Kernel Information Embeddings

Roland Memisevic ROLAND@CS.TORONTO.EDU
Department of Computer Science, University of Toronto, Toronto ON M5S3G4 Canada

Abstract

We describe a family of embedding algorithms that are based on nonparametric estimates of mutual information (MI). Using Parzen window estimates of the distribution in the joint (input, embedding)-space, we derive a MI-based objective function for dimensionality reduction that can be optimized directly with respect to a set of latent data representatives. Various types of supervision signal can be introduced within the framework by replacing plain MI with several forms of conditional MI. Examples of the semi-(un)supervised algorithms that we obtain this way are a new model for manifold alignment, and a new type of embedding method that performs 'conditional dimensionality reduction'.

1. Introduction

When high-dimensional data is governed by only a small number of degrees of freedom, dimensionality reduction can be used as a standard tool to identify the 'true' sources of variability and to overcome the 'curse of dimensionality'. By replacing the original dataset with a set of much lower dimensionality, standard algorithms, such as locally linear embedding (Roweis & Saul, 2000), kernel PCA (Schoelkopf et al., 1998) and others, construct a latent space that captures the underlying degrees of freedom and allows for data representations that can be processed more easily than the original data.

While these methods show impressive results on some difficult datasets, it is also becoming increasingly clear that their unsupervised objectives – usually the preservation of pair-wise point similarities – need to be supplemented by some sort of supervision signal in order to be useful in practice. The reason is that in practice the problem of dimensionality reduction is *task-dependent* rather than generic. If we consider embedding as a pre-processing step for classification, for example, we need a low-dimensional representation that captures variability that is highly 'correlated' with class-membership. If on the other hand the task is, say, the manipulation of lighting effects in images, we need a latent space that is good at capturing exactly these instead. However, since standard embedding methods are unsupervised, they typically represent every factor of variability in *all* latent space dimensions at the same time, and can therefore be sub-optimal for the task at hand. In the classification example, capturing lighting effects in the latent space will obviously deteriorate classification performance, while a tendency to cluster latent space elements according class structure would help. The exact opposite would be the case in the image manipulation task.

Several semi-supervised embedding approaches exist that partly resolve these issues. A classical method that aims specifically at classification is linear discriminant analysis (LDA). Recently, several modifications to LDA have been suggested (see *eg.* (Goldberger et al., 2005)), that consider *local* class membership criteria as opposed to the global one used by LDA. These modifications can at times greatly improve classification performance, even though they are usually based on iterative optimization for learning. Common to both LDA and its modifications is, that they are inherently linear. Even though nonlinear extensions are possible in principle, they require to exchange the parametric projection model by a non-linear one, such as a neural network. As a result the nonlinear extensions become complicated, and parameter estimation becomes difficult. For continuous valued settings, such as in regression problems, other embedding methods exist. Classical approaches are canonical correlation analysis and partial least squares regression. A recent nonlinear approach is described by (Ham et al., 2005).

Different kinds of supervision signal for use in embedding are discussed by (Tenenbaum & Freeman, 2000) and (Memisevic & Hinton, 2005), among others. These

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006.

approaches use extra-knowledge about the training data in order to *factor out* undesirable degrees of freedom and thereby to 'clean up' the embeddings. The idea is to use extra information to specify *irrelevant* modes of variability in the data, and thereby to help the embeddings capture the remaining, relevant statistics. One advantage of this kind of approach is that it can bring to use *any* information that is available about the data – not just the supervision signal that comes from the underlying supervised learning task (such as class information in a classification problem).

In this paper we describe a rather general framework for embedding that includes plain dimensionality reduction and semi-supervised extensions, such as the ones discussed and their combinations, in a single unifying framework. Our approach uses Parzen window estimates of mutual information and conditional mutual information. While nonparametric estimates of information theoretic quantities have been used in many different contexts (see *eg.* (Principe et al., 1999) for a general overview, or (Torkkola, 2003) and (Viola et al., 1996) for applications in embedding contexts), our method does in contrast to these not consider parametric transformations between the latent and the observable space. Instead, in the spirit of modern embedding methods such as locally linear embedding or kernel PCA, we parameterize an embedding objective directly in terms of the latent data representatives themselves. Advantages of this approach are that (a) it is naturally nonlinear, (b) it only depends on a single group of parameters (the latent representatives), and as a result does not require alternating optimization schemes akin to the EM-algorithm, (c) it allows us to include side-information and thereby to supplement the embeddings with supervision signals. Even though it lacks an explicit transformation, however, our approach does – in contrast to locally linear embedding, kernel pca, stochastic neighbor embedding (Hinton & Roweis, 2003), or the classic Sammon mapping (Sammon, 1969) – generalize learned embeddings to previously unseen inputs, and also to unseen latent space elements.

