Advances and Challenges in Machine Learning for High-Dimensional Data

Yoshua Bengio

Laboratoire d'Informatique

des Systèmes Adaptatifs

http://www.iro.umontreal.ca/~lisa

MITACS

Université de Montréal

Overview

- Some advances of recent years
  - Beyond logistic regression and decision trees?
  - Dimensionality reduction and spectral embedding
  - Semi-supervised learning: surveys++
  - Non-linear factor analysis on pairs of objects: preference prediction
  - Active learning and reinforcement learning: marketing campaigns sequential management
- Challenges ahead
  - Beyond local learning algorithms?
  - Non-local manifold learning

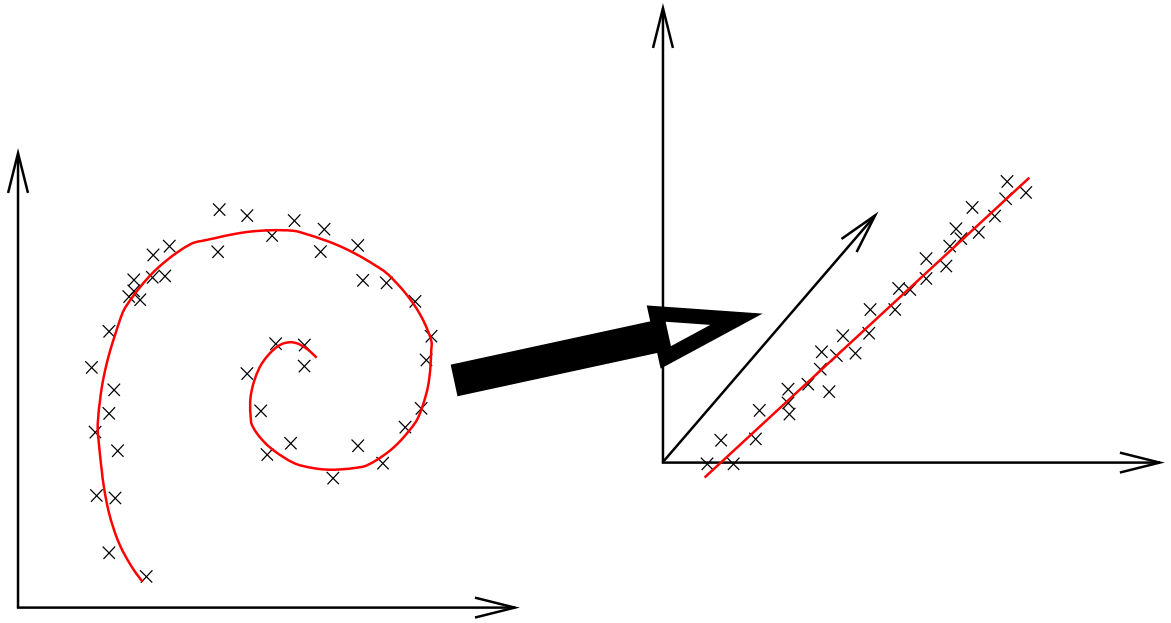
Beyond Logistic Regression and Decision Trees?

- Data-mining practionners tend to stick to a few learning algorithms that have worked well for them, such as logistic regression and decision trees.
- They are somewhat constrained by the software availability on platforms such as SAS, making it harder for them to try new algorithms.
- What they try is often framed in the same old paradigm, e.g. predict $P(\text{rare binary event } Y | \text{customer profile } X)$ using (X, Y) pairs.
- This talk is to suggest *paradigm shifts* to data-mining practitioners and discuss challenges for academics in this field.

Avoiding overfitting by dimensionality reduction

- **Overfitting** happens when the richness of the model class is too great, respective to the number of examples. Richness of a function class (capacity) is immediately related to the number of free parameters, e.g. the number of inputs.
- Data with **rare events** (e.g. a rare but important class), even with hundreds of thousands of records can actually get one in the overfitting regime even with models as simple as logistic regression.
- Unsupervised or semi-supervised learning can be used to reduce dimensionality and take advantage of structure in the distribution: **manifold structure**.
- Two main kinds of dimensionality reduction: linear (e.g. PCA, factor analysis) or non-linear. Much progress in recent years on non-linear dimensionality reduction.

Avoiding overfitting by dimensionality reduction



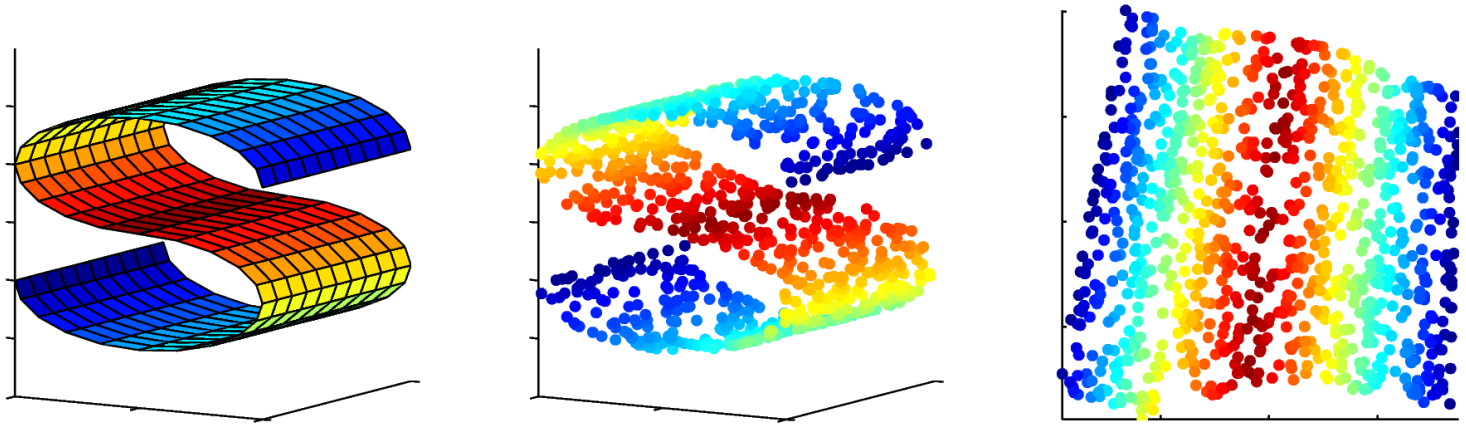
If we can find the non-linear coordinate system describing the position of points on this manifold we can describe the data more compactly and focussing on the actual degrees of freedom.

Spectral embedding algorithm perform this kind of transformation.

Spectral Embedding Algorithms

Non-parametric algorithms for estimating a training set embedding on the presumed data manifold from the *principal eigenvectors of a Gram matrix* M with $M_{ij} = K_D(x_i, x_j)$ from data-dependent kernel K_D.

