Recursive Sparse, Spatiotemporal Coding

Thomas Dean
Google Inc.
tld@google.com

Greg Corrado
Stanford University
gcorrado@stanford.edu

Rich Washington
Google Inc.
rwashington@google.com

Abstract

We present a new approach to learning sparse, spatiotemporal codes in which the number of basis vectors, their orientations, velocities and the size of their receptive fields change over the duration of unsupervised training. The algorithm starts with a relatively small, initial basis with minimal temporal extent. This initial basis is obtained through conventional sparse coding techniques and is expanded over time by recursively constructing a new basis consisting of basis vectors with larger temporal extent that proportionally conserve regions of previously trained weights. These proportionally conserved weights are combined with the result of adjusting newly added weights to represent a greater range of primitive motion features. The size of the current basis is determined probabilistically by sampling from existing basis vectors according to their activation on the training set. The resulting algorithm produces bases consisting of filters that are bandpass, spatially oriented and temporally diverse in terms of their transformations and velocities. The basic methodology borrows inspiration from the layer-by-layer learning of multiple-layer restricted Boltzmann machines developed by Geoff Hinton and his students. Indeed, we can learn multiple-layer sparse codes by training a stack of denoising autoencoders, but we have had greater success using L_1 regularized regression in a variation on Olshausen and Field's original SPARSENET. To accelerate learning and focus attention, we apply a space-time interest-point operator that selects for periodic motion. This attentional mechanism enables us to efficiently compute and compactly represent a broad range of interesting motion. We demonstrate the utility of our approach by using it to recognize human activity in video. Our algorithm meets or exceeds the performance of current state of the art activity-recognition methods.

Introduction

This work focuses on learning sparse, over-complete spatiotemporal codes in the spirit of Olshausen and Field (Olshausen and Field 1997), Hyvärinen *et al* (Hyvärinen, Hurri, and Väyrynen 2003), and others. Our initial investigation into this area was inspired by the work of Cadieu and Olshausen (Cadieu and Olshausen 2008) on learning transformational invariants from the statistics of natural movies. We adopt a generative model similar to that of Olshausen and Field (Olshausen and Field 1997) and an alternating-optimization algorithm analogous to the analysis-synthesis model proposed by Mumford (Mumford 1994) and used by Olshausen and field in their *SPARSENET* implementation.

Following the trend in sparse coding, we substitute L_1-regularized least-squares algorithms for the conjugate gradient solver used in SPARSENET. In particular, we develop several variants of the one-at-a-time coordinate-wise descent algorithm of Friedman *et al* (Friedman et al. 2007) and experiment with the feature-sign algorithm of Lee *et al* (Lee et al. 2007). We also employ various methods for *shaping* the variance of the activations — the coefficients of the basis vectors in solutions to the least-squares problem — to ensure that all of the basis vectors are contributing.

Spatiotemporal bases that span more than a few frames of video have large numbers of weights and are slow to train. We present an algorithm that accelerates sparse coding by recursively constructing basis vectors, adjusting only a fraction of the weights at any given time. The resulting bases exhibit a wide range of orientations, scales and velocities, and outperform bases trained in a conventional manner.

We found it to be the case that interesting motion is quite rare even in videos selected for illustrating motion, and so we experimented with several interest-point operators to extract 3-D patches more likely to contain motion of the sort characteristic of human behavior. We use the space-time interest-point operator of Dollár *et al* (Dollár et al. 2005) as a filter for extracting space-time volumes that exhibit periodic motion, and then use large collections of these volumes for learning sparse codes.

To evaluate our approach, we applied the resulting sparse codes to recognizing human activity. We adapted software developed by Piotr Dollár and performed the initial testing on his facial-expression dataset (Dollár et al. 2005) and the Weizmann human-action dataset (Gorelick et al. 2007). We then took codes trained on data from these two datasets and applied them to the KTH human-action dataset (Schuldt, Laptev, and Caputo 2004) achieving recognition error comparable to state-of-the-art methods. In leave-one-out experiments where we both trained and tested on the KTH dataset, our approach meets or exceeds the performance of current state-of-the-art methods.

