Unsupervised learning of visual invariance with temporal coherence

Will Y. Zou
Dept. of Electrical Engineering
Stanford University
wzou@stanford.edu

Andrew Y. Ng
Computer Science Dept.
Stanford University
ang@stanford.edu

Kai Yu
NEC Laboratories
America, Inc.
kyu@sv.nec-labs.com

Abstract

Natural scenes in a video stream contain rich collections of visual transformations. In this paper, a generic neural network is built to learn visual invariance from videos in an unsupervised manner. We use temporal coherence to learn both visual transformations and features with complex invariances. Without fine-tuning with labels, our invariant features are superior for classifying objects in still images. The learned features out-perform features learned with sparsity in vision benchmarks Caltech-101, STL-10 and COIL-100.

1 Introduction

Our visual system is able to recognize objects in various viewing angles, orientations, deformations, scales and lighting conditions. This ability relies on robust visual signals or invariant representations. In deep neural networks, invariances can be learned purely from discriminative training which requires large amounts of labeled data. We illustrate that alternatively, visual invariance can be learned by exposing a learning system to motion and transformations in unlabeled videos.

In modeling statistics of natural images, sparsity plays an important role in unsupervised learning methods such as Sparse Coding [1], Independent Component Analysis [2] and Sparse Restricted Boltzmann Machines [3]. As sparse representations, features learned by these algorithms share properties with simple cell receptive fields in early visual system V1. Another plausible principle specific to learning visual invariance is temporal coherence [4, 5, 6, 7]. Retinal projections of objects move smoothly across time. It is likely high-level visual representations become slow-changing over time by associating low-level visual templates [5].

With temporal coherence, a single layer of modules learns features similar to simple cell receptive fields [8, 9]. This evidence is interesting but shows the limitations of using only one layer of learning modules. We focus on learning a second layer with temporal coherence. Our work is closely related to motion models [10, 11, 12]. The second layer neurons combines signals from the first layer (V1) with an over-complete set of neurons. Instead of hard-coding the second layer weights [10], we optimize for an additional objective so that second layer activations are coherent over time. Motion patterns are learned implicitly by combining signals from the V1 layer, and second layer neurons produce signals invariant to those transformations. In both layers of the network, we follow the classic complex cell model to group neurons in group size 2 with energy pooling. The result is that features are represented efficiently with non-redundant quadrature pairs [11, 12].

Much prior work has been devoted to learning invariant features using image sequences [13, 8, 14, 12]. Despite efforts, there lacks indicative evidence showing invariant features learned with coherent image sequences are competitive in recognition tasks. To address this deficit, we aim to show that features learned using temporal coherence can improve object recognition on still images. In this work, a two-layer model is trained on a closed set of gray-scale natural videos in an unsupervised manner. The learned features are transferred to object recognition tasks using a linear classifier on

computer vision benchmarks. With our model, recognition performance increase consistently with temporal coherence. With cross-validation, the increase in accuracy by using video across three tested datasets is 4-5%. In particular, we achieve the best recognition accuracy to date (61%) on STL-10 dataset using video for unsupervised training. To validate the attribution of our results to invariance, we visualize and analyze second layer invariances.

2 Prior work

There has been long standing interest in unsupervised algorithms to learn visual invariance from image sequences. Foldiak [13] described a Hebbian learning rule for learning invariance to translation. Hyvarinen et. al. proposed the general 'Bubbles' framework [8] to encourage coherence both across features and across time. Hadsell et. al. [15] proposed a method to learn invariant mapping using convolutional networks. Slow Feature Analysis (SFA) [7] solves a constrained problem and optimize for temporal coherence by mapping data into quadratic expansion and performing eigenvector decomposition. Hashimoto [16] proposes a similar algorithm, but replaced L2 penalty in SFA with L1 penalty. Further, [14] applies SFA to extract object identity, position and rotation angles on an artificial dataset. Despite its elegance, SFA requires non-linear (quadratic) expansion and suffers from curse of dimensionality. There hasn't been clear successful application of SFA on computer vision benchmarks.

