Learning Features by Contrasting Natural Images with Noise

Michael Gutmann[1] and Aapo Hyvärinen[1,2]

[1] Dept. of Computer Science and HIIT, University of Helsinki,
P.O. Box 68, FIN-00014 University of Helsinki, Finland
[2] Dept. of Mathematics and Statistics, University of Helsinki
{michael.gutmann,aapo.hyvarinen}@helsinki.fi

Abstract. Modeling the statistical structure of natural images is interesting for reasons related to neuroscience as well as engineering. Currently, this modeling relies heavily on generative probabilistic models. The estimation of such models is, however, difficult, especially when they consist of multiple layers. If the goal lies only in estimating the features, i.e. in pinpointing structure in natural images, one could also estimate instead a discriminative probabilistic model where multiple layers are more easily handled. For that purpose, we propose to estimate a classifier that can tell natural images apart from reference data which has been constructed to contain some known structure of natural images. The features of the classifier then reveal the interesting structure. Here, we use a classifier with one layer of features and reference data which contains the covariance-structure of natural images. We show that the features of the classifier are similar to those which are obtained from generative probabilistic models. Furthermore, we investigate the optimal shape of the nonlinearity that is used within the classifier.

Keywords: Natural image statistics, learning, features, classifier.

1 Introduction

Natural scenes are built up from several objects of various scales. As a consequence, pictures that are taken in such an environment, i.e. "natural images", are endowed with structure. There is interest in modeling structure of natural images for reasons that go from engineering considerations to sensory neuroscience, see e.g. [1,2].

A prominent approach to model natural images is to specify a generative probabilistic model. In this approach, the probabilistic model consists of a parameterized family of probability distributions. In non-overcomplete ICA, for example, where each realization of the natural images $\mathbf{x} \in \mathbb{R}^N$ can be written as unique superposition of some basic features $\mathbf{a}_i$,

$$\mathbf{x} = \sum_{i=1}^{N} \mathbf{a}_i s_i, \tag{1}$$

C. Alippi et al. (Eds.): ICANN 2009, Part II, LNCS 5769, pp. 623–632, 2009.

the parameters in the statistical model are given by the $\mathbf{a}_i$, see e.g. [2]. In overcomplete models, either more than N latent variables s_i are introduced or the parameterization of the probability distribution is changed such that the parameters are some feature vectors $\mathbf{w}_i$ onto which the natural image is projected (non-normalized models, see e.g. [3,4]). Estimation of the features, i.e the $\mathbf{a}_i$ or $\mathbf{w}_i$, yields then an estimate of the probability distribution of the natural images.

The interest in this approach is threefold: (1) The statistical model for natural images can be used as prior in work that involves Bayesian inference. (2) It can be used to artificially generate images that emulate natural images. (3) The features visualize structure in natural images.

However, the estimation of latent variables, or non-normalized models, pose great computational challenges [2]. If the main goal in the modeling is to find features, i.e. structure in natural images, an approach that circumvents this difficult estimation can be used: For the learning of distinguishing features in natural images, we propose to estimate instead a discriminative probabilistic model. In other words, we propose to estimate a classifier (a neural network) that can tell natural images apart from certain reference data. The trick is to choose the reference data such that it incorporates some known structure of natural images. Then, the classifier teases out structure that is not contained in the reference data, and makes in that way interesting structure of natural images visible. We call this approach contrastive feature learning.

This paper is structured as follows: In Section 2, we present the three parts of contrastive feature learning: the discriminative model, the estimation of the model, as well as the reference data. The discriminative model has one layer of features. It further relies on some nonlinear function $g(u)$. In Section 3, we first discuss some properties which a suitable nonlinearity should have. Then, we propose some candidates and go on with presenting learning rules to optimize the nonlinearity. Section 4 presents simulation results, and Section 5 concludes the paper.

2 Contrastive Feature Learning

2.1 The Model

Since we want to discriminate between natural images and reference data, we need a classifier $h(.)$ that maps the data $\mathbf{x}$ onto two classes: $C = 1$ if $\mathbf{x}$ is a natural image and $C = 0$ if it is reference data.

