Centre for Theoretical Neuroscience
Technical Report

UW-CTN-TR-20090204-001
February 4, 2009

Intrinsic Divergence for Face Recognition

Yichuan Tang and Xuan Choo

Centre for Theoretical Neuroscience, Waterloo, ON. http://compneuro.uwaterloo.ca/cnrglab

Intrinsic Divergence for Face Recognition

Yichuan Tang
University of Waterloo
Waterloo, ON

Xuan Choo
University of Waterloo
Waterloo, ON

December 19, 2008

1. Introduction

Face and object recognition can be viewed as statistical classification due to the copious amounts of intra-class variations. Existing face and object recognition algorithms either attack it as a pattern classification problem, or use feature extraction as an intermediate step [28, 29, 46, 29]. The main challenge is that the probability density which defines a face or an object in $\Re^d$ is hard to model due to the extremely high dimensionality and to the highly nonlinear density distribution (as can be seen by translation/scaling [41]). Feature Extraction methods are a way of dealing with this problem, but they often create inter-class overlappings in the feature domain. Due to the high dimensionality of the data and the fact that there are only finite types of natural transformations that can occur to any given face, it is only reasonable to utilize dimensionality reduction techniques. By natural transformations, we take this to mean the changes to a face in our everyday environment, such as lighting intensity or direction changes, pose (3D rotation and 2D similarity), expression changes, as well as small occulsions such as glasses or hair placement. A finite number of transformations mean that intra-class points lie on a manifold with an intrinsic dimension equal the number of transformations [40]. Intuitively all face points that lie on an intra-class manifold should belong to the same person. We therefore deem the intrinsic divergence (henthforth refered to as *InDiv*), which is a measure of similarity, to be 0 for all points on an intra-class manifold.

In section 2, related work using distances is examined. Various other recognition algorithms are viewed in the framework of the intrinsic divergence. We also demonstrate why the intrinsic divergence is plausible for recognition by looking at 3 assumptions, which we show are most likely to be true. In section 3, we survey ways to learn the intra-class manifolds. The intricacies of the recent autoassociative network is also examined and we use it to find the intrinsic divergence. Section 4 proposes two possible improvements by using neural networks to find the *InDiv*.

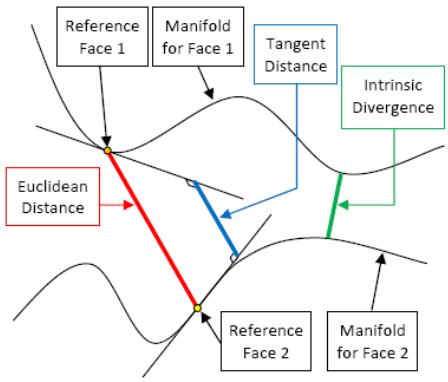


Figure 1. Visualizing euclidean distance, tangent distance, and intrinsic divergence in state space

2. Intrinsic Divergence

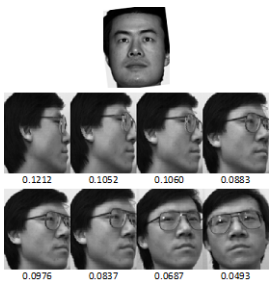
Distances and divergences provide a measure of dissimilarity between vectors and functions in high dimensional space and is often involved in statistical classification. In nonparametric cases, the distance measure between points is at the core of methods such as kNN, CART, SVM, and kernel density estimators. In parametric cases such as LDA and QDA, the Mahalanobis distance provides the first step in computing the likelihood [11, 16]. In information theory, KL-divergence is a functional that can measure the dissimilarity between two probability distributions, among other uses. In this project, we formulate the concept of intrinsic divergence as a measure of the dissimilarity between two manifolds, where each manifold is composed of class specific faces under various natural transformations. More formally, we define

$$InDiv = \min_{\hat{\mathbf{x}}} \|\mathbf{x} - \hat{\mathbf{x}}\|^2 \tag{1}$$

Where $\mathbf{x}$ is the test point and $\hat{\mathbf{x}} \in \mathcal{M}$, and $\mathcal{M}$ is an intra-class manifold. The intuition behind intrinsic divergence is simple: the real or intrinsic dissimilarity between two faces should be the minimum distance between all possible natural transformations. The $InDiv$ is always ≥ 0 and is 0 only if two points lie on the same class specific manifold. This concept is related to tangent distance [41], where the tangent distance between two data vectors is the minimum distance between their respective tangent hyperplanes. There are two major differences between intrinsic divergence and tangent distance: 1) the manifold is not restricted to be linear and 2) whereas known transformations such as translation, rotation, scaling, and distortion must be specified in order to construct the tangent hyperplane, manifold learning for $InDiv$ is unsupervised and is more flexible (to facial expression and nonlinear lighting changes, etc).