Closely related are unsupervised learning algorithms to learn visual invariance by factoring out 'what' and 'where' parts of representations. Hinton et. al. [17] formulated an auto-encoding neural network architecture that separates pathways for feature identity and feature position. Koster et. al. [18] introduced a horizontal product model based on Independent Component Analysis. [19, 20] proposed bi-linear models which represents natural images using a factorial code. Cadieu et. al. [12] modeled image transformations using interpolation between pooled features in quadrature pairs.

In computer vision, relatively few attempts have been made to improve vision features with video. Only recently, Stavens et. al. [21] show improvement in performance when SIFT/HOG parameters are optimized using tracked image patch sequences in specific application domains. In comparison, more work has been done to improve discriminative tasks such recognition or detection using videos as 'weakly supervised' signals. Leistner et. al. [22] used natural videos to improve random forest classifiers; Mobahi et. al. [23] used coherence in video to regularize a convolutional neural network and improve supervised object recognition task on COIL-100 dataset. It is worth noting that in [21], [22] and [23], a similar cost function or regularization technique is used in the discriminative task to enforce coherence between features in consecutive frames, and variation between features far-apart in time.

Prior work on unsupervised learning invariances with temporal coherence lack convincing application. In computer vision, there is need for unsupervised learning of complex invariances from temporal data. In this work, we illustrate a scalable feed-forward model that is able to learn visual invariances and applicable to real-world problems.

3 First layer algorithm

3.1 Linear Auto-encoder with Energy Pooling

We start by describing the algorithm for our basic learning module, based on the auto-encoder, an unsupervised learning architecture used to pre-train deep networks [24]. To learn features from data sample $x^{(i)}$, the conventional auto-encoder attempts to reconstruct the data by minimizing the following cost function:

$$\sum_{i=1}^{N} \|\mathbf{x}^{(i)} - W_2 g(W_1 \mathbf{x}^{(i)} + \mathbf{b}_1) + \mathbf{b}_2\|^2 + S(W_1 \mathbf{x}^{(i)} + \mathbf{b}_1) \tag{1}$$

g is a non-linear function such as sigmoid $\frac{1}{1+e^{-x}}$. S is usually a regularization function. To simplify the formulation, we use linear activation, no biases and tied weights ($W_2 = W_1^T$). Further, an energy pooling with group size of two is used. The cost function for the unsupervised learning algorithm is:

$$J = \sum_{t=1}^{N} \|\mathbf{x}^{(t)} - W^T W \mathbf{x}^{(t)}\|_2^2 + \lambda \sum_{t=1}^{N} \|\sqrt{H(W\mathbf{x}^{(t)})^2}\|_1 \tag{2}$$

In this equation, square and square-root operations are element-wise. The index t denotes data samples. H is a subspace-pooling matrix with group size of two [25]. More specifically, each row of H picks and sums tow neighboring feature dimensions in a non-overlapping fashion. The last term regularizes for sparsity in the pooling units. This auto-encoding algorithm has close ties with Independent Subspace Analysis (ISA) [25] and has the advantage of being able to learn over-complete hidden representations. Further, compared to ISA, the algorithm solves an unconstrained optimization problem.

To add temporal coherence, we give index t notion of time and add a simple L1-cost on temporal difference on the pooling units:

$$J = \sum_{t=1}^{N} \|\mathbf{x}^{(t)} - W^T W \mathbf{x}^{(t)}\|_2^2 + \lambda \sum_{t=1}^{N} \|\mathbf{p}^{(t)}\|_1 + \gamma \sum_{t=1}^{N-1} \|\mathbf{p}^{(t)} - \mathbf{p}^{(t+1)}\|_1 \tag{3}$$

The activations of pooling units are denoted as $\mathbf{p}^{(t)} = \sqrt{H(W\mathbf{x}^{(t)})^2}$.

When applied to natural images, the auto-encoder learns gabor-like features localized in orientation, space and frequency. An interesting property of this formulation is it is possible to learn a quadrature pair of gabor-like features when temporal coherence cost is added in the third term.

