computation of the *norms* of the projected frames in the 2-D-subspaces (second sum). When both frames and frame features are contrast-normalized, then the complex cell response is an encoding of motion. For example, if frame features $\vec{w}_{fs}, \vec{w}_{ft}$ are phase-shifted Fourier components, then the set of a_q encode translations in the form of phase-deltas between the projections of multiple frames (e.g., Fleet et al., 1996; Qian, 1994; Memisevic, 2012).

Because of the aperture problem, the representation of motion depends on the Fourier amplitudes of the input: a frequency component that is not present in the images turns off the response a_q, even if the motion is present in the video. As a result, the a_q show a dependency on image content through the Fourier amplitudes of the individual images. A solution is to let complex cells pool not only over quadrature features but over multiple subspaces to reduce this dependency (e.g., Fleet et al., 1996).

Formally, this amounts to replacing the quadrature pooling matrix $\boldsymbol{Q}$ in Eq. 1 by a full pooling matrix $\boldsymbol{P}$ that can be learned from data along with the features $\boldsymbol{W}$. See Figure 1 (b) for an illustration. Pooling across multiple components a_q yields motion features, $\vec{m}$, that are robust against the aperture problem because any motion affects multiple frequency components. Translation, for example, is visible in all Fourier frequencies whose orientation is the same as the direction of translation, and it is also visible in other, nearby orientations. $\boldsymbol{P}$ reduces the dependencies on image content by pooling *across* multiple 2-d subspaces. The pooled motion features $\vec{m}$ are also referred to as "mapping units" in the literature.

Various parameter estimation schemes have been used to learn complex cell models to encode videos (both with pooling across subspaces and without). They include maximizing sparsity (Hyvärinen & Hoyer, 2000; Le et al., 2011), maximizing likelihood (Memisevic & Hinton, 2010), subspace clustering (Bethge et al., 2007), predictive sparse coding (Memisevic, 2011), minimizing reconstruction error while enforcing amplitude constancy across frames (Cadieu & Olshausen, 2011; Zou et al., 2012), or training the closely related bilinear models (e.g., Rao & Ruderman, 1999; Olshausen et al., 2007; Miao & Rao, 2007; Grimes & Rao, 2005).

2.3. Generalizing the aperture problem

Learning features from data instead of hard-coding them was shown to extend the applicability of complex cell models beyond Fourier and Gabor components and beyond modeling translations. In particular, there exist features, $\boldsymbol{W}$, which can represent any orthogonal image transformations that form a *commutative group* (e.g., Memisevic, 2012; Lee & Soatto, 2011). Since orthogonal transformations are uniquely determined by a set of rotations in their eigenspaces and since the eigenspaces are shared among commuting transformations, extracting transformations from images amounts to extracting rotation angles in the eigenspaces. This makes it possible to learn video feature that encode rotations, local shifts or other, more complex motions. When training complex cell

