Looking around the Backyard Helps to Recognize Faces and Digits

Honghao Shan   Garrison W. Cottrell
Department of Computer Science and Engineering
University of California, San Diego
(hshan,gary)@cs.ucsd.edu

Abstract

Human beings have the ability to learn to recognize a new visual category based on only one or few training examples. Part of this ability might come from the use of knowledge from previous visual experiences. We show that such knowledge can be expressed as a set of "universal" visual features, which are learned from randomly collected natural scene images. Using these visual features, we have obtained state-of-the-art performance on several classification tasks using a single-layer classifier.

1. Introduction

One notable difference between the human visual system and most computer vision systems is that we human beings can learn a new visual category based on a few or even one example of the new category, while most computer systems require a large number of training samples to work reasonably well. For example, we can recognize the euro symbol (shown in Figure 1) under a wide variety of visual conditions after having seen a single example. Such an ability is necessary for the wide application of computer vision, as collecting training examples is generally expensive. In fact, due to the curse of dimensionality, almost all real-world pattern recognition problems can be regarded as problems of learning with few examples.

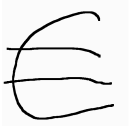

Figure 1. A handwritten euro (the European currency) symbol.

The problem of learning with scarce training samples has been addressed in different ways. We find two of these approaches related to our approach. One approach, namely semi-supervised learning [6], deals with the situation when labeled training samples are scarce by utilizing information from unlabeled samples. This approach reduces the amount of human intervention necessary to build a working system, since unlabeled data are generally plentiful and cheap to collect. It does not help, however, when unlabeled samples are also rare or expensive to collect, whereas human beings still exhibit reasonable performance in these cases. Miller *et al.* [15] adopted an alternative approach to address the problem of learning with few examples. They learned a set of transforms from handwritten English letters and then applied them to handwritten digits to regularize the input patterns, which allowed them to build a handwritten digit classifier using only several training examples per class. The limitations of this method, however, are that the forms of the transforms need to be predefined and that the visual knowledge can only be shared between highly similar visual categories.

These two types of techniques share one common characteristic – they both use knowledge obtained from a second information source to help building the classifier. They differ in how they define the knowledge and how the knowledge is learned and transferred, but they both pick a secondary information source which shares some common property with the labeled data, using either unlabeled data belonging to the same category as the labeled data, or labeled data belonging to a different but visually similar category, so that knowledge transfer is plausible.

The human visual system seems to adopt a similar approach. Low-level visual layers, such as retina, LGN (the lateral geniculate nucleus) and V1 (the primary visual cortex), are shared components that process all visual information we perceive. These layers develop and mature during childhood, and provide the basis for all the visual tasks encountered in the rest of life. If we loosely define the term "visual feature" as any function of the image pixel values, the above phenomenon can be interpreted as learning a set of *universal* visual features from the scenes encountered during childhood. These visual features are later applied to all visual stimuli and are used to perform a variety of visual tasks. Presumably these visual features provide one way to transfer the knowledge obtained from previous visual experiences and should help to build a classifier when labeled data are rare.

The idea of "universal visual features" might at first sight appear implausible, since this concept suggests that all visual stimuli share some characteristic in common such that knowledge obtained from one set of stimuli can be applied to a completely different set of visual stimuli. What is the common property shared by the appearance of your husband/wife's face and the view out your kitchen window that would allow you to better recognize the first by simply browsing the latter?

One observation is that they share similar local statistical structure. For example, if we take all of the $255,025$ 8×8 image patches from each of the two images shown on the top row of Figure 2, subtract the local mean from each image patch, and apply PCA to them, the resulting basis functions (i.e., eigenvectors) all resemble DCT filters. In fact, if we apply the PCA projection learned from image patches extracted from one image to the image patches from the other, and calculate the covariance matrix of the projected features, we will see that most off-diagonal cells have much smaller values than the diagonal cells. That is, although the projection matrix is calculated to capture the second-order local structure of one image, it also approximately captures the second-order local structure of the other image. The observation here is that, although the two images display very different visual content, they share very similar local/low-level statistical structure.