We visualize changes induced in the feature by the angle interpolating between two pooled neurons. Take two rows $\mathbf{w_1}$, $\mathbf{w_2}$ of matrix W which are in one group size, and denote each element $w_1^{(i)}$ in $\mathbf{w_1}$, $w_2^{(i)}$ in $\mathbf{w_2}$. For each angle value $\theta \in [0, 360]$ construct elements of interpolation $\hat{\mathbf{w}}$ by $\hat{w}^{(i)} = \cos(\theta)\sqrt{(w_1^{(i)})^2 + (w_2^{(i)})^2}$. In Figure 1 from left to right, we visualize $\hat{\mathbf{w}}$ by incrementing the angle θ by 45 degrees. The increment corresponds to translations in the localized Gaussian envelope in the gabor-like feature. In Figure 1, the translation is to the top-left, orthogonal to the feature orientation. Since the pooling units are invariant to the interpolation angle, we may say that the first layer units are invariant to translations in the local feature.

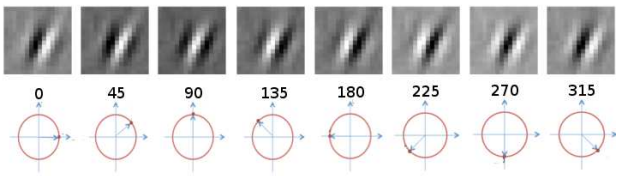

Figure 1: Visualization of change in one feature from the auto encoder by interpolation angle

3.2 Complex number interpretation

An interpretation with complex representations can be given to the above linear auto-encoder. Instead of having real number weights, we allow complex weight matrix and complex hidden activations, the auto-encoder objective would become (without energy pooling):

$$J = \sum_{i=1}^{N} \|\mathbf{x}^{(i)} - real(W_c^\dagger W_c \mathbf{x}^{(i)})\|_2^2 + \lambda \sum_{i=1}^{N} \|W_c \mathbf{x}^{(i)}\|_1 \tag{4}$$

$W_c \in \mathbf{C}^{m \times n}$ is a complex matrix. Note the hidden activations are decoded by a complex matrix-vector product $W_c^\dagger z$, where $W_c^\dagger$ is Hermitian of W_c. To form the objective function, we take the **real part** to calculate the reconstruction error between the decoded $\hat{x}$ and data x, and add a sparsity regularization term. The L1-norm indicates taking the L1-norm of the real vector obtained by taking complex norm in each element.

The objective in Equation 4 is equivalent to objective in Equation 2. In this respect, our algorithm has connections with the complex number formulation of sparse coding in Cadieu et. al. [12]. The visualization in Figure 1 can be seen as varying the phase of a complex feature. Later in the next section, we show that our model is able to produce similar results as Cadieu et. al. [12] to learn image transformations. For Figure 1, reviewers are encouraged to see a video visualization *layer1_motion.mov* [1] with multiple features.

4 Learning the second layer

4.1 Stack up to learn complex visual invariances

Each pooling unit on the first layer is only invariant to local translations within the local envelope. From natural videos, we expect to learn complex invariances such as rotational invariance, translational invariance across large scales, and perhaps invariance to out-of-plane rotation.

To obtain high-level invariance and implicitly model motion, second layer units should be able to see continuous switching between first layer features. More specifically, when a small object part moves coherently over time in video sequence, there should be one feature which describes this small part in frame 1, and another feature which describes the same part at a different location and pose in frame 2. In this manner, when the temporal coherence constraint is applied, higher layer units are able to associate the two abovementioned features across time thus learn from motion patterns. This observation is in accordance with models for the MT [10]. Therefore, the higher layer requires a highly over-complete set of first layer outputs. Also, the second layer feature should try to model larger-scale features than the first layer. We use convolution to replicate first layer neurons at multiple grid locations, and expand the receptive field of second layer neurons.

