IJCNN 2000, To appear.

Metrics that learn relevance

Samuel Kaski and Janne Sinkkonen

Neural Networks Research Centre, Helsinki University of Technology
P. O. Box 5400, FIN-02015 HUT, Finland
samuel.kaski@hut.fi, janne.sinkkonen@hut.fi

Abstract

We introduce an algorithm for learning a local metric to a continuous input space that measures distances in terms of relevance to the processing task. The relevance is defined as local changes in discrete auxiliary information, which may be for example the class of the data items, an index of performance, or a contextual input. A set of neurons first learns representations that maximize the mutual information between their outputs and the random variable representing the auxiliary information. The implicit knowledge gained about relevance is then transformed into a new metric of the input space that measures the change in the auxiliary information in the sense of local approximations to the Kullback-Leibler divergence. The new metric can be used in further processing by other algorithms. It is especially useful in data analysis applications since the distances can be interpreted in terms of the local relevance of the original variables.

1 Introduction

The success of unsupervised learning algorithms, including principal components analysis, various clustering algorithms, and Self-Organizing Maps, depends crucially on feature extraction, i.e., the choice and relative scaling of the input variables. Successful feature extraction stages are often tailored according to the task at hand using expert knowledge or heuristic rules of thumb.

There is, however, often some implicit auxiliary information available about the relevance of the input. A classification of the samples may be known and the goal is to discover characteristics of the classes, or to find suitable features for classification. Alternatively, for example in process monitoring, some indicator of the performance may be associated with each data vector and the goal is to find out what factors affect the performance. In a prediction task the goal may be to discover which features are important in successful prediction.

In this work we introduce an algorithm that learns to take such auxiliary information into account. In the first stage a network learns features that maximize the mutual information between the features and the auxiliary information.

Mutual information between the outputs of different processing modules has been used already previously as a criterion for building representations that are coherent over time or space [2, 3, 8]. Our approach is similar up to this point but we do not use the features as such. Instead, we explicitly transform the (local) metric of the original input space so that original proximity relations are preserved but the new distances measure the (local) change of the auxiliary information. Maximization of mutual information by itself is not enough to produce such a metric.

If the auxiliary information has been chosen to indicate relevance to the overall goals of the learning task then distances will be measured according to their relevance to the goals. Same distance always signifies equal relevance.

The relevance metric can be used by other algorithms. If the algorithms are unsupervised, the desirable properties of unsupervised algorithms like fast learning, generalizability, and visualization capability are preserved. A demonstration of Self-Organizing Map learning in the new metric will be presented in Sec. 6. The new metric is especially useful in data-analysis applications since it can be easily interpreted in terms of local relevance of the original input variables.

2 Networks that maximize mutual information

Consider a network of N neurons. Each neuron receives the same (stochastic) input $\mathbf{x} \in \mathbb{R}^n$, and the activation of the jth neuron in response to the input is denoted by $y_j(\mathbf{x}) \geq 0$, with $\sum_j y_j(\mathbf{x}) = 1$.

The learning method presented here is general in the sense that it does not depend on the type of neurons used (the exact form of the parameterization of the y_j). In the next section a detailed learning algorithm is derived for one suitable form of parameterization.

Unless the components of $\mathbf{x}$ already come from a carefully optimized feature extraction stage they can be noisy and some of them may even be completely irrelevant to the task at hand. A completely unsupervised network is unable to distinguish between relevant and irrelevant information. The question addressed in this section is how the network could be made to learn to utilize auxiliary information that is available about $\mathbf{x}$ so that after learning the responses of the neurons would reflect the auxiliary information as well as possible. Assuming that the auxiliary information is relevant to the task the network is intended to perform, the neurons would then have learned to extract the relevant features.

When an input $\mathbf{x}$ is presented to the network, the output is a distribution of activity over the neurons. Since the activations sum to unity, the distribution may be interpreted as the conditional probability distribution of a random variable called V: $p(v_j|\mathbf{x}) \equiv y_j(\mathbf{x})$, $j = 1, \ldots, N$. Here the values of V have been denoted by v_j. The marginal distribution of V is then $p(v_j) = \int y_j(\mathbf{x})p(\mathbf{x})d\mathbf{x}$, where $p(\mathbf{x})$ is the probability density function of the input.