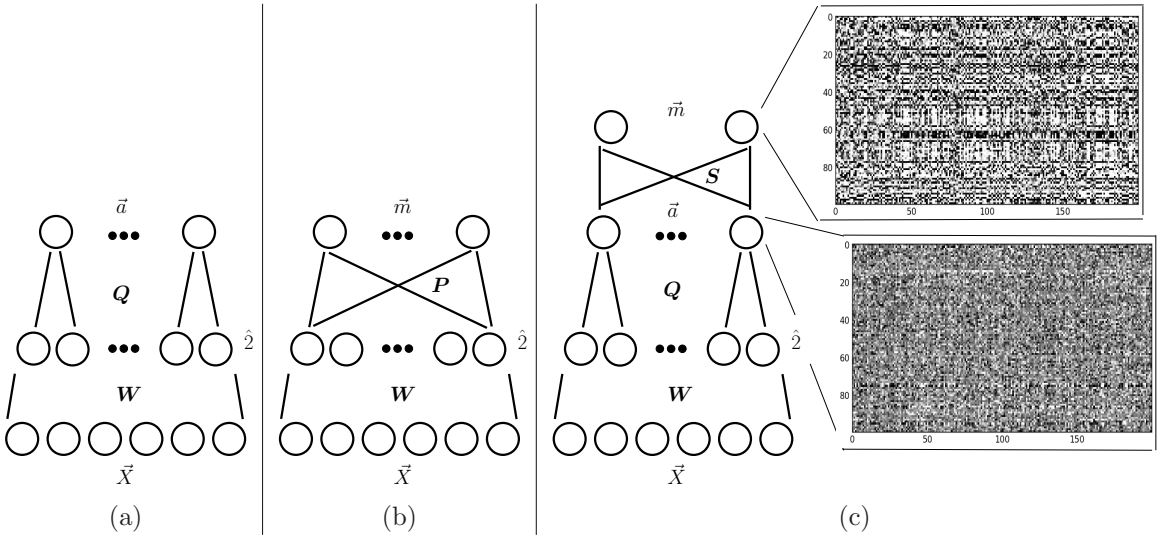

Figure 1. **(a)** Complex cell models, such as independent subspace analysis (ISA), extract energy in the invariant subspaces of transformations. **(b)** Due to the aperture problem, pooling across multiple subspaces is necessary to learn transformations from data. **(c)** By separating pooling *across* subspaces from pooling *within* subspaces we obtain features that are transformation invariant and content-dependent at the same time. **(Right-most panels)** In practice, top-level codes of simple motions (top panel) show more structure than intermediate level codes (bottom panel), because top-level codes do not vary as much when example images change (see text for details).

models on natural videos, it can be shown that the models decompose these into approximately group-structured components such as Gabor features (e.g. Cadieu & Olshausen, 2011; Memisevic, 2012). This allows them to learn complex motion patterns for tasks like activity recognition (Le et al., 2011; Taylor et al., 2010). It is important to note that, even in these more general settings, the aperture problem holds. In other words, motion estimates in the form of a population code of complex cell features, a_q, will depend on how well individual images are aligned with each subspace, unless across-subspace pooling is used to remove (or at least alleviate) that dependency.

2.4. Subspace segregation

Although the population code $\vec{a} = \left(a_q\right)_{q=1}^{F/2}$ is not an optimal representation of motion, it can be a good representation of *object identity*: Since it is affected by the aperture problem, it is sensitive to any transformation that the model was *not* trained on. By definition, these transformations change the magnitude of all subspace projections and thus change both the inner products and the norms in the subspaces.

More importantly, $\vec{a}$ will nevertheless tend to be *pose-invariant*, because a transformation applied to all frames of a video changes only the phase-angles of the subspace projections. In other words, the learned transformations, by definition, leave subspace norms and the relative phase-deltas between the projections constant.

Here, we suggest extracting these pose-invariant (but otherwise object sensitive) features using two-layer pooling as follows (cf., Figure 1 (c)): We use two-dimensional subspaces, as prescribed by the geometry of orthogonal transformations, and we use a *separate pooling layer* to overcome the aperture problem. Formally, we introduce a across-subspace pooling matrix $\boldsymbol{S} \in \mathbb{R}^{D \times F/2}$ of dimension $F/2 \times D$, and define a D-dimensional video code

$$\vec{m} = \boldsymbol{S}\vec{a} \tag{3}$$

The components of $\vec{a}$ are the same as in Eq. 1 (cf., Figure 1). This allows the learning of content-independent motion components, while leaving subspace features accessible to other pathways, such as those involved in object recognition.

As the lower-level features $\vec{a}$ encode image content via the aperture problem, we shall call them "aperture features" in the following. In the special case of translations, aperture features are similar to the amplitude spectrum. But they can learn other, more complex types of transformation as we shall show. For robustness, it can make sense to apply an elementwise nonlinearity to $\vec{m}$ in classification tasks, which we do in our experiments below.

It is interesting to note that performing invariant recognition by using a linear classifier on the aperture features amounts to computing a linear combination, thus it is a form of pooling, too. Computing a representation of motion (Eq. 3) amounts to a different kind of pooling over the same features. That way, aperture features provide a multi-view "substrate" for objects, which encodes information about both, transformations and invariances. Multiple higher-level pathways can selectively extract this information through different forms of pooling.