Preprocessing

We apply several preprocessing steps to the data prior to its use in learning sparse codes or inferring coefficients to reconstruct a 3-D patch as a sparse, linear combination of basis vectors. First, we apply a linear transform to each frame so that the pixels are uncorrelated (spatially) and their variances equal. This *whitening* step is used to remove correlations in the data that a learning algorithm would otherwise have to account for and typically are not of interest. Some neuroscientists believe the primate early visual system employs a gain-control mechanism whereby the response of each cell is normalized by the integrated activity of its neighbouring cells. Brady and Field (Brady and Field 2000) provide a good case that such a mechanism reduces cell-response variability both within and between scenes, as well as reducing the entropy of the response distribution resulting in a more efficient transfer of information. We therefore apply a method of local-contrast normalization which simulates this neural mechanism and thereby reduces undesirable variability in postprocessing. These two preprocessing steps, image whitening and local contrast normalization, provide a very rough approximation to the information being transferred by the optic tract leading from the retina and ultimately entering the striate cortex.

We found it difficult to efficiently learn useful spatiotemporal codes from random 3-D patches extracted from video, and so we experimented with several interest-point operators to extract space time volumes likely to contain motion of the sort characteristic of human behavior. We use the space-time interest-point operator of Dollár *et al* (Dollár et al. 2005) as a filter for extracting space-time volumes that exhibit periodic motion, and collect a large set of these volumes for learning sparse codes. Each video is first convolved with a 2-D Gaussian smoothing kernel applied along the spatial dimensions. The result is then convolved with a quadrature pair of 1-D Gabor filters applied temporally. A detector is tuned to respond whenever the variation in local image intensities contain periodic frequency components. The response function for the detector is defined as follows:

$$R = (I * g * h_1)^2 + (I * g * h_2)^2$$

where $g(x, y; \sigma)$ is the 2-D smoothing kernel, $\{h_1, h_2\}$ is the quadrature pair determined by $h_1(t; \tau, \omega) = -\cos(2\pi t\omega)e^{-t^2/\tau^2}$ and $h_2(t; \tau, \omega) = -\sin(2\pi t\omega)e^{-t^2/\tau^2}$. Following (Dollár et al. 2005), we set $\omega = 4/\tau$, and thus σ and τ correspond, respectively, to the spatial and temporal scale of the detector. A 3-D patch or *cuboid* is extracted at each local maxima of the filter response function using non-max-suppression to control for overlap.

We also experimented with an interest point detector developed by Laptev and Lindeberg (Laptev and Lindeberg 2003). Laptev and Lindeberg extend the idea of Harris-corner detectors in the spatial domain (Harris and Stephens 1988) to space-time interest points characterized by strong variation in both the spatial and temporal dimensions, *e.g.*, the abrupt change in velocity when a ball is kicked. They do so by expanding a linear scale-space representation of the image with a matrix of first-order spatial and temporal derivatives, and searching for regions with significant eigenvalues. The method is extended to be scale invariant by applying a normalized spatiotemporal Laplace operator. We found the Laptev and Lindeberg detector to be too restrictive for activity recognition and the Dollar *et al* detector to include most if not all of the points detected by Laptev and Lindeberg while excluding most irrelevant motion.

Sparse Coding

Olshausen and Field (Olshausen and Field 1996) present their method of learning a sparse basis to represent natural images as solving the following optimization problem:

$$B^* = \arg\min_B \langle \min_A \ \|X - AB\|_2^2 + \lambda S(A) \rangle \qquad (1)$$

where X is an $M \times L$ data matrix consisting of M cuboids of L pixels each, B is an $N \times L$ matrix of N basis vectors, A is an $M \times N$ matrix of coefficients intended to reconstruct X as a linear combination of the basis vectors, $S(A)$ is a sparsity penalty, and λ is a constant that trades reconstruction error for sparsity. For many standard penalty functions, the objective function is convex in A if we hold B fixed and convex in B if we hold A fixed, and so solutions to Equation 1 are often solved using an iterative process in which each iteration consists of two steps: In the first step, we fix the coefficients and solve for the basis vectors subject to a set of linear constraints that control for the size of the basis weights. In the second step, we fix the basis vectors and solve for the coefficients in an attempt to reconstruct the input modulo some variant of weight penalty designed to encourage sparsity. Learning consists of alternating between these two steps until convergence or some performance threshold is achieved.