Figure 2 illustrates the second layer architecture. Once the first layer network with m hidden units are trained, convolution is performed on a larger image patch to obtain m response maps. The second layer, with a larger visual field in the original image space, is built on top of the all the convolutional response maps of the first layer neurons. We apply PCA with whitening to this high-dimensional input data and reduce to acceptable dimensions for the learning algorithm. This is similar to Le et. al [26] for learning high-dimensional spatial-temporal features. We use the auto-encoder algorithm described in the last section to learn the second layer. The only difference is a much stronger temporal coherence cost is placed on the pooling units. In the coming sections, we illustrate that temporal coherence cost is important to learning a meaningful second layer.

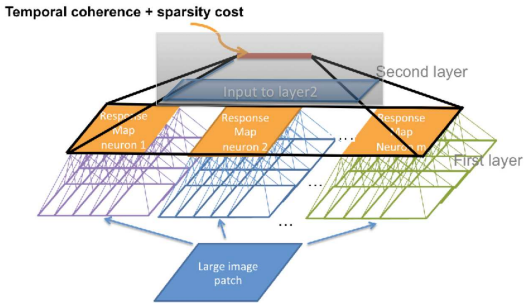

Figure 2: Illustration for learning the second layer on convolutional response map of layer 1 norm units.

4.2 Visualization of invariances

Similar to the first layer, second layer invariance can be visualized by adding increments to interpolation angle in each pooling group and observe the resulting transformations. To visualize changes that correspond to increasing the interpolation angle, second layer weight vectors are multiplied by the corresponding norm, projected onto the larger receptive field. By incrementing the interpolation angle in each element of the weight vector, we can visualize a motion sequence to which the second layer norm unit is invariant to. This visualization is shown on the right of Figure 3. Each row

[1] http://ai.stanford.edu/~wzou/avi/layer1_motion.mov

corresponds to one second layer feature. The video of these invariance visualization are supplied in video *layer2_motion.mov* [2].

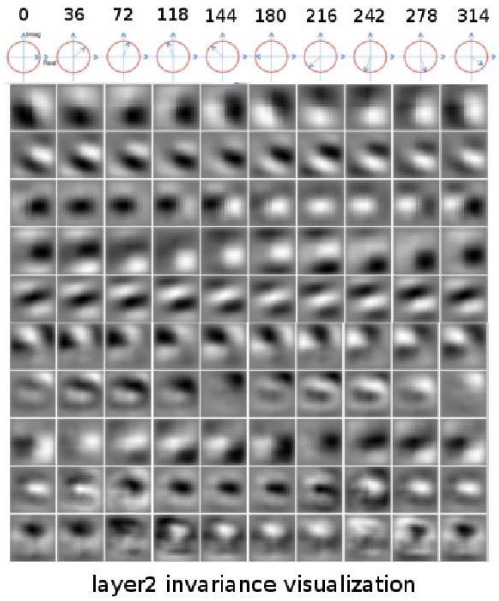

Figure 3: Visualization of layer 2 features (patch size 32x32): We can observe complex invariances with a number of variations in the second layer, such as different kinds of warping (rows 9 and 10), rotation (first row), local non-affine changes (rows 3, 4, 6, 7), large scale translations (rows 2 and 5).

4.3 Tracking

Of these learned invariances, the least interesting is translation. First, translation are the most common transformations in patch sequences extracted from videos; second, it is easy to hand-craft translation invariance with convolution and pooling. Instead of learning on video blocks cut out from videos, we attempt to learn our features with tracked sequences. The difference is illustrated in Figure 4. Tracking reduces translation invariance learned in the second layer. Instead, the features encode more interesting invariance such as warping. In object recognition tasks, we track our training data. This increases performance and makes the our method more stable against changes in hyper-parameters.

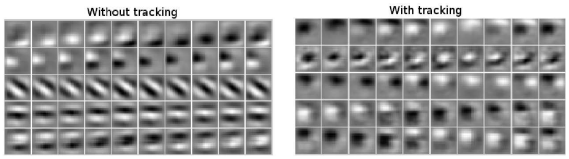

Figure 4: Comparison of layer 2 invariances learned with data obtained with tracking and without tracking.

4.4 Importance of temporal coherence for layer 2

Figure 5 shows different second layer features visualized using methods described in past sections. On the left are second layer features learned using only sparsity regularization. On the right are features learned using a strong temporal coherence cost. From this illustration together with classification results in Section 5, we observe that temporal coherence is key to learning a meaningful second layer in our model.