2.5. Learning

Training the model (Figure 1 (c)) seems hard at first sight because of the presence of squaring non-linearities. However, there is a wide variety of complex cell models as discussed is Section 2.2, which can learn in the presence of these. Here, we use the approach described by (Memisevic, 2011), who show that a denoising autoencoder (Vincent et al., 2008) can learn in the presence of such squaring non-linearities by training to reconstruct multiple noisy copies of the input from each other.

More specifically, consider a decoder corresponding to the encoder network shown in Figure 1 (c). We use "tied weights", so the decoder parameters are the transpose of the encoder parameters. By combining Eqs. 1 and 3, and writing the square in Eq. 1 as a product, we may write the encoder activation in the form

$$\vec{m} = \boldsymbol{S}\boldsymbol{Q}\left(\boldsymbol{W}\vec{X}\right) \cdot \left(\boldsymbol{W}\vec{X}\right) \tag{4}$$

where $\cdot$ denotes elementwise multiplication. This shows that we can interpret the encoder as a three-layer linear autoencoder $\boldsymbol{SQW}\vec{X}$ whose second layer is *gated* (multiplied elementwise) by another projection of the input, $\boldsymbol{W}\vec{X}$. Likewise, we can define the decoder as

$$\vec{X}_{\text{rec}} = \boldsymbol{W}^{\text{T}}\left(\boldsymbol{W}\vec{X}\right) \cdot \left(\boldsymbol{Q}^{\text{T}}\boldsymbol{S}^{\text{T}}\vec{m}\right) \tag{5}$$

which is another three-layer linear network, whose next-to-last layer is again gated by $\boldsymbol{W}\vec{X}$. In contrast to a standard denoising autoencoder, the input occurs twice in both the encoder and decoder. Using independent (factorial) noise as the corruption process (Vincent et al., 2008) thus implies that the two copies of $\vec{X}$ have to be corrupted independently for training the model (see also, Memisevic, 2011).

Formally, we thus generate two noisy copies $\vec{X}_{\text{noise1}}$, $\vec{X}_{\text{noise2}}$ for each training case $\vec{X}$ and train the model by minimizing the reconstruction error that results from reconstructing one noisy copy of the input from the other:

$$\left(\vec{X}_{\text{noise2}} - \boldsymbol{W}^{\text{T}}\left(\boldsymbol{W}\vec{X}_{\text{noise1}}\right) \cdot \left(\boldsymbol{Q}^{\text{T}}\boldsymbol{S}^{\text{T}}\text{sigmoid}(\vec{m})\right)\right)^2 \tag{6}$$

with

$$\vec{m} = \boldsymbol{S}\boldsymbol{Q}\left(\boldsymbol{W}\vec{X}_{\text{noise2}}\right) \cdot \left(\boldsymbol{W}\vec{X}_{\text{noise1}}\right) \tag{7}$$

Since gating is symmetric, one may also switch the roles of $\vec{X}_{\text{noise1}}$ and $\vec{X}_{\text{noise2}}$ for each training example in Eqs. 6 and 7, add up the two resulting costs.

There are many kinds of noise-process one can use for training denoising autoencoders (Vincent et al., 2008). In our experiments we use "zero-mask" noise which amounts to independently setting to zero a certain fraction (typically 50% in our experiments) of the components of the inputs (Vincent et al., 2008).

Rather than reconstruction one noisy copy of the image from another noisy copy in Eq. 6, one may also reconstruct the original input. We experimented with both approaches in practice and did not observe a significant difference across any of these approaches.

2.6. Relationship with factored bilinear models

Computing a weighted sum of squared filter responses is closely related to factored bilinear models, such as the factored gated Boltzmann machine (Memisevic & Hinton, 2010). One can show that both squaring non-linearities and factored bilinear models encode relationships between images by recovering rotation angles in the invariant subspaces of the transformation class (Memisevic, 2012; Fleet et al., 1996).

It is interesting to note that we can recover the gated autoencoder (Memisevic, 2011) as a special case from Eqs. 6 and 7 using a particular corruption process: consider a video consisting of only two frames. Now define $\vec{X}_{\text{noise1}}$ as the input video with the first frame blanked out (set to zero), and $\vec{X}_{\text{noise2}}$ as the input video with the second frame blanked out. Inference during learning (Eq. 7) now amounts to multiplying the filter responses from two frames, and reconstruction amounts to multiplying filter responses from one frame to reconstruct the other. Both these operations are exactly the same as as in a factored bilinear model (e.g. Memisevic & Hinton, 2010; Memisevic, 2011).