We choose a classification approach where we first estimate the regression function $r(\mathbf{x}) = E(C|\mathbf{x})$, which is here equal to the conditional probability $P(C = 1|\mathbf{x})$. Then, we classify the data based on Bayes classification rule, i.e. $h(\mathbf{x}) = 1$ if $r(\mathbf{x}) > 1/2$ and $h(\mathbf{x}) = 0$ if $r(\mathbf{x}) \leq 1/2$.

Our model for $r(\mathbf{x})$ is a nonlinear logistic regression function:

$$r(\mathbf{x}) = \frac{1}{1 + \exp(-y(\mathbf{x}))}, \tag{2}$$

where

$$y(\mathbf{x}) = \sum_{m=1}^{M} g(\mathbf{w}_m^T \mathbf{x} + b_m) + \gamma \tag{3}$$

for a suitable nonlinearity $g(u)$ (see Section 3), and where M is not necessarily related to the dimension N of the data $\mathbf{x}$.

In a neural network interpretation, $g(\mathbf{w}_m^T \mathbf{x} + b_m)$ is the output of node m in the first layer. The weights of the second layer are all fixed to one. The second layer pools thus the outputs of the first layer together and adds an offset γ, the result of which is $y(\mathbf{x})$. The network has only one output node which computes the probability $r(\mathbf{x})$ that $\mathbf{x}$ is a natural image.

Parameters in our model for $r(\mathbf{x})$ are the features $\mathbf{w}_m$, the bias terms b_m and the offset γ. The features $\mathbf{w}_m$ are the quantities of interest in this paper since they visualize structure that can be used to tell natural images apart from the reference data.

2.2 Cost Function to Estimate the Model

Given the data $\{\mathbf{x}_t, C_t\}_{t=1}^T$, where $C_t = 1$ if the t-th input data point $\mathbf{x}_t$ is a natural image and $C_t = 0$ if it is reference data, we estimate the parameters by maximization of the conditional likelihood $L(\mathbf{w}_m, b_m, \gamma)$. Given $\mathbf{x}_t$, class C_t is Bernoulli distributed so that

$$L(\mathbf{w}_m, b_m, \gamma) = \prod_{t=1}^{T} P(C_t = 1|\mathbf{x}_t)^{C_t} P(C_t = 0|\mathbf{x}_t)^{1-C_t} \tag{4}$$

$$= \prod_{t=1}^{T} r_t^{C_t} (1 - r_t)^{1-C_t}, \tag{5}$$

where we have used the shorthand notation r_t for $r(\mathbf{x}_t; \mathbf{w}_m, b_m, \gamma)$. Maximization of $L(\mathbf{w}_m, b_m, \gamma)$ is done by minimization of the cost function $J = -1/T \log L$,

$$J(\mathbf{w}_m, b_m, \gamma) = \frac{1}{T} \sum_{t=1}^{T} \left(-C_t \log r_t - (1 - C_t) \log(1 - r_t)\right). \tag{6}$$

This cost function J is the same as the cross-entropy error function [5]. Furthermore, minimizing the cost function J is equivalent to minimization of the Kullback-Leibler distance between $P(C|\mathbf{x})$ and an assumed true conditional probability $P_{\text{true}}(C|\mathbf{x}) = C$.

2.3 Reference Data

In contrastive feature learning, we construct the reference data set such that it contains the structure of natural images which we are familiar with so that the features of the classifier can reveal novel structure.

A simple way to characterize a data set is to calculate its covariance matrix. For natural images, the covariance-structure has been intensively studied, see

e.g. [2] (keyword: approximate $1/f^2$ behavior of the power spectrum). Hence, we take data which has the same covariance matrix as natural images as reference data. Or equivalently, we take white noise as reference data and contrast it with whitened natural images.

3 Choice of the Nonlinearity

3.1 Ambiguities for Linear and Quadratic Functions

We show here that the function $g(u)$ in Equation (3) should not be linear or quadratic.