2.1. InDiv Assumptions

There are 3 assumptions that must be true in order for $InDiv$ to make sense. First, for any 2 faces from 2 classes (2 persons), there are no natural transformations that makes the $InDiv$ between them 0. Second, for any 2 faces within the same class, there exist a natural transformation which would make the $InDiv$ close to 0. Third, for any 2 faces from 2 classes, the $InDiv$ is the smallest when the pose, lighting conditions, and facial expression are approximately the same. Figure 2 provides some demonstration to these three assumptions respectively. They only provide



(a) Assumption 1

(b) Assumption 3

(c) Assumption 2

Figure 2. The number below each face is the mean pixel squared error between the test (top) face and training faces. Note that the value of $InDiv$ cannot be treated absolutely. For example, the correct match in (c) has higher error than the wrong matches in (b).

illustrations to support the validity of these 3 assumptions, and the more sound proofs will be left for future work.

2.2. Project-Out Operation

If the class specific manifold $\mathcal{M}$ is linear, then

$$InDiv = \|\mathbf{x} - \hat{\mathbf{x}}\|^2 \tag{2a}$$

$$= \|\mathbf{x} - U_d U_d^T \mathbf{x}\|^2 \tag{2b}$$

where U_d is column matrix of the orthonormal eigenvectors which span $\mathcal{M}$. In order to extend $InDiv$ to a nonlinear $\mathcal{M}$, we define a ProjOut operator $g(\mathbf{x}) : \mathbf{x} \rightarrow \hat{\mathbf{x}}$, where $\mathbf{x} \in \Re^d$, $\hat{\mathbf{x}} \in \Re^d$, and $\hat{\mathbf{x}}$ is the closest point to $\mathbf{x}$ on $\mathcal{M}$. Therefore,

$$InDiv = \|\mathbf{x} - g(\mathbf{x})\|^2 \tag{3}$$

In essence, to find the $InDiv$, we first need to find $\hat{\mathbf{x}}$ using g($\mathbf{x}$). Figure 3 provides a graphical visualization of the ProjOut operation. The top right plot shows a 1D linear manifold (pink) embedded in $\Re^2$. All regions in $\Re^2$ which have the same colour should project to the same $\hat{\mathbf{x}}$ on the 1D linear manifold. The top left plot shows what an optimum ProjOut operator would look like. The z-axis is the latent variable that must be estimated. The regions of same colour must all be level since they need to be mapped onto the same latent variable. section 3 discuss

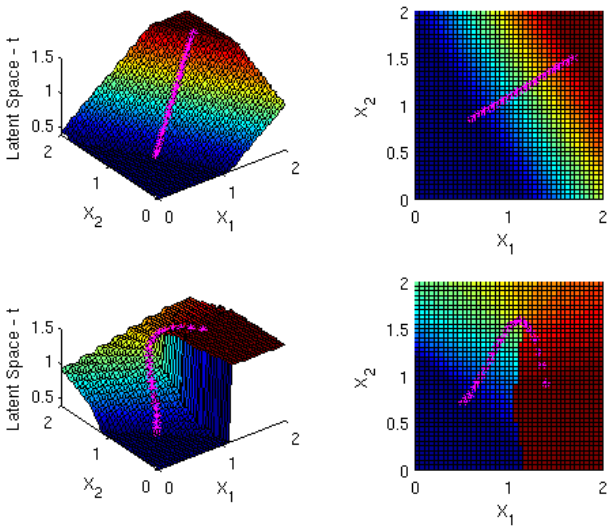


Figure 3. Optimum ProjOut mapping for linear and nonlinear 1D manifold in 2D space. See text for details

the ProjOut operation as it corresponds to decoding. Notice that eq. 2 performs this operation perfectly with the exception of the end regions of the manifold. The bottom plots shows the mapping of the optimum ProjOut operator on a nonlinear 1D manifold. The function is much more complicated. The goal of this project is to look at the plausibility of using autoassociative encoders to approximate $g(\mathbf{x})$.

2.3. Related Algorithms

There are numerous other algorithms used for face recognition. We briefly discuss the 3 big genres and how they are essentially trying to approximate the $InDiv$. The first genre is the subspace methods that include Eigenfaces and Fisherfaces [45, 3]. These methods find a linear subspace to project the high dimensional data to, and then perform classification in the lower dimensional space. The Eigenface approach performs PCA on face data, and after discarding the first 3 eigenvectors (found by trial and error), classification is performed on the subspace composed by the remaining eigenvectors. The reason for discarding the first 3 eigenvectors is because they typically comprise of variations in lighting and pose changes. The method basically assumes that natural transformations results in linear variations of the data. If the variations on all faces were linear and span only 3 dimensions, then the Eigenface approach would find the $InDiv$. The second genre consists of alignment based approaches. This includes the 3D morphable model [7] and Active Appearance Models [9]. These methods require iterative nonlinear optimization to align or match the model to the test image. This method corresponds to the traversal along the intra-class manifold to minimize the $InDiv$. The third genre is based on feature extraction applied at interesting locations within the face. This includes algorithms such as Elastic Bunch Graph Matching [46], and local descriptor methods [28, 29]. The key characteristic of these approaches is that the filters are local (over small region, e.g. corner of eyes, nose, mouth) and the features extracted are invariant to classes of variation such as illumination, rotation, scaling, and small deformations. This corresponds to transforming the data to the feature space and this will be further discussed in section 4.

