Feature Learning for Image Data: from Dictionary Learning to Deep Learning

Jeremy Watt
jermwatt@gmail.com

Reza Borhani
borhani@u.northwestern.edu

Today's talk overview

Part I. An overview of features / unsupervised feature learning- principles of feature engineering, unsupervised feature learning, dictionary learning / sparse coding

Part II. Supervised feature learning – elements of function approximation, neural networks and fixed basis kernels, learning features for regression and classification

Part III. Deep learning and nonlinear optimization – a host of reasons, convolutional networks, a primer on the backpropogation algorithm

Part III

The resurgence of neural networks

The most recent resurgence in neural nets is largely due to

1. **Enormous datasets – esp. image, speech, and text**
2. Huge advancement in computer speed / lowering prices
3. Neural nets scale more gracefully than fixed basis kernels
4. Neural nets naturally permit embedment of formalized knowledge
5. Small collection of critical technical improvements
6. (conjecture) Compositional bases may be superior to fixed ones for representing natural data

Enormous datasets previously unavailable

- e.g., a major paper in pedestrian detection [22] in 2005 uses 2500 examples (prior papers often used a smaller set of only 700),
- by 2012 researchers used datasets with tens or hundreds of thousands of examples [23]

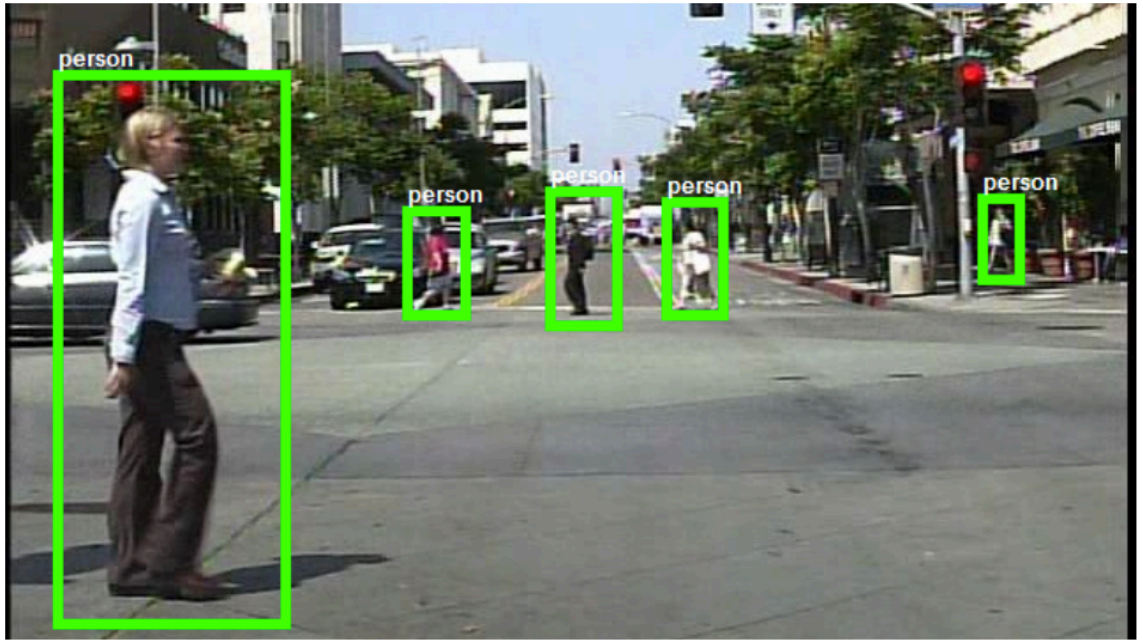


Image taken from [23]

Enormous datasets previously unavailable

- e.g., a major paper in face detection [24] in 2004 uses 5000 facial images
- by 2014 researchers at Facebook use a dataset with 4 million facial images [25]