[2] http://ai.stanford.edu/~wzou/avi/layer2_motion.mov

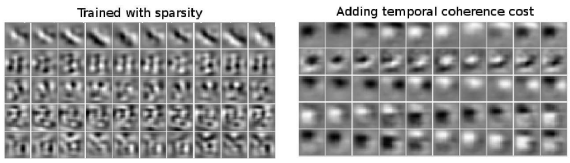

Figure 5: Comparison of layer 2 invariances learned with only sparsity, and adding temporal coherence cost

4.5 Learning motion and transformations

As in [12], we take the difference in interpolation angle (or complex phase) across time in the first layer units as representation for motion, and apply an unsupervised learning algorithm. In this experiment, we used the van Hateren video dataset as training data.

PCA, Sparse Coding, and the same auto-encoder were applied on the phase difference across time. All algorithms returned similar qualitative results and able to learn bases vectors that correspond to meaningful image transformations. To visualize, we take the complex coefficients of any image patch, and take the dot-product of this complex vector with a complex vector $\mathbf{c}$. $\mathbf{c}$ is constructed with unit norm and increment the phase of each element in proportion to learned component directions; the image patch for each increment step is then reconstructed to form a motion sequence corresponding to that second layer feature. Reviewers are encouraged to view this in video (*pca_transformations.mov*[3]). The video shows 10 first principle components of motion. Our model is able to pick up translation, and warping (local rotation, or non-linear affine transformations) similar to [12].

5 Object Recognition Experiments

In our experiments, we motivate self-taught and transfer learning from unlabeled data by learning features unsupervised on high-quality natural scene videos, and evaluate these features with classification on object recognition datasets. We only use gray-scale features in the experiments. For unsupervised learning of features, we use van Hateren video dataset of natural scenes. COIL-100 is first experimented to illustrate that the learned invariances work in concept to recognize the same object from different perspectives. Next we run classification experiments on realistic object recognition datasets STL-10 and Caltech 101 with the same features.

5.1 Training

Tracked image patch sequences (patch size 16x16 and 32x32) were extracted from the van Hateren video dataset. First layer is learned on 16x16 patches with 128 features (pooled with a group size of 2 from 256 linear bases). After pre-training the first layer, the bases are convolved with in the larger 32x32 image patches with a stride of 2 to obtain 128 response maps (each 9x9). The second layer is trained on all these response maps. In practice, PCA is used to first reduce the dimensions of the response maps to 300 before learning the second auto-encoder layer. The second layer learns 150 features (pooled from 300 bases). 16x16 patch size for the first layer is chosen similar to typical vision features (such as SIFT). For layer2 we expand the patch size to 32x32, twice layer 1 patch size to learn features at a significantly larger scale. The number of features on each layer and the convolution stride are chosen without much tuning to meet computational and memory requirements in training.

5.2 Feature extraction

To obtain first layer features for an image patch of size 16x16 (vectorized as a column vector x), perform a feed-forward operation on data x through the trained network: (the square and square-root operations are element-wise, H is a group size 2 pooling matrix)

$$f_1 = \sqrt{\epsilon + H(W_1 x)^2} \tag{5}$$

[3] http://ai.stanford.edu/~wzou/avi/pca_transformations.mov

To obtain second layer features for an image patch of size 32x32: perform convolution with the first layer network on 16x16 sub-images with a stride of 2, combine all response maps in a vector z. Obtain feed-forward activations of the network for second layer features:

$$f_2 = \sqrt{\epsilon + H(W_2 z)^2} \tag{6}$$

For large images, features are extracted using convolution: the first layer features are extracted by sampling the image in strides of 2, and second layer features extracted in the same manner on layer 1 response maps. The recognition architecture is similar to [27]. We use a two-layer feature extraction step, contrast-normalize, average pool the features and apply a standard classifier on the concatenation of both first and second layer features. This method is simpler than supervised fine-tuning a conventional convolutional network [23]. The feature extraction stage is illustrated in Figure 6 for a 72 by 72 image.