Our method shares this ability with the methods described in (Meinicke et al., 2005) and (Lawrence, 2004), both based on non-parametric regression. In the special case where we do *not* make use of any kind of side-information, our method can be interpreted similarly as performing a kind of non-parametric regression. A main difference of this work is that we consider also the introduction of several kinds of supervision signal to perform dimensionality reduction in task-dependent ways.

2. Kernel Information Embedding

Dimensionality reduction can be defined as the task of finding a set of low-dimensional representatives $\{z_i\}_{i=1\ldots N}$ for a set $\{y_i\}_{i=1\ldots N}$ of high-dimensional, input data-points. A standard way to solve this problem is to define an objective function that captures the preservation of inter-point similarities and to optimize this objective wrt. the z_i. Once an embedding is found, depending on the application, it might be necessary to generalize it to unseen elements. In (inductive) classification tasks, for example, we need a 'backward'-mapping $g(y)$ from the data-space to the latent space in order to get the embeddings for test points. In applications such as noise-reduction an additional 'forward'- mapping $f(z)$ is required, that can map latent space elements back to the data-space. Not all existing embedding methods naturally provide these mappings, but several heuristics have been proposed for both (see *eg.* (Bengio et al., 2003) for backward-, and (Kwok & Tsang, 2003) for forward-mappings).

Let us assume that the data set is a sample from a random variable Y and assume in addition that there exits a (possibly nonlinearly) related variable Z, that captures the main underlying degrees of freedom in Y. Then an intuitively appealing criterion for embedding is the *mutual information* $I(Y;Z)$ between the data distribution and the distribution over low-dimensional codes:

$$I(Y;Z) = \int p(y,z) \log \frac{p(y,z)}{p(y)p(z)} \; dy \; dz, \qquad (1)$$

with $p(y), p(z)$ the data- and latent space-densities, respectively. What makes MI appealing, is that it captures exactly what we want a low-dimensional data representation to preserve when using it to replace the data. Intuitively, MI answers the question: "What, on average, can the latent representation tell us about the data and vice versa?" Recall, that we can express MI also in terms of entropies (Cover & Thomas, 1990) as:

$$I(Y;Z) = H(Y) + H(Z) - H(Y,Z), \qquad (2)$$

where $H(\cdot)$ denotes (differential) Shannon entropy. In practice, MI and other entropy based quantities are not often used for embedding, because they are hard to evaluate for all but simple distributions, such as the Gaussian.

Instead of trying to evaluate the involved high-dimensional integrals directly, we suggest using an estimate of the entropies, based on a kernel estimate of the underlying densities: Using some spherical kernel

functions $k(x, x')$ and $k(y, y')$, we can obtain an estimate of the joint density over the input and latent space as:

$$\hat{p}(Y = y, Z = z) = \sum_i k(z, z^i)k(y, y^i) \tag{3}$$

with marginals $\hat{p}(Z = z) = \sum_i k(z, z^i)$ and $\hat{p}(Y = y) = \sum_i k(y, y^i)$. (Note that for convenience we include any normalization constants in the kernel functions themselves.) Using these density estimates we can obtain estimates of an *entropy* as follows:

$$\begin{aligned} H(Z) &= -\int p(z) \log p(z)\, dz && (4)\\ &\approx -\frac{1}{N}\sum_i \log p(z^i) && (5)\\ &\approx -\frac{1}{N}\sum_i \log \sum_j k(z^i, z^j) =: \hat{H}(Z) && (6) \end{aligned}$$

We are making two approximations: First, we approximate the entropy by the negative log-likelihood of a sample (Eq. 5), and secondly, we plug in our kernel estimate $\hat{p}(z)$ for the true probability density function $p(z)$ (eq. 6). Analogously, we obtain estimates $\hat{H}(Y)$ and $\hat{H}(Y, Z)$ for the entropy of Y and the joint entropy of the latent/observable space, respectively. In particular, for the latter we have $\hat{H}(Y, Z) = -\frac{1}{N}\sum_i \log \sum_j k(y^i, y^j)k(z^i, z^j)$.