An illustration of features learned using the gating approach is shown in the two right-most panels in Figure 1 (c). They show subsets of the mapping units $\vec{m}$ computed for test-data after training the model on videos showing 3-D rotations of objects. The figure shows that top-level units are more structured and less noisy than the intermediate level units, which is consistent with the fact that the data set consists of many

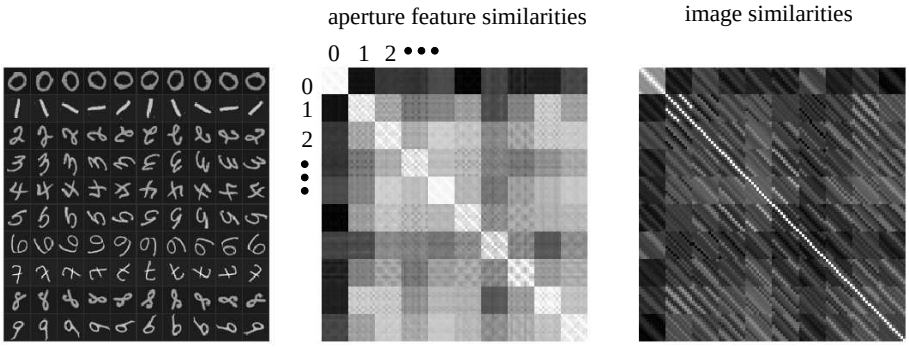

Figure 2. **Left-to-right:** Example subset of the mnistrot data set; similarity matrix for 100 example digits (10 in each class); similarity matrix for images

different objects which are transformed in a small number of ways (rotations in 3-D) For details, see Section 3.

3. Learning invariant features from videos

3.1. Learning rotation invariant features

We performed a quantitative comparison to evaluate the utility and the degree of invariance of the aperture features. We trained the three-layer model (Figure 1 (c)) using 2000 video features that are pooled into 1000 aperture features and subsequently into 200 mapping units. For training, we used videos of length 2 showing rotations of random dot images to learn rotation-invariant aperture features.

To test the model used the *mnistrot* data set described in (Larochelle et al., 2007). The data set consists of 60000 training images and 12000 test images showing rotated MNIST digits of size 28×28 pixels. Some example images are shown in Figure 2 (left). The two right-most plots in the figure show the cosine-similarity between aperture features trained on rotations and raw images for a random subset of 100 images (10 from each class). They demonstrate that aperture features, although not trained on digits, are better at capturing the similarity structure inherent in the data, as representations of digits from the same class tend to be more similar than representations of digits from different classes.

Figure 3 shows classification error rates for several training set sizes, obtain using raw images (blue), subspace features trained with a gating model (Memisevic, 2012) (dark blue), and subspace features trained with an energy model (black). We used logistic regression to predict digit class from the aperture features. On the images, we tried logistic regression and k-nearest neighbors, and we found that k-nearest neighbors works best. The number of neighbors, as well as weight-decay for logistic regression are determined using cross-validation. The figure shows that aperture features based on subspace norms perform best, and clearly outperform raw digits. Note that aperture features were trained on rotating random dot images, not digits.

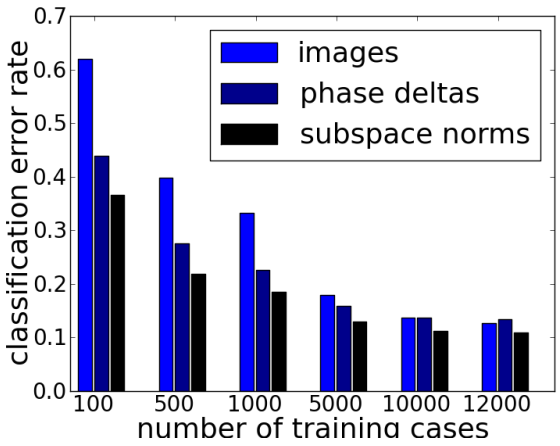

Figure 3. Classification error rates on mnistrot.