Plugging a linear $g(u) = u$ into the formula for $y(\mathbf{x})$ in Equation (3) leads to

$$y(\mathbf{x}) = \sum_{m=1}^{M} \left(\mathbf{w}_m^T \mathbf{x} + b_m\right) + \gamma = \left(\sum_{m=1}^{M} \mathbf{w}_m^T\right) \mathbf{x} + \left(\sum_{m=1}^{M} b_m\right) + \gamma, \tag{7}$$

so that instead of learning M features $\mathbf{w}_m$ one could also learn only a single one, namely $\sum_m \mathbf{w}_m$ with bias $\sum_m b_m$. In other words, having more than a single feature introduces into the cost function J an ambiguity regarding the values of the parameters $\mathbf{w}_m$ and b_m.

There are also ambiguities in the cost function if $g(u) = u^2$. In that case, $y(\mathbf{x})$ equals

$$y(\mathbf{x}) = ||\tilde{\mathbf{W}}^T \tilde{\mathbf{x}}||^2 + \gamma, \tag{8}$$

where $\tilde{\mathbf{x}} = [\mathbf{x}; 1]$ and the $m-th$ column of $\tilde{\mathbf{W}}$ is $[\mathbf{w}_m; b_m]$. As

$$||\tilde{\mathbf{W}}^T \tilde{\mathbf{x}}||^2 = ||\mathbf{Q}\tilde{\mathbf{W}}^T \tilde{\mathbf{x}}||^2 \tag{9}$$

for any orthogonal matrix $\mathbf{Q}$, choosing a quadratic nonlinearity leads to a rotational ambiguity in the cost function. Again, many different sets of features will give exactly the same classifier.

While the arguments just given show that the features are ambiguous for linear and quadratic $g(u)$ for any data set, there is another reason why they are not suitable for the particular data sets used in this paper. In this paper, the natural image data and the reference data have, by construction, exactly the same mean and covariance structure. Thus, any linear or quadratic function has, on the average, the same values for both data sets. Therefore, any linear or quadratic classifier is likely to perform very poorly on our data. Note that such poor performance is not logically implied by the ambiguity in the features discussed above.

3.2 Candidates for the Nonlinearity

In the neural network literature, two classical choices for $g(u)$ in Equation (3) are the tanh and the logistic function $\sigma(u)$,

$$\sigma(u) = \frac{1}{1 + \exp(-u)}. \tag{10}$$

For zero-mean natural images, it seems reasonable to assume that if $\mathbf{x}$ is a natural image then also $-\mathbf{x}$. Thus, the regression function should verify $r(\mathbf{x}) \approx r(-\mathbf{x})$ if $\mathbf{x}$ is a natural image. This holds naturally if we omit the bias terms b_m and choose $g(u)$ even-symmetric. A symmetric version of the logistic function is

$$g(u) = \sigma(u - u_0) + \sigma(-u - u_0), \tag{11}$$

where $2u_0$ is the length of the "thresholding zone" where $g(u) \approx 0$. Other simple symmetric functions are obtained when we add a thresholding zone to the linear and quadratic function, i.e.

$$g(u) = [\max(0, u - u_0)]^{1+\epsilon} + [\max(0, -u - u_0)]^{1+\epsilon} \tag{12}$$

where we added $\epsilon \ll 1$ in the exponent to avoid jumps in the derivative $g'(u)$, and

$$g(u) = [\max(0, u - u_0)]^2 + [\max(0, -u - u_0)]^2. \tag{13}$$

In the following, we call this two functions linear-thresholding nonlinearity and squared-thresholding nonlinearity, respectively.

3.3 Optimizing the Nonlinearity

Instead of using a fixed nonlinearity, one can also learn it from the data. For instance, $g(u)$ can be written as weighted superposition of some parameterized functions $g_i(u; \theta)$,

$$g(u) = \sum_{i=1}^{I} \alpha_i g_i(u; \theta). \tag{14}$$

Then, we can optimize the conditional likelihood, or in practice the cost function J of Equation (6), also with respect to α_i and the parameters θ.