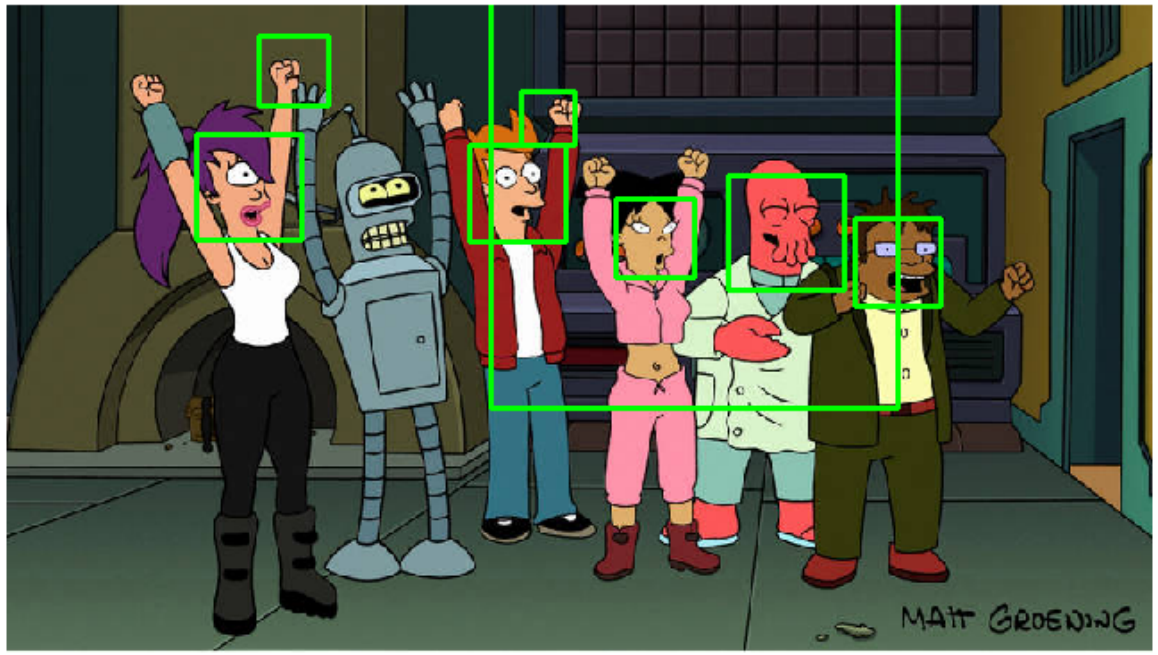
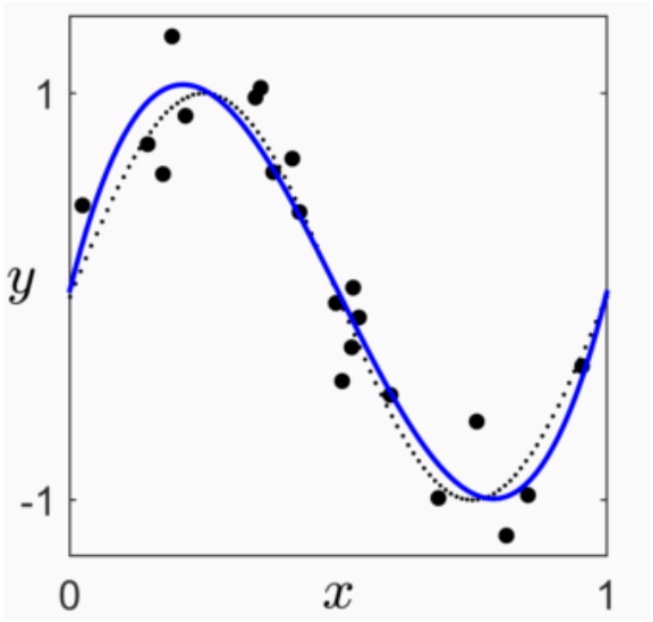


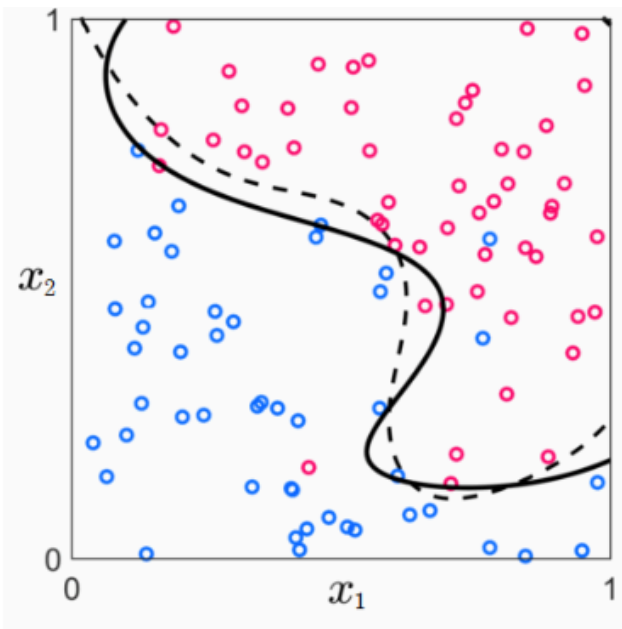
Image taken from [26]

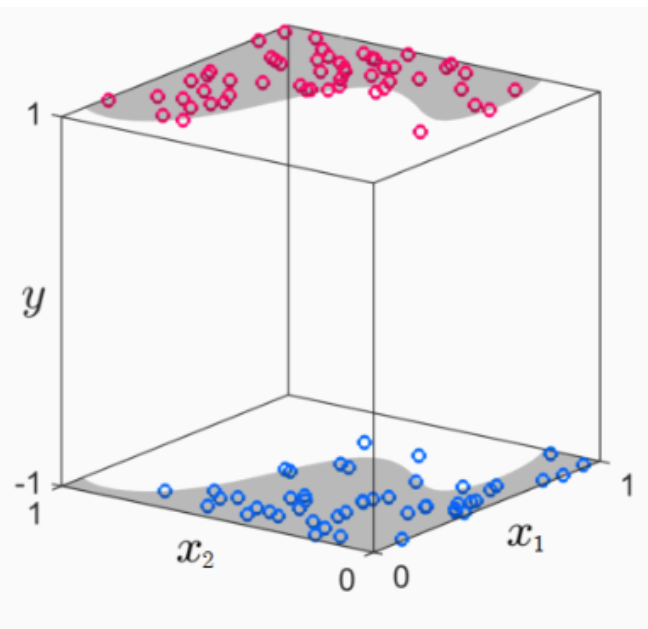
Enormous datasets previously unavailable