3.2. The "NORBvideos" data set

To evaluate our approach, and to facilitate experimentation with learning-based 3-D invariance, we created a data set of objects that rotate in 3-D. The data set is available at `www.iro.umontreal.ca/~memisevr/aperture`

The data set is derived from the NORB data set (LeCun et al., 2004), which consists of images of objects shown from various viewpoints (by changing az-

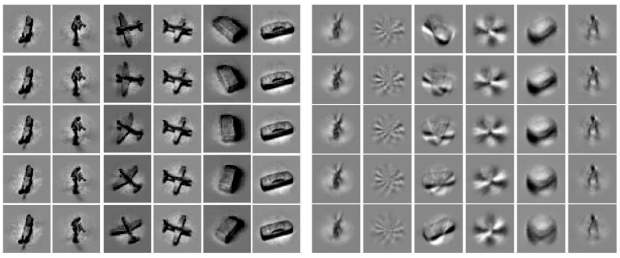

Figure 4. Example videos and extracted features. **Left:** A random selection of videos from the training set. In each frame we get a view from a different angle, with various degrees of change. **Right:** Samples of the filters after training.

imuth and camera elevation) and under various lighting conditions. Each object belongs to one of 5 *classes* (*four-legged animals*, *human figures*, *airplanes*, *trucks*, and *cars*), and to one of 9 *instances* per class. From the NORB data, we generate video sequences that show "fly-overs" of each object by incrementally changing viewpoints, while maintaining the same object class, instance and lighting. We generate 109350 videos for training, using only NORB object instances 1 through 8 and 12150 videos for testing, using instance 9. This ensures that the sets of objects shown in the training and test sets are disjoint. Some examples are shown in Figure 4 (left).

All images are PCA-whitened, retaining 95% of the variance, and subsequently projected onto a basis learned using a contractive autoencoder (Rifai et al., 2011) with a complete basis. We abbreviate the resulting "autoencoder sparse coding" representation ASC in the following. To learn mid-level features of 3-D structure, we concatenate the preprocessed images to obtain videos of 5 frames. Note that a subset of the transformations in this data set are 2-D rotations, since for some views the rotation angle will be parallel to the image plane.

3.3. Learning a 3-D invariant representation

To test complex cell video features, we used the autoencoder described in Subsection 2.4 with 2 frames, 1000 filters, 500, intermediate-level units and 100 mapping units. We trained the model using "gating" noise (cf., Section 2.6). Figure 4 (right) shows that some features resemble a circular Fourier basis (Memisevic & Hinton, 2010; Bethge et al., 2007), that can encode 2-D rotations, while others seem to be responsible for encoding out-of-plane rotations. Many filters look distinctly different than 2-D rotation filters.

We use analogy making as a way to assess qualitatively if the model can extract 3-D structure from the data: given two source images $\vec{x}_{\text{source}}$ and $\vec{y}_{\text{source}}$, we infer the top-level poolings. Clamping a different input image $\vec{x}_{\text{target}}$ and using the decoder of the model then allows us to infer a prediction $\vec{y}_{\text{target}}$ based on the transformation seen in the source image pair (see, e.g. Memisevic & Hinton, 2010).

Figure 5 (bottom right) shows output images from a random subset of target plane images taken from the *test data*, not seen during training. Source input and output images, as well as target input images, are shown on the top left, top right, and bottom left, respectively. The figure shows that the model is able to infer the correct transformation in most (not all) cases. More importantly, the correctly inferred transformations indicate that the model correctly infers some aspects of the 3-D structure of the test data to produce output images of the right shape (cf., for example, third from the right, top row).