Now, plugging the entropy estimates into equation 2, we obtain an estimate of the mutual information itself:

$$\begin{aligned} \hat{I}(Y, Z) &= \hat{H}(Y) + \hat{H}(Z) - \hat{H}(Y, Z) && (7)\\ &= \Omega_0 - \frac{1}{N}\sum_i \log \sum_j k(z^i, z^j) && (8)\\ &\quad + \frac{1}{N}\sum_i \log \sum_j k(z^i, z^j)k(y^i, y^j) \end{aligned}$$

where we absorb constants that do not depend on latent space elements into Ω_0.

The MI estimate (Eq. 7) is a function of the set of latent data representatives. In order to perform dimensionality reduction, we can therefore maximize $\hat{I}(Y, Z)$ with respect to the latent space elements themselves. Any gradient-based optimization method could be used for this purpose. The gradient of $\hat{I}(Y, Z)$ decouples into the sum of two terms which are readily shown to be (we show the gradient wrt. to a single element z^l):

$$\frac{\partial \hat{H}(Z)}{\partial z^l} = -\frac{1}{N}\sum_j \left(\kappa_Z^l + \kappa_Z^j\right) \frac{\partial k(z^l, z^j)}{\partial z^l}$$

where we abbreviate $\kappa_Z^i = \frac{1}{\sum_k k(z^i, z^k)}$; and

$$\frac{\partial \hat{H}(Y, Z)}{\partial z^l} = -\frac{1}{N}\sum_j \left(\kappa_{YZ}^l + \kappa_{YZ}^j\right) k(y^l, y^j) \frac{\partial k(z^l, z^j)}{\partial z^l}$$

where $\kappa_{YZ}^i = \frac{1}{\sum_k k(y^i, y^k)k(z^i, z^k)}$. If we use RBF kernels $k(z^l, z^j) = \exp(-\frac{1}{h}\|z^l - z^j\|^2)$, we have furthermore: $\frac{\partial k(z^l, z^j)}{\partial z^l} = -\frac{2}{h}k(z^l, z^j)(z^l - z^j)$.

Note that, since the latent space elements are free to move, any change in the latent space kernel bandwidth can be compensated by rescaling the elements. The choice of the latent space bandwidth h is therefore arbitrary, and we set $h = 1.0$ in the following. (Note, however, that the data-space bandwidth is *not* arbitrary and has to be set by hand, cross-validation, or some heuristics).

2.1. Regularization

A closer inspection of Eq. 7 shows that there is a trivial way to maximize the objective, which is to drive all latent representatives infinitely far apart from one another. To obtain a maximum that is meaningful, we therefore need to constrain the range of the latent elements to a finite region of the latent space, or to impose some kind of power constraint on Z. A natural constraint is to require $\mathrm{tr}(C_Z) <= R$, where C_Z denotes the latent covariance matrix and R is a maximally allowable power. In practice, setting $C_Z = \frac{1}{N}Z^T Z$ then simply amounts to an L2-penalty on the matrix of latent representatives. In practice, the effect of this penalty is that it controls the overall scale of the latent elements and thereby regularizes the model. Interestingly, such a restriction on the allowable latent space power also parallels the power restriction on the input variables in the context of optimizing a channel capacity (Cover & Thomas, 1990). Instead of using an L2-penalty, however, depending on how we assume the data is distributed along the manifold, we can also restrict higher order moments, or force the latent space elements in other ways to reside within a confined region of the latent space. A simple way to implement such a constraint in practice is by adding $\frac{\lambda}{N}\mathrm{tr}(C_Z)$ (or similarly for other constraints) to the objective function, where λ controls the amount of regularization.

