The Curse of Dimensionality for Local Kernel Machines

Yoshua Bengio, Olivier Delalleau & Nicolas Le Roux

April 7th 2005

Perspective

Most common non-parametric approaches based on smoothness prior, which leads to "local" learning algorithms, e.g. kernel-based.

Perspective

Most common non-parametric approaches based on smoothness prior, which leads to "local" learning algorithms, e.g. kernel-based.

Smoothness may not be the only way to obtain "simple functions" : e.g. According to Kolmogorov complexity, $sinus(x)$ and $parity(x)$ are simple yet they are highly variable (apparently complex) functions.

Perspective

Most common non-parametric approaches based on smoothness prior, which leads to "local" learning algorithms, e.g. kernel-based.
Smoothness may not be the only way to obtain "simple functions" : e.g. According to Kolmogorov complexity, $sinus(x)$ and $parity(x)$ are simple yet they are highly variable (apparently complex) functions.

Let us clarify the notion of "locality" which leads to the **curse of dimensionality** even to learn simple but highly variable functions, and probably to learn what is required for true AI.

Perspective

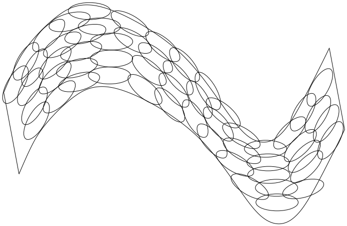
Most common non-parametric approaches based on smoothness prior, which leads to "local" learning algorithms, e.g. kernel-based. Smoothness may not be the only way to obtain "simple functions" : e.g. According to Kolmogorov complexity, *sinus*(x) and *parity*(x) are simple yet they are highly variable (apparently complex) functions.

Let us clarify the notion of "locality" which leads to the **curse of dimensionality** even to learn simple but highly variable functions, and probably to learn what is required for true AI.

Already established for classical non-parametric learning ⇒ generalize it to modern kernel machines.

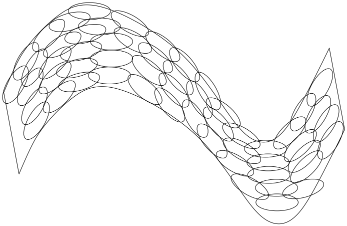
Geometric Intuition

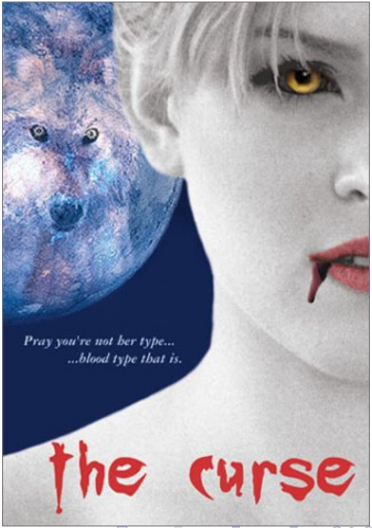
If we have to tile the space or the manifold where the bulk of the distribution is concentrated, then we will need an **exponential number of "patches"** :



Geometric Intuition

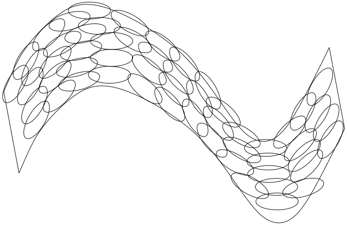
If we have to tile the space or the manifold where the bulk of the distribution is concentrated, then we will need an **exponential number of "patches"** :



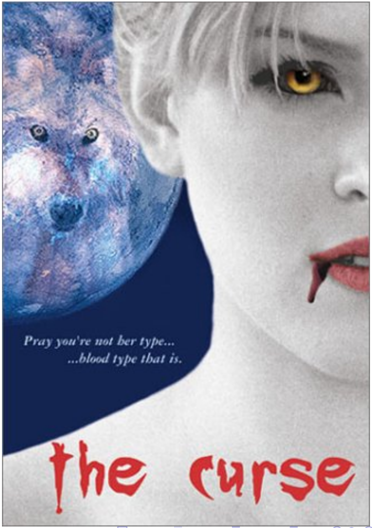


Geometric Intuition

If we have to tile the space or the manifold where the bulk of the distribution is concentrated, then we will need an **exponential number of "patches"** :

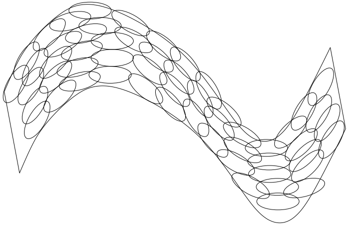


For classification problems no need to cover the whole space/manifold, only decision surface, but still has dim. $d - 1$.