3.4. Quantitative evaluation

We compared the representation from our model with the autoencoder (ASC) features on single frames by training a logistic regression classifier on the inferred representations. Since aperture features should tend to be invariant to 3-D transformations by design, we expect them to perform better than image features, in particular on small training set sizes. We use as training, validation and test data sets using the NORB object instances $0-4$, $5-8$, and 9, respectively (cf., LeCun et al., 2004). This makes it possible to obtain a sufficiently large training set for learning features, and still a sufficiently large test set of about 12000 videos to obtain significant results. Only the training subset is used for learning features. We learn features on the full 5-frame videos using the independent noise across all frames (cf., Section 2.5) Figure 6 shows classification results using regularized logistic regression on various training set sizes. We cross-validate the regularization parameter on the training subset. The figure shows that aperture features significantly outperform the ASC features, as well as classification based on mapping units. More importantly, classification performance is much more consistent as the training set size gets very small. Both ASC and aperture features perform much better than random, which is 20% correct in this data. The plot also shows that representing objects using videos improves recognition performance both over images in isolation and over videos consisting just of *pairs* of frames.

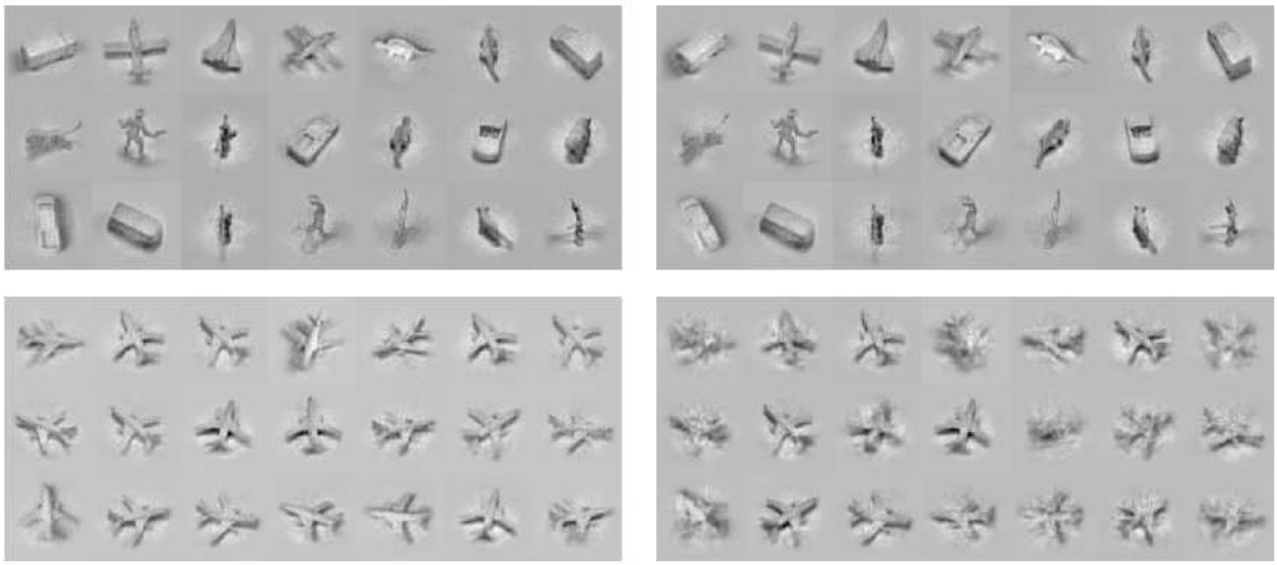

Figure 5. Example analogies using random test data. **Left-to-right, top-to-bottom:** Source input images, source output images, target input images, target output images rendered by the model. Target output images are generated by inferring the transformation from objects in the images shown on the top, and applying the transformation to the images on the bottom-left.

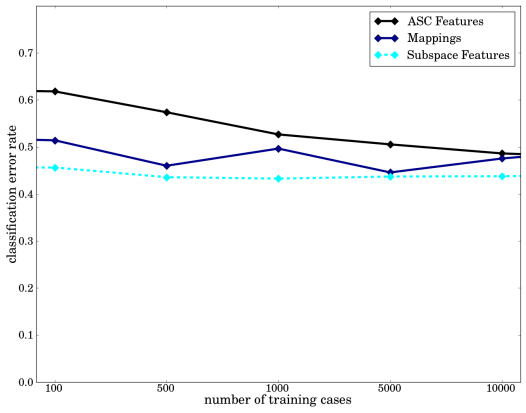

Figure 6. Classification performance for "NORBvideos" object recognition.