• Examples: **LLE** (Roweis & Saul 2000), **Isomap** (Tenenbaum et al 2000), **Laplacian Eigenmaps** (Belkin & Niyogi 2003), **spectral clustering** (Weiss 99), **kernel PCA** (Schölkopf et al 98). *Each corresponds to different* K_D. (fig. *Roweis & Saul*)



• all optimize a criterion wrt embedding of each training example.

• Attractiveness: represent non-linear manifolds with analytic solution.

Out-of-Sample Embedding = Induction

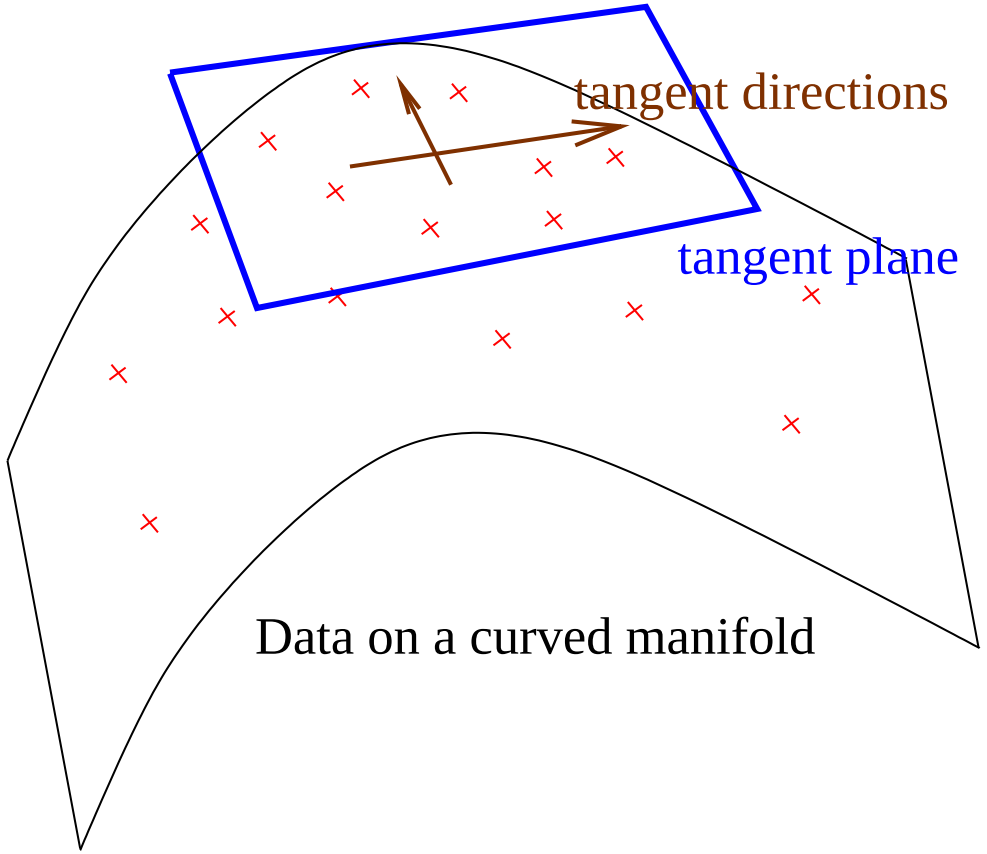
- How to generalize to new examples without recomputing eigenvectors?
- Are there corresponding **induction algorithms**?
- Out-of-sample generalization with the Nyström formula:

$$e_k(x) = \frac{1}{\lambda_k} \sum_{i=1}^{n} v_{ki} K_D(x, x_i)$$

for k-th coordinate, with (λ_k, v_k) the k-th eigenpair of M.

- This is an estimator of the eigenfunctions of K_D as $|D| \to \infty$ (see recent Neural Comp. paper, on my web page).

Tangent Plane $\Longleftrightarrow$ Embedding Function

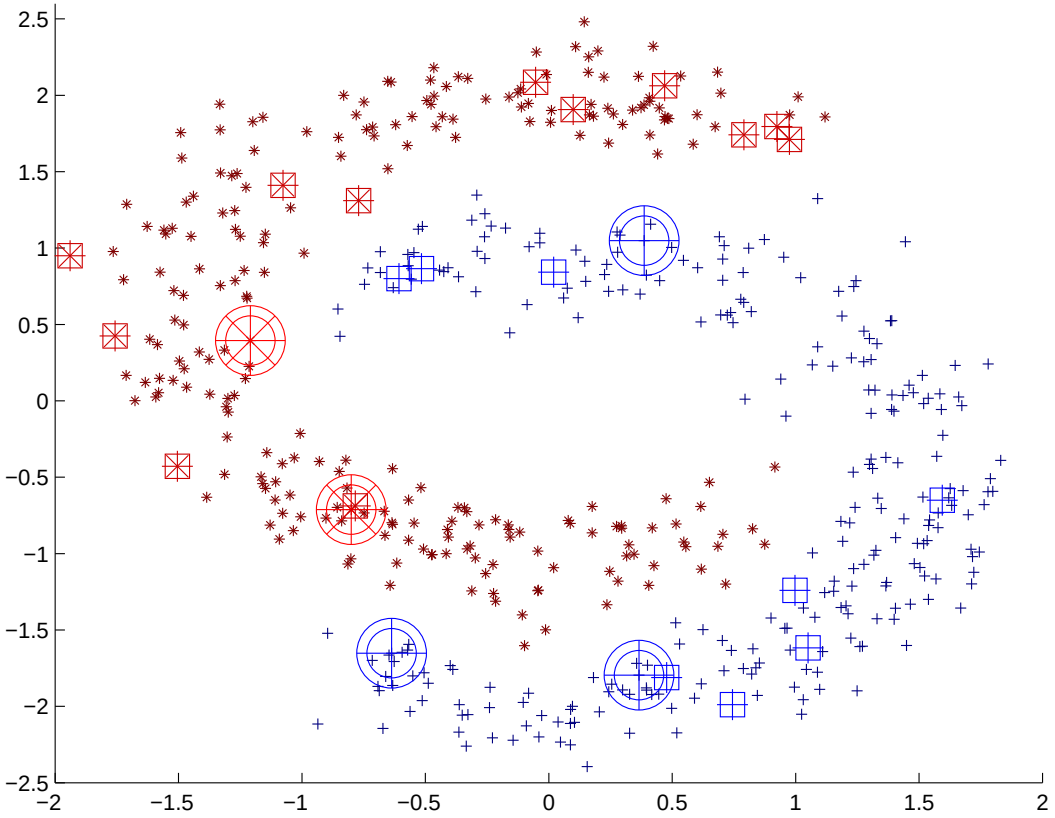


Important observation:

The tangent plane at x is simply the subspace spanned by the gradient vectors of the embedding function:

$$\frac{\partial e_k(x)}{\partial x}$$

Semi-Supervised Learning



In this example there are two curved manifolds, one for each class.

Discovering them can greatly help to build a classifier.

Big circles are the only labeled training examples. Small $*$ and $+$ are unlabeled training examples.