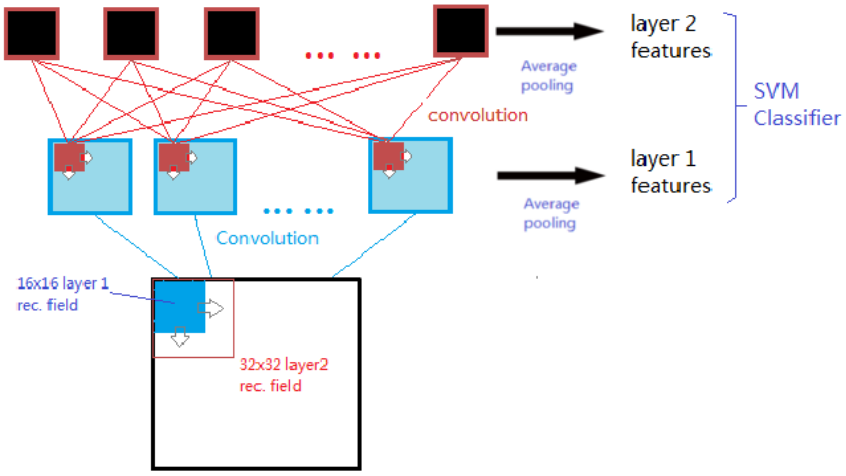

Figure 6: Feature extraction using a 2-layer model

5.3 Datasets

We evaluate our features learned from video using 3 publicly available datasets: COIL-100, Caltech 101 and STL-10.

COIL-100 contains 100 objects placed on turntables. Each object has 72 views. In our experiments we followed testing protocols in [28] and use 0, 90, 180, 270 degree perspectives of objects as supervised training set, and the rest as test set. Since we follow the protocol of training only on related videos, the classification experiment is performed on all 100 objects.

Caltech 101 contains images of 100 different objects and 1 background class. Our experiments were performed following the original protocol (30 objects per class) by [29]: we take 5 images per class as validation set, randomly pick 30 images per class (from the rest) for training, and randomly pick 50 per class (from the rest) as test set. This is performed 10 times and we report the average classification accuracy.

STL-10 is a newly proposed dataset by Coates et. al. [30] for developing algorithms for unsupervised feature learning and self-taught learning. There are 10 pre-defined folds of training images, 500 in each fold. The experiment protocol requires that in each fold, a classifier is trained on the pre-defined 500 training images, and tested on all 8000 images in the test set. Reported accuracy is averaged across 10 folds. Since the number of supervised training labels are small, this procedure tests the discriminative power of unsupervised learned features.

5.4 Results

We report the results on COIL-100, Caltech 101 and STL-10 dataset in tables 1, 2 and 3 respectively.

Table 1: Results on COIL-100 dataset trained with unrelated video

VTU [28]	79.1%
ConvNet regularized with video [23]	79.77%
Our results without video	82.0%
Our results using video	**87.0%**

Table 2: Results on Caltech 101 dataset

Two-layer ConvNet [31]	66.9%
ScSPM [32]	73.2%
Hierarchical sparse-coding [33]	74.0%
Our results without video	66.5%
Our results using video	**74.6%**

In all three datasets, the hyper-parameters (temporal coherence cost, sparsity cost multipliers) are cross-validated. Despite the features are transferred from completely unrelated videos, training with temporal coherence offers a consistent and significant 4-5% increase in recognition accuracy, bringing our results to state-of-the-art and competitive levels.

6 Conclusion

We have shown an unsupervised learning architecture with feed-forward neural network to learn invariant features from video using the temporal coherence principle. The model is able to learn hierarchically invariant features by convolution and stacking to two layers. We show that learning from videos, the newly proposed features achieve competitive results on standard object recognition benchmarks. The learned features are invariant a collection of complex transformations. Using a feed-forward network, our features are easy to use and fast to extract from images.

References

[1] B. Olshausen and D. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 1996.

[2] A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley Interscience, 2001.

[3] H. Lee, C. Ekanadham, and A. Y. Ng. Sparse deep belief net model for visual area v2. In *NIPS*, 2007.