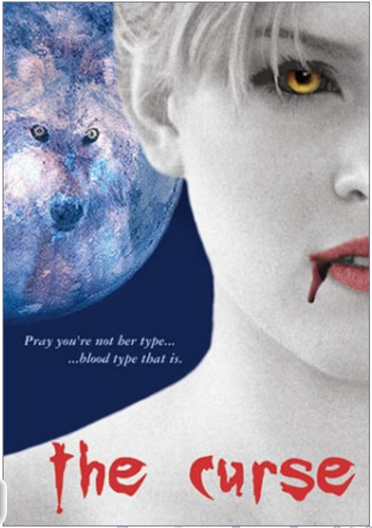
Geometric Intuition

If we have to tile the space or the manifold where the bulk of the distribution is concentrated, then we will need an **exponential number of "patches"** :



For classification problems no need to cover the whole space/manifold, only decision surface, but still has dim. $d - 1$.

Number of required examples $\propto \mathrm{const}^d$



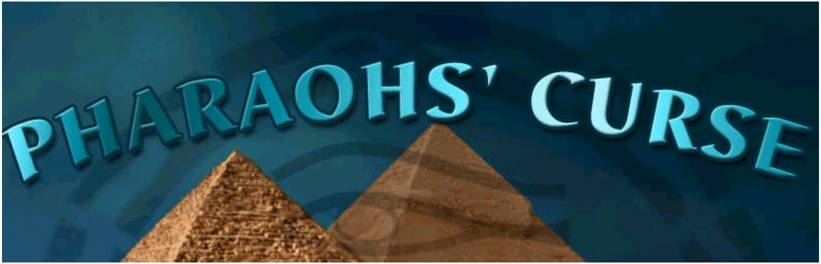
Kernel Density Estimation

For a wide class of kernel density estimators (Härdle et al., 2004), the generalization error converges in $n^{-4/(4+d)}$, i.e.

Kernel Density Estimation

For a wide class of kernel density estimators (Härdle et al., 2004), the generalization error converges in $n^{-4/(4+d)}$, i.e.

The required number of examples to reach a given error level is exponential in d



K nearest neighbors

In the context of K nearest neighbors with weighted L^p metrics of the form $dist(x, y) = \|A(x - y)\|_p$, (Snapp and Venkatesh, 1998) show the generalization error can be written as a series expansion of the form

$$E_n = E_\infty + \sum_{j=2}^{\infty} c_j n^{-j/d}$$

under smoothness constraints on the class distributions, i.e. again

K nearest neighbors

In the context of K nearest neighbors with weighted L^p metrics of the form $dist(x, y) = \|A(x - y)\|_p$, (Snapp and Venkatesh, 1998) show the generalization error can be written as a series expansion of the form

$$E_n = E_\infty + \sum_{j=2}^{\infty} c_j n^{-j/d}$$

under smoothness constraints on the class distributions, i.e. again

The required number of examples to reach a given error level is exponential in d

Kernel Methods

$$f(x) = b + \sum_{i=1}^{n} \alpha_i K_D(x, x_i)$$

Used in classification (KNN, SVM, ...), dimensionality reduction (kernel PCA, LLE, Isomap, Laplacian eigenmaps, ...). May be training data (D) dependent.

