Scaling Learning Algorithms Towards AI

Authors: Yoshua Bengio, Yann LeCun
Presenter: Marilyn Vazquez

George Mason University

February 10, 2017

Outline

1. Curse of Dimensionality
2. Shallow Learning
3. Deep Learning
4. Results
5. Conclusion

Curse of Dimensionality

The curse of dimensionality can be viewed either as the limitation on data analysis due to the large amount of data or parameters needed to analyze the data.

Curse of Dimensionality: Example 1

Kernel density estimation: At point $x \in \mathbb{R}^d$, the kernel density $\hat{q}$ estimates the real density q with high probability, i.e.

$$\mathbb{E}(\hat{q}(x_i)) = \mathbb{E}\left(\frac{\sigma^{-d}}{N}\sum_{j=1}^{N} K_\sigma(x_i, x_j)\right)$$

$$= \mathbb{E}\left(\frac{1}{N}\sum_{j=1}^{N}\frac{e^{-\frac{||x_i - x_j||}{2\sigma^2}}}{(2\pi\sigma^2)^{\frac{2}{d}}}\right) \rightarrow q(x_i) + \mathcal{O}(\sigma^2, N^{-\frac{1}{2}}\sigma^{-\frac{d}{2}}\sqrt{q(x)})$$

where the bias error, σ^2, is dominant with large data and variance error, $N^{-\frac{1}{2}}\sigma^{-\frac{d}{2}}$, blows up if we take $\sigma \rightarrow 0$

Error

To find an optimal bandwidth σ, we can balance errors:

$$\sigma^2 = c_1 N^{-\frac{1}{2}} \sigma^{-\frac{d}{2}} \implies \sigma^{\frac{4+d}{2}} = c_1 N^{-\frac{1}{2}} \implies \sigma = c_1 N^{-\frac{1}{4+d}}$$

$$\implies \text{error} = c_2 N^{-\frac{2}{4+d}}$$

So that if for $d = 1$ we need n_1 points to achieve the fixed error e_1, then increasing the dimension to d, we need $n_1^{\frac{4+d}{5}}$ data points i.e. exponential in dimension!

Curse of Dimensionality: Example 2

Smooth function representation: A gaussian kernel machine is a representation

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

where x_i are the base points, w_i are weights found through regression, and $K(x_i, x)$ is a Gaussian kernel.

Theorem

Let $f : \mathbb{R} \to \mathbb{R}$ computed by a Gaussian Kernel machine with k base points (k non-zero w_i's). Then f has at most $2k$ zeros.

Curse of Dimensionality: Example 2

Smooth Function Representation

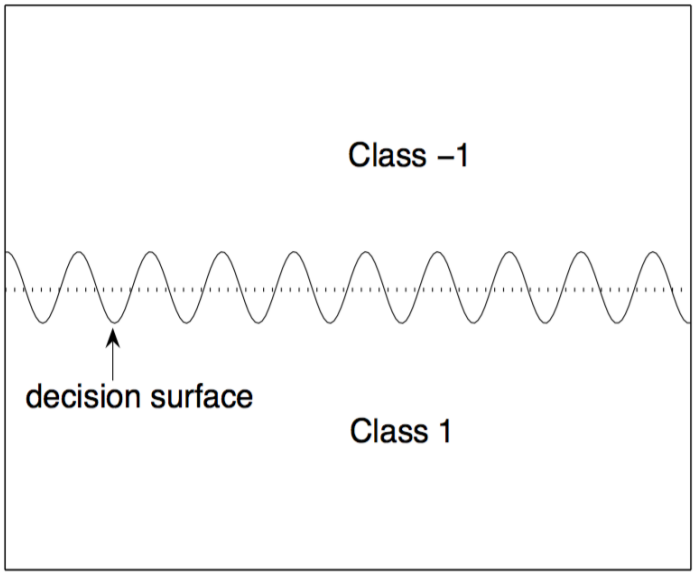


Figure 2: The dotted line crosses the decision surface 19 times: one thus needs at least 10 Gaussians to learn it with an affine combination of Gaussians with same width.