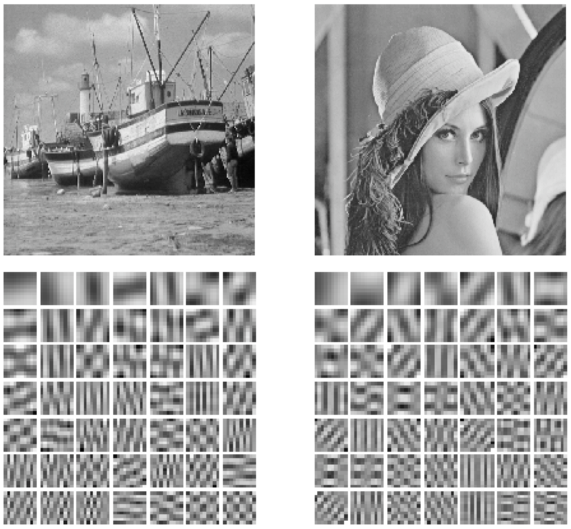

Figure 2. Applying PCA on image patches. For each 512×512 image on the top row, we sample $255,025$ 8×8 image patches and apply PCA to them. The top 49 eigenvectors are shown in the lower panels.

In this paper we apply the hypothesis of universal visual features to several classification tasks and achieve state of the art results. We begin by briefly reviewing neuroscience theories of low-level human vision in Section 2 from a computational perspective. Then we show in Section 3 how to learn a set of visual features from ten randomly collected natural images by simulating visual information processing up to the simple cells, the first layer of visual information processing in the cerebral cortex using ICA. The main difference between our approach and previous approaches is that we apply a non-linearity to the ICA features on a component-wise basis, that converts them from having a sparse distribution to having a gaussian distribution. We then apply PCA to that representation to reduce the dimensionality and feed the resulting vectors into a single layer classifier with a softmax activation function. Using this approach, we have achieved recognition performance comparable to recently proposed computer vision techniques on the Yale face dataset, the ORL face dataset, and the MNIST dataset. We then show through an example when the technique will seriously fail, as well as a solution to deal with such cases. We end the paper by discussing possible approaches to further improve this technique.

2. Theory of Low-level Human Vision

2.1. The Efficient Coding Hypothesis

What is the utility of unsupervised visual feature extraction from images? A similar question has been considered in the neuroscience community for years: what is the functional role of low-level visual layers in the human visual pathway which appear to receive little top-down (*i.e.* task driven) influence?

One hypothesis is that they capture the statistical structure of sensory inputs so that corresponding high-level decisions can be made accordingly (see [3] for a brief review). This hypothesis engenders two questions: (1) how to (quantitatively) define the statistical structure; and (2) how to capture the statistical structure. In 1954, Attneave [1] pointed out that whether we perceive structures in an image depends on how well we can predict a missing part of the image by its remaining parts. This insight suggests that we can use the dependency among the input features (*i.e.* dimensions), or the redundancy of the inputs, as a quantitative measurement of the statistical structure provided by the sensory inputs. Based on this observation, Barlow [2] hypothesized that one plausible way for a neural system to capture the statistical structure of its inputs was to remove their redundancy in its outputs, because to do so, the neural system must have a complete knowledge about statistical structures contained in its inputs. This hypothesis is later referred to as the redundancy reduction principle or the efficient coding principle. Although the name appears to suggest pursuing an economic coding, its essence is still focused upon capturing the statistical structure of the sensory inputs.

2.2. Linear Efficient Coding

Linear implementations of the efficient coding hypothesis, such as independent component analysis [5] and sparse coding [16], have been used to explain the functional role of simple cells in the primary visual cortex, the first layer of visual information processing in the cerebral cortex. These algorithms are best described by a generative model, in which the observed data $\vec{x} \in R^d$ is assumed to be generated by linearly mixing underlying signal source $\vec{s} \in R^D$:

$$\vec{x} = A\vec{s} + \vec{\epsilon} \tag{1}$$

where $A \in R^{d \times D}$ is the linear mixing matrix, $\vec{\epsilon} \in R^d$ denotes additive gaussian noises. The signal sources s_j's are assumed to be statistically independent, which incorporates the desire of capturing statistical structure in $\vec{x}$. For natural image statistics studies, a sparse marginal distribution for each s_j is assumed, which is characterized by a peak at zero and two heavy tails symmetrically residing on both sides of zero, such as the Student-t distribution or the Laplacian distribution. It was argued [16] that such a distribution incorporates the need to transfer more information with minimum energy cost, which is very important for a biological system.