This way of controlling the model complexity can also be used during optimization as a way to find a good local optimum of the objective function: By setting λ to a large value in the beginning and decreasing slowly during optimization, we can perform to a kind of deterministic annealing to find and track a good optimum of the objective function. We have used this procedure in some of our experiments, and have found

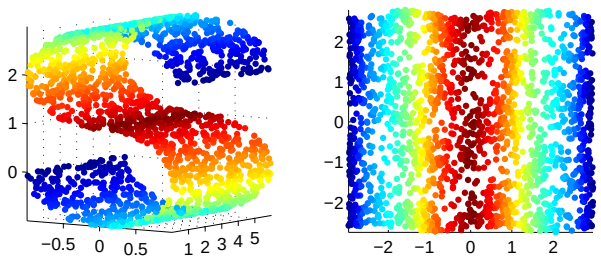

Figure 1. 'S-curve' dataset (left). Embedding (right).

that it is a very effective way to overcome problems of bad local maxima, avoiding the necessity, for example, to perform multiple restarts or other tricks to obtain good solutions. Annealing makes the experiments more or less invariant to initial conditions (up to a rotational invariance inherent in the objective function). For more details on the optimization, see also the respective sections themselves.

Figure 1 shows a proof-of-concept example of KIE applied to a two-dimensional 'S'-curve dataset embedded in three dimensions. The dataset consists of 2000 points sampled uniformly from the 'S'-curve. We trained a two-dimensional embedding (shown on the right, using a color-coding scheme that reveals the underlying mapping), with an L4-penalty on the latent space, that encourages alignment with the coordinate axes[1]. We used an RBF kernel with $h = 10.0$. To train the model, we have initialized the z_i to small random values, and we have used the annealing procedure described above, with λ set to 0.1 initially and multiplied by 0.8 for 20 steps. In each step we optimized Eq. 7 using simple unconstrained optimization. We have repeated the experiment ten times with different random initializations and with no noticeable difference in the result (except that in some cases the embedding is rotated by 90 degrees in the latent space).

We have also experimented with a wide range of kernel bandwidths, also without significant differences in the performance. Note that, regardless of the bandwidth, the kernel density estimate necessarily underestimates the true density in this dataset (which is infinite within a two-dimensional 'sheet' of the 3-dimensional space). Even though the estimate is bad, however, KIE performs well and finds the underlying structure. The reason for this robustness wrt. the choice of bandwidth, and the ability to cope with this dataset in general, is that the embedding does not care for an accurate *density estimate* itself, as much as it cares for *similarities* in data-space. For the same reason, KIE is relatively immune against problems with density estimation in high-dimensional spaces, as we show in Sections 3 and 4. As long as the kernel is able to reflect the similarity structure in the data, arranging the latent elements accordingly will improve the objective – which is all we need for this task.

[1] We would like to point out, that this would be rather hard to achieve using, for example, a spectral method.

2.2. Generalization

It is straightforward to generalize the embeddings to previously unseen data and to new latent space elements. To derive the backward-mapping g, first note that our entropy-estimate (Eq. 6) is simply the sample-average of the *information content* based on a kernel estimate of the underlying density. Likewise, the embedding of the training data is computed by maximizing the average 'mutual information'-content (Eq. 7) between the training sample and its corresponding set of latent space elements. Therefore, to find the optimal embedding for an unseen training point y^{test}, we can search for that latent space element, whose estimated mutual information-content wrt. y^{test} is maximal. That is, we can define

$$g(y^{\text{test}}) = \arg\max_z \hat{I}(y^{\text{test}}, z) \tag{9}$$

where

$$\hat{I}(y^{\text{test}}, z) := \log \sum_j k(z, z^j) k(y^{\text{test}}, y^j) - \log \sum_j k(z, z^j).$$

Analogously, to generalize the forward mapping to a new latent space element z^{test}, we can set:

$$f(z^{\text{test}}) = \arg\max_y \hat{I}(y, z^{\text{test}}). \tag{10}$$

Both the backward- and the forward-mappings require the solution of a nonlinear, but rather mild (in particular low-dimensional), optimization problem. At test time, the same restrictions on the latent space power apply as before, and they can be transfered without change from the corresponding training problem.