Curse of Dimensionality: Example 2

Smooth function representation: A gaussian kernel machine is a representation

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

where x_i are the base points, w_i are weights found through regression, and $K(x_i, x)$ is a Gaussian kernel.

Corollary

In $\mathbb{R}^d$, if the learning problem requires f to change sign at least $2k$ times along some straight line, then the kernel machine must have at least k base points (k non-zero w_i's).

Curse of Dimensionality: Example 3

Local Derivative : For a Gaussian Kernel classifier, the normal of the tangent of the decision surface at x is constrained to approximately lie in the span of the vectors $(x - x_i)$, where $||x - x_i||$ is small compared to σ and x_i are in the training set.

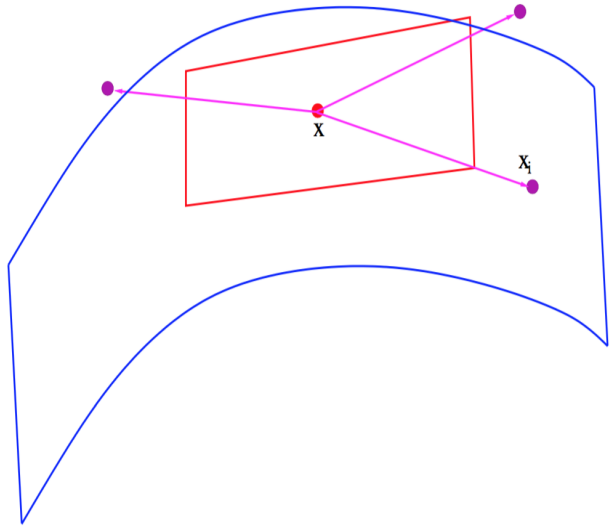


Figure 3: For local manifold learning algorithms such as LLE, Isomap and kernel PCA, the manifold tangent plane at x is in the span of the difference vectors between test point x and its neighbors x_i in the training set. This makes these algorithms sensitive to the curse of dimensionality, when the manifold is high-dimensional and not very flat.

Local Derivative

Brief explanation:

$$\text{For } f(x) = b + \sum_{i=1}^{n} w_i K(x, x_i) = b + \sum_{i=1}^{n} w_i e^{-\frac{||x-x_i||^2}{\sigma^2}}$$

$$\text{We get } \frac{\partial f(x)}{\partial x} = -\sum_{i=1}^{n} \frac{2(x-x_i)w_i}{\sigma^2} e^{-\frac{||x-x_i||^2}{\sigma^2}}$$

Note that the dominant terms are those for which x_i is a near neighbor of x, so that we approximately get

$$\frac{\partial f(x)}{\partial x} \approx -\sum_{i=1}^{m} w_i' e^{-\frac{||x-x_i||^2}{\sigma^2}}$$

where $w_i' = \frac{2(x-x_i)w_i}{\sigma^2}$ and $m \leq n$.