Assume that the auxiliary information related to the sample is represented by a discrete-valued random variable denoted by C. Denote its values by c_i; the indices i may denote, for example, the possible classes of data, alternative contexts of $\mathbf{x}$, or the possible outcomes in a prediction task. If the space of the outcomes is continuous it may be discretized suitably. Note that usually instead of the distribution $p(c_i|\mathbf{x})$ itself, only values c associated with the inputs $\mathbf{x}$ are known.

The aim of learning is to optimize the parameters of the neurons so that the information that the activities of the neurons mediate of C is maximized, i.e., to *maximize the mutual information $I(C;V)$ between the distribution of activity over the neurons and the distribution of the auxiliary random variable C*. The mutual information is

$$\begin{aligned} I(C;V) &= \sum_{i,j} p(c_i, v_j) \log \frac{p(c_i, v_j)}{p(c_i)p(v_j)} \\ &= \int \sum_{i,j} \log \frac{p(c_i, v_j)}{p(c_i)p(v_j)} p(c_i|\mathbf{x})p(v_j|\mathbf{x})p(\mathbf{x})d\mathbf{x} \ . \end{aligned} \quad (1)$$

Above we have decomposed the joint probability $p(c_i, v_j)$ using the fact that the auxiliary information does not directly affect the activity of the neurons. The variables C and V are therefore conditionally independent, given $\mathbf{x}$.

In the next section we will introduce an algorithm that maximizes the mutual information (1) by adjusting the parameters of the network.

3 Maximization of mutual information for Gaussian neurons

It can be shown that the gradient of $I(C;V)$ with respect to the parameters $\boldsymbol{\theta}$ of the neurons is (derivation omitted)

$$\nabla_{\boldsymbol{\theta}} I(C;V) = \sum_{i,j} \log \frac{p(c_i, v_j)}{p(c_i)p(v_j)} \int p(c_i|\mathbf{x}) \nabla_{\boldsymbol{\theta}} y_j(\mathbf{x}) p(\mathbf{x}) \ d\mathbf{x} \ . \quad (2)$$

The expression (2) is valid for any kinds of parameterized neurons. In this section we show how to maximize $I(C;V)$ for normalized Gaussian neurons with an *a priori* set common width σ. The response of such a neuron to the input $\mathbf{x}$ is

$$y_j(\mathbf{x}) = \frac{G_j(\mathbf{x};\mathbf{w}_j)}{\sum_k G_k(\mathbf{x};\mathbf{w}_k)} \ , \ \text{where} \quad G_j(\mathbf{x};\mathbf{w}_j) = e^{-\|\mathbf{x}-\mathbf{w}_j\|^2/2\sigma^2} \quad (3)$$

where the $\mathbf{w}_j$ are the parameters to be optimized. (Note that although we denote $y_j(\mathbf{x})$ for brevity, each $y_j(\mathbf{x})$ actually depends on all the $\mathbf{w}$'s). For these Gaussian neurons the gradient of $I(C;V)$ is

$$\nabla_{\mathbf{w}_j} I(C;V) = \frac{1}{\sigma^2} \sum_i \sum_{l \neq j} \log \frac{p(c_i|v_j)}{p(c_i|v_l)} \int (\mathbf{x} - \mathbf{w}_j) y_j(\mathbf{x}) y_l(\mathbf{x}) p(c_i, \mathbf{x}) d\mathbf{x} \ . \quad (4)$$

It is straightforward to maximize $I(C;V)$ using (4). The integrals in (4) and in $p(c_i|v_j) = \int v_j(\mathbf{x})p(c_i, \mathbf{x})d\mathbf{x}$ can be estimated as weighted sums over the data which is supposedly drawn from the distribution $p(c_i, \mathbf{x})$. The parameters $\mathbf{w}_j$ can then be readily optimized using a general-purpose optimization algorithm.

In order to achieve on-line learning we have, however, used stochastic approximation to maximize $I(C;V)$. For the stochastic approximation the responses $y_j(\mathbf{x})$ and $y_l(\mathbf{x})$ of neurons j and l, respectively, are interpreted as the densities $p(v_j|\mathbf{x})$ and $p(v_l|\mathbf{x})$ of two discrete random variables that are conditionally independent of each other and of C. Then, the expression

$$y_j(\mathbf{x})y_l(\mathbf{x})p(c_i, \mathbf{x}) = p(v_j|\mathbf{x})p(v_l|\mathbf{x})p(c_i|\mathbf{x})p(\mathbf{x}) = p(v_j, v_l, c_i, \mathbf{x})$$

in (4) can be used as the sampling function for stochastic approximation. This leads to the following algorithm: At the step t of stochastic approximation, draw an input $(\mathbf{x}(t), c_i)$, and then two distinct neurons j and l from the multinomial distribution $\{y_k(\mathbf{x}(t))\}$. Adapt the parameters according to

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \alpha(t) \log \frac{\hat{p}(c_i|v_j)}{\hat{p}(c_i|v_l)} \left(\mathbf{x}(t) - \mathbf{w}_j(t)\right) \ , \quad (5)$$

where $\alpha(t)$ is the gradually decreasing step size. The estimates $\hat{p}(c_i|v_j)$ can be adapted simultaneously with leaky integration at a rate of change larger than $\alpha(t)$ in (5).