Why is this important? Because remember:

supervised learning (regression and classification) are *noisy subsampled function approximation problems*







Images taken from [29]

So unless

a. The phenomenon under study is relatively simple/low dimensional

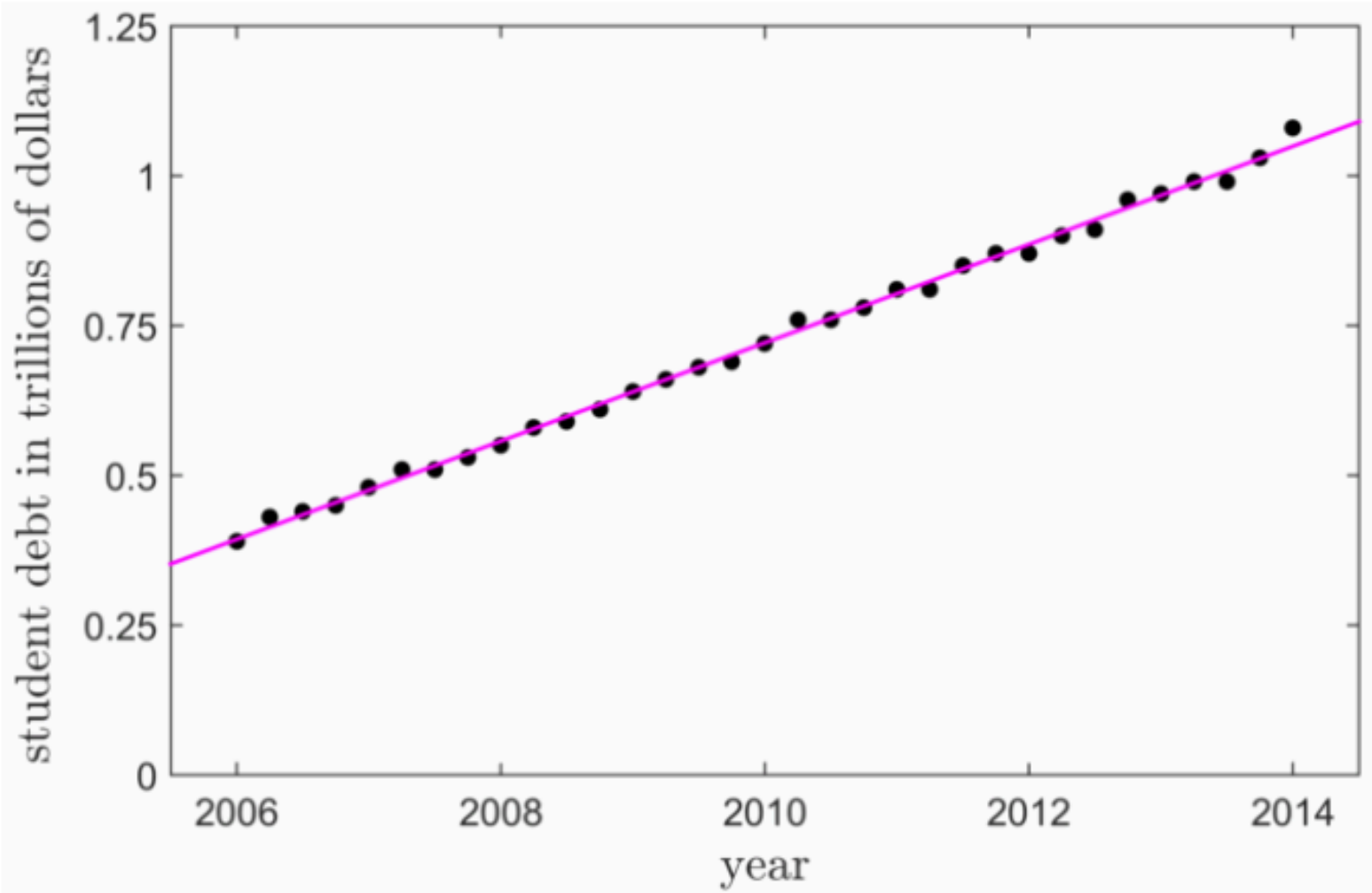
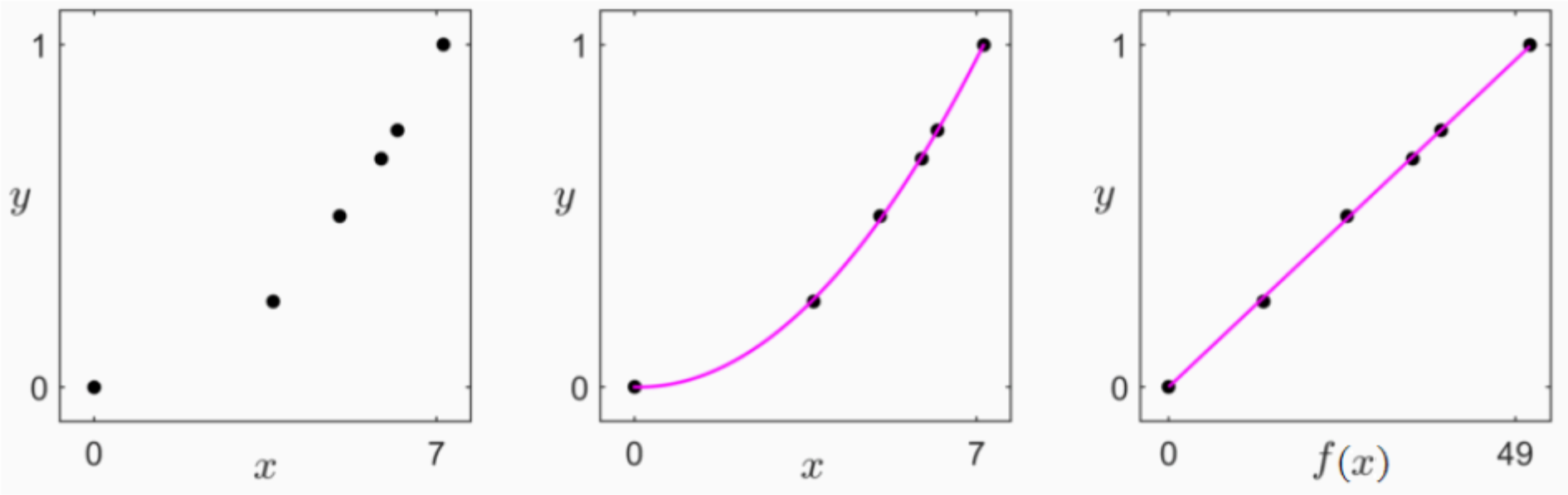