Principles of Non-Parametric Semi-Supervised Learning

Many approaches proposed. Most rely on these principles:

- Assign a (soft) label $f(x_i)$ to each unlabeled example x_i, which is a free parameter (hence the non-parametric nature).
- *Two nearby examples should have a similar (soft) label*. "Nearby" is given by a similarity function $W(x_i, x_j)$. Disagreement between (soft) labels is given by a function $D(f(x_i), f(x_j))$.
- The soft label $f(x_i)$ assigned to labeled example $i \in L$ should be close to the actual label y_i (can be forced to be equal).

Simple formalization: choose $\{f(x_i)\}$ to minimize

$$\sum_{i,j \in U \cup L} W(x_i, x_j) D(f(x_i), f(x_j)) + \lambda \sum_{i \in L} D(f(x_i), y_i)$$

Taking D quadratic yields a linear system, easily solved analytically (but $O(n^3)$ computation required, for n examples). We find better results with $\lambda < \infty$.

Applications of Semi-Supervised Learning

- Many companies collect information about customers through surveys.
- Surveys are expensive, one or few 1000 "labeled" examples.
- Semi-supervised learning: combine survey data with non-surveyed population data.
- Important: techniques to speed-up computation to avoid the $O(n^3)$ growth (see recent work in my lab with Delalleau and Le Roux, submitted). Our technique reduces computation to $O(nm^2)$ with $m \ll n$.
- We have shown how to naturally generalize from $f(x_i)$ on unlabeled training example x_i to $f(x)$ for arbitrary test point x.

Non-Linear Factor Analysis on Pairs of Objects

Another new paradigm in machine learning: many tasks involve two types of objects, e.g.

1. users, customers, etc...
2. products, etc...

Co-occurence of these two types of objects are observed, sometimes with some response variable Z (e.g. customer X liked or bought product Y or rated it with value Z).

Goals:

1. **predict response variable Z on new (X, Y) pairs**
2. **qualitatively learn about the semantic relations between X and Y objects**

Associate each X and each Y with a point embedding in $\mathbf{R}^d$.

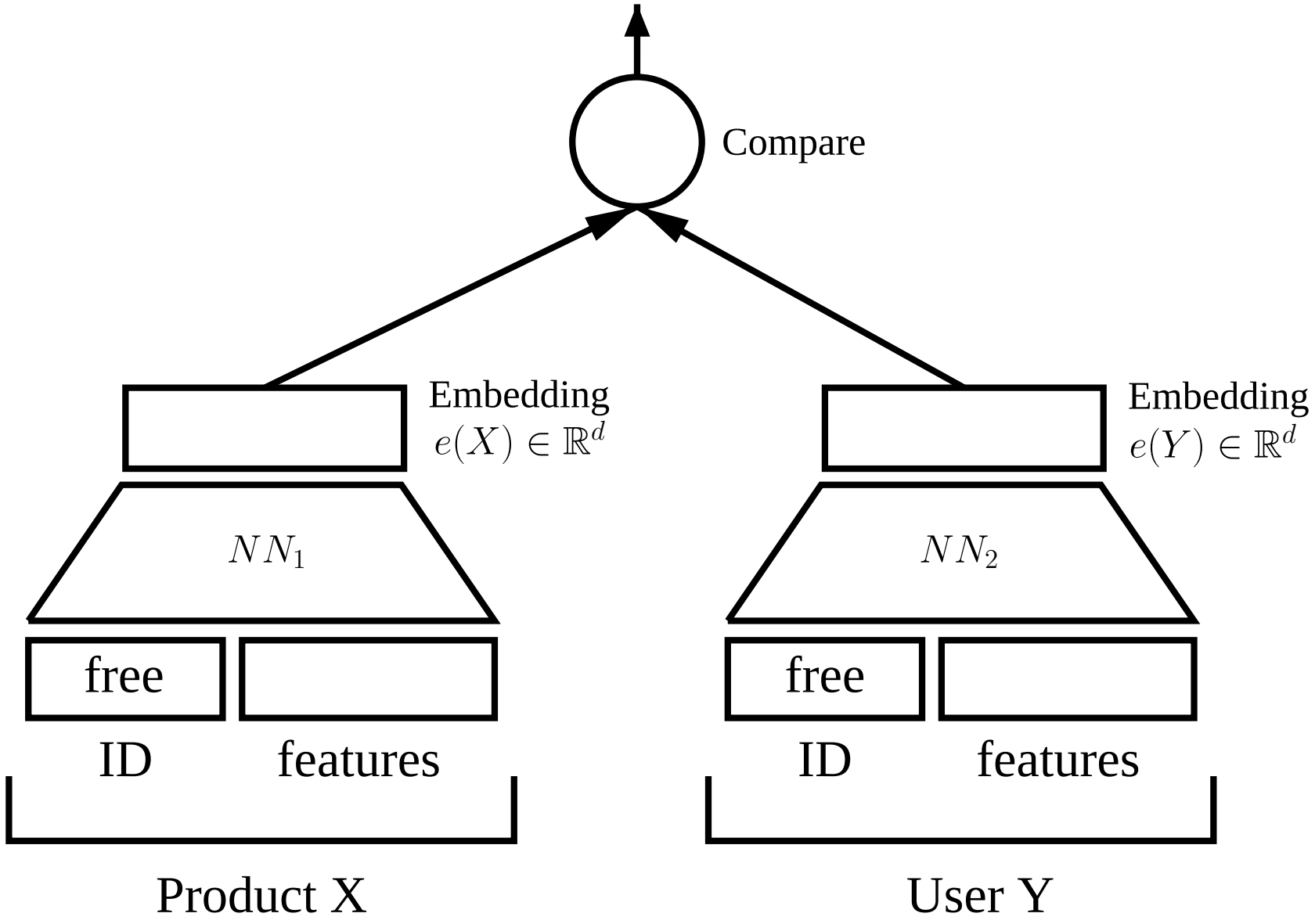
Object Pairs Embedding: Examples

- *Collaborative filtering*: music, books, movies.
  Predict who is likely to like what based on previous responses on other products.
- Extensions of collaborative filtering: **in addition to product ID and user ID**, use hand-crafted features on each user (questionnaire answers, profile, web-site clicks, etc...) and each product (e.g. acoustics of a song, features of a movie, etc...).
- Industrial data-mining application: match molecules X that are candidate drugs (the **keys**) with molecules Y that are protein receptors (the **locks**) in the same space. Measure Z = how well key X fits lock Y for some (X, Y) pairs. Generalize to new pairs.

This approach most interesting when the number of products is greater than a handful. Capitalize on the **learned similarity betweeen products**.

Ongoing projects in my lab: music preference prediction and virtual screening for candidate drugs.