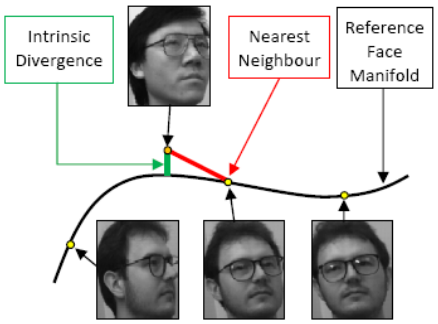


Figure 4. Nearest Neighbor suffers from the curse of dimensionality and is generally not the same as $InDiv$.

3. Manifolds

As the amount of training sample data approaches infinity, the $InDiv$ becomes nearest neighbor distance. However, since a large number of prototypes is both hard to obtain as well as computationally expensive to search, a nonlinear manifold must be learned from the data in order to find the $InDiv$. Figure 4 shows that when data points are limited, NN distance $\neq InDiv$ in general. There are many techniques for learning manifolds such as LLE [35], Isomap [44], SOM [24], GTM [6], Eigenmaps [32], SNE [18], NCA [14], and Autoassociative encoders [17]. Bengio et al. have nicely fit most of these methods into the same eigendecomposition framework [4]. Qualitatively speaking, most of these algorithms could be divided into the generative or non-generative types. The non-generative approaches LLE, Isomap(MDS), and SNE tackle the problem by learning some property in original data space (the local linear reconstruction weight matrix W in LLE, the geodesic distance in Isomap, the $\Pr(\cdot)$ of being a neighbor in SNE), and then put each point in a lower dimensional space such that these properties are as best preserved as possible. On the other hand, generative approaches look at the manifold that has been generated by a set of latent variables which are in low dimensional space. This assumption guarantees a generation/decoding step which allows the latent variable to generate a data vector in the original space. PCA, GTM, and Autoassociative Encoders (henceforth referred to as AE) can all be classified as the generative type. Since non-generative manifold algorithms lack the ability to decode back into the original space, it makes them poor choices for the ProjOut algorithm. Since PCA is only linear, we will focus on utilizing AE to compute the $InDiv$.

3.1. Autoassociative Encoders

Autoassociative networks originated in works in Connectionism and neural network research [20, 2, 23, 25]. If however, a feed-forward architecture is used with a bottleneck layer (a layer with a smaller number of nodes than the inputs and outputs), then an compression/encoding followed by decoding is possible. Applications of these networks include content addressable memory, noise suppression, and associative memory. It is known that a multilayer perceptron with 2 layers of weights can be trained to minimize reconstruction SSE and therefore is equivalent to PCA [8, 33]. These types of networks are essentially linear AEs. When presented an unseen vector in original data space, the network projects the vector onto the subspace spanned by the 1st layer weights. The 2nd layer weights decode back into the original data space, and the reconstructed vector is the

closest point to the original vector constrained by the subspace. Therefore, for linear manifolds, a PCA network would be a perfect ProjOut operator. If more layers with nonlinear activation nodes are added to the 2 layer PCA network, nonlinear autoassociative encoding can be achieved [5]. See [5, 26, 10] for graphical visualizations of these networks.

These AEs (nonlinear PCA networks) have been used to reduce the dimensionality of data [26, 10]. Backpropagation and regularization were used to train the network to minimize reconstruction error. However, it was noted that due to the number of layers and nodes in the network, optimization was very prone to local minima. Reconstruction for face data was noticeably bad [10]. Recently, using unsupervised methods in a pre-training procedure, an new type of AE was found to perform superior dimensionality reduction [17].

3.2. Intricacies of RBM based AE

The new approach to AE in [17] and the reason for its superior performance lies with the fact that it uses an unsupervised learning stage called pre-training to try and find initial weights close to the global minimum. Boltzmann machines (BM) are a fully symmetrically connected recurrent network with stochastic binary units that can be used to model the joint probability density of data [1]. For any given network, an energy can be defined as:

$$E = -\sum_{i,j} W_{ij} s_i s_j \tag{4}$$

where W is the weight matrix and s_i, s_j are the nodes. The probability of any configuration or state γ is:

$$\Pr(\gamma) \propto \exp(-E_\gamma) \tag{5}$$

The dynamic update rule for the stochastic node s_i is defined as:

$$\Pr(s_i = 1) = \frac{1}{1 + \exp(-\Delta E)} \tag{6}$$

where $\Delta E = E_{s_i=0} - E_{s_i=1}$. It turns out that the stochastic updating of all the nodes or the "running" of the network is actually performing Gibbs sampling on the joint probability density defined by the network [13]. Learning is achieved by maximizing the likelihood of the data or minimizing the KL-Divergence [1]:

$$\Delta W_{ij} = \varepsilon(\langle s_i s_j \rangle_{data} - \langle s_i s_j \rangle_{model}) \tag{7}$$

Therefore, the BM is an MCMC generalization of the Hopfield network [20] and are well suited to represent the distribution of data. Modelling the distribution of data is unsupervised and it is this ability which pre-training for AE exploits.