Image taken from [29]

So unless

a. The phenomenon under study is relatively simple/low dimensional
b. You have a very solid formal understanding of the phenomenon (i.e., you can formulate very good features)

You don't need many datapoints to be confident you are capturing an underlying phenomenon when you understand pretty well how it works.
e.g., gravity: the distance an object falls ~ (time it has been falling)^2

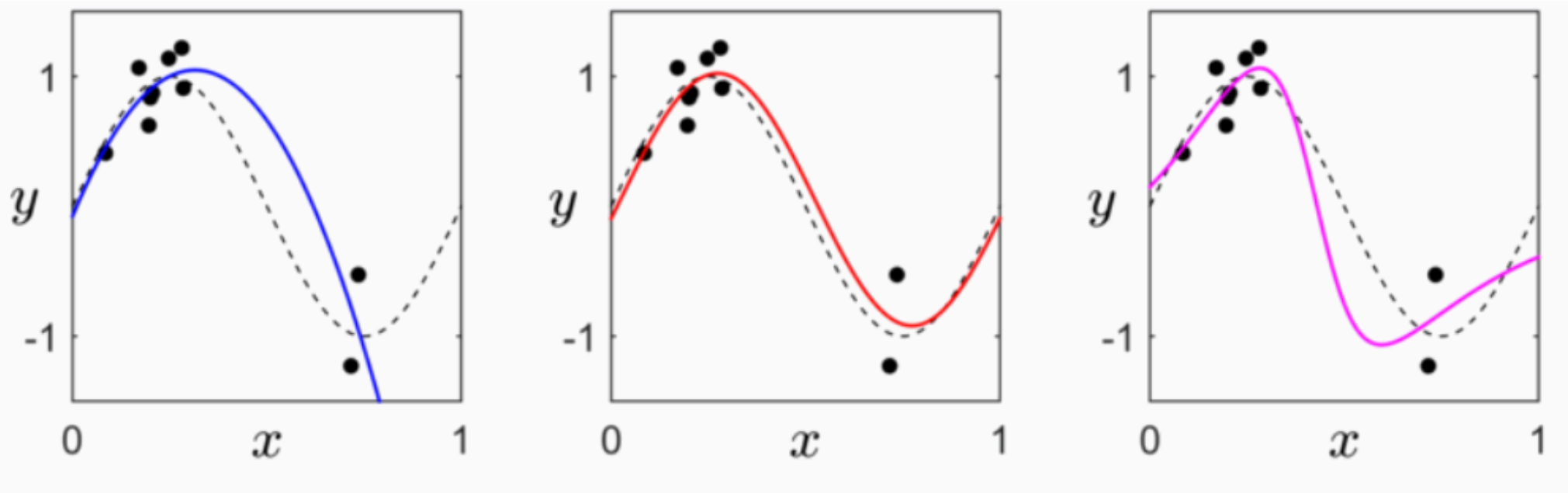


And so the feature $f(x) = x^2$, where x denotes time, should create a linear relationship in the corresponding feature space (it does!)