SVM's α_i's may depend on x_j far from x_i

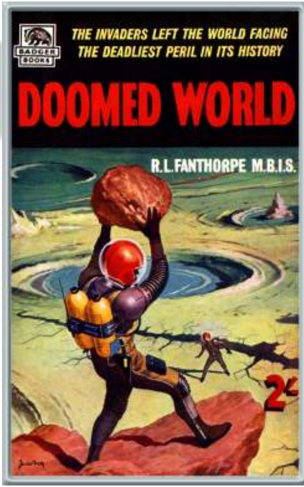
Kernel Methods

$$f(x) = b + \sum_{i=1}^{n} \alpha_i K_D(x, x_i)$$

Used in classification (KNN, SVM, ...), dimensionality reduction (kernel PCA, LLE, Isomap, Laplacian eigenmaps, ...). May be training data (D) dependent.

SVM's α_i's may depend on x_j far from x_i

This talk = **independent of the way the α_i are learned**



When a Test Example is Far from Training Examples

If the kernel is **local**, i.e.

$$\lim_{||x-x_i||\rightarrow\infty} K(x,x_i) \rightarrow c_i$$

then when x gets farther from the training set

$$f(x) \rightarrow b + \sum_i \alpha_i c_i$$

After becoming approx. linear, *the predictor becomes either constant or (approximately) the nearest neighbor predictor (e.g. with the Gaussian kernel)*

When a Test Example is Far from Training Examples

If the kernel is **local**, i.e.

$$\lim_{||x-x_i||\rightarrow\infty} K(x,x_i) \rightarrow c_i$$

then when x gets farther from the training set

$$f(x) \rightarrow b + \sum_i \alpha_i c_i$$

After becoming approx. linear, *the predictor becomes either constant or (approximately) the nearest neighbor predictor (e.g. with the Gaussian kernel)*

In high dimensions, a random test point tends to be **equally far** from most training examples.

Local-Derivative Kernels

SVM : $f(x)$ not local (depends on x_i far from x) through α_i's !
The derivative of f is

$$\frac{\partial f(x)}{\partial x} = \sum_{i=1}^{n} \alpha_i \frac{\partial K(x, x_i)}{\partial x}$$

Local-Derivative Kernels

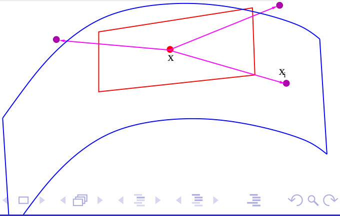
SVM : $f(x)$ not local (depends on x_i far from x) through α_i's !
The derivative of f is

$$\frac{\partial f(x)}{\partial x} = \sum_{i=1}^{n} \alpha_i \frac{\partial K(x, x_i)}{\partial x}$$

Local-derivative kernel

When $\partial f / \partial x$ is (approximately) contained in **the span of the vectors** $(x - x_j)$ **with** x_j **a neighbor of** x

$$\frac{\partial f(x)}{\partial x} \simeq \sum_{x_j \in \mathcal{N}(x)} \gamma_j (x - x_j)$$



Tangent Planes and Decision Surfaces

Manifold learning : $\partial f / \partial x$ span manifold's **tangent plane**

Tangent Planes and Decision Surfaces

Manifold learning : $\partial f/\partial x$ span manifold's **tangent plane**
Classification : $\partial f/\partial x$ is the **decision surface normal vector**

Tangent Planes and Decision Surfaces

Manifold learning : $\partial f / \partial x$ span manifold's **tangent plane**
Classification : $\partial f / \partial x$ is the **decision surface normal vector**

Constraining $\partial f / \partial x$ in the span of the neighbors is a very strong constraint, possibly leading to **high-variance** *estimators.*

Tangent Planes and Decision Surfaces

Manifold learning : $\partial f / \partial x$ span manifold's **tangent plane**
Classification : $\partial f / \partial x$ is the **decision surface normal vector**

Constraining $\partial f / \partial x$ in the span of the neighbors is a very strong constraint, possibly leading to **high-variance** *estimators.*

SVMs with Gaussian kernel are **local-derivative**

Tangent Planes and Decision Surfaces

Manifold learning : $\partial f / \partial x$ span manifold's **tangent plane**
Classification : $\partial f / \partial x$ is the **decision surface normal vector**

Constraining $\partial f / \partial x$ in the span of the neighbors is a very strong constraint, possibly leading to **high-variance** *estimators.*