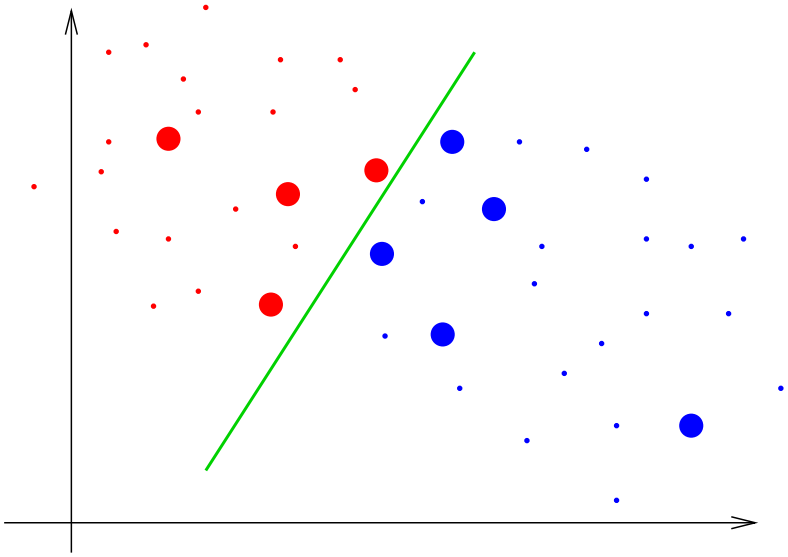
Object Pairs Embedding: Neural Net Implementation



Active Learning

Active learning: **learner chooses where to sample**! different paradigm.

Can obtain a potential exponential speed-up in the convergence rate of the learning algorithm wrt the number of training samples to reach a given classification accuracy!



Big circles are the examples chosen by the active learning algorithm, which focusses more and more near the decision surface.

Reinforcement Learning

- Consider a sequence of decisions, and an unknown environment.
- Goal is to maximize total expected "reward". Earlier decisions can impact on reward associated with later decisions.
- Decisions affect a "state variable", which may be observed (Markov Decision Process) or not (Partially Observable Markov Decision Processes = **POMDP**).
- Optimal algorithms are not computationally feasible for even moderately dimensional state, especially for POMDPs, but many approximate and heuristically motivated algorithms have been proposed and successfully applied.
- **Exploration-exploitation trade-off**: if we only go in places (states) that we know to be good we might miss the best spots. If we only go to unknown places (states) we don't get much reward.

Active + Reinforcement Learning: Campaign Management

- Marketing campaigns usually in **multiple phases**.
- Each phase brings more information on the population.
- Who to reach? two competing objectives
  1. Maximize short term profit: customers most likely to respond positively.
  2. Maximize model performance: collect information to better model population.

Active + Reinforcement Learning: Campaign Management

- This is a **sequential decision taking / learning** task. Sequence of actions impact on future data distribution!
- Traditional dilemma in reinforcement learning: **exploitation vs exploration**.
- Overall goal: maximize CUMULATIVE profit, must compromise between the two.

Much recent progress in reinforcement learning from a probabilistic standpoint can be applied this task.

Serious Challenges Ahead!

- The curse of dimensionality in modeling data distribution.
- The curse of dimensionality in reinforcement learning.
- Trends: mathematically sophisticated work but generally not ambitious from an AI perspective!
- Is our research community going to get stuck in a local minimum? e.g. speech recognition community seems to be!

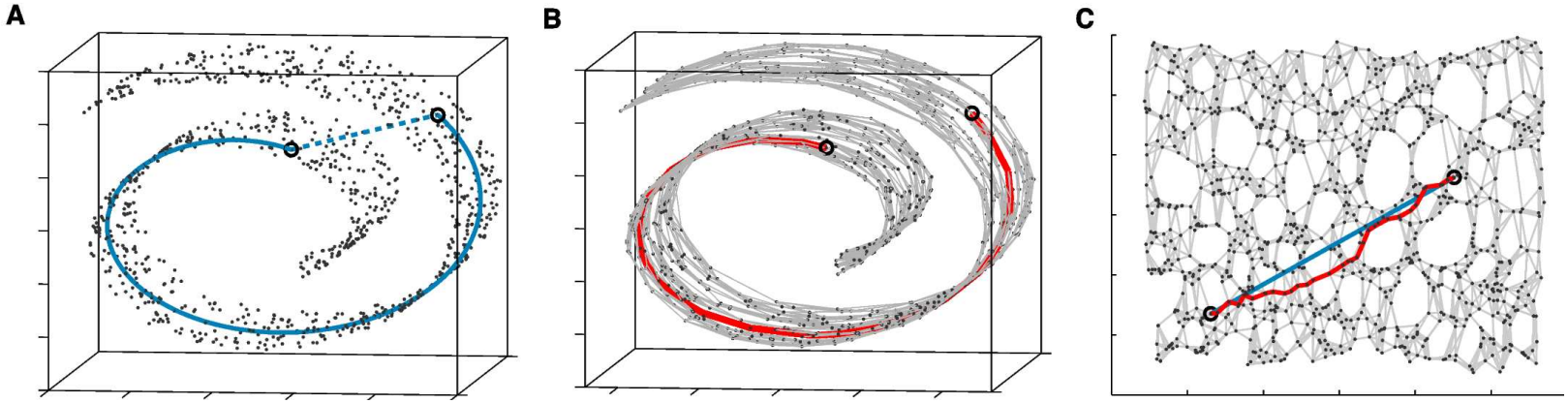
Local Manifold Learning

- Local Manifold Learning Algorithms: derive information about the manifold structure near x using mostly the neighbors of x.
- For LLE, kernel PCA with Gaussian kernel, spectral clustering, Laplacian Eigenmaps $K_D(x, y) \to 0$ for x far from y, so $e_k(x)$ only depends on the neighbors of x.
- Therefore the tangent plane

$$\frac{\partial e_k(x)}{\partial x} = \frac{1}{\lambda_k} \sum_{i=1}^{n} v_{ki} \frac{\partial K_D(x, x_i)}{\partial x}$$

  also only depends on the neighbors of x.
- $\Rightarrow$ can't say anything about the manifold structure near a new example x that is "far" from training examples!

ISOMAP is also a Local Estimator



Isomap estimates the **geodesic distance** along the manifold using the shortest path in the nearest neighbors graph.