2.3. Related Methods

It is interesting to note that we can view Eqs. 9 and 10 as maximizing *conditional log-likelihoods*, which shows that we can interpret KIE as a kind of non-parametric regression method. The objective itself (Eq. 7) can then be viewed as an estimate of $\log p(Y|Z)$, and defines a kind of regression on latent variables.

Using non-parametric regression for embedding has been suggested originally by (Meinicke et al., 2005) and (Lawrence, 2004). A difference to the former method is that KIE maximizes likelihood instead of a reconstruction error, and therefore does not postulate

an explicit noise model in the data space. A difference to the latter method is that the KIE objective function (Eq. 7) scales only quadratically (instead of cubically) with the number of data-points and can therefore be applied to much larger datasets. An obvious advantage over both these methods, however, is that KIE's information theoretic view suggests several ways of introducing supervision signals into the embeddings, as we discuss in more detail in the following sections.

3. Conditional Embeddings

A well-known problem with standard embedding methods is that they typically compute representations that capture generic similarity structure in the data, but do not necessarily reflect the kind of variability that is important for a specific task. In this case, supervision signals can help adjust the methods so that they actually capture the kind of variability we are interested in (see *eg.* (Tenenbaum & Freeman, 2000)). One way we can apply extra information is by using it to *factor out* the kind of variability we are *not* interested in, and thereby to focus the embedding on the kind of variability that is important to the actual task at hand.

To cast this idea in information theoretic terms, let us introduce a random variable X that captures any available extra information about the data. Then the task of computing an embedding that reflects all variability in the data *except* for the variability in X has a natural solution: We need to maximize the *conditional* mutual information between Y and a latent variable Z, *given* X. Conditional MI is defined as (Cover & Thomas, 1990):

$$\begin{aligned} I(Y;Z|X) &= H(Y|X) - H(Y|Z,X) \qquad (11) \\ &= H(X,Z) - H(X,Y,Z) + \Omega_1, \end{aligned}$$

where Ω_1 contains terms that do not depend on Z. Conditional MI also comes with an appealing intuition; it answers the question: "What, on average, can the random variable Z tell us about Y, and vice versa, *given* that we know X." By minimizing 11, we obtain factors that reflect the kind of variability in the data that is *not* represented by, or correlated with, X.

Practically, we can reserve one or more latent space dimensions as the conditioning (the 'X')-space and deliberately position the latent elements x_i in this space according to the kind of variability that we want to encode. To encode, for example, grouping structure, we can cluster the elements x_i accordingly, or use an orthogonal ('one-hot'-) encoding; to express knowledge about a natural ordering in the data we can arrange the elements according to this ordering; etc. After

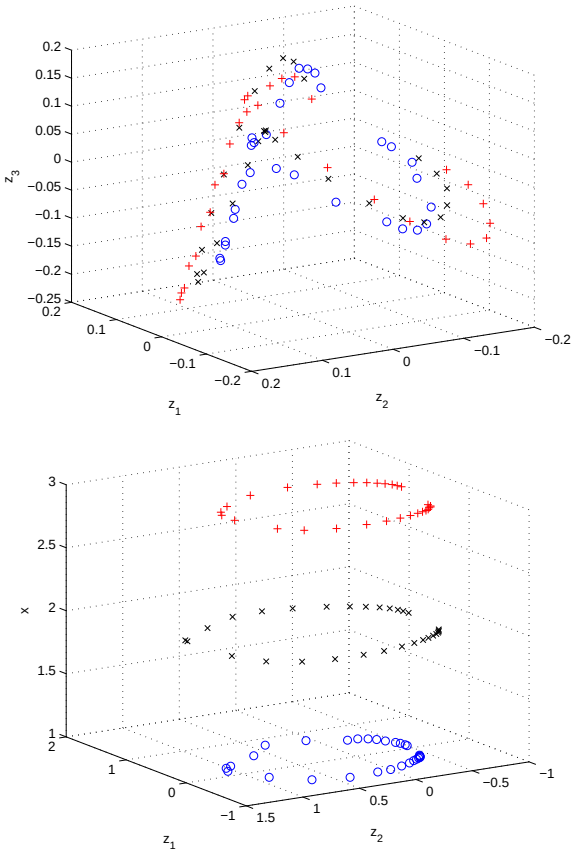

Figure 2. Uninformed embeddings (top) vs. informed embeddings (bottom).