Local Derivative Example

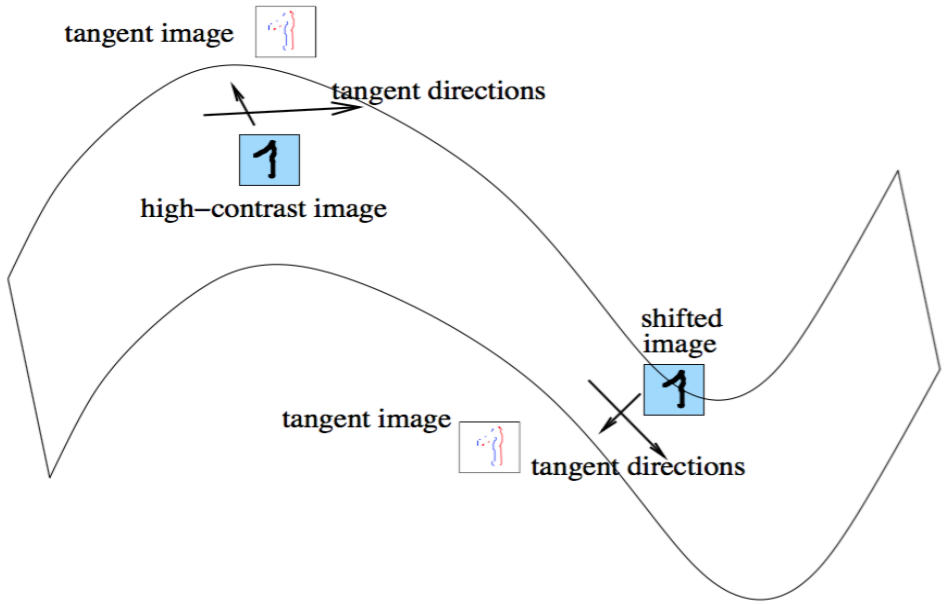


Figure 4: The manifold of translations of a high-contrast image has high curvature. A smooth manifold is obtained by considering that an image is a sample on a discrete grid of an intensity function over a two-dimensional space. The tangent vector for translation is thus a *tangent image*, and it has high values only on the edges of the ink. The tangent plane for an image translated by only one pixel looks similar but changes abruptly since the edges are also shifted by one pixel. Hence the two tangent planes are almost orthogonal, and the manifold has high curvature, which is bad for local learning methods, which must cover the manifold with many small linear patches to correctly capture its shape.

Getting Around the Curse of Dimensionality

There is no universal solution to the curse of dimensionality; however, for particular purposes you can make assumption that may help.

"We hypothesize that many tasks in the AI set may be built around common representations, which can be understood as a set of interrelated concepts"

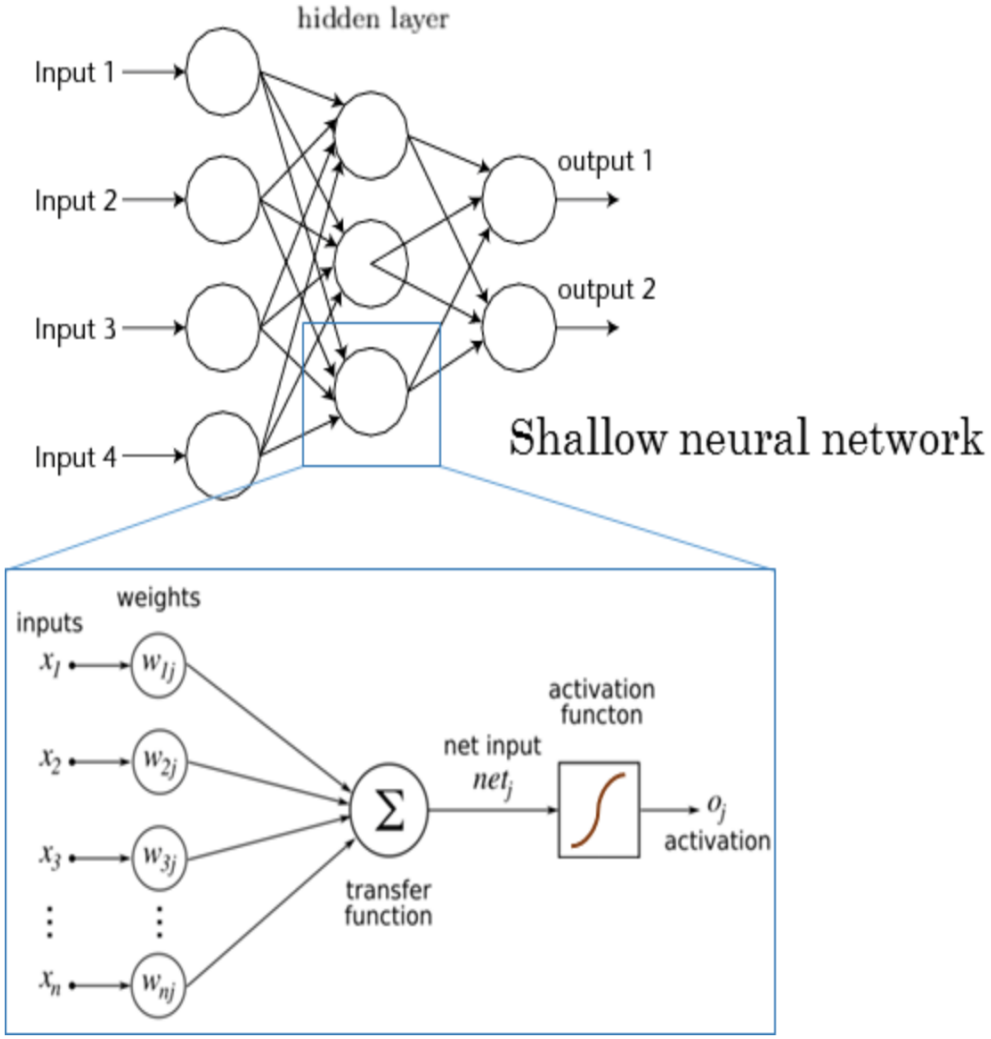
Translation: Use the mathematical idea of composition of functions to build a complicated function from simple parts (common representation), such as Gaussian kernels or any basis functions.