Images taken from [29]

So unless

a. The phenomenon under study is relatively simple/low dimensional
b. You have a very solid formal understanding of the phenomenon (i.e., you can formulate very good features)
c. Your underlying data generating function lies in a span of your chosen basis elements (e.g., polynomials, Fourier elements, neural nets)



Polynomial Fourier Single-hidden layer NN

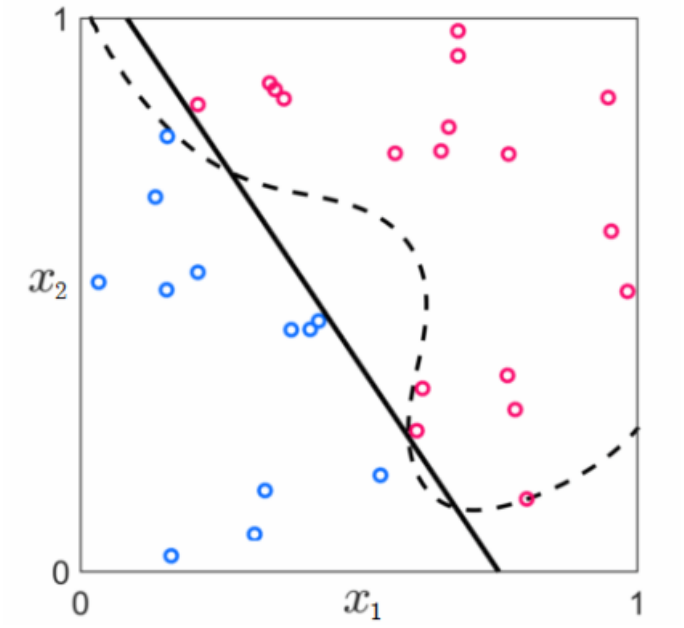
Images taken from [29]

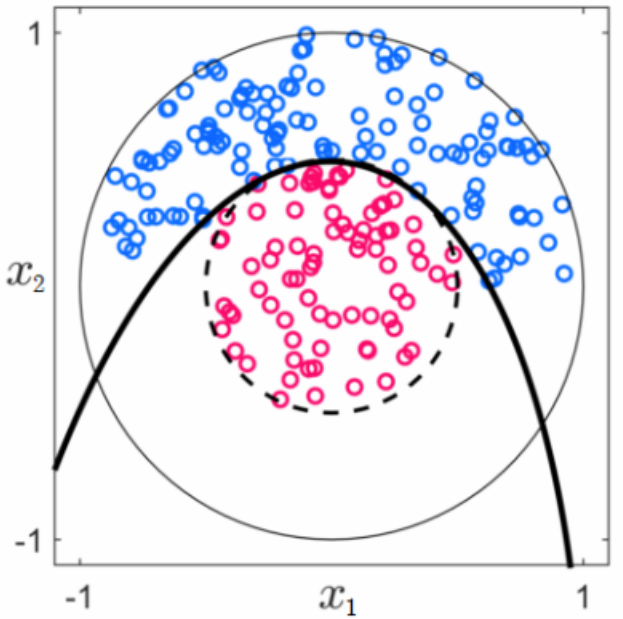
So unless

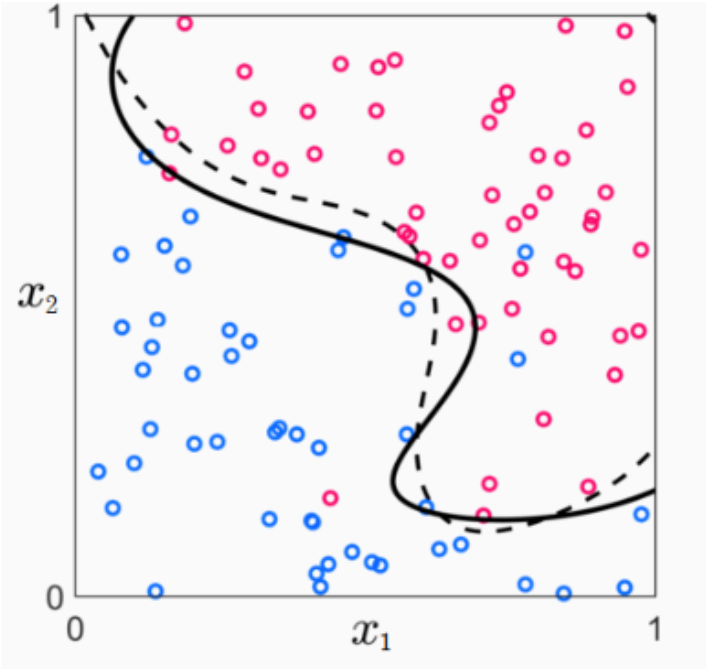
a. The phenomenon under study is relatively simple/low dimensional
b. You have a very solid formal understanding of the phenomenon (i.e., you can formulate very good features)
c. Your underlying data generating function lies in a span of your chosen basis elements (e.g., polynomials, Fourier elements, neural nets)