4 Metric that measures relevance

The mutual information $I(C;V)$ is a measure of the statistical dependency between the outputs of the network and the auxiliary information C. The outputs $y_j(\mathbf{x})$ form a 'representation' of the input x which is our own construction intended to capture the dependency between $\mathbf{x}$ and C, in the limits allowed by the parametrization of V. The auxiliary information C, on the other hand, has been selected so that changes in its distribution signify relevance.

In this section our first aim is to make the dependency between $\mathbf{x}$ and C, as mediated by V, explicit, and then to use this knowledge to judge the relevance of *local* changes of $\mathbf{x}$ in various *directions of the input space*. Assuming that C has been selected suitably the relevance can be measured as changes in the estimated distribution of C as a function of $\mathbf{x}$, and formulated as a new metric to the input space.

Let us start by making the dependency between $\mathbf{x}$ and C explicit. The mutual information $I(C;V)$ can be written as an integral over the probability $p(\mathbf{x})$:

$$I(C;V) = -\int \sum_i p(c_i|\mathbf{x}) \log \frac{p(c_i|\mathbf{x})}{\hat{p}(c_i|\mathbf{x})}\, p(\mathbf{x}) d\mathbf{x} + \text{const.}, \tag{6}$$

where

$$\hat{p}(c_i|\mathbf{x}) = \exp \sum_j y_j(\mathbf{x}) \log p(c_i|v_j)\ . \tag{7}$$

The integral in (6) is equal to the average Kullback-Leibler divergence between the distributions $\{p(c_i|\mathbf{x})\}$ and $\{\hat{p}(c_i|\mathbf{x})\}$. Therefore, if we wish to simultaneously maximize $I(C;V)$ *and* construct good density estimates (in the sense of the Kullback-Leibler divergence), we need to use $\hat{p}(c_i|\mathbf{x})$ of equation (7) as our density estimate.

Now we could measure the relevance of the difference between any pair of input samples by the difference between the corresponding $\{\hat{p}(c_i|\mathbf{x})\}$. Such distance measures are useful but they have a disadvantage: they generate a topology which may be different than the original topology of the input space. Two points originally far away may have zero distance if the density estimates $\{\hat{p}(c_i|\mathbf{x})\}$ are identical!

Preservation of the topology of the input space is, of course, important unless we want the auxiliary information to completely override the original identity of the data. Therefore we introduce one additional constraint: the topology of the input space may not change, which can be guaranteed by measuring the distances locally and defining non-local distances as path integrals along the minimal paths in the input space. This fixes the new metric to be locally similar to the original one up to a local scaling, that is, the new distances will be of the form $d^2(\mathbf{x}, \mathbf{x}+d\mathbf{x}) = d\mathbf{x}^T \mathbf{J}(\mathbf{x}) d\mathbf{x}$.

If the differences between the estimates of posterior distributions $\hat{p}(c_i|\mathbf{x})$ are measured in terms of the Kullback-Leibler divergence, it is well known [6] that the distance can be computed locally as

$$D(p(c_i|\mathbf{x}) \| p(c_i|\mathbf{x}+d\mathbf{x})) = d\mathbf{x}^T \mathbf{J}(\mathbf{x}) d\mathbf{x}\ , \tag{8}$$

where

$$\mathbf{J}(\mathbf{x}) = E_{C|\mathbf{x}} \left\{ \left(\nabla_{\mathbf{x}} \log P(C|\mathbf{x})\right) \left(\nabla_{\mathbf{x}} \log P(C|\mathbf{x})\right)^T \right\}$$

is the Fisher information matrix.

In the new metric the posterior density $\{p(c_i|\mathbf{x})\}$ changes evenly everywhere and in all directions of the input space. The new metric can be used for visualization by unsupervised methods, or as a distance measure for later supervised learning.