[4] N. Li and J.J. DiCarlo. Unsupervised natural experience rapidly alters invariant object representation in visual cortex. *Science*, 2008.

[5] D. Cox, P. Meier, N. Oertelt, and J. DiCarlo. ̀breaking' position-invariant object recognition. *Nature Neuroscience*, 2005.

[6] T. Masquelier and S.J. Thorpe. Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS Computational Biology*, 2007.

[7] P. Berkes and L. Wiskott. Slow feature analysis yields a rich repertoire of complex cell properties. *Journal of Vision*, 2005.

[8] A. Hyvarinen, J. Hurri, and J. Vayrynen. Bubbles: a unifying framework for low-level statistical properties of natural image sequences. *Optical Society of America*, 2003.

[9] J. Hurri and A. Hyvarinen. Temporal coherence, natural image sequences and the visual cortex. In *NIPS*, 2006.

Table 3: Results on STL-10 dataset

Sparse Coding features, Kenn's encoding [34]	56.0%
Sparse Coding features, sparse coding encoding [34]	59.0%
Our result without video	56.5%
Our result using video	**61.0%**

[10] N.C. Rust, V. Mante, E. P. Simoncelli, and J. A. Movshon. How mt cells analyze the motion of visual patterns. *Nature Neuroscience*, 2006.

[11] E.Eh. Adelson and J.R. Bergen. Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America*, 1985.

[12] C. Cadieu and B. Olshausen. Learning tranformational invariants from natural movies. In *NIPS*, 2009.

[13] P. Foldiak. Learning invariance from transformation sequences. *Neural Computation*, 1991.

[14] M. Franzius, N. Wilbert, and L. Wiskott. Invariant object recognition with slow feature analysis. In *ICANN*, 2008.

[15] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping, 2006.

[16] W. Hashimoto. Quadratic forms in natural images. *Network: Computation in Neural systems*, 2003.

[17] G.E. Hinton, A. Krizhevsky, and S. Wang. Transforming auto-encoders, 2011.

[18] M. Gutmann U. Koster, J. Lindgren and A. Hyvarinen. Learning natural image structure with a horiontal product model. In *Proc. Int. Conf. on Independent Component Analysis and Blind Source Separation*.

[19] B. Olshausen, C. Cadieu, J. Culpepper, and D.K. Warland. Bilinear models of natural images. In *Proc. SPIE 6492*, 2007.

[20] R. P. N. Rao D. B. Grimes. Bilinear sparse coding for invariant vision.

[21] S. Thrun D. Stavens. Unsupervised learning of invariant features using video. In *CVPR*, 2010.

[22] C. Leistner, M. Godec, S. Schulter, M. Werlberger, A. Saffari, and H. Bischof. Improving classifiers with unlabeled weakly-related videos. In *CVPR*, 2011.

[23] H. Mobahi, R. Collobert, and Jason Weston. Deep learning from temporal coherence in video. In *ICML*, 2009.

[24] G. E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 2006.

[25] A. Hyvarinen, J. Hurri, and P.O. Hoyer. *Natural Image Statistics*. Springer, 2009.

[26] 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 *CVPR*, 2011.

[27] M.A. Ranzato K. Jarrett, K. Kavukcuoglu and Y. LeCun. What is the best multi-stage architecture for object recognition? In *ICCV*, 2009.

[28] H. Wersing and E. Kröner. Learning optimized features for hierarchical models of invariant object recognition. *Neural Computation*, 2003.

[29] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories.

[30] A. Coates, H. Lee, and A. Ng. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS 14*, 2010.

[31] K. Kavukcuoglu, P. Sermanet, Y. Boureau, K. Gregor, M. Mathieu, and Y. LeCun. Learning convolutional feature hierarchies for visual recognition. In *NIPS*, 2010.

[32] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, 2009.

[33] K. Yu, Y. Lin, and J. Lafferty. Learning image representations from the pixel level via hierarchical sparse coding. In *CVPR*, 2011.

[34] A. Coates and A. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *ICML*, 2011.