There are two optimization problems associated with this model. One is to learn the most probable A given n observations $\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n$ (see [14] on the learning algorithm). Once the optimal A is learned, the inference problem is to infer the most probable signal source $\vec{s}$ given the mixing matrix A and an observation $\vec{x}$. When the marginal distribution is assumed to be a Laplacian distribution, the inference problem is a convex optimization problem and can be solved efficiently.

If we apply the linear efficient coding algorithms to natural image patches, the resulting basis functions (*i.e.*, columns of A) resemble simple cell receptive fields, as shown in Figure 4. It is widely observed [5, 13, 16] that the basis functions learned from different image datasets are qualitatively similar, which suggests even though these images display very different global contents they share similar local statistical structure.

2.3. Before Linear Efficient Coding

The visual information passes through retina and LGN before reaching the primary visual cortex. What happens there and why?

The classical theory about what happens before V1 is the whitening theory [9], which states that retina and LGN serve to flatten the magnitudes of the images on the frequency domain. It was observed that natural images approximately follow the $1/f$ law in the frequency domain. That is, if we apply 2D Fourier transform on a natural image, most of the time we will observe that the magnitude of each component decreases with increasing component frequency. Supposedly the visual system removes the redundant information by flattening the magnitudes on the frequency domains. Later it was pointed out that this operation also approximately makes pixel values uncorrelated [10]. In a recent paper [22] we also noted that this operation regulates the distribution of the inputs to V1 so that the linear efficient coding algorithm can work more efficiently. We showed that when ICA is treated as a generative model, the marginal distributions of the x_i's are zero-mean Gaussians. Hence the whitening process is "formatting" the inputs for the ICA procedure.

2.4. After Linear Efficient Coding

A further development in the [22] paper was that by applying a component-wise nonlinearity to the absolute value of the s_js (the sign is redundant), one can convert them into Gaussian distributions, and apply ICA again to obtain an efficient encoding of the first layer. This process may be repeated multiple times to obtain a multilayer ICA. We applied this algorithm to develop a two layer ICA, and the resulting second-layer basis functions appeared to code for texture boundaries and corners. In this paper, we stop at one layer, but apply the nonlinear function to the outputs. We have found this improves performance over simply using linear ICA.

The procedure for computing the nonlinearity is as follows: For each dimension of the layer-1 outputs s_j, the cdf $\mathcal{F}$ of $\|s_j\|$ is estimated. Then the coordinate-wise activation function G was defined as:

$$G(s_j) = \mathcal{G}(\mathcal{F}(\|s_j\|)) \tag{2}$$

where $\mathcal{G}$ denotes the inverse cdf function of a standard normal distribution. This coordinate-wise activation function works to discard the signs of layer-1 outputs, and then transform the marginal distributions to be Gaussian distributions.

Here is an intuitive explanation of the activation functions. The layer-1 basis function can be roughly considered as edge/bar detectors. Taking the absolute values of s_j introduces some shift invariance. Figure 3 plots one actual activation function learned from natural images. As shown in the figure, the activation function can be roughly divided into segments. Between zero and the red dots, the activation function amplifies the $\|s_j\|$ values that are highly peaked near zero from their small range on the x-axis to a large range on the y-axis. This segment serves to increase the distance between two $\|s_j\|$ values and may help the classifier to distinguish their differences. When the $\|s_j\|$ value is bigger than where the red dot indicates, few $\|s_j\|$'s are for any one image, and the activation function flattens the s_j's.

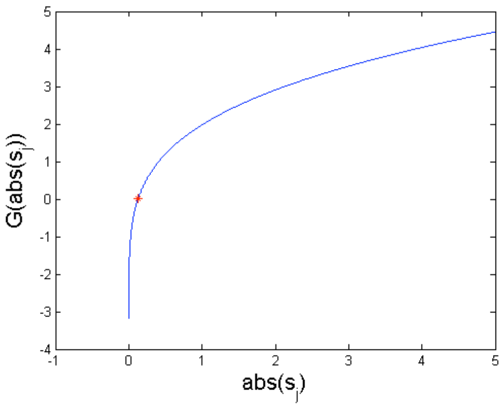

Figure 3. One actual activation function $\mathcal{G} \cdot \mathcal{F}$ learned from natural image patches. The red dot indicates where $\mathcal{F}(\|s_j\|) = 0.5$, *i.e.*, half of the s_j's responses in natural image patches are smaller than this value.