We consider here the special case where

$$g(u) = \alpha_1[\max(0, u - \beta_1)]^{\eta_1} + \alpha_2[\max(0, -(u - \beta_2))]^{\eta_2}, \tag{15}$$

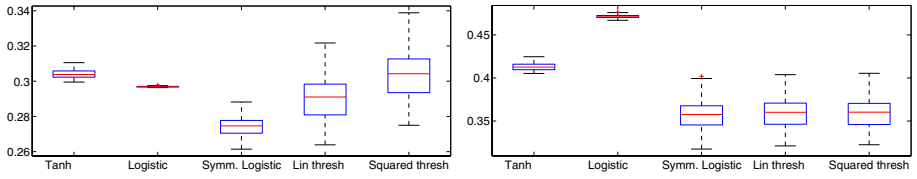
for $\alpha_i \in \mathbb{R}$, $\beta_i \in \mathbb{R}$, and $\eta_i \in (1, 4]$. We optimize thus the type of nonlinearity of Equation (12) and (13) with respect to the size of the thresholding zone and the power-exponent. Furthermore, the signs of the α_i control whether the two power functions in (15) are each concave-up or concave-down.

4 Simulations

4.1 Settings

We estimate the classifier with a steepest descent algorithm on the cost function J of Equation (6), where we sped up the convergence by using the rprop algorithm [6].[1] Preliminary simulations with a fixed stepsize yielded similar results. The classifier was estimated several times starting from different random

[1] The multiplicative factors were $\eta_+ = 1.2$ and $\eta_- = 0.5$, maximal allowed change was 2, and minimal change 10^{-4}.



(a) Cross entropy error J **(b)** False classification rate

Fig. 1. Distributions of the cross entropy error J and the false classification rate for the nonlinearities of Section 3.2. The distributions were obtained from the validation sets and are shown as box plots. The central red line is the median, the edges of the box are the 25th and 75th percentiles, and the whiskers extend to the most extreme data points. Outliers are marked with a cross. The settings were as follows: Bias terms b_m were only included for the tanh and the logistic function. The shift amount u_0 was $u_0 = 8$ for the symmetric logistic function (Symm. Logistic, see Equation (11)). For the linear-thresholding nonlinearity (Lin thresh, see Equation (12)) and the squared-thresholding nonlinearity (Squared thresh, see Equation (13)), we used $u_0 = 2$. On the training set, the minimal cross entropies J and false classification rates ("er") for each nonlinearity were $J = 0.282$, $er = 0.372$ for Tanh; $J = 0.294$, $er = 0.457$ for Logistic; $J = 0.231$, $er = 0.223$ for Symm. Logistic; $J = 0.202$, $er = 0.223$ for Lin tresh; $J = 0.199$, $er = 0.220$ for Squared thresh.

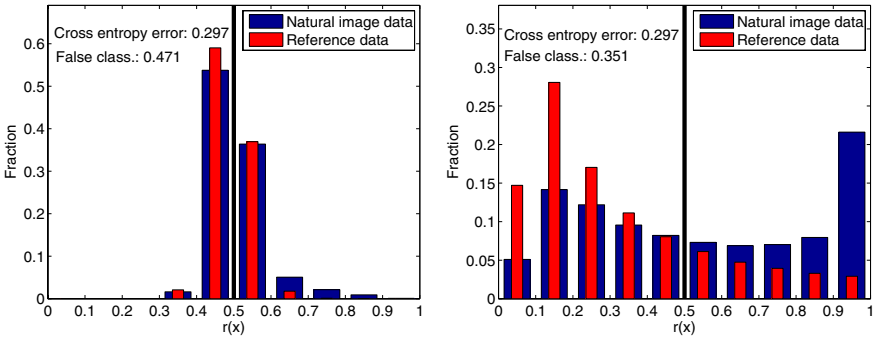
initializations. For computational reasons, we used only 5 initializations for the simulations of Section 4.2 and 20 for those of Section 4.3. We stopped optimization when the average change in the parameters was smaller than 10^{-3}. The classifier that had the smallest cost was selected for validation. The number of features M was set to 100.

Each training sample $\mathbf{x}_t$ was normalized to have an average value (DC component) of zero and norm one to reduce the sensitivity to outliers. The training set consisted of 80000 patches of natural images (size: 14× 14 pixels), and an equal number of reference data. For validation, we used 50 data sets of the same size as the training set. We also reduced the dimensions from $14 \times 14 = 196$ to 49, i.e. we retained only 25% of the dimensions.