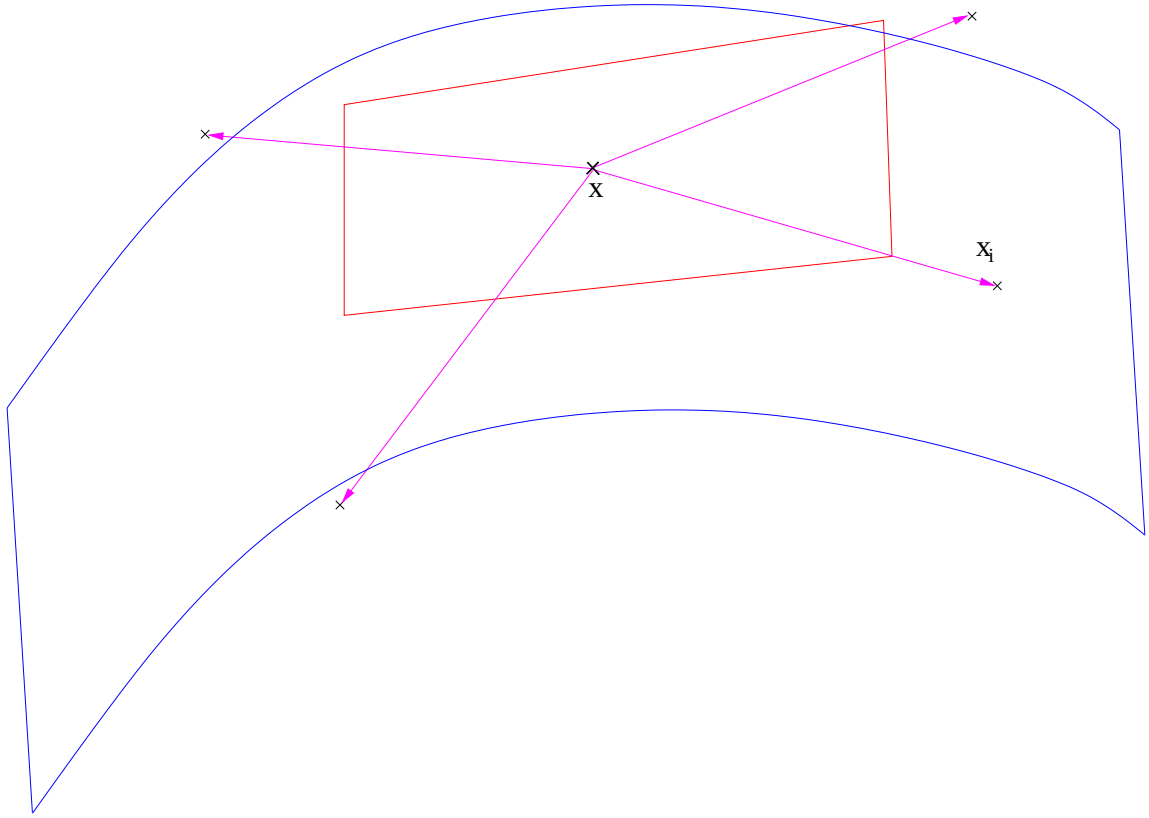
It then looks for a low-dimensional representation that approximates those geodesic distances in the least square sense (MDS).

Lemma: the tangent plane at x of the manifold estimated by Isomap are included in the span of the vectors $x - x_j$ where x_j are training set neighbors of x (in the sense of being the first neighbor on the path from x to one of the training examples).

Isomap is also a local manifold learning algorithm!

Tangent Plane Defined from Neighhbors

In conclusion the above non-parametric manifold learning algorithms define the local tangent plane at x mainly on the span of the vectors of neighbor differences $x_i - x$.



This may be inadequate because of: noise, curvature, non-trivial dimension of the manifold, finite data.

Pancake Mixture Models

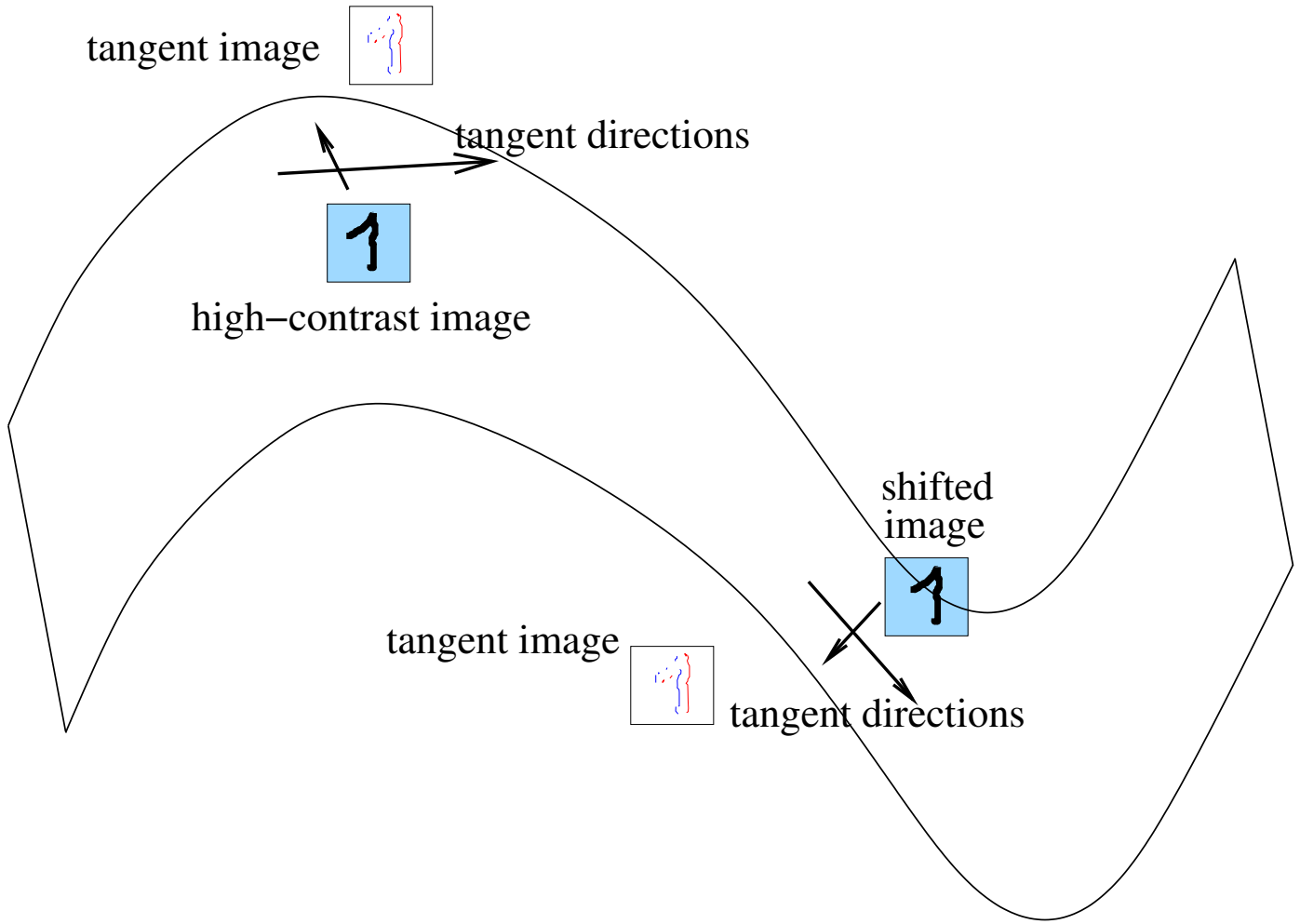
Other local manifold learning algorithms, density mixture models of flattened Gaussians:

- Mixtures of factor analyzers (Ghahramani & Hinton 96)
- Mixtures of probabilistic PCA (Tipping & Bishop 99)
- Manifold Parzen Windows (Vincent & Bengio 2003)
- Automatic Alignment of Local Representations (Teh & Roweis 2003)
- Manifold Charting (Brand 2003)