placing the conditioning latent elements and defining a kernel for this space, maximizing the estimate $\hat{I}(Y;Z|X)$ yields the elements z_i that capture the remaining degrees of freedom. Since, similar to MI, conditional MI decomposes into entropy terms (Eq. 11), we obtain an estimate $\hat{I}(Y;Z|X)$ in complete analogy as before, with a gradient that decouples in the same way, too. (Note, that the same restrictions regarding latent space power etc. apply). Similarly as before, the model is trained by simple gradient based optimization.

Figure 2 illustrates the advantage of an 'informed' latent space as opposed to an 'uninformed' one, using images from the COIL-database (Nene et al., 1996). The data consists of images of objects with varying orientations. (See figure 3 for some examples.) The top row of figure 2 shows a three-dimensional embedding obtained by applying kernel PCA (Schoelkopf et al., 1998), using an RBF kernel with $h = 5{\cdot}10^7$, on a subset of images depicting toy-cars. The three different types of car are shown by using different symbols. The plot shows that the embedding captures the presence of three individual one-dimensional degrees of freedom. However, variability across classes is represented si-

multaneously in the same latent space as variability that is due to rotation. As a result, both degrees of freedom are intermixed, and in particular the fact that all three objects vary simply by rotation is not obvious. The bottom row of the figure shows the embedding of the same objects, where class-membership has been *factored out* prior to computation of the latent representation. The 'X'-space (the vertical axis in the figure) was obtained by simply placing representatives for the objects in three different clusters (located at $1.0, 2.0$ and 3.0). For training we initialized to small random values, fixed $\lambda = 0.6$ and used the same h as for kernel PCA. (Annealing was not used in this experiment.) As before, we used simple gradient based optimization to maximize the objective. The resulting conditional embedding, given by the remaining two axes, captures the rotational degree of freedom that remains after class-membership has been factored out.

In a similar experiment, we trained the model on a subset of 30 images for each of the first 5 objects from the same database (ref. figure 3 for the objects). In one setting we used plain dimensionality reduction, in the other we informed the model about grouping structure by using an orthogonal ('one-hot'-)encoding for class-membership and a Gaussian kernel with bandwidth 1.0 to obtain a latent 'class'-space. We then picked for each object from the first class (the 'duck'-images) the nearest neighbors in each of the remaining four classes in the 'Z'-space. The four chosen neighbors are shown below each duck-image in figure 3. Since class membership has been factored out, the embedding-space captures the remaining, rotational, degree of freedom very cleanly. We repeated the same neighborhood-picking procedure in the original data-space and in an uninformed latent space, that we obtained running plain KIE with the same settings as before. For one of the classes ('wood block'-images) data-space similarity also captures the rotational degree of freedom. However, when the degree of freedom is too subtle (as in the other three object classes), it is not reflected in the data-space anymore. While it is not surprising that data-space similarity fails in this context, note that any uninformed latent space does, too, (see figure 3, bottom). The reason is that without class-membership factored out, any generic embedding algorithm tries to capture all factors underlying the overall similarity structure in the data. In this setting, for example, it tends to cluster the data according to class membership, while at the same time trying to capture the rotational degree of freedom in the *same* latent space. Ironically, neighborhood in the plain latent space consequently fails entirely to capture *any* of the structure present in the data. Neighborhood in the the informed latent space on the other hand is meaningful, after some of the 'nuisance' structure has been factored out.

Figure 3. Nearest neighbors in {informed latent space (top), data-space (middle), plain latent space (bottom)}.

4. Joint Embeddings and Feature Extraction

A different kind of semi-supervised embedding problem is the following: We are given pairs of points from two different manifolds in correspondence (possibly in two different spaces), and our goal is to find a single embedding for these, ie. we are looking for one latent representative for each *pair* of input points. (Ham et al., 2005) discuss this kind of problem in the context of spectral methods; the setting is also related to canonical correlation analysis and many existing non-linear extensions.