3. Methods

3.1. Learning Visual Features

We apply the sparse coding algorithm [17] on ten 512×512 natural images available from Olshausen's homepage. The images are whitened using the whitening filter described in [16], whose matlab code is also available from Olshausen's homepage. As discussed earlier, this whitening process is supposed to simulate the processing in retina and LGN. We normalize each image to have zero mean and unit variance. After whitening, six pixels off the boundary are discarded to avoid boundary effects. Each image is now 500×500 in size. We extract all the $2,430,490$ 8×8 possible image patches, subtract the local mean from each image patch, then calculate the PCA projection matrix. Now each image patch is represented as a 63 dimensional vector $\vec{x}$. While this does not reduce the dimensionality, it appears to make the ICA algorithm learn better features. After the PCA projection, we scale each dimension of $\vec{x}$ to have unit variance.

The sparse coding algorithm is applied on these image patches. It corresponds to the linear efficient coding model described in Equation 1, with the following marginal prior:

$$p(s_j) \propto \frac{1}{(1 + (s_j/\sigma))^{\beta}} \qquad (3)$$

In our experiments, we set $\sigma = 1$ and $\beta = 0.4$. We have found a relatively wide range of these values all work well. The variances of the noise $\vec{\epsilon}$ (see Equation 1) is also set to 1.

We initialize the linear mixing matrix A with gaussian random variables. Then, on each iteration, we randomly pick 100 image patches, and calculate their PCA projections $\vec{x}_1, \dots, \vec{x}_{100}$. The mixing matrix A is then updated by:

$$A = A + \frac{\eta}{100} \sum_{i}^{100} (\vec{x}_i - A\vec{s}_i)\vec{s}_i' \qquad (4)$$

where η denotes the learning rate, $\vec{s}_i$ is the most probable underlying signal given the observation $\vec{x}_i$ and the current mixing matrix A (see [17] for the inference algorithm). After each update, the columns of A are normalized to unit length to speed up the learning process. We repeat this process for $100,000$ iterations with $\eta = 0.1$, followed by another $100,000$ iterations with $\eta = 0.01$.

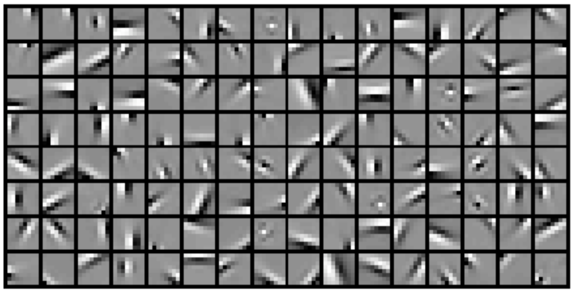

Figure 4. 128 basis functions (*i.e.* columns of A) learned on 8×8 natural image patches. See Section 3.1 for the learning procedure. Each columns of A is reconstructed to the original image space by reversing the PCA projection.

To evaluate the effect of overcompleteness (*i.e.*, the ratio between the dimensionality of $\vec{s}$ and the dimensionality of $\vec{x}$) on classification performance, we learn three different sets of features, with the dimensionality of $\vec{s}$ equal to 64, 128, or 192. Figure 4 displays the basis functions learned when the dimensionality of $\vec{s}$ is set to 128.

After that, we estimate the cdf function of $\|s_j\|$ for each dimension. This is done by first inferring the underlying signal representation $\vec{s}$ for all the $2,430,490$ extracted image patches. We calculate the histogram of $\|s_j\|$ with bins between 0 and 15 and a bin size of 0.0001. The empirical cdf function of $\|s_j\|$ is generated from this histogram, and then fitted by the following function:

$$\mathcal{F}_j(\|s_j\|) = \Gamma((\|s_j\|/\tau)^{\theta}, 1/\theta) \qquad (5)$$

where Γ denotes the incomplete Gamma function. Figure 5 displays the empirical cdf function of one $\|s_j\|$ as well as the fitted cdf function, when the dimensionality of $\vec{s}$ equals 128. This activation function converts the highly sparse distribution learned via ICA into a Gaussian distribution, which in previous work we have used to apply ICA again. However, in this work, we simply investigate the utility of this nonlinear function applied to one layer of ICA features.