Due to the fact that Simulated Annealing needs to be applied for proper MCMC convergence [31, 22], the BM training algorithm is extremely slow. First introduced in [42] as "Harmoniums", the restricted BM or RBM is a type of BM which has two sets of nodes, visible and hidden. It can be visualized as a 1 layer network with the hidden nodes occupying the hidden layer. Visible nodes are connected to data (e.g. pixels); while hidden nodes encode interactions or discover features among visible nodes. In an RBM, there are no visible to visible or hidden to hidden

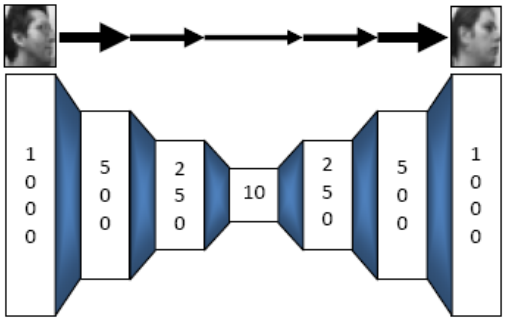


Figure 5. How AE performs the ProjOut operation. The AE is trained using the training set of person on the right. When a test face (on the left) is presented to the network, the reconstructed output will lie on the training class manifold.

connections. This property gives conditional independence to all hidden nodes given the activation of all visible nodes and vice versa. This essentially allows $\langle v_i h_j \rangle_{data}$ to be calculated without using sampling. The need for sampling to find $\langle v_i h_j \rangle_{model}$ is eliminated by replacing $\langle v_i h_j \rangle_{model}$ with an approximation $\langle v_i h_j \rangle_{recon}$. The new term is known as Contrastive Divergence and is described in detail by Hinton [19]. Experiments show that this approximation performs well and has huge reduction in learning time.

Contrastive Divergence training methods allow RBMs to model the distribution of data vectors presented to its visible nodes. In order to allow for more complexity of distribution modelling, it is proposed that a hierarchy of RBMs be learned where the inputs to the visible nodes of the higher level RBMs are the activations of hidden nodes of the level below [17]. This is exactly the pre-training process for the AE. After pre-training, the RBMs are stacked on top of one another, forming a feed-forward network from the input to the hidden layer of the top RBM. This feed-forward network forms an encoding process; where the top layer nodes can be viewed as codes or latent variables. For the decoding process, all the weights and nodes below the code layer are mirrored up to form the decoding layers. This step creates the basic autoassociative network and is called the unrolling process [17]. In order to achieve low reconstruction error, the second part of AE training consists of fine-tuning the weights. Due to the large number of weights, conjugated gradient methods are typically used to fine-tune the weights.

3.3. AE for ProjOut

Inspired by how linear PCA networks can perform ProjOut, it is natural to think that nonlinear PCA networks can do the same thing for nonlinear manifolds. However, ProjOut using neural networks might suffer from overfitting and the out-of-sample problem experienced by all neural networks. The overfitting problem can occur if the AE performs reconstruction perfectly but has an undefined behaviour for points in-between training data points. The out of sample problem exist because neural network essentially performs regression and therefore are great interpolator but terrible extrapolators. Since the RBM-based AE uses an unsupervised pre-training stage, we want to use them to see how it performs the ProjOut operation. Figure 5 demonstrates how the AE network is used to perform ProjOut operation. The training code for the AE were mainly from code provided by Hinton [36]. Modification and related optimization of code layer nodes were implemented by us.

Figure 6. Facial data reconstruction of person A with indepth rotation. The top row is the original and the bottom is the reconstructed faces. A 1000-500-250-10 (# of nodes from the 1st hidden to the code layer) AE was used with a mean pixel squared error of .014.

Figure 7. The 1st, 3rd, and 5th row contain unseen test faces. The 2nd, 4th, and 6th row contains the faces found by the AE. Notice that the encoding+decoding process constrains the newly found face to be on the manifold. See text for discussions.

To see how the AE performs on this problem, an AE was trained on 38 images of person A from UMIST face data set [15]. The variations consist of in-depth rotations. Figure 6 shows the original faces and the reconstructed faces using a 7 layer AE.

Now, we need to find out how this AE performs with unseen test faces. This is done by simply encoding and decoding the unseen test face using the AE learned earlier. The hope is that the encoding+decoding process will result in a reconstructed face which is closest to the test face but is still on the manifold defined by the AE.