We can phrase this problem in information theoretic terms as follows: Given two observed random variables X and Y, that are known to share some (unknown) common structure, we are looking for a latent variable Z that *minimizes* the conditional mutual information $I(X;Y|Z)$. This type of conditional MI answers the question: "What can the (observed) variable X tell

us on average about the (observed) variable Y, and vice versa, *given* that we know Z?" Minimizing it with respect to Z yields latent factors that capture the remaining information, *ie.* the kind of information that is *shared* between X and Y. We have (Cover & Thomas, 1990):

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z) \quad (12)$$
$$= H(X,Z) + H(Y,Z) - H(Z) - H(X,Y,Z).$$

Again, we can get an estimate $\hat{I}(X;Y|Z)$ by plugging in the corresponding entropy estimates, making use of the decomposition of conditional MI. The gradient again decouples, too, and we can use gradient based optimization, with same power-constraint as before[2].

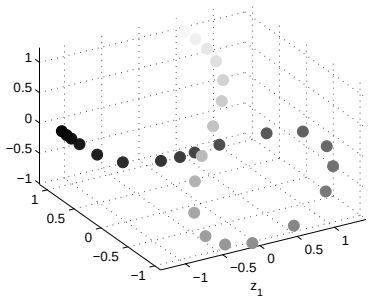

Figure 4. A joint embedding.

Note that, by minimizing this objective, we obtain latent factors Z that make $X - Z - Y$ become approximately a 'Markov chain'. Cast in terms of feature extraction, we are looking for features, such that given these, the data is not needed to predict the output, and vice versa. In contrast to other information theoretic approaches to feature extraction, such as the information bottleneck method (Tishby et al., 1999), or continuous valued extensions, we obtain arbitrary continuous nonlinear and non-Gaussian features.

As an example of a manifold alignment, we show a three dimensional representation of a *joint* embedding of the first two object classes from the previous experiment in figure 4. As before, training is performed simply with gradient based optimization, with λ being annealed, in this case starting with 1.0 and by multiplying with 0.8 for ten steps. We used RBF kernels with $h = 3 \cdot 10^7$. The single, one-dimensional but 'curly', degree of freedom underlying the variability in the data becomes visible. Note that here each latent space element is the representative of *two* data-points in correspondence.

One potential application of this kind of joint embedding is feature extraction in supervised learning. We consider a simple regression task in figure 6. The figure shows the training data, consisting of 4 (input, output)-pairs. In general, this training set is obviously

[2]Note in particular, that without any restrictions on Z, we can minimize $I(X;Y|Z)$ by making Z equal to X or Y.

Figure 6. Training set for regression task.

ridiculously low for the task of learning any reasonably mapping between the two high-dimensional image spaces. However, since the data is very strongly structured, we can make progress, if we can make use of the unlabeled data in some way. In this experiment we have used a transductive setting: We have embedded both the testing- and training-data in a joint three-dimensional latent space, where we have simply added the objectives Eq. 7 and (the estimate of) Eq. 12 in order to achieve an *alignment* for the training-set and an *embedding* (in the same space) for the test set. For training we initialized to small random values, and annealed λ by multiplying by 0.9 for 10 steps, starting with $\lambda = 1.0$ and performing simple gradient based optimization in each step. We used the same kernel bandwidth as in the previous experiment. Given the joint embedding, we can now define a function by first mapping from the input- to the embedding space and then from the embedding to the output-space. We have used a simple non-parametric regression function for this purpose, with the result shown in figure 5. The top row depicts the input-, the row below the output-data. Note that the model has confused two elements (5^{th} from the left, and rightmost) by essentially assuming an object of the opposite orientation at the corresponding position. While this solution is not quite correct it does help reduce squared error. The bottom row of the figure shows the application of a nonparametric regression function directly on the high-dimensional input-output data for comparison. The performance is necessarily bad, since without the availability of unlabeled examples, the best any regression model can do here is trying to interpolate the (output-) training cases. We obtain a reduction in average reconstruction error from approximately 2936 to 2630 with the joint embedding.