After the linear mixing matrix A and the nonlinear activation functions $G_j = \mathcal{G}(\mathcal{F}_j)$ are learned from the ten natural images, we apply them on various classification tasks

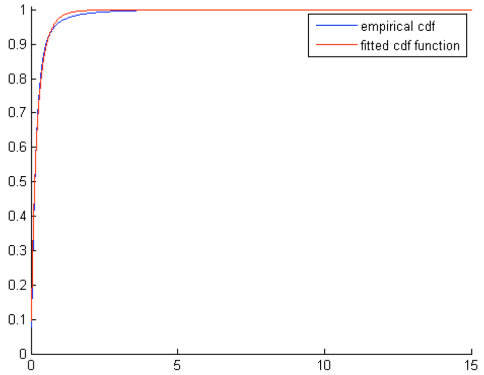

Figure 5. The empirical cdf curve of one $\|s_j\|$ as well as the fitted cdf function. The fitted parameters for this curve are $\theta = 0.7027$ and $\tau = 0.1119$.

without any adjustment of the parameters. The classification performance based on these features outperforms many recently proposed computer vision techniques, even if we just use a single layer classifier.

3.2. Experiments on Yale Face Dataset

We first test the features on the Yale dataset [4], which contains 165 gray-scale images of 15 individuals. Each individual has 11 images. The manually aligned and cropped images can be downloaded from the homepage of the first author of reference [7]. We downloaded the 64 × 64 processed images, and then downscaled the images to 32 × 32 using the imresize matlab function. This procedure does not suffer from the pixellation artifacts that occur in the 32×32 images provided on the website. Then we whitened each image using the whitening filter discussed in Section 3.1 and normalize each image to have zero mean and unit variance. For each image, we extract all the 625 8 × 8 image patches, and infer the most probable $\vec{s}$ for each image patch. After that, we apply the nonlinear activation G_j on each dimension of s_j. When the dimensionality of $\vec{s}$ is 64 (1 times overcomplete), each facial image is represented by a $625 \times 64 = 40,000$ dimensional vector. For 2 times overcompleteness, each image is represented by a $80,000$ dimensional vector, *etc*.

We followed the method in [7] to randomly divide the images into the training and the testing sets. For each experiment, we randomly select $M = 2, 3, \ldots, 8$ images from each individual as the training images, and use the rest as testing images. For each setting of M, we tested 50 random splits.

D	M=2	M=3	M=4	M=5	M=6	M=7	M=8
64	28.86	19.55	13.71	9.96	7.71	6.43	4.93
128	27.13	17.78	12.04	8.36	6.96	5.50	4.04
192	27.11	17.38	11.71	8.16	6.27	5.07	3.82
Ref [7]	42.4	27.7	22.2	18.3	-	-	-
Ref [12]	-	-	-	13.2	-	-	-
[7] update	37.5	25.5	19.3	14.7	12.3	10.3	8.7

Table 1. Recognition error rates (in percentage) on the Yale dataset, using $G(\|s\|)$ as features. D denotes the dimensionality of $\vec{s}$. M denotes the number of training images selected from each individual.

When the training set and the testing set are selected, we take the 40,000 dimensional representation (for 1 times overcomplete, *etc*.) and reduce its dimensionality using PCA. The number of principal components are chosen so that 95% of the variance is captured. For example, as M increases from 2 to 8, the number of principal components using this criterion range from 27 to 105. We then use the projected training examples to train a one layer network using the softmax activation function. The network is updated for $1,000$ epochs or until the network weights have converged, using the scaled conjugate gradient algorithm (the glm + netopt function in the netlib library implement this algorithm). The test images are projected using the PCA projection matrix trained on the training images, and fed to the one layer network.

Table 1 lists the recognition performance on the test images, with different overcompleteness and different numbers of training images from each individual. The results are extremely good given the small number of training examples. Results reported in CVPR2007 in [7] and [12] are given for comparison. We noticed that on the homepage of the first author of reference [7], they reported new results. These are given on the last line. However, there is still an obvious gap between their results and our results, and our results are based on first layer features not adapted to the training set.