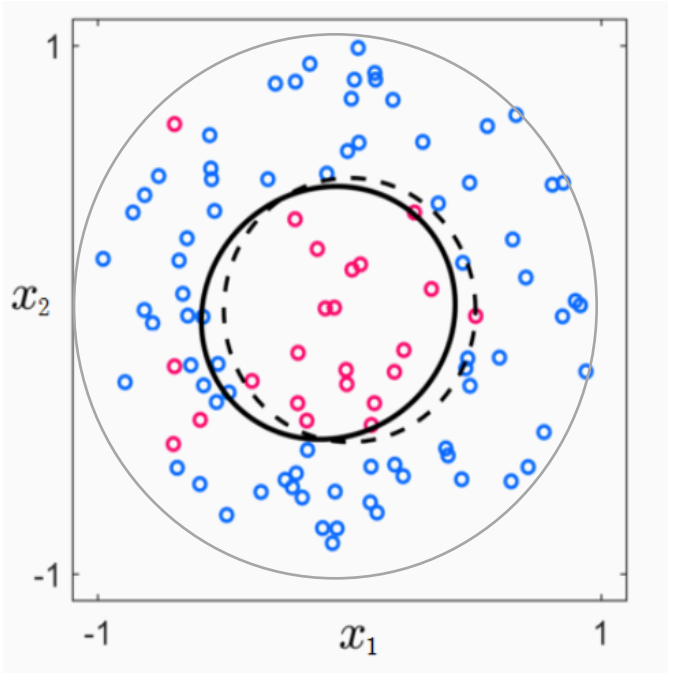
For high dimensional data (e.g., image, audio, text) you will always need a very large dataset in order to learn the function underlying a dataset

No amount of cross-validation using nonlinear bases will learn e.g., the correct boundary between two classes









Images taken from [29]

So unless

a. The phenomenon under study is relatively simple/low dimensional
b. You have a very solid formal understanding of the phenomenon (i.e., you can formulate very good features)
c. Your underlying data generating function lies in a span of your chosen basis elements (e.g., polynomials, Fourier elements, neural nets)

For high dimensional data (e.g., image, audio, text) you will always need a very large dataset in order to learn the function underlying a dataset

No amount of cross-validation using nonlinear bases will learn e.g., the correct boundary between two classes

Increasing a dataset massively improves the predictive power of supervised methods applied to phenomenon we understand poorly