Note about an invariance property of the constructed metric. The distance d^2 is invariant to certain mappings of the input space. If we have a differentiable, one-to-one mapping from $\mathbf{x}$ to $\mathbf{x}' = s(\mathbf{x})$, then $p(c|\mathbf{x}') = p(c|\mathbf{x})$ and $ds(\mathbf{x}) = \mathbf{K} d\mathbf{x}$, where $\mathbf{K}(\mathbf{x}) \equiv \partial s(\mathbf{y})/\partial \mathbf{y}|_{\mathbf{x}}$ is the Jacobian matrix of the mapping. The distance between $\mathbf{x}$ and $\mathbf{x} + d\mathbf{x}$ after the mapping then becomes

$$d^2(\mathbf{x}', \mathbf{x}' + d\mathbf{x}') = E_C \left\{ \left(d\mathbf{x}^T \mathbf{K}^T (\mathbf{K}^T)^{-1} \nabla_{\mathbf{x}} \log p(c|\mathbf{x}) \right)^2 \right\} = d^2(\mathbf{x}, \mathbf{x} + d\mathbf{x})\ ,$$

i.e., the d^2-distance is invariant under invertible, smooth mappings, if computed from the correct posterior probabilities $p(c|\mathbf{x})$. In reality we have only estimates for them, but if the estimates are good, any further processing stages based on d^2 would be insensitive to a large class of topology-preserving, nonlinear transformation of the input space.

5 Unsupervised learning in the relevance metric

We will next discuss how the new relevance metric can be used as a distance measure in subsequent processing. In this paper we will use the Gaussian neurons discussed in Sec. 3 and derive the explicit metric generated by them. Similar metrics could be derived for other types of representations as well.

When Gaussian neurons are used to estimate the conditional probability distribution $p(c_i|\mathbf{x})$ according to (7), and the estimate is used in the distance measure (8), we get the approximation

$$d_G^2(\mathbf{x}, \mathbf{x} + \delta\mathbf{x}) = \sum_i \hat{p}(c_i|\mathbf{x})[(\delta\mathbf{x})^T \mathbf{b}_i(\mathbf{x})]^2 \tag{9}$$

for the squared Kullback-Leibler distances. In the formula for d_G, we have denoted

$$\mathbf{b}_i(\mathbf{x}) = \sum_j y_j(\mathbf{x}) \log \hat{p}(c_i|v_j)(\mathbf{w}_j - \hat{\mathbf{x}}) \tag{10}$$

and $\hat{\mathbf{x}} = \sum_j y_j(\mathbf{x})\mathbf{w}_j$. This distance measure can in principle be used in any further processing task. However, since the metric was originally derived for differential $\delta\mathbf{x}$, it is most appropriate to use methods that rely mainly on distances between close-by points of the input space if (9) is used as such.

We will demonstrate the use of the relevance metric by computing Self-Organizing Maps [5] in the new metric. The Self-Organizing Map (SOM) is a regular grid of units in which a model vector $\mathbf{m}_i$ is associated with each unit i. During the learning process the model vectors are modified so that they learn to follow the distribution of the input data in an ordered fashion: model vectors that are close-by on the map lattice attain close-by locations in the input space as well. If the map grid is chosen to be two-dimensional the resulting map display can be used for visualizing various properties of the input data which is useful in data analysis applications.

The SOM algorithm consists of iterative application of two steps. The winning unit that is closest to the current input sample is first sought, and thereafter the winner and its neighbors on the map lattice are adapted towards the input sample.

We will carry out these steps in the relevance metric. The winner is defined to be the unit for which $d_G^2(\mathbf{x}, \mathbf{m}_c) \leq d_G^2(\mathbf{x}, \mathbf{m}_i)$, and the model vectors adapt at time step t according to

$$\begin{aligned} \mathbf{m}_i(t+1) &= \mathbf{m}_i(t) - h_{ci}(t)\nabla_{\mathbf{m}_i} d_G^2(\mathbf{x}, \mathbf{m}_i) \\ &= \mathbf{m}_i(t) - h_{ci}(t) \sum_i \hat{p}(c_i|\mathbf{x})(\mathbf{m}_i - \mathbf{x})^T \mathbf{b}_i(\mathbf{x})\mathbf{b}_i(\mathbf{x}) \ . \end{aligned}$$

Here $h_{ci}(t)$ is the so-called neighborhood function, a decreasing function of the distance between the units c and i on the map lattice. The height and width of $h_{ci}(t)$ decrease gradually in time.