Mumford (Mumford 1994) describes this iterative process as an *analysis-synthesis* loop, which he conjectures plays an important role in early visual processing. In the case of learning a sparse code for natural images, the first step — solving for the basis vectors — constitutes a form of analysis which produces an explanation in the form of a generative model of the data; we call this the *analysis step*. The second step —- solving for the coefficients — corresponds to synthesizing the data from a given fixed basis; we call this the *synthesis step*.

Olshausen and Field (Olshausen and Field 1997) implemented a version of this analysis-synthesis loop which learns a sparse, over-complete code whose basis vectors correspond to filters that are bandpass and oriented and that resemble Gabor functions. The original Olshausen and Field work assumed a Cauchy prior on the coefficients, introduced a differentiable regularization term to implement this prior, and used a conjugate-gradient solver in the synthesis step:

$$\begin{aligned}&\text{minimize}_A \ J(A|B) = \\ &\qquad \|X - AB\|_2^2 + \lambda \sum_{i,j} \log(1 + A_{i,j}^2) \qquad (2)\end{aligned}$$

In the analysis step, they fix the coefficients and take one step of gradient descent toward solving the following minimization:

$$\text{minimize}_B \ J(B|A) = \|X - AB\|_2^2$$

Without imposing some constraint, this procedure will cause the basis weights to grow without bound. Olshausen and

Field deal with this by adapting the L_2 norm of each basis vector independently so the coefficients are maintained at an appropriate level and the separate variances over the training data in the activation of basis vectors are approximately equal.[1] Their implementation of this algorithm is called *SPARSENET*.

L_1 Regularization

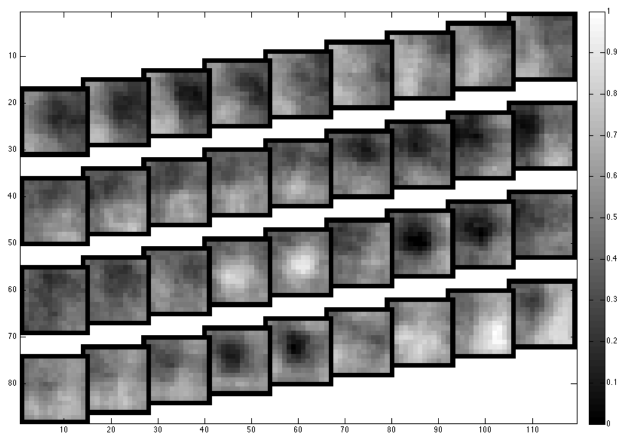
In the last few years, there has been a great deal of work in machine learning and statistics using L_1 regularization to induce sparsity. Since the L_1 term is not differentiable, much of the effort has gone into developing new algorithms for solving the corresponding optimization problem. We have experimented extensively with the method of one-at-a-time coordinate-wise descent (Friedman et al. 2007) to solve for the coefficients in the synthesis step. Instantiations of this method are called *coordinate-descent* algorithms and solve the following alternative to the optimization in Equation 2:

$$\text{minimize}_A \; J(A|B) = \|X - AB\|_2^2 + \lambda\|A\|_1 \qquad (3)$$

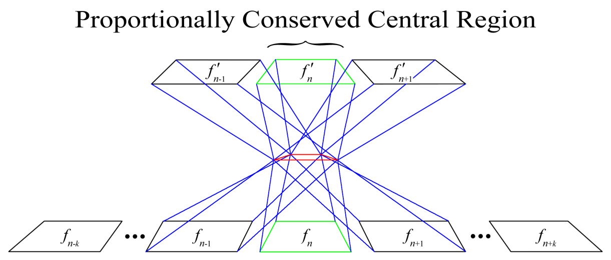
where $\|A\|_1 = \sum_{i,j} |A_{i,j}|$. In these experiments, we started with the basic SPARSENET algorithm and substituted a L_1-regularized coordinate-descent algorithm for the conjugate-gradient solver used by Olshausen and Field. We call the algorithm *LASSONET* in homage to SPARSENET and the acronym *LASSO* (for *Least Absolute Selection and Shrinkage Operator*), which has become a catchall term describing a class of methods for estimating least-squares parameters subject to an L_1 penalty. Convergence using coordinate descent is much faster than the original Olshausen and Field algorithm, but still relatively slow in working with large datasets and thousands of basis vectors.

Lee *et al* (Lee et al. 2007) describe a method for learning sparse codes that works by alternating between solving an L_1-regularized least-squares problem and an L_2-constrained least-squares problem. To solve the first they introduced the *feature-sign* algorithm based on the insight that once the signs of the coefficients are known, the problem reduces to a standard, unconstrained quadratic optimization problem, which can be solved efficiently. They *guess* the signs by performing line searches using a conjugate gradient solver. To solve the L_2-constrained least-squares problem, they reduce the number of optimization variables considerably by solving the Lagrange dual using Newton's method.

We ran experiments substituting the feature-sign and Lagrange-dual algorithms for the analysis and synthesis steps in LASSONET. Solving for the optimal basis vectors B given fixed coefficients A reduces to minimizing $\|X-AB\|_F^2$ — where $\|.\|_F$ is the Frobenius norm — subject to the constraint that $\sum_{i=1}^{L} B_{i,j}^2 < c$ for all $1 \leq i,j \leq N$ assuming N basis vectors. This constraint on the basis vectors was introduced for numerical stability and to avoid degenerate solutions. In our experience, incorporating the constraint directly into the objective function, as in the Lagrange-dual algorithm, is more robust than the adaptive methods used



(a)



(b)

Figure 1: Graphic (a) depicting a sample of four of the 2048 basis vectors learned by LASSONET in a trial that performed well on our activity-recognition task. RECURSIVE_LASSONET (b) consists of learning a sequence of models with each successive model having a larger temporal extent and an increased number of basis vectors.

in SPARSENET, and no additional normalization step is required to keep the activation variance approximately equal over all of the basis vectors.

Recursive Sparse Coding

We have considerable experience with learning sparse spatiotemporal codes using different combinations of coding algorithm, interest-point operators for extracting space-time volumes, and basis vectors of varying spatial and temporal extent. If the basis vectors are too large, learning is slow and the resulting codes tend to generalize poorly; if they are too small, the codes tend to be too general and they discriminate poorly. We are not the first to generate sparse, spatiotemporal codes from video; van Hateren and Ruderman (van Hateren and Ruderman 1998) apply independent components analysis to obtain codes resembling moving sinusoids windowed by Gaussian envelopes (Gabors), and Olshausen (Olshausen 2003) was able to compute convolution codes that exhibit similar characteristics by applying matching-pursuit in space and time. The best-performing codes obtained using LASSONET tend to look very different (see Figure 1.a) from those reported by Olshausen and van Hateren and Ruderman. Spectral analysis of the basis

[1]A basis vector is said to be *activated* with respect to a given cuboid if the associated coefficient in the linear combination of basis vectors reconstructing the cuboid is non-zero.