Shallow Learning

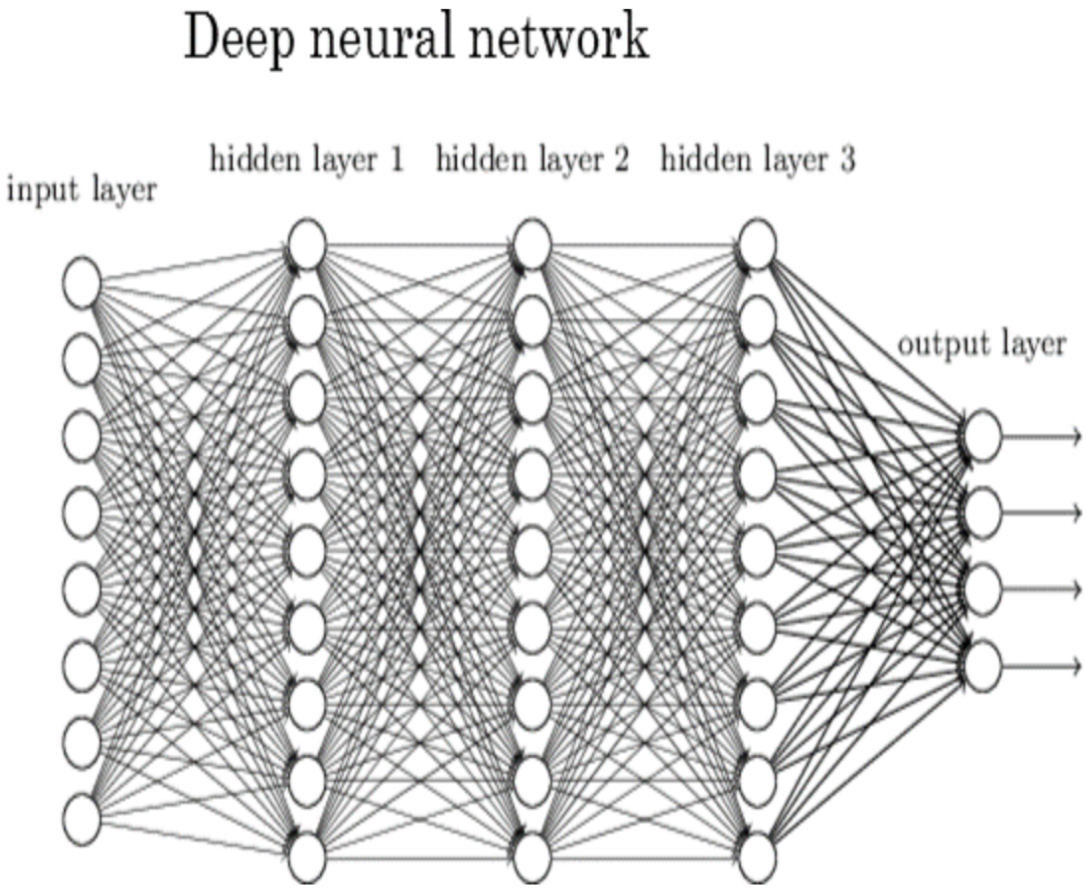
$$f(x) = b + \sum_{i=1}^{N} w_i \phi_i(x)$$