5. Discussion

MI is a useful concept for embedding, because it captures exactly the intuition of what we want to preserve when performing dimensionality reduction. *Conditional* mutual information has to the best of our knowledge not been applied in this way before, but it can be useful, since it provides elegant ways to include side-information. Although conditional MI poses the same estimation problems as MI for general distribu-

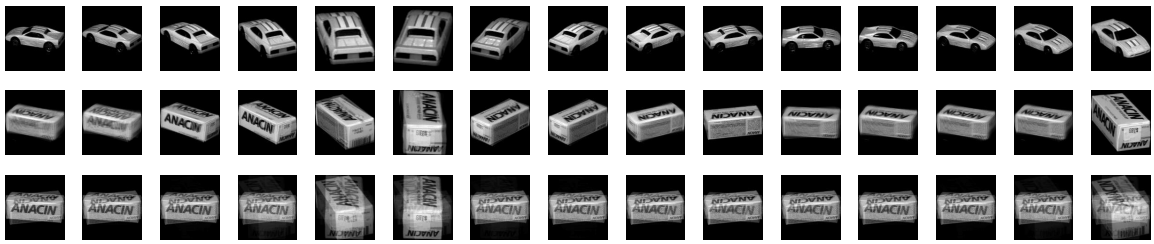

Figure 5. Semi-supervised regression example.

tions, the completely nonparametric kernel estimation framework that we suggest provides a simple way to overcome this problem. Further ways of making use of conditioning information, and also of combining the ones we have described here are possible. The modular structure, owed to the decomposition into entropy terms, can be useful for exploring these, since it makes it easy to combine the involved information theoretic quantities.

Acknowledgments

We thank Geoff Hinton, Jenn Listgarten, Ted Meeds and Nati Srebro for helpful discussions. This work was supported in part by a Government of Canada Award.

References

Bengio, Y., Paiement, J., Vincent, P., Delalleau, O., Roux, N. L., & Ouimet, M. (2003). Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. *Adv. in Neural Information Processing Systems 16*.

Cover, T., & Thomas, J. (1990). *Elements of information theory*. John Wiley and Sons.

Goldberger, J., Roweis, S., Hinton, G., & Salakhutdinov, R. (2005). Neighbourhood components analysis. *Adv. in Neural Information Processing Systems 17*.

Ham, J., Lee, D., & Saul, L. (2005). Semisupervised alignment of manifolds. *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*.

Hinton, G., & Roweis, S. (2003). Stochastic neighbor embedding. *Adv. in Neural Information Processing Systems 15*.

Kwok, J. T., & Tsang, I. W. (2003). The pre-image problem in kernel methods. *Proc. of the Twentieth Intern. Conf. on Machine Learning*.

Lawrence, N. D. (2004). Gaussian process latent variable models for visualisation of high dimensional data. *Adv. in Neural Information Processing Systems 16*.

Meinicke, P., Klanke, S., Memisevic, R., & Ritter, H. (2005). Principal surfaces from unsupervised kernel regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*, 1379–1391.

Memisevic, R., & Hinton, G. (2005). Multiple relational embedding. *Adv. in Neural Information Processing Systems 17*.

Nene, S. A., Nayar, S. K., & Murase, H. (1996). *Columbia object image library (coil-20)* (Technical Report).

Principe, J., Xu, D., & Fisher, J. (1999). Information theoretic learning. *Unsupervised Adaptive Filtering* (pp. 265–319). John Wiley & Sons.

Roweis, S., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*, 2323–2326.

Sammon, J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, *18*, 401–409.

Schoelkopf, B., Smola, A., & Mueller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, *10*, 1299–1319.

Tenenbaum, J. B., & Freeman, W. T. (2000). Separating style and content with bilinear models. *Neural Computation*, *12*, 1247–1283.

Tishby, N., Pereira, F., & Bialek, W. (1999). The information bottleneck method. *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*.

Torkkola, K. (2003). Feature extraction by nonparametric mutual information maximization. *Journal of Machine Learning Research*, *3*, 1415–1438.

Viola, P., Schraudolph, N. N., & Sejnowski, T. J. (1996). Empirical entropy manipulation for real-world problems. *Adv. in Neural Information Processing Systems 8*.