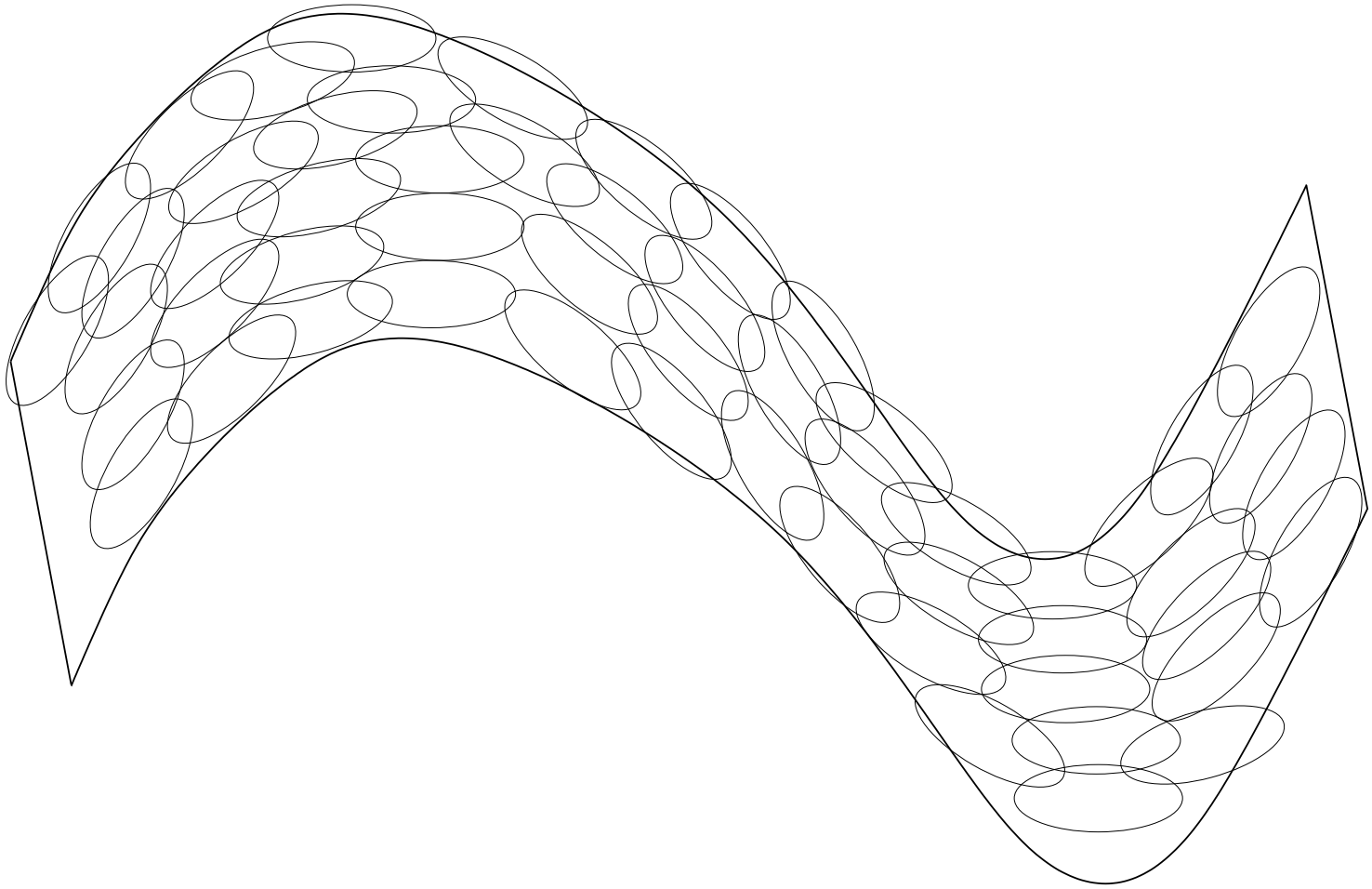
Some provide both density and embedding.

Local Manifold Learning: Local Linear Patches

Current manifold learning algorithms cannot handle highly curved manifolds because they are based on locally linear patches estimated locally (possibly aligned globally).



The Curse of Dimensionality on a Manifold



It is similar to the ordinary curse of dimensionality for classical non-parametric statistics, but where d = dimension of the manifold.

It hurts all the local manifold learning methods.

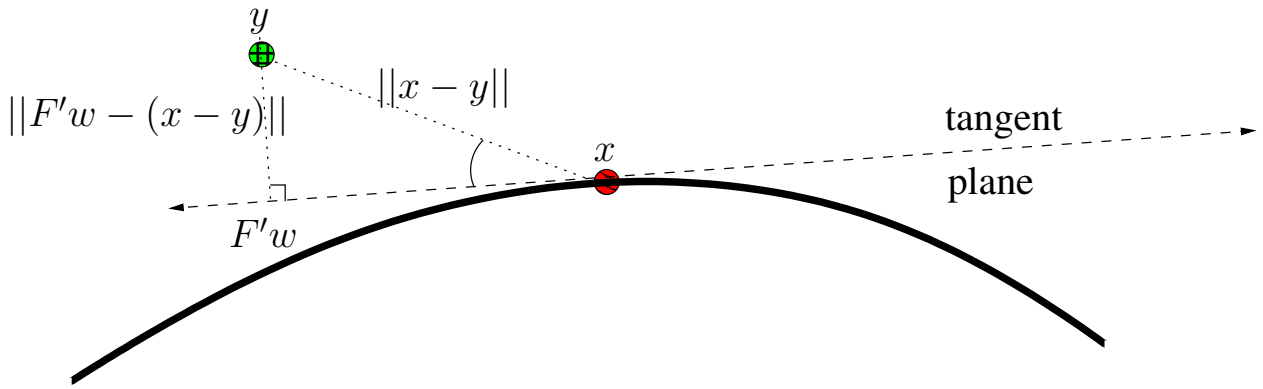
Fundamental Problems with Local Manifold Learning

- **High Noise:** constraints not perfectly satisfied. Data not strictly on manifold. More noise $\rightarrow$ more data needed per local patch.
- **High Curvature:** need more smaller patches O($(1/r)^d$) with $r =$ patch radius decreasing with curvature.
- **High Manifold Dimension:** $O((1/r)^d)$ patches are needed (curse of dimensionality), at least $O(d)$ examples per patch ($\propto$ noise).
- **Many manifolds:** e.g. images of transformed object instances = 1 manifold per instance or per object class. Local manifold learning can't take advantage of shared structure across multiple manifolds.

Non-Local Tangent Plane Predictors

Proposed approach: estimate tangent plane basis vectors as a function of position x in input space, with flexibly parametrized matrix-valued $d \times n$ function $J(x)$.