Figure 7 shows the results. The 1st, 3rd, and 5th row contain various test faces. Each face is applied to the same AE that was shown in figure 6. The immediate image below is the reconstructed face. Since the AE represent the manifold of person A, all reconstructed faces look similar to person A. A striking resemblance in in-depth rotation is observed for most of the test images. This remarkable result indicates that the encoding+decoding process actually found a close face to the test face which also lies on the manifold. This suggest the applicability of using the AE to find the $InDiv$, since the $InDiv$ is the distance between a test face and the closest face to it on the manifold. The result is perfect for the test subject in the 1st row and the result is worst for the test subject in the 5th row. Possible explanation for why the result is bad for the 5th row is that the face in 5th row is very different from the face of person A. This can create an out of sample problem in which the AE suffers from a lack of ability to extrapolate very far away from training data. In particular, compare the frontal image of test subject 3 (Row 5 Col 12) and frontal of person A (Row 4, Col 12), the bright forehead of person A and the plethora of dark hair for test subject 3 makes test subject 3 really far away from the manifold of person A.

3.4. AE for Face Recognition

We tested face recognition on the UMIST and Olivetti [37] data sets by using the AE to find the $InDiv$. The UMIST data set consists of 20 individuals and the main variations are in-depth rotation. The data set is randomly divided in various portions to form the training and testing sets. 20 AEs were trained using the training data to form 20 nonlinear manifolds. For every test face, 20 ProjOut operations were performed by encoding and decoding using all 20 AEs. The $InDiv$ between the test face and a manifold is the Euclidean distance between the test face and the reconstructed face from that AE. The classifier chooses the manifold with the smallest $InDiv$ as output. A 1-NN classification results is obtained for comparison.

Table 1 shows the results between AE and NN classifiers. Except for one set of parameters, NN is better or equal to the AE. We can think of two reason for this result. First is that the transformations represented in the training set is not large enough. For example, the UMIST images are mostly in-depth rotation and Olivetti images are mostly expressions and poses. The

Table 1. Recognition results: AE based $InDiv$ and NN classifiers.

Column Labels	
PCNT	% of face data given to training
CLN	# of code layer nodes
EBP	# of epochs for backpropagation during fine-tuning
TF	# of test faces
AEM	# missed by AE
AER	AE error rate
NNM	# missed by NN
NNR	NN error rate

UMIST Data Set							
PCNT	CLN	EBP	TF	AEM	AER	NNM	NNR
50	5	10	290	10	3.45%	8	2.76%
50	10	5	290	40	13.79%	8	2.76%
50	10	10	290	9	3.10%	8	2.76%
50	10	20	290	13	4.48%	8	2.76%
50	10	30	290	10	3.45%	8	2.76%
50	15	10	290	8	2.76%	8	2.76%
30	10	10	413	27	6.54%	27	6.54%
40	10	10	353	21	5.95%	13	3.68%
60	10	10	239	3	1.26%	6	2.51%
70	10	10	180	4	2.22%	4	2.22%

Olivetti Data Set							
PCNT	CLN	EBP	TF	AEM	AER	NNM	NNR
50	10	10	200	27	13.50%	20	10.00%
60	10	10	160	21	13.13%	16	10.00%
70	10	10	120	11	9.17%	6	5.00%
80	10	10	80	8	10.00%	4	5.00%
90	10	10	40	3	7.50%	3	7.50%
70	15	10	120	20	16.67%	7	5.83%
70	5	10	120	13	10.83%	6	5.00%
70	10	30	120	10	8.33%	6	5.00%
70	10	20	120	10	8.33%	6	5.00%
70	10	5	120	11	9.17%	6	5.00%

small amount of training set data therefore caused bad generalization of the AE and thus worse results. The second reason might be due to the fact that the manifold built from raw image pixels is simply too nonlinear for the AE to be a good ProjOut operator.

4. Improvements and Extensions

In this section we propose two extensions which could improve finding $InDiv$ using the AE. The first extension uses research in computational neuroscience to build a hierarchical representation which results in more linear class-specific manifolds. The second extension looks at finding a smaller $InDiv$ by using nonlinear optimizations in the code layer of the AEs.

4.1. Vision Science

Human vision has not yet been surpassed by computer vision systems. The ease at which we can recognize the identity of family, old friends, and foe under extreme lighting, pose, and expression variations suggest that algorithms can be inspired by biological vision. However, due to the highly complex nature of the visual cortex, viable models only appeared after the seminal work by Hubel and Wiesel exploring the cortical cells of cats [21]. Their work revealed that neurons in the primary visual cortex are selective or tuned towards stimulus of different orientations, blob sizes, and spatial frequencies. It was subsequently discovered that there is a massive feed-forward pathway from the primary visual cortex (V1) to the V2, V4, and Inferotemporal (IT) cortex, forming a hierarchy [39]. Along this pathway, neurons are selective towards more and more complex shapes. In the IT, neurons that are selective towards faces has been discovered [43].