3.3. Experiments on ORL Face Dataset

The Olivetti Research Laboratory (ORL) database[20] contains 400 face images of 40 persons, with 10 per person, taken at different time, under different lighting conditions, and with different facial expressions. We downloaded the manually aligned and cropped 64 × 64 images from the homepage of the first author of reference [7], and then downsampled them to 32 × 32 size using the imresize matlab function. We implemented the same experiments as on the Yale datasets. We randomly select $M = 2, 3, \ldots, 8$ images per person for training and the rest for testing. The average recognition error rates are reported in Table 2, with the results from two CVPR 2007 papers for comparison.

D	M=2	M=3	M=4	M=5	M=6	M=7	M=8
64	15.99	8.64	5.08	2.91	2.14	1.40	1.10
128	15.46	8.20	4.75	2.70	1.78	1.20	0.90
192	15.11	8.02	4.46	2.43	1.78	1.12	0.90
Ref [7]	14.8	7.7	4.2	2.8	-	-	
Ref[12]	-	-	-	3.0	-	-	

Table 2. Recognition error rates (in percentage) on the ORL dataset, using $G(\|s\|)$ as features. D denotes the dimensionality of $\vec{s}$. M denotes the number of training images selected from each individual. The last two rows are from CVPR 2007 papers

D	M=10	M=20	M=30	M=40	M=50
64	12.62	8.73	7.26	6.49	5.92
128	12.20	8.20	6.84	6.08	5.52
192	12.05	8.12	6.73	5.93	5.32
Invar-SVM	14.9	8.9	7.8	5.8	5.5

Table 3. Recognition error rates (in percentage) on the MNIST dataset, using $G(\|s\|)$ as features. D denotes the dimensionality of $\vec{s}$. M denotes the number of training images selected from each digit. The bottom row displays the results reported in [23].

3.4. Experiments on MNIST Digit Dataset

In this experiment, we apply our features derived from natural scene statistics to the MNIST handwritten digits dataset. The MNIST dataset has a training set of $60,000$ examples and a test set of $10,000$ examples. The digits have been size-normalized and centered in 28×28 images. We downsampled each image to 18×18 size by the imresize matlab function. Each image is whitened and normalized to have zero mean and unit variance. 121 8×8 image patches are extracted from each image for the one times overcomplete case, so each image is represented by a $121 * 64 = 7744$ dimensional vector. We also give the results for two and three times overcomplete.

We randomly select $M = 10, 20, 30, 40, 50$ training examples for each digit from the training dataset. For each M, we try 40 different sets of training examples. As with the experiments on face datasets, we apply PCA to reduce the dimensionality while retaining 95% of the variance, and use a one-layer softmax classifier. We test the classifier on the $10,000$ test images, and the average recognition error rates are reported in Table 3.

These results are comparable to recently proposed computer vision techniques. For example, in NIPS2007, the authors of [23] reported how to learn a set of kernels which are invariant to 20 transforms: 1-pixel and 2-pixel shift in 4 and 8 directions, rotations by ± 10 degrees, scaling by ± 0.15, and shearing in vertical or horizontal axis by ± 0.15. SVM based on these kernels yields results shown on the bottom row of Table 3. In CVPR2007, the authors of [19] trained a four-layer network to extract visual features. They used all the $60,000$ unlabeled training examples to train the network, and then use a subset of labeled training examples to train a two-layer neural network, using the visual features extracted by the four-layer network. They reported an average error rate of 7.18% on the test dataset, using just 300 training examples, as we do here. However, they achieved much better results when they used more training examples (less than a percent). It remains to be seen how well we can do with more training examples.

3.5. Why This Works

Our above experiments raise an interesting question: why should these visual features help classification tasks in general? We have noticed that when we apply the visual features to the images, we in fact map the image to a much higher dimensional space nonlinearly. The nonlinearity comes from two operations. First, given input $\vec{x}$, the optimal $\vec{s}$ is inferred by minimizing the reconstruction error $(\vec{x} - A\vec{s})^2$ while minimizing the sparsity penalty $\log p(\vec{s})$. As a result, $\vec{s}$ is a nonlinear function $\vec{x}$ when $D > d$. Instead of mapping to a linear subspace of the higher dimensional space, as a linear transform $W\vec{x}$ would do, the inference process spreads the optimal $\vec{s}$ in the high dimensional space. Another nonlinearity comes from the activation function $G_j(\|s_j\|)$ we apply on each dimension of $\vec{s}$. We hypothesize by mapping the data to a higher dimensional space, we can benefit from what kernel SVM's have benefited from – the samples are much more likely to be linearly separable. However, since the nonlinear functions are learned from natural images and capture some meaningful structure, we suffer little from the problem of over-fitting.