4.2 Results for Fixed Nonlinearities

Performance. First, we validated our reasoning of Section 3.1 that a linear or quadratic $g(u)$ is not suitable to discriminate between whitened natural images and white Gaussian noise. Indeed, the false classification rate for the validation sets were distributed around chance level for the linear function (mean 0.5) and above chance level for the quadratic nonlinearity (mean 0.52).

Then, we performed simulations for the nonlinearities discussed in Section 3.2. The generalization performance as measured by the cross entropy error function J for validation sets is summarized in Figure 1a. The classifier with the symmetric logistic function has the best generalization performance. The squared-thresholding nonlinearity, which attained the minimal value of J for the training



(a) Logistic **(b)** Squared-threshold

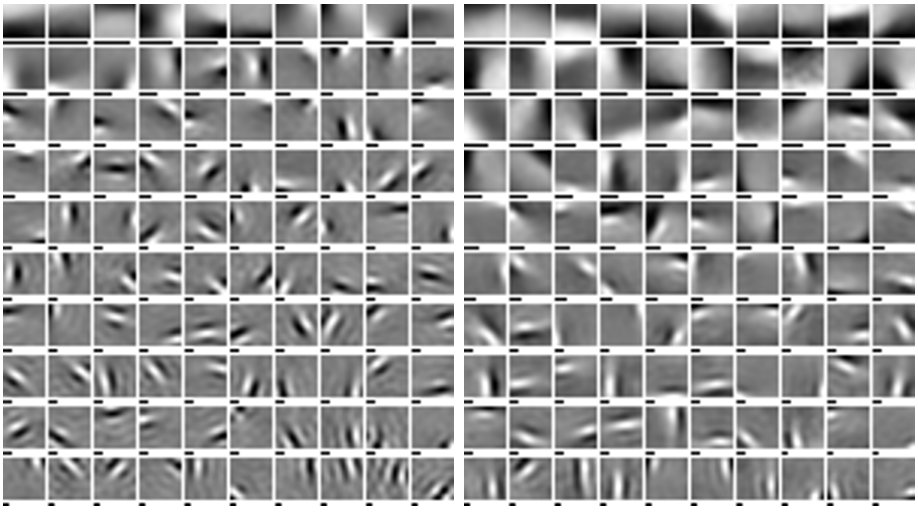
Fig. 2. Conditional probability distributions $r(\mathbf{x}) = P(C = 1|\mathbf{x})$ when the input is natural image data (blue) or reference data (red). For natural image input, $r(\mathbf{x})$ should be 1. For reference data, $r(\mathbf{x})$ should be 0. The data set was chosen from the validation sets such that the cross entropy error J was approximately the same for the logistic nonlinearity and the squared-threshold nonlinearity. It is intuitively clear that the distribution in (b) is better for classification, although the cross-entropies are equal in the two cases. This seems to be because the cross-entropy gives a lot of weight to values near 0 or 1 due to the logarithmic function.

set (see caption of Figure 1), leads to the distribution with the highest median and a large dispersion, which seems to indicate some overlearning.

Figure 1b shows the false classification rates for the validation data. The symmetric nonlinearities, i.e. the symmetric logistic function and the nonlinearities with a thresholding zone, perform all equally well. Furthermore, they outperform the tanh and the logistic function. The performance as measured by the false classification rate and the cross entropy lead thus to different rankings.

Figure 2 gives a possible explanation for the discrepancy between the cross-entropies and false classification rates. The figure shows that two distributions of the conditional probability $r(\mathbf{x})$ can be rather different but, nevertheless, attain the same cross entropy error J. For the logistic nonlinearity in Figure 2a, $r(\mathbf{x})$ is clustered around chance level 0.5. The false classification rate is therefore also close to 0.5. On the other hand, for the same cross entropy error, the squared-thresholding nonlinearity leads to a false classification rate of 0.35. The reason behind its high cross entropy is that natural images (reference data) which are wrongly assigned a too low (high) conditional probability $r(\mathbf{x})$ enter logarithmically weighted into the calculation of the cross entropy.