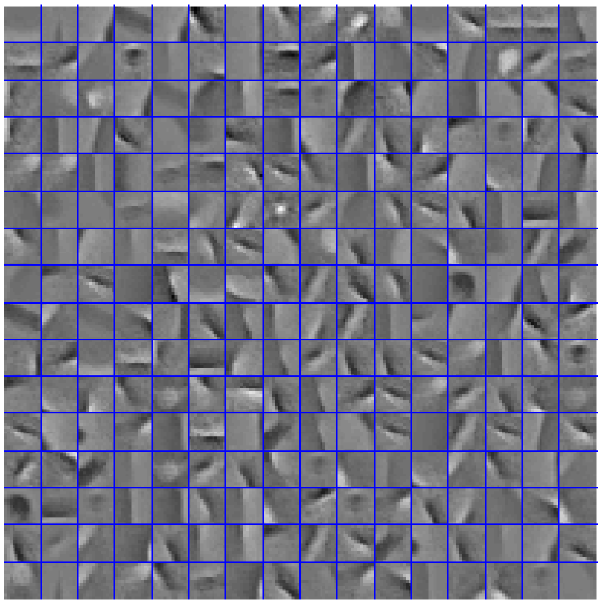


Figure 2: The center frames of the receptive fields of 256 out of 2048 basis vectors learned by RECURSIVE_LASSONET. The basis vectors capture a wide range of oriented filters undergoing diverse transformations.

vectors reveals an entirely appropriate distribution of velocities but hardly any spatial orientation — the individual frames resemble symmetric Gaussians rather than Gabors.

Using our algorithms and training on random 3-D patches for a considerable duration, we were able to obtain sparse, spatiotemporal codes that appeared roughly similar to those reported by Olshausen and van Hateren and Ruderman; they did not, however, perform particularly well on the activity recognition task. We noticed, however, that when we applied any of the three algorithms to learning a (temporally) degenerate code consisting of a single frame trained using 3-D cuboids extracted using the Dollár *et al* filter, we observed the usual variety of oriented, bandpass filters, but with the difference that due to the max-suppression step in extracting interest points a Gaussian envelope fit to the filters tended to be centrally located in the frame (see Figure 2). This suggested a simple iterative training method that learns a basis one frame at a time. The function *RECURSIVE_LASSONET* implements such a frame-by-frame scheme as follows:

- *base case* — we start a basis B_1 consisting of $|B_1|$ basis vectors with a temporal extent of one that code for a single frame; we use just the center frames from a sample of cuboids for training;
- *recursive step* — given the basis B_{n-1} which consists of $|B_{n-1}|$ vectors, we create a new basis with $|B_n| > |B_{n-1}|$ basis vectors; the default method uses a simple exponential growth model $|B_n| = |B_{n-1}|^G$, where $G > 1$ is the exponential growth factor;
- *temporal expansion* — each basis vector v in B_n is constructed from a randomly selected basis vector u in B_{n-1}; we increase the temporal extent of u to span $2n-1$ frames by adding two new frames to those of u with randomly initialized weights that *sandwich* the frames of u; the weights from u that reside within this sandwich are called the *conserved region* of v;
- *proportional weight conservation* — apply the chosen sparse-coding algorithm to adjust the new weights with the following twist; in the analysis step, hold the weights in the conserved region constant, but then prior to the synthesis step, rescale all of the basis vectors — including the weights in the conserved region — to have unit magnitude.

In a somewhat more sophisticated variant, we use coefficients obtained from fitting the current basis to the training data as pseudo counts to construct a proposal distribution for generating the basis vectors in the temporal expansion step.

Experiments

For exploratory experiments, we used the facial-expression dataset described by Dollár *et al* (Dollár et al. 2005). Our initial experiments in learning to recognize human activity were performed using the Weizmann human-action dataset described by Gorelick *et al* (Gorelick et al. 2007). For comparing the performance of our features against other published results, we used the KTH human-activity dataset described by Schuldt *et al* (Schuldt, Laptev, and Caputo 2004). Each dataset is divided into two or more disjoint subsets of video clips for testing, and each clip is assigned a *label* that characterizes its associated activity. The facial-expression dataset consists of 192 clips, 5 expressions and 4 subsets divided according to subject and lighting conditions. The Weizmann dataset consists of 93 clips and 10 activities divided into two subsets featuring different subjects. The KTH dataset consists of 599 clips of 25 subjects performing six activities (walking, jogging, running, boxing, handwaving and hand-clapping) in four different scenarios (outdoors, outdoors with scale variation, outdoors with subjects wearing different clothes and indoors); the KTH dataset is divided into 25 subsets according to subject.