6 Experiments

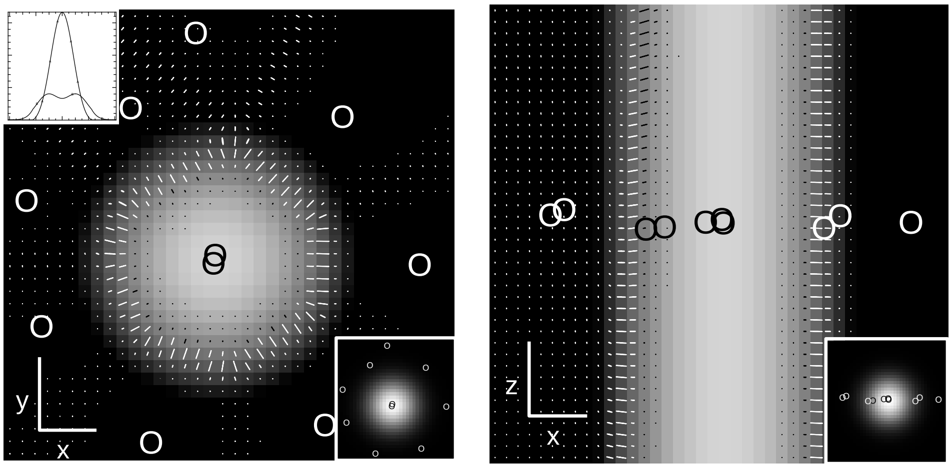
In this section we demonstrate what kind of a metric ensues when a network learns to extract the essential characteristics of a three-dimensional, easily visualizable artificial data set. The metric is then used in the SOM algorithm to visualize the class distribution of the data. The experiment is intended to serve as an illustration only; we will report on results with practical data in later papers.

The data was sampled from a spherically symmetric normal distribution (cf. the insets in Fig. 1). The available auxiliary information consisted of the class of each data sample. The two classes were highly overlapping and distributed in a "cylindrical" fashion: the conditional probability distribution $p(c_i|\mathbf{x})$ was constant in each column directed along the z-axis. On each plane orthogonal to the z-axis the first class was concentrated somewhat more in the middle of the xy-plane (cf. the inset in the top left corner of Fig. 1**a**).

In summary, although the distribution of the data was spherically symmetric only the radial direction in the xy-plane was relevant from the viewpoint of auxiliary information.

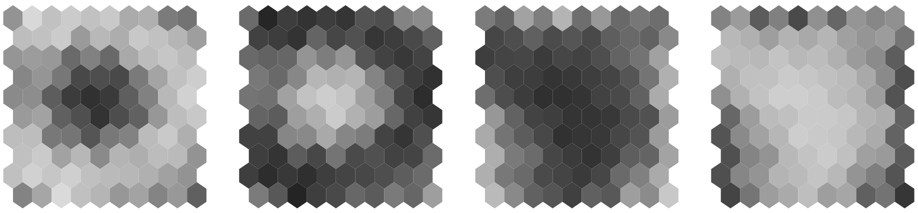
Fig. 1 shows the weight vectors (parameters) of neurons that have learned according to the stochastic approximation method described in Sec. 3. The vectors are located approximately on a plane that is orthogonal to the z-axis; they do not represent the z-axis at all. On the xy-plane there are two neurons in the center and the others are located in an almost regular symmetric configuration around the center to be able to best represent the important radial direction.

In the information distances d_G estimated using the converged network, and illustrated with the small line segments in Fig. 1, the radial direction on the xy-plane dominates. Distances are largest where the class distribution changes most rapidly. In the direction of the z-axis the distances are practically zero.



a **b**

Figure 1: Locations of weight vectors of neurons (here centers of Gaussian kernels) in a network that has learned to maximize mutual information with auxiliary class information. Two different cross sections of the three-dimensional space are shown; the circles denote the weight vectors projected on the cross sections. The gray levels indicate the conditional probability $p(c_1|\mathbf{x})$ of the first class in each location, and the small line segments (or dots) depict the dominant direction and relative distances in the local relevance metric. The insets in the bottom right corner depict the distribution of the data, and the two curves in the inset in the top left corner of **a** depict the conditional probability distributions of the two classes on a cross-section of the plane.



a **b** **c** **d**

Figure 2: Distribution of the classes of data on the map units of two SOMs. **a** Class 1, SOM 1: relevance metric. **b** Class 2, SOM 1: relevance metric. **c** Class 1, SOM 2: Euclidean metric. **d** Class 2, SOM 2: Euclidean metric. Light shades denote high density and dark shades low density, respectively.