Features. The estimated features $\mathbf{w}_m$ when the nonlinearity $g(u)$ is the symmetric logistic function are shown in Figure 3a. They are localized, oriented, and indicate bright-dark transitions. They are thus "gabor-like" features. For the linear-thresholding and squared-thresholding function, the features were similar. For the tanh and the logistic function, however, they did not have any clear structure.



(a) Symm. Logistic **(b)** Optimized nonlinearity

Fig. 3. Features $\mathbf{w}_m$, $m = 1 \dots M = 100$. The features are shown in the original image space, i.e. after multiplication with the dewhitening matrix. For visualization purposes, each feature was normalized to use the full range of the color-map. The black bar under each image panel indicates the euclidean norm $||\mathbf{w}_m||$ of the feature.

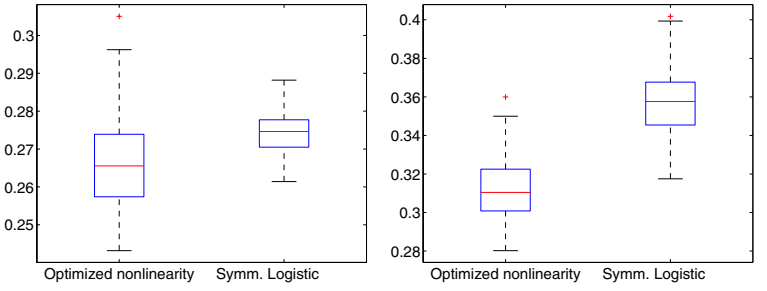
For the symmetric logistic function the learned offset γ in Equation (3) is -3.54. For the linear- and squared-thresholding functions, we also have $\gamma < 0$. For natural image input, $y(\mathbf{x})$ in Equation (3) must be as large as possible so that $r(\mathbf{x}) \to 1$. For reference data, $y(\mathbf{x})$ should be as negative as possible. Since the nonlinearities $g(u)$ attain only positive values, $y(\mathbf{x}) < 0$ is only possible when $\mathbf{w}_m^T\mathbf{x}$ falls into the thresholding zone of the nonlinearity. The negative γ leads then to a negative $y(\mathbf{x})$. Hence, the classifiers work by thresholding the outputs of gabor-like features. Large outputs are indicators for natural image input while small outputs indicate the presence of reference data.

4.3 Results for Optimized Nonlinearity

Optimization of the nonlinearity in Equation (15) leads to a classifier with a better performance than the fixed nonlinearities, see Figure 4.

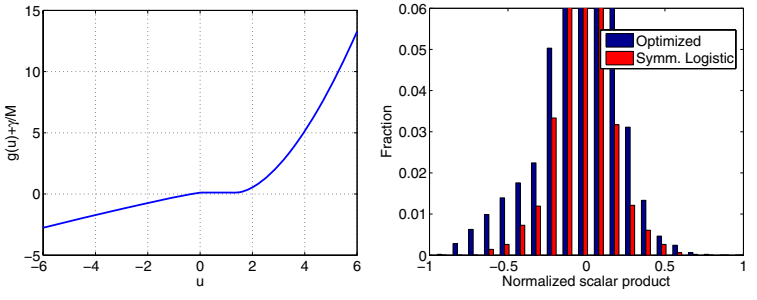
Figure 5a shows the optimal nonlinearity where the offset per feature has been added, i.e. $g_{\text{eff}}(u) = g(u) + \gamma/M$ is shown. If $g_{\text{eff}}(\mathbf{w}_m^T\mathbf{x}) > 0$, feature m signals the presence of natural image data. Negative outputs indicate the presence of reference data. The outputs are negative when, approximately, $\mathbf{w}_m^T\mathbf{x} < 0$. This is in contrast to the fixed nonlinearities where for reference data $\mathbf{w}_m^T\mathbf{x}$ had to be in the thresholding zone.