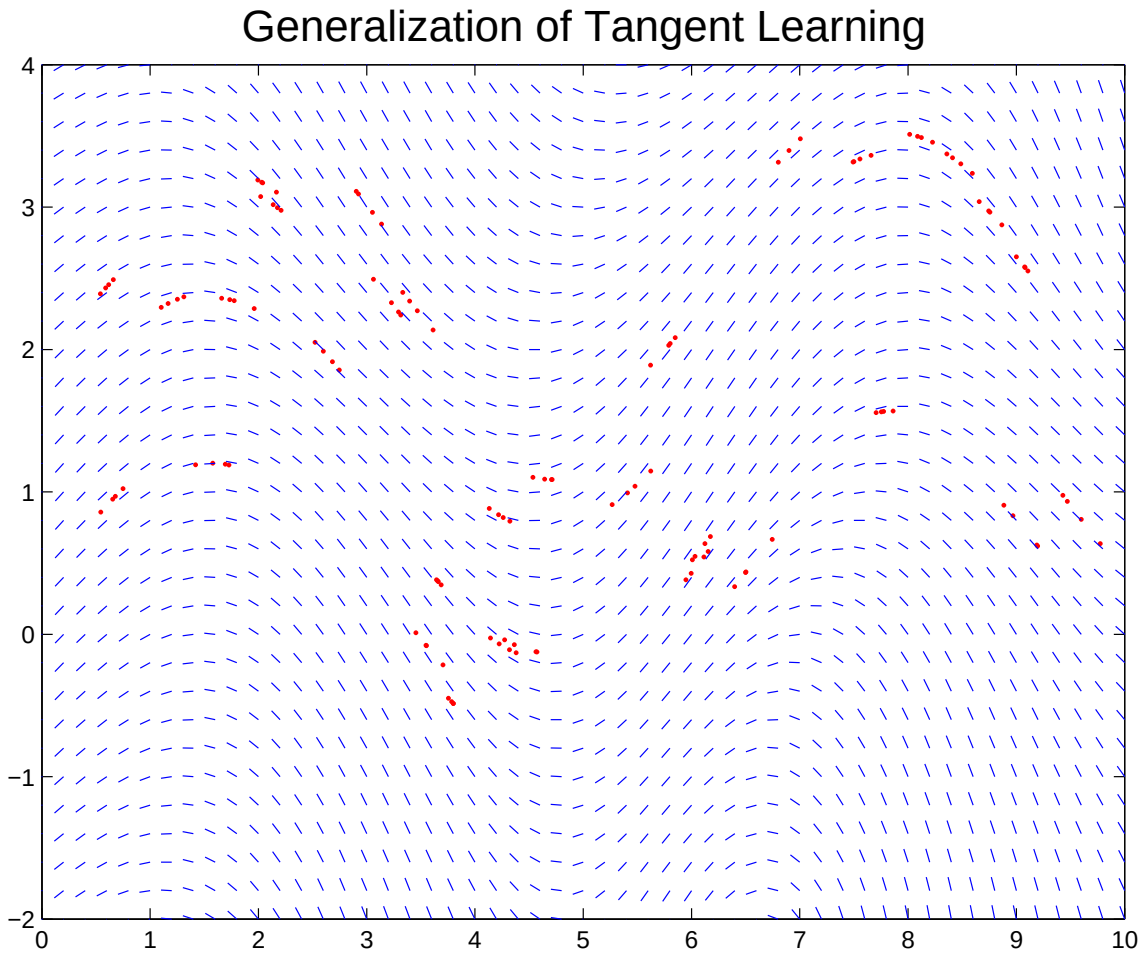
Train $J(x)$ to approximately span the differences between x and its neighbors y.



Training criterion = relative projection error at examples x_t and their neighbors x_i:

$$\min_{F,\{w_{tj}\}} \sum_t \sum_{j \in \mathcal{N}(x_t)} \frac{||F'(x_t)w_{tj} - (x_t - x_j)||^2}{||x_t - x_j||^2}$$

Results with Tangent Plane Predictors



Task 1: 2-D data with 1-D sinusoidal manifolds: the method indeed captures the tangent planes. Small blue segments are the estimated tangent planes. Red points are training examples.

Results with Tangent Plane Predictors

1000 digit images + image with rotation = 2 examples / manifold.

Images are 14×14 of 10 digits from MNIST database.

testing on MNIST digits	Average relative projection error
analytic tangent plane	0.27
tangent learning	0.43
Dim-NN or **Local PCA**	1.50

Tangent vector on test image of 8:

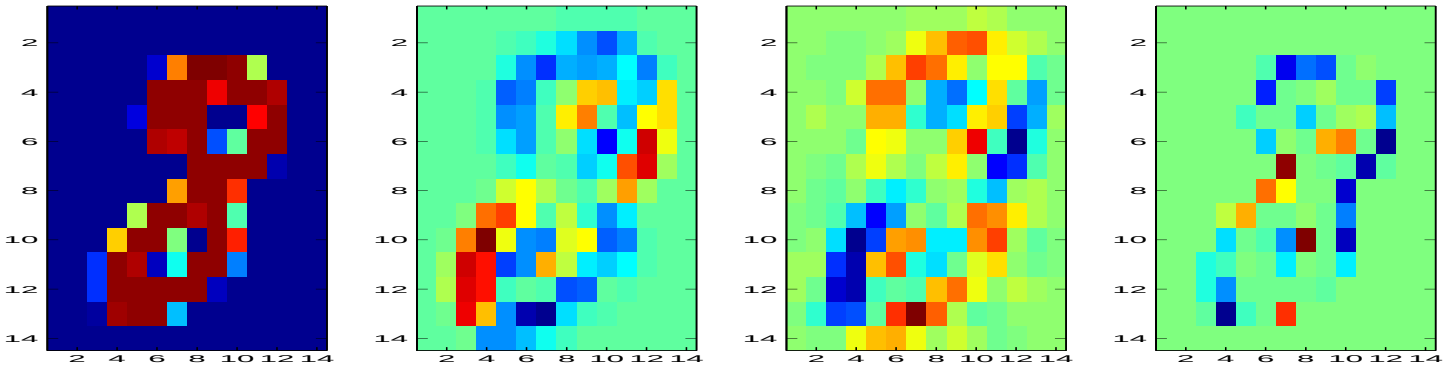
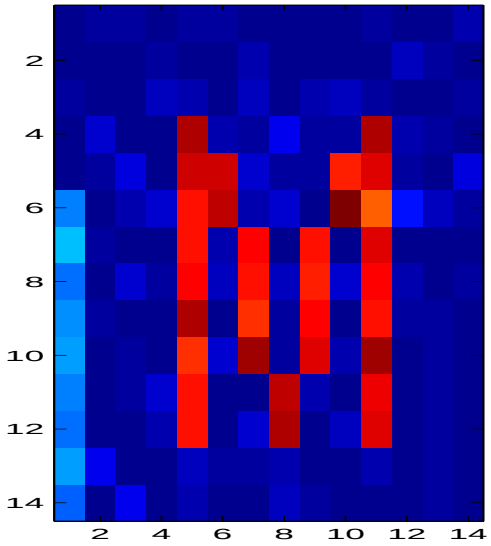