Over the decades, various computational models embodying the abovementioned ideas were developed to be applied to recognition and classification, starting with the Neocognitron for digit recognition [12]. Convolutional Neural Nets were developed for handwritten digits [27]; HMAX model for 3D recognition [34]; state of the art object matching is achieved by SIFT [28]; GLOH [29], which extended SIFT into log-polar form; the most neurobiologically plausible model was built in [39]; and the recent generalization for multi-class recognition is described in [30]. What the visual cortex has inspired is the hierarchy of increasingly more complex representation, along with increasingly more tolerance to variances (small changes in position, lighting, or deformation).

Given that the intra-class manifold for faces is highly nonlinear and makes ProjOut operation hard, it is natural to ask what if the manifold of some higher feature level is more linear and has reduced variance. It certainly means that we can train the AE to perform the ProjOut operation better. For example, we can reduce the complexity of the AE by reducing the number of hidden nodes since the manifold will be of a lower complexity as well. If the manifold become linear enough then it is even possible to use PCA. The question then becomes, does a hierarchical model of the visual system achieve these two goals that we aspire to achieve? The hypothesis is yes and is tested next.

4.2. Testing Hypotheses

To test whether or not the manifold becomes more linear we will measure the reconstructive error $\sum_i^n \|\mathbf{x_i} - U_d U_d^T \mathbf{x_i}\|^2$, where we fix the numer of columns of U_d to be the estimated number of intrinsic dimensions. Basically the idea is that if the manifold is linear, then we should have small SSE, whereas for highly nonlinear distribution of data, SSE would be large. To test the

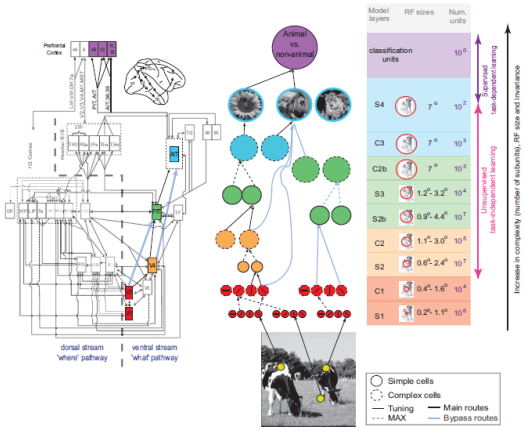


Figure 8. Visual cortex hierarchy model. The bottom level represent neurons from the primary visual cortex, the top layer represent neurons from the Inferotemporal cortex. As level increases in the hierarchy, neurons become selective of more complex pattern as well as become more invariant to natural transformations. Diagram is from [38].

spread of the manifold, we will use $Tr(\Sigma)$, where Σ is the covariance matrix of the data. $Tr(\Sigma)$ is a good measure since $\Sigma = U\Lambda U^T$, $Tr(\Sigma) = Tr(\Lambda) = \sum_{i=1}^{n} \lambda_i$, where λ_i is the eigenvalues of Σ and is the variance in the direction of the i_{th} eigenvector. Other types of measure such as the total edge weights in a minimum spanning tree could also be used.

The model hierarchical network tested is the one described by Serre et al. [38], see figure 8. This network has four layers, consisting of alternating 'S' and 'C' layers. An S layer contains neurons which detect small localized features within the big image. A C layer contains neurons which are invariant to small transformation in the S layer below. The bottom input layer is basically the retina or image pixel space. The first hidden layer is occupied by the so called S1 neurons. They are basically Gabor wavelets selective toward oriented edges. Pixels are connected to (innervates) the S1 layer. The second hidden layer are occupied by the so called C1 neurons, which will be active if its preferred type S1 neurons are active within a local window of the C1 neuron. The C1 neurons essentially perform a max operation and this step is vital to ignore some small variances such as shift, scale, and rotation. The next layer is another S layer known as S2, which is followed by the C2 layer. This type of alternating layering could continue but for the purpose of the hypothesis, we compare the linearness and spread of the manifold at the image, C1, and C2 layers. At C1 and C2 level, the measure is on the spread and linearness of the activation of the neurons.

Table 2 shows the result performed on the Olivetti face dataset [37]. The C1 layer did not show much reduction of both linearness and spread, but in the C2 layer, the reduction was dramatic. Similar results were observed for the UMIST faces. The results for UMIST were averaged for 20 subjects. Due to the large number of neurons and the time consuming process of training AEs, we are in the process of running experiments to test the recognition performance of $InDiv$ with using C1 or C2 layer neuron activities as input.

Figure 9. Visualizing what transformations the code layer activity represent. Images are decoded from the 1st principle component of the code layer nodes. The leftmost and the rightmost image correspond to 1 standard deviation away from code layer mean along the principle component.