SVMs with Gaussian kernel are **local-derivative**

LLE is **local-derivative**

Tangent Planes and Decision Surfaces

Manifold learning : $\partial f / \partial x$ span manifold's **tangent plane**
Classification : $\partial f / \partial x$ is the **decision surface normal vector**

Constraining $\partial f / \partial x$ in the span of the neighbors is a very strong constraint, possibly leading to **high-variance** *estimators.*

SVMs with Gaussian kernel are **local-derivative**

LLE is **local-derivative**

Isomap is **local-derivative**

Tangent Planes and Decision Surfaces

Manifold learning : $\partial f/\partial x$ span manifold's **tangent plane**
Classification : $\partial f/\partial x$ is the **decision surface normal vector**

Constraining $\partial f/\partial x$ in the span of the neighbors is a very strong constraint, possibly leading to **high-variance** *estimators.*

SVMs with Gaussian kernel are **local-derivative**

LLE is **local-derivative**

Isomap is **local-derivative**

Kernel PCA with Gaussian kernel is **local-derivative**

Tangent Planes and Decision Surfaces

Manifold learning : $\partial f / \partial x$ span manifold's **tangent plane**
Classification : $\partial f / \partial x$ is the **decision surface normal vector**

Constraining $\partial f / \partial x$ in the span of the neighbors is a very strong constraint, possibly leading to **high-variance** *estimators.*

SVMs with Gaussian kernel are **local-derivative**

LLE is **local-derivative**

Isomap is **local-derivative**

Kernel PCA with Gaussian kernel is **local-derivative**

Spectral clustering with Gaussian kernel is **local-derivative**

General Curse of Dimensionality Argument

Locality. Show that crucial properties of $f(x)$ (e.g. tangent plane, decision surface normal vector) depend mostly on examples in ball $\mathcal{N}(x)$.

General Curse of Dimensionality Argument

Locality. Show that crucial properties of $f(x)$ (e.g. tangent plane, decision surface normal vector) depend mostly on examples in ball $\mathcal{N}(x)$.

Smooothness. Show that within $\mathcal{N}(x)$, crucial property of $f(x)$ must vary slowly ($=$ smoothness within $\mathcal{N}(x)$).

General Curse of Dimensionality Argument

Locality. Show that crucial properties of $f(x)$ (e.g. tangent plane, decision surface normal vector) depend mostly on examples in ball $\mathcal{N}(x)$.

Smooothness. Show that within $\mathcal{N}(x)$, crucial property of $f(x)$ must vary slowly ($=$ smoothness within $\mathcal{N}(x)$).

Complexity. Consider targets that vary sufficiently so that one needs to consider $O(\mathrm{const}^d)$ different neighborhoods, with significantly different properties in each neighborhood.

Spectral Manifold Learning Algorithms

Many manifold learning algorithms can be seen as kernel machines with data-dependent kernel (LLE, Isomap, kernel PCA, Laplacian Eigenmaps, charting, etc...).

Locality

Shown for the estimated tangent plane.

Spectral Manifold Learning Algorithms

Many manifold learning algorithms can be seen as kernel machines with data-dependent kernel (LLE, Isomap, kernel PCA, Laplacian Eigenmaps, charting, etc...).

Locality

Shown for the estimated tangent plane.

Smoothness of $f(\cdot)$

The tangent plane varies slowly within $\mathcal{N}(x)$, since it is in the span of vectors $x - x_i$.

Spectral Manifold Learning Algorithms

Many manifold learning algorithms can be seen as kernel machines with data-dependent kernel (LLE, Isomap, kernel PCA, Laplacian Eigenmaps, charting, etc...).

Locality

Shown for the estimated tangent plane.