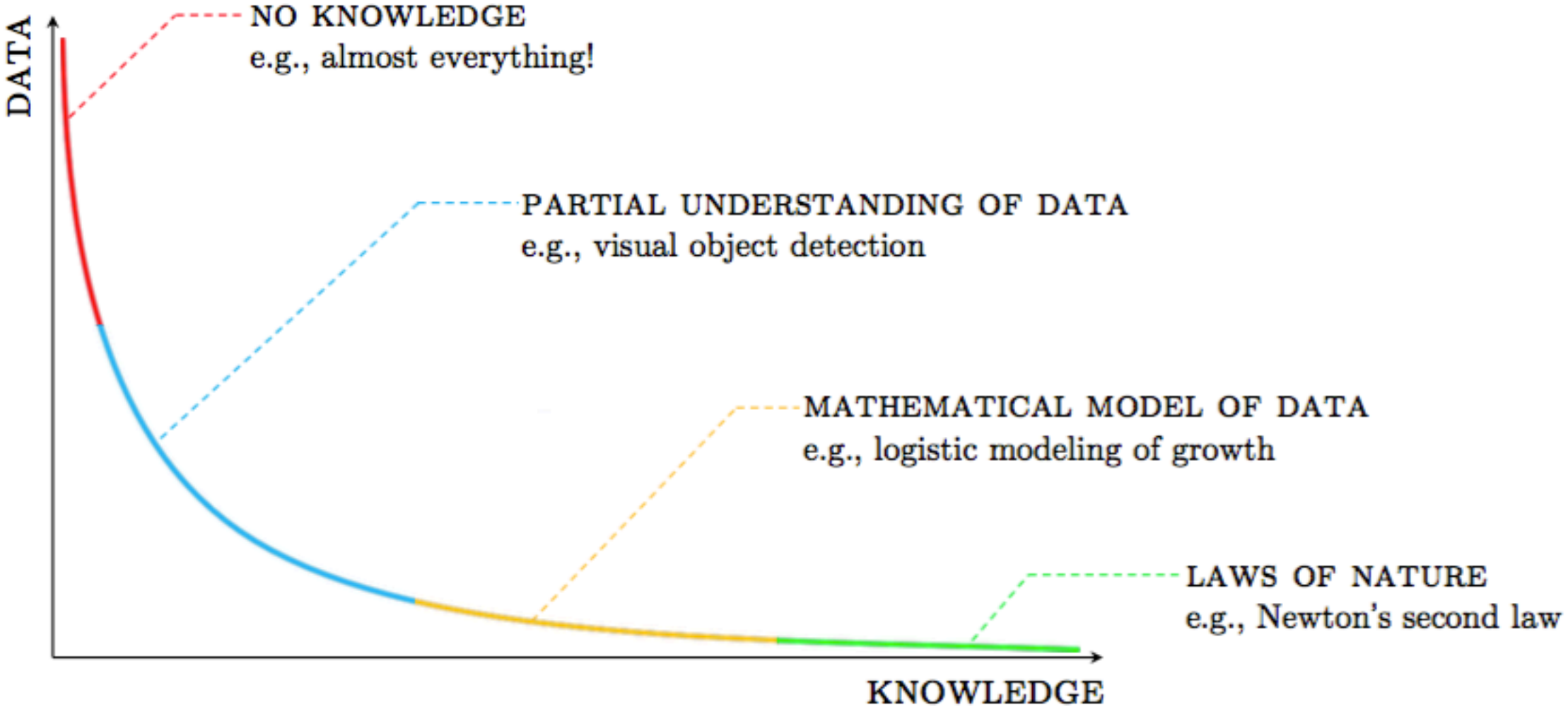


Image taken from [29]

The resurgence of neural networks

The most recent resurgence in neural nets is largely due to

1. Enormous datasets – esp. image, speech, and text
2. **Huge advancement in computer speed / lowering prices**
3. Neural nets scale more gracefully than fixed basis kernels
4. Neural nets naturally permit embedment of formalized knowledge
5. Small collection of critical technical improvements
6. (conjecture) Compositional bases may be superior to fixed ones for representing natural data

Moore’s law has held very well!

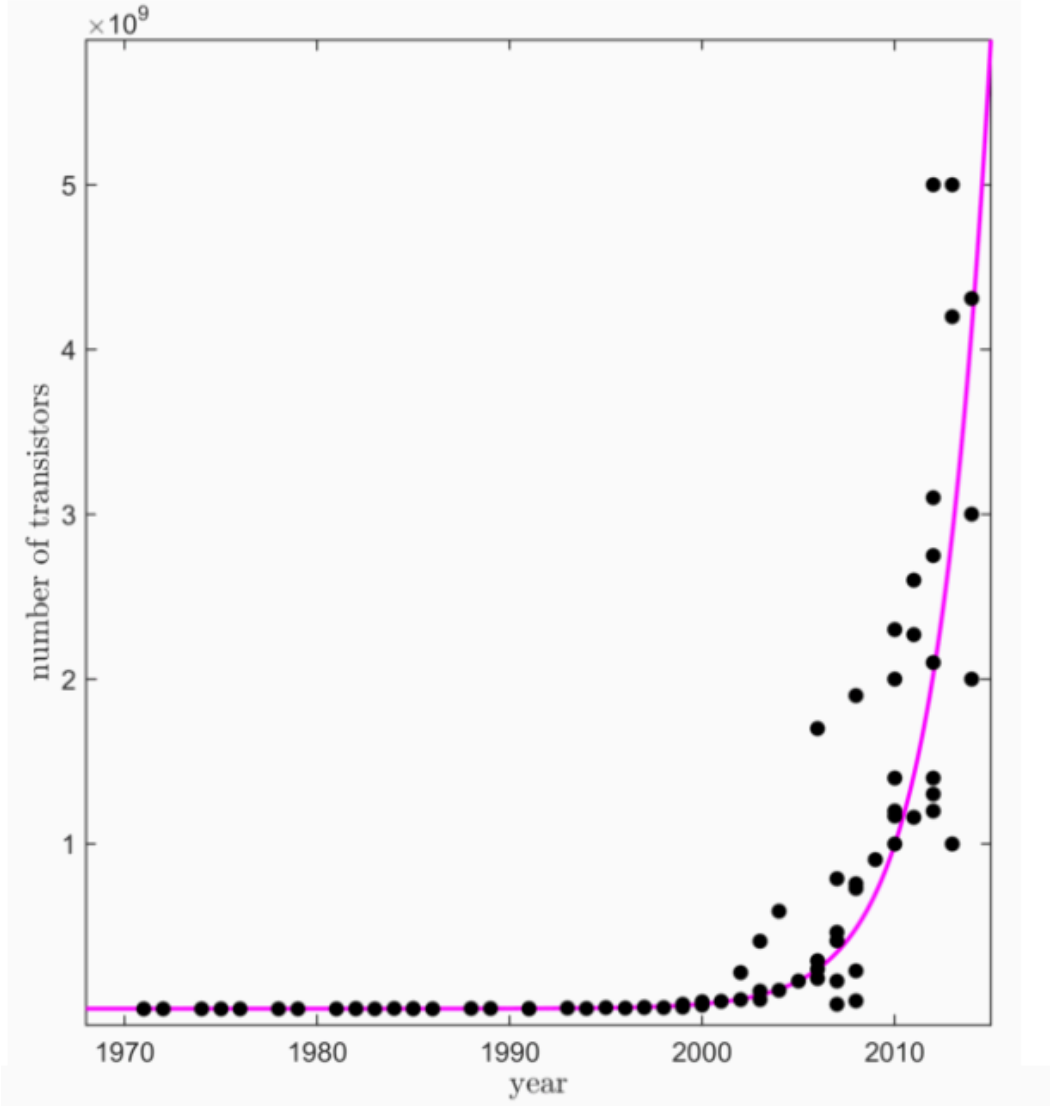


Image taken from [29]