4. Discussion

One of the defining properties of a 3-D model is that it is a representation that is invariant to 3-D rotations. Both theory and experiments suggest that subspace energies computed from short videos are capable of providing a representation from which such a model can be easily derived, for example through pooling. Our work shows that one way to learn suitable subspace energies is by making use of a twist on deep learning, where adding a layer is not used to learn more abstract features but to change the meaning of a *lower layer* of the model.

Our work parallels some recent work in computer vision on learning features for recognition using videos (e.g., Lee & Soatto, 2011). A common finding in this line of work is that current object recognition benchmarks are somewhat unrealistic, because they provide training data in the form of still images not videos. Unlike practically all common object recognition systems that are based on local descriptors, humans have no problem distinguishing a picture of a chair from a real chair, and one of the reasons for this is probably that humans learn about objects not from still images but from multiple views of the objects (as well as from interacting with the objects).

Acknowledgments

This work was supported in part by the German Federal Ministry of Education and Research (BMBF) in the project 01GQ0841 (BFNT Frankfurt). This work was done in part while the authors were at the University of Frankfurt. The authors would also like to thank the reviewers for several useful suggestions.

References

Adelson, E.H. and Bergen, J.R. Spatiotemporal energy models for the perception of motion. *J. Opt. Soc. Am.*

A, 2(2):284–299, 1985.

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

Bethge, M, Gerwinn, S, and Macke, JH. Unsupervised learning of a steerable basis for invariant image representations. In *Human Vision and Electronic Imaging XII*. SPIE, February 2007.

Cadieu, Charles F. and Olshausen, Bruno A. Learning Intermediate-Level Representations of Form and Motion from Natural Movies. *Neural Computation*, 24(4):827–866, December 2011.

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

Foldiak, Peter. Learning invariance from transformation sequences. *Neural Computation*, 3(2):194–200, 1991.

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

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

Hoyer, Patrik and Hyvärinen, Aapo. A multi-layer sparse coding network learns contour coding from natural images. *Vision Research*, 42:1593–1605, 2002.

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

Hyvärinen, Aapo, Hurri, J., and Hoyer, Patrik O. *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision*. Springer Verlag, 2009.

Larochelle, Hugo, Erhan, Dumitru, Courville, Aaron, Bergstra, James, and Bengio, Yoshua. An empirical evaluation of deep architectures on problems with many factors of variation. In *ICML*, 2007.

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

LeCun, Yann, Huang, Fu-Jie, and Bottou, Leon. Learning Methods for generic object recognition with invariance to pose and lighting. In *CVPR*, 2004.

Lee, T. and Soatto, S. Video-based descriptors for object recognition. *Image and Vision Computing*, 29(10):639–652, September 2011.

Memisevic, Roland. Gradient-based learning of higher-order image features. In *ICCV*, 2011.

Memisevic, Roland. On multi-view feature learning. In *ICML*, 2012.

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

Miao, Xu and Rao, Rajesh. Learning the lie groups of visual invariance. *Neural Computation*, 19(10):2665–2693, 2007.

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

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

Rao, Rajesh and Ruderman, Daniel. Learning lie groups for invariant visual perception. In *In Advances in Neural Information Processing Systems 11*. MIT Press, 1999.

Rifai, Salah, Vincent, Pascal, Muller, Xavier, Glorot, Xavier, and Bengio, Yoshua. Contractive Auto-Encoders: Explicit Invariance During Feature Extraction. In *ICML*, 2011.

Taylor, Graham, W., Fergus, Rob, LeCun, Yann, and Bregler, Christoph. Convolutional learning of spatio-temporal features. In *ECCV*, 2010.

Vincent, Pascal, Larochelle, Hugo, Bengio, Yoshua, and Manzagol, Pierre-Antoine. Extracting and composing robust features with denoising autoencoders. ICML, 2008.

Wiskott, Laurenz. How does our visual system achieve shift and size invariance? In *23 Problems in Systems Neuroscience*, chapter 16, pp. 322–340. Oxford University Press, New York, 2006.

Wiskott, Laurenz and Sejnowski, Terrence. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.

Zou, Will, Ng, Andrew, Zhu, Shenghuo, and Yu, Kai. Deep learning of invariant features via simulated fixations in video. In *Advances in Neural Information Processing Systems 25*. 2012.