To evaluate the utility of the nonlinear activation function G, we repeat the experiments on the Yale dataset. The experiment settings are the same as described in Section 3.2, except that now we only keep the absolute values of $\vec{s}$ without applying the nonlinear activation functions. As shown in the table, the nonlinear activation functions make a significant difference in the final classification performance.

We also tried using Gabor magnitudes (3 scales and 8 orientations, followed by PCA), as in previous work [8] instead of our ICA features with the nonlinear activation function. The results were worse than Cai *et al.*'s results, and much worse than our results. Hence the ICA features, with the nonlinear activation function, are an important part of the model.

Another observation in all the tables is that generally the higher the dimensionality, the better the recognition performance. This accords with our hypothesis that we are benefitting from mapping the data to a higher dimensional space nonlinearly. Would this explain the observation that the pri-

D	M=2	M=3	M=4	M=5	M=6	M=7	M=8
64	36.92	26.63	19.37	15.24	12.77	11.00	10.22
128	36.00	25.72	18.65	13.82	11.87	9.87	9.02
192	34.00	23.78	16.72	12.49	11.04	8.37	8.31
64	28.86	19.55	13.71	9.96	7.71	6.43	4.93
128	27.13	17.78	12.04	8.36	6.96	5.50	4.04
192	27.11	17.38	11.71	8.16	6.27	5.07	3.82

Table 4. Recognition error rates (in percentage) on the Yale dataset, without using the nonlinear activation functions. The bottom three rows replicate the first three of Table 1 to ease comparison.

D	M=2	M=3	M=4	M=5	M=6	M=7	M=8
64	48.73	36.73	30.72	24.22	19.73	16.27	13.69
128	46.46	34.30	28.93	22.13	17.81	13.97	11.69
192	45.78	33.92	28.34	21.16	17.41	13.63	11.69

Table 5. Recognition error rates (in percentage) on the Yale dataset, using $G(\|s\|)$ as features. However, now we are directly using the 32×32 version images downloaded from the first author of [7].

mary visual cortex constitutes an about 100 times overcomplete representation of the sensory inputs? Will the performance stop improving or even start to degenerate as we keep increasing the dimensionality of $\vec{s}$? We leave these questions for future work.

3.6. When It Will Fail

As discussed at the beginning of this paper, the whole idea of the universal visual feature hypothesis is based on the observation that visually quite different images share similar local statistical structures. Hence, the method will greatly fail when the local statistics of the images under consideration are very different from the images on which the visual features are learned. This may occur when some artificial statistics are introduced during the image capture process. This is exactly why we always make the pixel values of each image to have zero mean and unit variance.

Here we show how the method will seriously fail. We download the processed 32×32 Yale face images from Ref [7]'s first author's homepage. And then apply the same procedure as described in Section 3.2. That is, everything is the same except that we are directly using the 32×32 version images, instead of using the 64×64 version followed by a downsampling operation. As shown in Table 5, the performance degenerates greatly.

It turns out that the software the authors used to downsample the images introduced pixellation effects, as shown in Figure 6. As the original images on which we have learned the visual features contains are smooth outdoor scenes, the pixellation errors make the images being tested to have quite different local statistical structure. This effect, however, can be easily counteracted by simply downscaling or blurring the images slightly. This is also why we down-scaled the handwritten digits from 28×28 to 18×18 size.

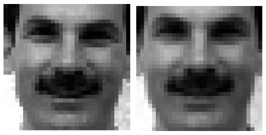

Figure 6. The pixellation effect. The left figure is the original 32×32 version. The right figure is generated by down-scaling the 64×64 image to 32×32 using the imresize matlab function.

This, however, raises an interesting question – how to avoid/remove artificial statistics introduced during image capture? For the human visual system, there is a unique set of image capturing devices. Any artificial statistics introduced by these devices will be applied to future testing images, and should not cause much trouble. However, for a computer vision system in which the images are captured using different devices, artificial statistics introduced during image capture might be a serious problem. Although in our experiments we found that simply shrinking/blurring the images will suffice to yield good performance, how to detect and remove artificial statistical structures is still an important research topic in the study of these universal visual features.