where w_i results from the training, the basis could be something fixed such as $K(x_i, x)$ or also a result of the training.

Shallow Learning



Deep Learning as Neural Network



Deep Learning as Function Composition

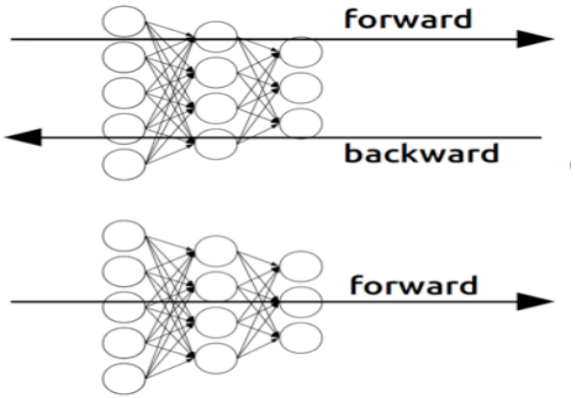
Let f_{jk} represent the jth feature in the k layer

$$
\begin{aligned}
f_{j1}(x) &= b_{j,1} + \sum_i w_{ij1} K(x_{ij1}, x) \\
f_{j,2}(x) &= b_{j,2} + \sum_i w_{ij2} K(x_{ij2}, f_{j1}(x)) \\
&\vdots
\end{aligned}
$$

They conjecture that by allowing these compositions, we will need fewer parameters to fit compared to a shallow representation

Deep Learning Steps

- **Step 1** Initialization via unsupervised learning with a feedback that helps reconstruct input from output
- **Step 2** Refine via gradient-descent supervised learning



Deep Learning

$$c_{ijxy} = \tanh\left(b_{ij} + \sum_{k} \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} w_{ijkpq} c_{(i-1),k,(x+p),(y+q)}\right)$$