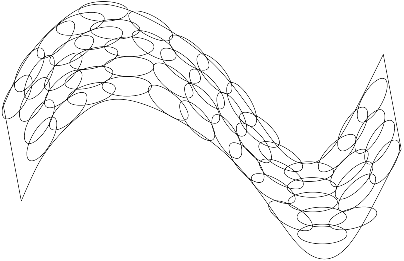
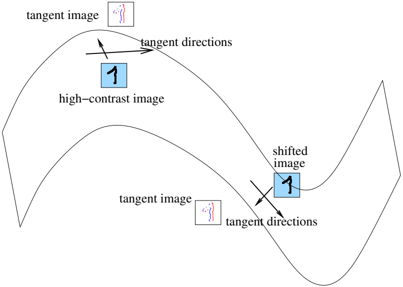
Smoothness of $f(\cdot)$

The tangent plane varies slowly within $\mathcal{N}(x)$, since it is in the span of vectors $x - x_i$.

Non-Smoothness of Target

If the underlying manifold has high curvature in many places, we are doomed...

Ex : Translation of a High Contrast Image

N.B. ∃ examples of **non-local learning** with **no domain-specific prior knowledge** which worked on learning such manifolds (rotations and translations), (Bengio and Monperrus, 2005), generalizing far from training examples.

The 1-Norm Soft Margin SVM with Gaussian Kernel

Locality

As shown in (Keerthi and Lin, 2003), the SVM becomes constant when $\sigma \rightarrow 0$ or $\sigma \rightarrow \infty \Rightarrow$ notion of locality w.r.t σ.
Local-derivative : Locality of normal vector of decision surface.

The 1-Norm Soft Margin SVM with Gaussian Kernel

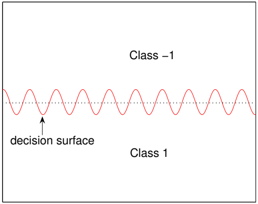
Locality

As shown in (Keerthi and Lin, 2003), the SVM becomes constant when $\sigma \to 0$ or $\sigma \to \infty \Rightarrow$ notion of locality w.r.t σ.
Local-derivative : Locality of normal vector of decision surface.

Smoothness of $f(\cdot)$

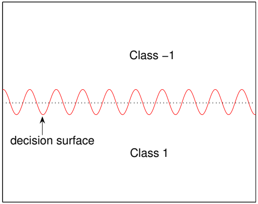
When there are training examples at a distance of the order of σ, the normal vector is almost constant in a ball whose radius is small with respect to σ.

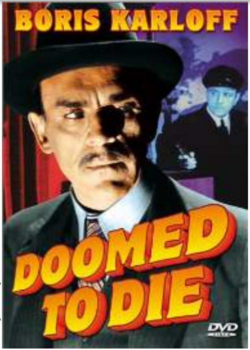
Simple but Highly Variable Functions : Difficult to Learn



This "complex" sinusoidal decision surface cannot be learned with less than 10 Gaussians. However, in "C" language, it has a high prior.

Simple but Highly Variable Functions : Difficult to Learn



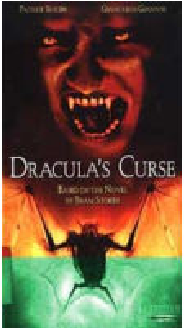


This "complex" sinusoidal decision surface cannot be learned with less than 10 Gaussians. However, in "C" language, it has a high prior.

Corollary of (Schmitt, 2002)

If $\exists$ a line in $\mathbb{R}^d$ that intersects m times with the decision surface S (and is not included in S), then one needs at least $\lceil \frac{m}{2} \rceil$ Gaussians (of same width) to learn S with a Gaussian kernel classifier.

The Parity Problem



parity :

$$(b_1, \ldots, b_d) \in \{0,1\}^d \mapsto \begin{cases} 1 \text{ if } \sum_{i=1}^{d} b_i \text{ is even} \\ -1 \text{ otherwise} \end{cases}$$

Theorem

A Gaussian kernel classifier needs at least 2^{d-1} Gaussians (i.e. support vectors) to learn the parity function (when Gaussians have fixed width and are centered on training points).

Then What ?

- Local Kernel machines won't scale to highly variable functions in high manifold dimension. Good news : SVMs interpolate between very local and very smooth (vary σ). Bad news : if target function structured but not smooth...

Then What ?