image analytic tan. learn. local PCA

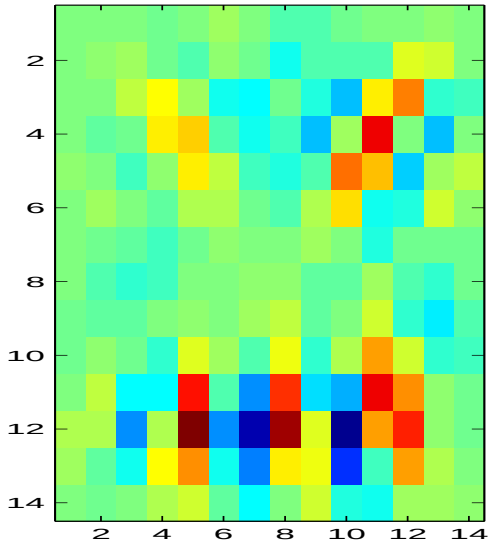
Truly Out-of-Sample Generalization

Model was trained on digits 0 to 9: test it on letter M

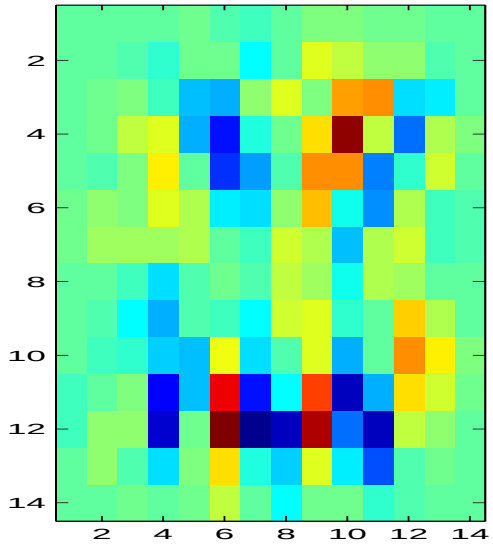
Compare predicted tangent vectors:



image



tan. learn.

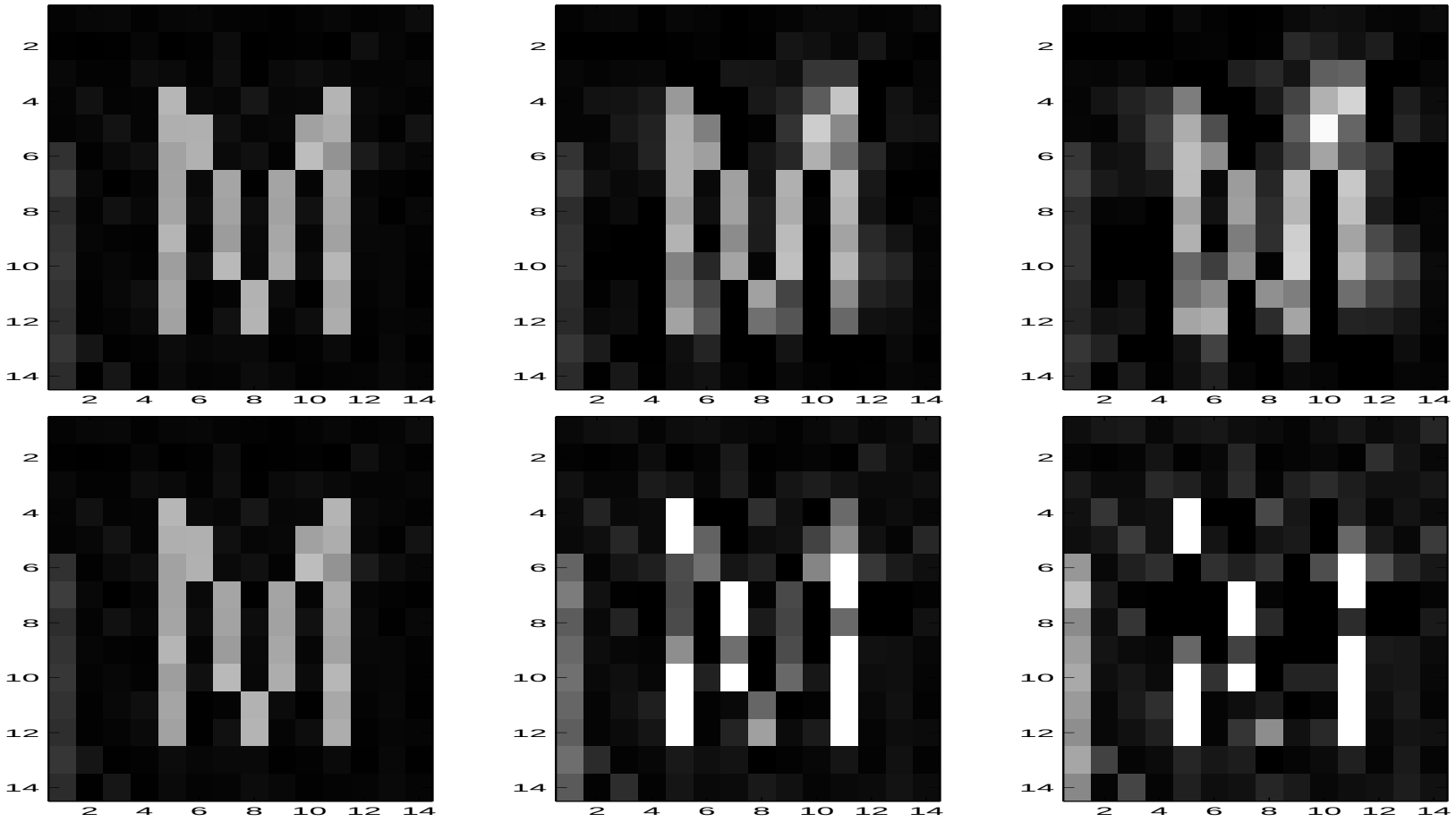


local PCA

Truly Out-of-Sample Generalization

Left: original image, center: moving along the manifold, right: moving further away.

Top: with tan.learn., bottom: with local PCA



Not surprisingly, local manifold learning fails, whereas the globally estimated tangent plane predictor generalizes correctly to a very different image!

Discussion of Non-Local Learning

Advantages:

- It is possible to generalize to examples that are truly not near neighbors of the training examples.
- Can **take advantage of multiple manifolds whose structure has something in common**.
- If there is structure, number of degrees of freedom remains small wrt dimensionality and data $\rightarrow$ generalization to truly different examples.

Achieved because the parameters of $J(.)$ (e.g. the neural network weights) are shared across space (independent of x). "Sharing" is the only way to face the generalization challenges raised earlier!

Disadvantages

- Optimization may be more difficult! (but so far, so good)

Discussion of Non-Local Learning

Other ways to get non-local learning

- Learn a function as a composition of reusable components
- Assume a prior on the class of distribution function: complex structure can be written down as a combination of simpler structures that are predictive by themselves.
- These are not easy things to do and we not be able to express these algorithms as convex optimization...
- But at least we should try to develop algorithms that have the potential to learn to generalize non-locally, so as to be able to capture complex structure with a finite amount of data, even if high-dimensional.