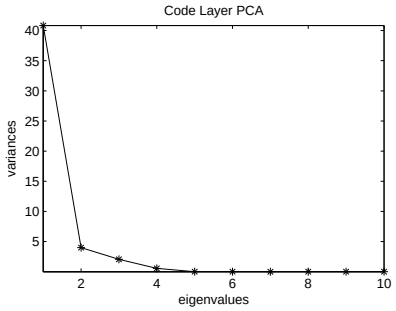


Figure 10. The sharp "knee" shows that the AE has found that the data probably has only 1 intrinsic dimension.

4.3. The Mind of AE

One criticism of neural networks is that they are like black boxes; it's not easy to perceive the representation and computation of the hidden nodes. With AEs, this is not a problem since the network is an also a decoder. Therefore, the activities of the hidden nodes in the code layer and its respective decoded image can give us a visualization of what the different values of the code layer nodes represent. A 1000-500-250-10 AE was trained to model the manifold of 38 images of person A from UMIST. The 10 code layer node's activity was collected and the principle component found. 11 locations spanning 1 standard deviation were found along the principle component. Decoding is performed on all 11 points and their decoded or reconstructed images are displayed in figure 9.

Figure 10 shows the eigenvalues of code layer node activations. The "knee" is visible at the 2nd eigenvalue, suggesting that there is only 1 intrinsic variation in the input data. Given some intuition regarding the behaviour of the code layer neurons, a natural and simple extension is to fine-tune the activation of code layer nodes to find a more accurate InDiv between test face and training manifold. It should be noted that the fine-tuning when AE is trained is with respect to the weights. Here, the weights are fixed, since they represent the training faces manifold. What

Table 2. The linearness and spread measure of data at different hierarchy levels. The results are averaged across 40 persons, each with 10 images. The entries are relative ratio with respect to the image layer measures.

Olivetti faces	Image Layer	C1 Layer	C2 Layer
Linearness	1.0	0.829	0.0334
Spread	1.0	0.830	0.0758
UMIST faces	Image Layer	C1 Layer	C2 Layer
Linearness	1.0	0.885	0.0182
Spread	1.0	0.961	0.0333

we are proposing is that the feed-forwarding encoding process may be slightly off and optimization is used to find a closer point to the test face which still lies on the manifold. Let the distance between output face $\mathbf{y}$ and test face $\mathbf{t}$ be:

$$d(\mathbf{t}) = \frac{1}{2}(\mathbf{y}-\mathbf{t})^T(\mathbf{y}-\mathbf{t}) \tag{8}$$

The soft constraint error to minimize can be defined as:

$$E = \frac{1}{2}(\mathbf{y}-\mathbf{t})^T(\mathbf{y}-\mathbf{t}) + \xi\big[\sigma((\mathbf{z}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{z}-\boldsymbol{\mu})\big)\big] \tag{9}$$

where $\mathbf{z}$ is the activation of the code layer nodes when $\mathbf{t}$ is presented to the input of AE, $\boldsymbol{\mu}$ is the mean vector activation of the code layer nodes averaged over all class specific faces, Σ is the covariance matrix of the code layer nodes, and $\sigma(\tau) = 1/(1+\exp(-50(\tau-3)))$. By placing a soft constraint on the deviation of $\mathbf{z}$, we make sure the resulting optimized $\mathbf{z}$ stays on the manifold. The $\sigma(\cdot)$ is used to allow no penalty for $\mathbf{z}$ as long as it is within the 3 standard deviation contour. If we let $\eta = \sigma((\mathbf{z}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{z}-\boldsymbol{\mu}))$, Differentiating with respect to $\mathbf{z}$,

$$\frac{\partial E}{\partial \mathbf{z}} = \frac{\partial d(\mathbf{t})}{\partial \mathbf{z}} + 2\xi\eta(1-\eta)\Sigma^{-1}(\mathbf{z}-\boldsymbol{\mu}) \tag{10}$$

The conjugate gradient method was used for optimization using the first-order partial derivatives. On the UMIST data set, it was found that on average, the ratio of InDiv distances after optimization compared to before is 0.929. Recognition results using the newly optimized InDiv have produced no significant changes currently. However, we believe that it is due to the restricted types of transformations in training set and not due to the general concept of optimization in the code layer.

5. Conclusion

In this project an intuitive measure of dissimilarity between class specific manifold is introduced. The difficulty in its calculation has led to the idea of using the power for AE framework to serve as a ProjOut operator. The inner workings of the recent RBM AE was explored. Good results were obtained where the correct pose was found simply by an encoding and decoding process. However, recognition rates were not great and didn't perform as well as NN. This is somewhat expected since the manifolds at the image level is very nonlinear. Two extensions were proposed to help find better $InDiv$s and they show great promise. The hypothesis that the manifolds becomes more linear and less dispersed was verified. For future work, large collections of face data with most of the natural transformations need to be collected into the training set. Recognition using AE based $InDiv$ at higher hierarchical levels need to be fully tested. With respect to the 2nd extension idea, information from the higher levels can also be projected top-down to bias the $InDiv$ search.