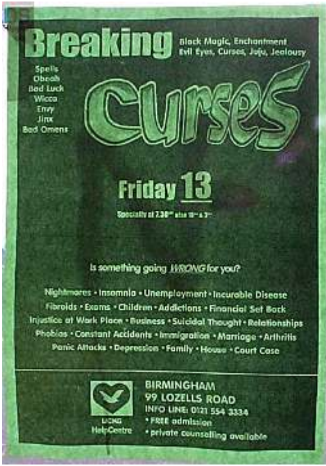
- Local Kernel machines won't scale to highly variable functions in high manifold dimension. Good news : SVMs interpolate between very local and very smooth (vary σ). Bad news : if target function structured but not smooth...
- The no-free-lunch thm : no universal recipe without appropriate prior.

Then What ?

- Local Kernel machines won't scale to highly variable functions in high manifold dimension. Good news : SVMs interpolate between very local and very smooth (vary σ). Bad news : if target function structured but not smooth...
- The no-free-lunch thm : no universal recipe without appropriate prior.
- Is there hope ?

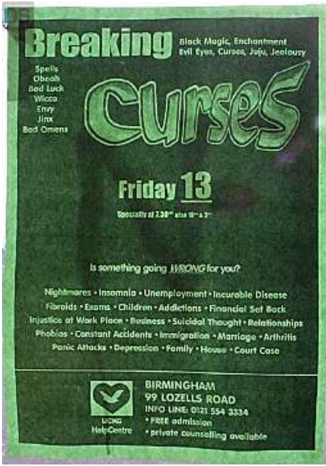
Then What ?

- Local Kernel machines won't scale to highly variable functions in high manifold dimension. Good news : SVMs interpolate between very local and very smooth (vary σ). Bad news : if target function structured but not smooth...
- The no-free-lunch thm : no universal recipe without appropriate prior.
- Is there hope ?
- Humans seem to do learn such functions !
- There might be loose enough priors on general classes of functions that allow non-local learning algorithms to learn them.

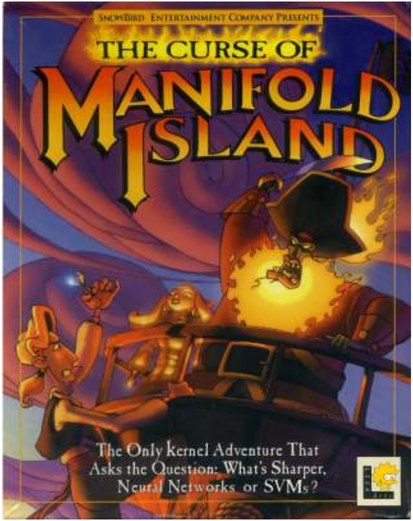


Then What ?

- Local Kernel machines won't scale to highly variable functions in high manifold dimension. Good news : SVMs interpolate between very local and very smooth (vary σ). Bad news : if target function structured but not smooth...
- The no-free-lunch thm : no universal recipe without appropriate prior.
- Is there hope ?
- Humans seem to do learn such functions !
- There might be loose enough priors on general classes of functions that allow non-local learning algorithms to learn them.
- **Let us explore priors / learning algorithms beyond the smoothness prior.**



~~Questions~~ Coffee time !



Bibliography

Bengio, Y., Delalleau, O., and Le Roux, N. (2005).
The curse of dimensionality for local kernel machines.
Technical Report 1258, Département d'informatique et recherche opérationnelle, Université de Montréal.

Bengio, Y. and Monperrus, M. (2005).
Non-local manifold tangent learning.
In Saul, L., Weiss, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 17*. MIT Press.

Härdle, W., Müller, M., Sperlich, S., and Werwatz, A. (2004).
Nonparametric and Semiparametric Models.
Springer, http ://www.xplore-stat.de/ebooks/ebooks.html.

Keerthi, S. S. and Lin, C.-J. (2003).
Asymptotic behaviors of support vector machines with Gaussian kernel.
Neural Computation, 15(7) :1667–1689.

Schmitt, M. (2002).
Descartes' rule of signs for radial basis function neural networks.
Neural Computation, 14(12) :2997–3011.

Snapp, R. R. and Venkatesh, S. S. (1998).
Asymptotic derivation of the finite-sample risk of the k nearest neighbor classifier.
Technical Report UVM-CS-1998-0101, Department of Computer Science, University of Vermont.