Feature detection layer

Feature pooling layer

Deep Learning

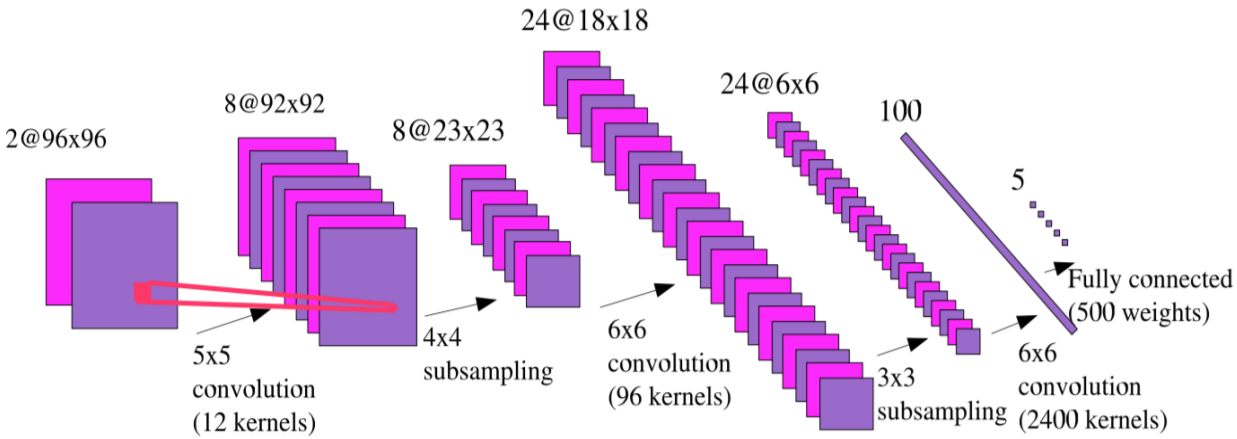


Figure 5: The architecture of the convolutional net used for the NORB experiments. The input is an image pair, the system extracts 8 feature maps of size 92×92, 8 maps of 23×23, 24 maps of 18×18, 24 maps of 6×6, and 100 dimensional feature vector. The feature vector is then transformed into a 5-dimensional vector in the last layer to compute the distance with target vectors.

Results

Classifier	Defor-mations	Error %	Reference
Knowledge-free methods			
2-layer NN, 800 hid. units		1.60	Simard et al. 2003
3-layer NN, 500+300 units		1.53	Hinton et al. 2006
SVM, Gaussian kernel		1.40	Cortes et al. 1992
Unsupervised stacked RBM + backprop		0.95	Hinton et al. 2006
Convolutional networks			
Convolutional network LeNet-5		0.80	Ranzato et al. 2006
Convolutional network LeNet-6		0.70	Ranzato et al. 2006
Conv. net. LeNet-6 + unsup. learning		0.60	Ranzato et al. 2006
Training set augmented with affine distortions			
2-layer NN, 800 hid. units	Affine	1.10	Simard et al. 2003
Virtual SVM, deg. 9 poly	Affine	0.80	DeCoste et al. 2002
Convolutional network,	Affine	0.60	Simard et al. 2003
Training set augmented with elastic distortions			
2-layer NN, 800 hid. units	Elastic	0.70	Simard et al. 2003
SVM Gaussian Ker. + on-line training	Elastic	0.67	this volume, chapter 13
Shape context features + elastic K-NN	Elastic	0.63	Belongie et al. 2002
Convolutional network	Elastic	0.40	Simard et al. 2003
Conv. net. LeNet-6	Elastic	0.49	Ranzato et al. 2006
Conv. net. LeNet-6 + unsup. learning	Elastic	0.39	Ranzato et al. 2006

Table 1: Test error rates of various learning models on the MNIST dataset. Many results obtained with deslanted images or hand-designed feature extractors were left out.

Sample Data

Figure 7: Some of the 291,600 examples from the *jittered-cluttered* training set (left camera images). Each column shows images from one category. A 6-th background category is added

Results

	SVM	Conv Net			SVM/Conv
test error	43.3%	16.38%	7.5%	7.2%	5.9%
train time (min*GHz)	10,944	420	2,100	5,880	330+
test time per sample (sec*GHz)	2.2	0.04			0.06+
#SV	5%				2%
parameters	σ=10^4 C=40	step size = 2×10^{-5} - 1×10^{-6}			dim=100 σ=5 C=1

Table 3: Testing error rates and training/testing timings on the *jittered-cluttered* dataset of different methods. The timing is normalized to hypothetical 1GHz single CPU. The convolutional nets have multiple results with different training passes due to its iterative training.

Summary

- The curse of dimensionality can limit the amounts of data that can be analyzed
- We can not completely get rid of the curse of dimensionality, but we can go around it if we make some assumptions
- Shallow learning assumes that we can represent functions with the smooth functions such as gaussian kernels
- Deep learning assumes that complicated functions can be build by composing simple functions such as Gaussian kernels
- Deep learning is composed of several two-layer sequences, the feature detection layer and the feature pooling layer, in which each layer a non-linear supervision step is performed.
- The authors show successful results in image classification

References


Yoshua Bengio and Yann LeCun, *Scaling Learning Algorithms towards AI*, Large-Scale Kernel Machine, 2007.


Leslie Lamport, *Deep Learning and Convolutional Neural Networks*, RSIP Vision Blogs http://www.rsipvision.com/exploring-deep-learning/.


Jianxin Wu, *Introduction to Convolutional Neural Networks*, National Key Lab for Novel Software Technology, Nanjing University, China 2016.