4. Discussion

In this work, we have shown how to learn a set of visual features from randomly collected images, and apply these features on different datasets without special adjustment of the features. These ideas are inspired by the structure of human visual system, and based on the observation that visually different images share similar local statistical structures. We show that these features help to yield recognition performance comparable to state-of-the-art computer vision techniques.

There is a great deal of recent work using deep networks for recognition [11, 21]. The work we report here suggests that shallow but wide networks may be sufficient for recognition. This observation is consistent with another recent article using shallow networks with a nonlinear activation function for object recognition [18].

However, as we have shown in Section 3.6, we need to apply this technique with caution. Although we showed in the paper that simply shrinking the images being tested will help to remove some artificial statistics, it remains an interesting question how to automatically detect and remove such structures.

We feel that the best way to interpret the linear efficient encoding hypothesis as instantiated by ICA is that it is trying to provide a set of universal visual features. This is plausible because globally different visual stimuli in fact share similar local statistical structure. This is beneficial because all we need to train the features is a set of natural images. These images do not need to be labeled, and apparently, they do not need to share similar properties, as evidenced by the application of our approach to hand written digits.

Acknowledgement

We thank the GURU members for valuable discussions. We thank Eric Christiansen for helping to prepare some experiments. This research was funded by the NSF (grant SBE-0542013 to the Temporal Dynamics of Learning Center) and the NIH (grant MH57075) to G.W.C.

References

[1] F. Attneave. Some informational aspects of visual perception. *Psychological Review*, 61(3):183–193, 1954.

[2] H. B. Barlow. Possible principles underlying the transformation of sensory messages. In W. A. Rosenblith, editor, *Sensory Communication*, pages 217–234. MIT Press, Cambridge, MA, USA, 1961.

[3] H. B. Barlow. Redundancy reduction revisited. *Network: Computation in Neural Systems*, 12:241–253, 2001.

[4] P. Belhumeur, J. Hespanha, and D. Kriegman. Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[5] A. J. Bell and T. J. Sejnowski. The 'independent components' of natural scenes are edge filters. *Vision Research*, 37(23):3327–3338, 1997.

[6] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pages 92–100, 1998.

[7] D. Cai, X. He, Y. Hu, J. Han, and T. Huang. Learning a spatially smooth subspace for face recognition. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[8] M. N. Dailey, G. W. Cottrell, C. Padgett, and R. Adolphs. EMPATH: A neural network that categorizes facial expressions. *Journal of Cognitive Neuroscience*, 14(8):1158–1173, 2002.

[9] D. J. Field. Relations between the statistics of natural images and the response properties of cortical cells. *Journal of the Optical Society of American, A*, 4(12):2379–2394, 1987.

[10] D. J. Field. What is the goal of sensory coding? *Neural Computation*, 6(4):559–601, 1994.

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

[12] G. Hua, P. A. Viola, and S. M. Drucker. Face recognition using discriminatively trained orthogonal rank one tensor projections. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[13] A. Hyvarinen and P. O. Hoyer. A two-layer sparse coding model learns simple and complex cell receptive fields and topography from natural images. *Vision Research*, 41(18):2413–2423, 2001.

[14] M. S. Lewicki and T. J. Sejnowski. Learning overcomplete representations. *Neural Computation*, 12(2):337–365, 2000.

[15] E. G. Miller, N. E. Matsakis, and P. A. Viola. Learning from one example through shared densities on transforms. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 464–471, 2000.

[16] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

[17] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, 37(23):3311–3325, 1997.

[18] N. Pinto, D. Cox, and J. DiCarlo. Why is real-world visual object recognition hard? *PLoS Computational Biology*, 4(1), 2008.

[19] M. Ranzato, F.-J. Huang, Y.-L. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[20] F. Samaria and A. Harter. Parametrization of a stochastic model for human face identification, in proc. of ieee workshop on allocations of computer vision. *Sarasota, Florida*, 1994.

[21] T. Serre, L. Wolf, and T. Poggio. Object recognition with features inspired by visual cortex. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 994–1000, 2005.

[22] H. Shan, L. Zhang, and G. W. Cottrell. Recursive ICA. In *Advances in Neural Information Processing Systems*, Cambridge, MA, USA, 2007. MIT Press.

[23] C. H. Teo, A. Globerson, S. Roweis, and A. Smola. Convex learning with invariances. In *Advances in Neural Information Processing Systems*, volume 20, 2008.