References

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9:147–169, 1985. 6

[2] J. A. Anderson. Cognitive and psychological computation with neural models. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13:799–815, 1983. 5

[3] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class-specific linear projection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(7):711–720, July 1997. 4

[4] Y. Bengio, J. franois Paiement, and P. Vincent. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. In *In Annual Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference*, pages 177–184. MIT Press, 2004. 5

[5] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, August 2006. 6

[6] C. M. Bishop, M. Svensén, and C. K. I. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998. 5

[7] V. Blanz and T. Vetter. A morphable model for the synthesis of 3-D faces. In *SIGGraph-99*, pages 187–194, 1999. 4

[8] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, 59(4):291–294, 1988. 5

[9] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(6):681–685, June 2001. 4

[10] D. Demers and G. Cottrell. Non-linear dimensionality reduction. In *Advances in Neural Information Processing Systems 5*, pages 580–587, 1993. 6

[11] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*, volume November. John Wily & Sons, Inc., New York, second edition, 2000. 2

[12] K. Fukushima. Neocognitron: A neural model for a mechanism of visual pattern recognition. *IEEE Trans. SMC*, 13(5):826–834, 1983. 10

[13] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, 1984. 6

[14] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighborhood component analysis. In *NIPS*. 5

[15] D. B. Graham and N. M. Allinson. Face recognition using virtual parametric eigenspace signatures, 1997. 8

[16] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001. 2

[17] Hinton and Salakhutdinov. Reducing the dimensionality of data with neural networks. *SCIENCE: Science*, 313, 2006. 5, 6, 7

[18] G. Hinton and S. Roweis. Stochastic neighbor embedding. In *Advances in Neural Information Processing Systems 15*, pages 833–840. MIT Press, 2003. 5

[19] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:2002, 2002. 7

[20] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. of the National Academy of Science, USA*, 79:2554–2558, 1982. 5, 6

[21] D. Hubel and T. Wiesel. Receptive fields of single neurons in the cats striate cortex. *Journal of Physiology*, 148:574–591, 1959. 10

[22] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983. 6

[23] T. Kohonen. Associative memory: A system-theoretical approach. Technical report, Berlin, BRD, 1977. 5

[24] T. Kohonen. The self-organizing map. *Proc. IEEE*, 78(9):1464–1480, 1990. 5

[25] B. Kosko. Bidirectional associative memories. *Applied Optics*, 18:49–60, 1988. 5

[26] M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AICHE Journal*, 37:233–243, 1991. 6

[27] Y. LeCun, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998. 10

[28] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004. 1, 4, 10

[29] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 257–263, 2003. 1, 4, 10

[30] J. Mutch and D. G. Lowe. Multiclass object recognition with sparse, localized features. In *CVPR*, pages I: 11–18, 2006. 10

[31] R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, Sept. 1993. 6

[32] P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15:1373–1396, 2003. 5

[33] E. Oja. Principal components, minor components, and linear neural networks. *Neural Networks*, 5:927–935, 1992. 5

[34] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2(11):1019–1025, Nov. 1999. 10

[35] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, Dec. 2000. 5

[36] R. Salakhutdinov and G. Hinton. http://www.cs.toronto.edu/ hinton/matlabforsciencepaper.html. 7

[37] F. Samaria and A. Harter. Parameterisation of a stochastic model for human face identification. In *IEEE Workshop on Applications of Computer Vision*, Sarasota (Florida), Dec. 1994. 9, 11

[38] T. Serre, M. Kouh, C. Cadieu, U. Knoblich, G. Kreiman, and T. Poggio. A theory of object recognition: Computations and circuits in the feedforward path of the ventral stream in primate visual cortex. In *AI Memo*, page 2005, 2005. 11

[39] T. Serre, L. Wolf, and T. Poggio. Object recognition with features inspired by visual cortex. In *In CVPR*, pages 994–1000, 2005. 10

[40] H. S. Seung and D. D. Lee. The manifold ways of perception. *Science*, 290:2268 – 2269. 1

[41] P. Y. Simard, Y. A. LeCun, J. S. Denker, and B. Victorri. Transformation invariance in pattern recognition — tangent distance and tangent propagation. *Lecture Notes in Computer Science*, 1524, 1998. 1, 2

[42] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, volume 1, chapter 6, pages 194–281. MIT Press, Cambridge, 1986. 6

[43] K. Tanaka. Representation of visual features of objects in the inferotemporal cortex. *Neural Networks*, 9(8):1459–1475, 1996. 10

[44] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, Dec. 2000. 5

[45] M. Turk and A. P. Pentland. Eigenfaces for recognition. *Journal Cognitive Neuroscience*, 3(1):71–96, 1991. 4

[46] L. Wiskott, J. M. Fellous, N. Kruger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(7):775–779, July 1997. 1, 4