Methods

As prolog to testing, we learn a basis and then use it to generate a set of descriptors, one set of descriptors for each video clip. The following steps are performed in preparation for testing:

1. whiten and apply local contrast normalization to each frame of each video clip;
2. extract a sample of cuboids from each video clip using the method of Dollár *et al*;
3. using this sample as training data, learn a basis using LASSONET or RECURSIVE_LASSONET;
4. generate a descriptor for each cuboid corresponding to the coefficients inferred using the L_1-regularized least-squares solver used in the sparse-coding algorithms;
5. optionally, apply singular-value decomposition to a sample of descriptors to generate a set of principal components to reduce the dimensionality of the descriptors;

For testing, we adopt the leave-one-out (LOO) protocol which was used in the evaluation of the other methods with which we compare ours. In each round of LOO evaluation on the KTH dataset, we train on 24 of the 25 subsets and test on the remaining one. We run ten trials, where each trial is composed of 25 rounds such that for each disjoint subset s_j of video clips we perform the following steps:

1. use k-means to cluster the descriptors in the complement $C_j = \cup_{i \neq j} s_i$ of s_j returning K centroids indexed $1, ..., K$, where K is the number of so-called *visual words*;
2. for each cuboid in each clip find the centroid closest to the cuboid's descriptor and return the centroid's index;
3. for each clip, construct a vector of length K whose kth component is the number of cuboids in the clip that map to the kth centroid; this vector (histogram) is used to represent the clip as a *bag of visual words*;
4. to label a clip c in s_j with associated histogram h_c use the label of the clip c' in C_j whose histogram $h_{c'}$ is nearest h_c according to the χ^2 distance metric;
5. the *recognition error* is just the number of incorrectly labeled clips divided by the number of clips in s_j;

We use early stopping to avoid over fitting. Our goal is not to find a better classifier, but rather to find better features for classification. For this reason, we use a simple nearest-neighbor classifier for all of our comparisons.

Results

The following table compares the reported performance of several approaches to activity recognition on the KTH dataset using the LOO protocol with our best model obtained by training with LASSONET and using the feature-sign algorithm to compute descriptors:

Action	Dean *et al*, 2009	Schuldt *et al*, 2004	Dollár *et al*, 2005	Niebles *et al*, 2008	Wang and Li, 2009
Box	81	98	80	98	88
Clap	80	60	82	86	81
Wave	86	74	84	93	83
Jog	69	60	63	53	70
Run	89	55	73	88	73
Walk	81	84	89	82	91
Mean	81.1	71.8	78.5	83.3	81.0

Our results are most appropriately compared with Dollár *et al* (Dollár et al. 2005) and Wang and Li (Wang and Li 2009) since their work primarily concerns feature selection, and both our results and their results shown here are based on a 1-NN classifier. Schuldt *et al* (Schuldt, Laptev, and Caputo 2004) use the space-time interest points of Laptev and Lindeberg (Laptev and Lindeberg 2003) and support-vector machine rather than 1-NN for classification. Dollár *et al* (Dollár et al. 2005) use the same interest-point operator as we do — they introduced the operator in this paper — and a descriptor based on histograms of gradients in x, y and t analogous to Lowe's (Lowe 2004) 2-D descriptor. Wang and Li may gain some advantage from the fact that they crop the region of the video containing the person performing the activity. Niebles *et al* (Niebles, Wang, and Fei-Fei 2008) also use Dollár's features, and their results are included as an example of how more sophisticated classifiers can leverage spatiotemporal features.