Finally, we will demonstrate that the relevance metric can be used as a preprocessing stage for further processing. We computed a self-organizing map in both the original Euclidean metric and in the relevance metric, and visualized the class distribution on the resulting maps. It can be seen in Fig. 2 that the classes are more distinctly separated on the SOM computed in the relevance metric, and that the border between the classes is well represented.

Note that in a data analysis application it would be possible to use the same SOM displays for visualizing other aspects of the data. For instance the distribution of the original variables could be visualized to reveal which variables contribute to the separation of the classes *locally* in the input space.

7 Discussion

In this paper we have shown that maximization of the mutual information between an auxiliary random variable and a feature set derived from the input can be seen as a form of density estimation. The estimate defines a metric in the input space that is additionally constrained to preserve the original topology of the space although it rescales the local distances. If the auxiliary variable is chosen so that changes in its value signify relevance to the task at hand, then the metric measures the relevance. The measure is optimal given the restrictions posed by the parametrization of the network.

According to our knowledge the principle is new. Works in which some aspects resemble our approach exist, however. Amari and Wu [1] have augmented support vector machines by making an isotropic change to the metric near the class border. In contrast to this, our metric is non-isotropic and global. Jaakkola and Haussler [4] induced a distance measure into a discrete input space using a generative probability model. The crucial differences are that they do not use external information, and that they do not constrain the metric to preserve the topology. We have also recently become aware of the Information Bottleneck framework of Tishby et al. [9]. Their setup is discrete, and therefore does not aim at finding local metrics. The approach is related to ours in that the goal is to maximize mutual information between a representation and a relevance indicator.

If an unsupervised algorithm learns using the relevance metric, or if the metric as such is used as the output of a data analysis, then the learning process is somewhere in between supervised and unsupervised. The topology of the input space is preserved as is typical to unsupervised methods, while the metric (local scaling of the space) is induced in a supervised manner. Compared to using a standard separate feature extraction stage, the change of the metric defines a manifold which cannot in general be projected to a Euclidean space of the same dimension. Therefore, no dimensionality-preserving mapping with the same local properties exists which means that the change of the metric is a more general operation than feature selection by a dimensionality-preserving (or dimensionality-reducing) nonlinear mapping.

The most obvious applications of the method are in exploratory data analysis. Because the metric of the *original input space* is transformed, interpretation of the discovered relevant factors is straightforward. For high-dimensional inputs, the results can be visualized using a dimensionality-reducing method such as the Self-Organizing Map.

Forming the relevance metric can additionally be considered as a kind of a nonlinear discriminant analysis. The linear discriminant analysis finds a linear transformation that maximizes class separability. Our metric transforms the input space locally such that the change in the class distribution becomes isotropic, the same in every direction, which allows inspection of the class distributions even more closely.

Classical canonical correlation analysis has been generalized by replacing the linear combinations by nonlinear functions [2, 7]. It would be possible to use our metric for the same task, finding statistical dependencies between two data sets, by replacing the discrete auxiliary random variable with a parametrized set of features computed of an auxiliary continuous random variable. The advantage of our method would then be that it creates an easily interpretable metric.

Acknowledgments

This work was supported by the Academy of Finland.

References

[1] S. Amari and S. Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12:783–789, 1999.

[2] S. Becker. Mutual information maximization: models of cortical self-organization. *Network: Computation in Neural Systems*, 7:7–31, 1996.

[3] S. Becker and G. E. Hinton. Self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, 355:161–163, 1992.

[4] T. S. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems 11*. Morgan Kauffmann Publishers, San Mateo, CA. To appear.

[5] T. Kohonen. *Self-Organizing Maps*. Springer, Berlin, 1995. (Second, extended edition 1997).

[6] S. Kullback. *Information Theory and Statistics*. Wiley, New York, 1959.

[7] P. L. Lai and C. Fyfe. A neural implementation of canonical correlation analysis. *Neural Networks*, 12:1391–1397, 1999.

[8] W. A. Phillips, J. Kay, and D. Smyth. The discovery of structure by multi-stream networks of local processors with contextual guidance. *Network: Computation in Neural Systems*, 6:225–246, 1995.

[9] N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In *37th Annual Allerton Conference on Communication, Control, and Computing*, Illinois, 1999. To appear.