The features for the optimal nonlinearity are shown in Figure 3b. They are also "gabor-like". Visual inspection, as well as a histogram of the normalized



(a) Cross entropy error J **(b)** False classification rate

Fig. 4. Distribution of the cross entropy error J and the false classification rate for the learned nonlinearity and, for reference, the symmetric logistic function. The optimized nonlinearity achieves better performance both in terms of the cross entropy error J and the false classification rate. We tested if the distributions give enough evidence to conclude that the mean cross entropy error and the mean false classification rate are different for the two nonlinearities. For the cross entropy error, the p-value was 0.0014. For the false classification rate, the p-value was 0 (below machine precision). Hence, there is statistically significant evidence that their means are different, i.e. that, on average, the classifier with the optimized nonlinearity performs better than the symmetric logistic function. On the training set, the cross entropy error J was 0.186, and the false classification rate 0.203, cf. Figure 1.



(a) Optimal nonlinearity **(b)** Similarity of features

Fig. 5. (a) The optimal nonlinearity of Equation (15) has the parameters $\alpha_1 = 1.00$, $\alpha_2 = -0.40$, $\beta_1 = 1.40$, $\beta_2 = -0.01$, $\eta_1 = 1.69$, $\eta_2 = 1.10$, and $\gamma = 11.76$. The negative α_2 makes the nonlinearity highly asymmetric. Note that $\eta_2 = 1.10$ is the smallest exponent which was allowed in the optimization. (b) In the calculation of the scalar product between the features, we first normalized them to unit norm.

dot-products between the features in Figure 5b, shows, however, that the features are more similar to each other than the features of the symmetric logistic function. Together with the shape of the optimal nonlinearity, this suggests that the classifier is using a different strategy than with the fixed (symmetric) nonlinearities. The negative part of the nonlinearity can be interpreted as leading

to an interaction between features. An input is likely to be a natural image if some of the features have with the same sign large dot-products with $\mathbf{x}$, and not with opposite signs.

4.4 Relation to Other Work

Features that are obtained from a generative probabilistic model of natural images lead also to gabor-like features as in Figure 3, see e.g. [2]. This might reflect the relation between nonlinear neural networks and ICA [7]. However, sign-dependent interactions between the features (see Section 4.3) has not been found so far in generative models of natural images. Other work where learning a discriminative model led to gabor-like features is [8] where the features emerged from learning shape-from-shadings.

5 Conclusions

We presented an alternative to generative probabilistic modeling for the learning of features in natural images. The features are learned by contrasting natural image data with reference data that contains some known structure of natural images. Here, we used a classifier with only one layer of features and reference data with the same covariance-structure as natural images to validate the concept. The learned features were similar to those of generative models. When we optimized the nonlinearity in the classifier, we obtained a function which seems to facilitate interaction between the features.

The presented approach can easily be extended to multi-layer architectures, which is difficult for generative models, and also reference data that contain more structure than the one used here. Furthermore, the method can also be used on other kinds of data, and is not at all restricted to natural images.

References

1. Srivastava, A., Lee, A., Simoncelli, E., Zhu, S.: On advances in statistical modeling of natural images. J. Math. Imaging and Vision 18(1), 17–33 (2003)
2. Hyvärinen, A., Hurri, J., Hoyer, P.: Natural Image Statistics - A probabilistic approach to early computational vision. Springer, Heidelberg (2009)
3. Teh, Y., Welling, M., Osindero, S., Hinton, G.: Energy-based models for sparse overcomplete representations. J. Mach. Learn. Res. 4(7-8), 1235–1260 (2004)
4. Hyvärinen, A.: Estimation of non-normalized statistical models using score matching. J. Mach. Learn Res. 6, 695–709 (2005)
5. Bishop, C.: Neural networks for pattern recognition. Oxford University Press, Oxford (1995)
6. Riedmiller, M., Braun, H.: A direct adapatative method for faster backpropagation learning: The rprop algorithm. In: Ruspini, H. (ed.) Proc. IEEE Int Conference on Neural Networks (ICNN), pp. 586–591 (1993)
7. Hyvärinen, A., Bingham, E.: Connection between multilayer perceptrons and regression using independent component analysis. Neurocomp. 50(C), 211–222 (2003)
8. Lehky, S., Sejnowski, T.: Network model of shape-from-shading: neural function arises from both receptive and projective fields. Nature 333, 452–454 (1988)