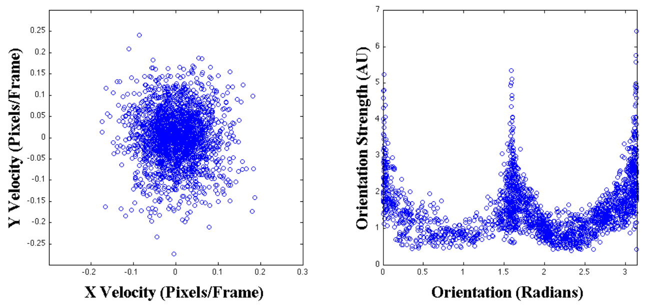


Figure 3: The left plot shows the distribution of velocities obtained from a spectral analysis of the basis vectors. The plot on the right approximates the distribution of orientations.

Wang and Li (Wang and Li 2009) is noteworthy as their *weighted sequence descriptor* takes into account the temporal ordering of features, which we currently do not in our simpler bag-of-visual-words model. The results for Dollár *et al* and Wang and Li are based on the use of a 1-NN classifier for comparison purposes. Niebles *et al* (**?**) is included as it also uses the Dollár *et al* features, but with more sophisticated, unsupervised learning.

We learn competitive models with as many 2048 $13 \times 13 \times 9$ basis vectors (80%) and as few as 128 $13 \times 13 \times 5$ vectors (79%). The larger bases can be reduced in dimension to produce descriptors constructed from between 100 and 200 visual words with no appreciable reduction in performance. We conclude that the intrinsic dimensionality of the best codes is around 100. The generality of the features learned in our approach is evident from the fact we can achieve good human activity classification even when we learn codes from data sets of a very different character. Following Cadieu and Olshausen (Cadieu and Olshausen 2008), we learned sparse codes using video from the BBC Motion Gallery — primarily clips of stampeding wildebeests and stalking leopards — and then applied these codes to achieve 80% accuracy on the KTH dataset of human behaviors. We also learned sparse codes using the considerably smaller Weizmann dataset and then used these codes to achieve respectable performance (78.5%) on the KTH dataset.

Since each recursive step generates a new model, RECURSIVE_LASSONET produces a sequence of models; each model in the sequence provides features that can be used for activity recognition. Here are the results of evaluating each model in such a sequence on the KTH dataset:

% Errors	0.189	0.202	0.231	0.252	0.289
Residual	0.056	0.062	0.066	0.073	0.065
% Zeros	94.13	92.13	91.21	90.73	92.76
# Vectors	2048	1024	512	256	128
Field Width	13	13	13	13	13
Field Depth	9	7	5	3	1

Each model in this particular sequence was produced by LASSONET using the feature-sign algorithm for the synthesis step, and the size of the basis was doubled in each recursive call starting with 128 basis vectors. Each model in the sequence was trained on 10,000 cuboids organized in ten epochs over 10 batches of 1,000 cuboids. For testing, we used 256 visual words and, where appropriate, 128 principal components for reducing the dimensionality of the descriptors. Each model was evaluated on 10 trials of 25 leave-one-out tests with the average recognition error shown in the first column. The sparsity parameter (λ in Equation 3) for the first model in the sequence was set to 0.90 and reduced by 0.10 for each subsequent model reflecting our experience in avoiding an ill-conditioned Hessian in computing the Lagrange-dual algorithm. These parameter settings were chosen to demonstrate that the method does not require careful tuning to produce sparse codes that look good and perform well.

Figure 3 summarizes the spectral analysis of the basis vectors in terms of velocities and orientations. It seems reasonable to conjecture that this approach works as well as it does in part for the same reasons that greedy layer-by-layer learning works in so-called deep networks (Hinton and Salakhutdinov 2006; Bengio et al. 2007). Weights trained early in the process continue to do a good job of reconstructing frames in the core of the basis vector, while gradient adjustments concentrate on the outermost frames added in the temporal expansion step. Vectors that don't serve to represent new aspects of the data can be weeded out by analyzing the degree to which they are activated during reconstruction.

Acknowledgments Piotr Dollár's Computer Vision and Image Processing Toolkit was enormously useful, and we are also grateful to him for allowing us to use his code for activity recognition. Thanks to Charles Cadieu and Pierre Garrigues for their suggestion on solvers for sparse coding. Anand Madhavan and Rajat Raina working with Andrew Ng let us experiment with their GPU-based algorithms for sparse coding. Rajat provided timely feedback on an early draft of this paper and useful advice guiding our use of graphics processors to speed learning.

References

Bengio, Y.; Lamblin, P.; Popovici, D.; and Larochelle, H. 2007. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press. 153–160.

Brady, N., and Field, D. J. 2000. Local contrast in natural images: normalisation and coding efficiency. *Perception* 29(9):1041–1055.

Cadieu, C., and Olshausen, B. 2008. Learning transformational invariants from time-varying natural images. In Schuurmans, D., and Bengio, Y., eds., *Advances in Neural Information Processing Systems 21*. Cambridge, MA: MIT Press.

Dollár, P.; Rabaud, V.; Cottrell, G.; and Belongie, S. 2005. Behavior recognition via sparse spatio-temporal features. In *Second Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 65.

Friedman, J.; Hastie, T.; Höfling, H.; and Tibshirani, R. 2007. Pathwise coordinate optimization. *Annals of Applied Statistics* 1(2):302–332.

Gorelick, L.; Blank, M.; Shechtman, E.; Irani, M.; and Basri, R. 2007. Actions as space-time shapes. *Transactions on Pattern Analysis and Machine Intelligence* 29(12):2247–2253.

Harris, C., and Stephens, M. 1988. A combined corner and edge detector. In *Alvey Vision Conference*, 147–152.

Hinton, G., and Salakhutdinov, R. 2006. Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507.

Hyvärinen, A.; Hurri, J.; and Väyrynen, J. 2003. Bubbles: a unifying framework for low-level statistical properties of natural image sequences. *Journal of the Optical Society of America* 20(7):1237–1252.

Laptev, I., and Lindeberg, T. 2003. Space-time interest points. In *Proceedings of the ninth IEEE International Conference on Computer Vision*, volume 1, 432–439.

Lee, H.; Battle, A.; Raina, R.; and Ng, A. Y. 2007. Efficient sparse coding algorithms. In Schölkopf, B.; Platt, J.; and Hoffman, T., eds., *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press. 801–808.

Lowe, D. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60(2):91–110.

Mumford, D. 1994. Neuronal architectures for pattern-theoretic problems. In *Large Scale Neuronal Theories of the Brain*. MIT Press. 125–152.

Niebles, J.; Wang, H.; and Fei-Fei, L. 2008. Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision* 79(3):299–318.

Olshausen, B. A., and Field, D. J. 1996. Natural image statistics and efficient coding. *Computation in Neural Systems* 7(2):333–339.

Olshausen, B. A., and Field, D. J. 1997. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research* 37(23):3311–3325.

Olshausen, B. 2003. Learning sparse, overcomplete representations of time-varying natural images. In *Proceedings of the IEEE International Conference on Image Processing*, volume 1, 41–44. IEEE Computer Society.

Raina, R.; Madhavan, A.; and Ng, A. 2009. Large-scale deep unsupervised learning using graphics processors. In *Proceedings of the 25th Annual International Conference on Machine Learning*.

Schuldt, C.; Laptev, I.; and Caputo, B. 2004. Recognizing human actions: A local SVM approach. In *Proceedings of the International Conference on Pattern Recognition*. IEEE Computer Society.

van Hateren, J. H., and Ruderman, D. L. 1998. Independent component analysis of natural image sequences yields spatiotemporal filters similar to simple cells in primary visual cortex. *Proceedings Royal Society London B* 265:2315–2320.

Wang, Z., and Li, B. 2009. Human activity encoding and recognition using low-level visual features. In *Proceedings of the 21th International Joint Conference on Artificial Intelligence*.