The resurgence of neural networks

The most recent resurgence in neural nets is largely due to

1. Enormous datasets – esp. image, speech, and text
2. Huge advancement in computer speed / lowering prices
3. **Neural nets scale more gracefully than fixed basis kernels**
4. Neural nets naturally permit embedment of formalized knowledge
5. Small collection of critical technical improvements
6. (conjecture) Compositional bases may be superior to fixed ones for representing natural data

kernels: final notes

✓In addition to polynomial and Fourier bases, several other fixed bases can be defined directly via the kernel matrix itself (see e.g., [29])

- e.g., radial basis functions RBF, single layer tanh net

Ⓧ The kernel matrix **H** is of size PxP where P is the size of the dataset

- e.g., if P = 10,000 **H** has 10^8 entries, and we want to use $P \approx$ millions

✓ Classic methods exist [49] and promising research currently underway to ameliorate this issue [47-48]

The resurgence of neural networks

The most recent resurgence in neural nets is largely due to

1. Enormous datasets – esp. image, speech, and text
2. Huge advancement in computer speed / lowering prices
3. Neural nets scale more gracefully than fixed basis kernels
4. **Neural nets naturally permit embedment of formalized knowledge**
5. Small collection of critical technical improvements
6. (conjecture) Compositional bases may be superior to fixed ones for representing natural data

conv nets

- Convolutional networks are a generalization of taking pooled fixed edges as a feature
  - The basic generalization is that instead of choosing a particular set of edge detecting filters, we *learn* them based on a (large) dataset

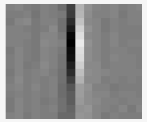
Input image

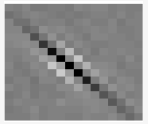
Filters

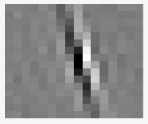
Convolved images

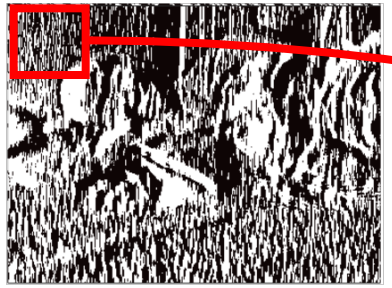
Pooling e.g., histogram or max val

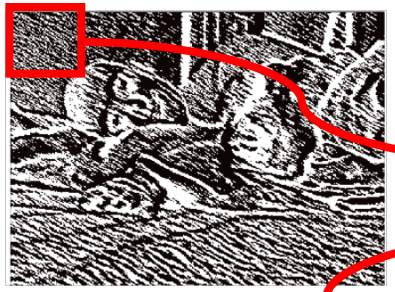
New feature vector

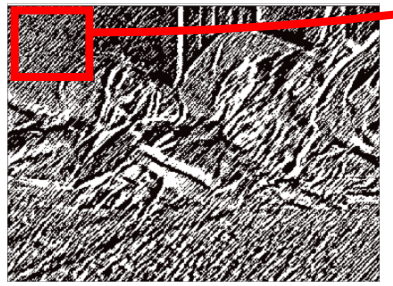


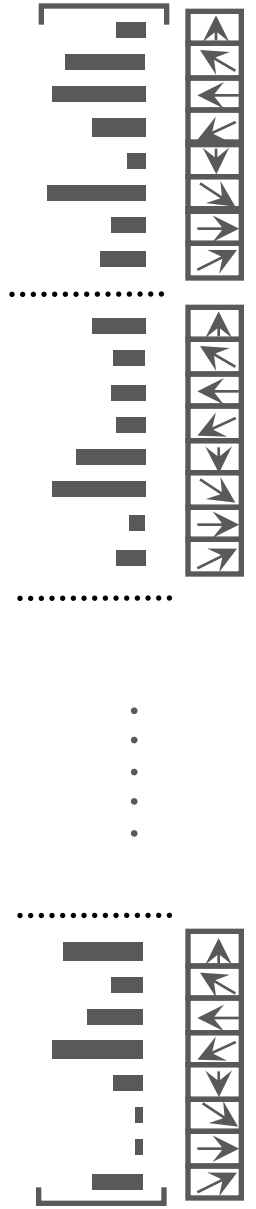












Often called Histogram of Oriented Gradients [22], many similar schemes exist (e.g., SIFT [43]) as well as extensions e.g., spatial pyramid matching [40-41]

conv nets

- Convolutional neural network is a natural generalization of taking pooled edges as a feature
  - The basic generalization is that instead of choosing a particular set of edge detecting filters, we *learn* them based on a (large) dataset
  - Via biological inspiration we stack several layers of convolution/pooling (also biologically inspired) and compose with nonlinear activations

conv nets

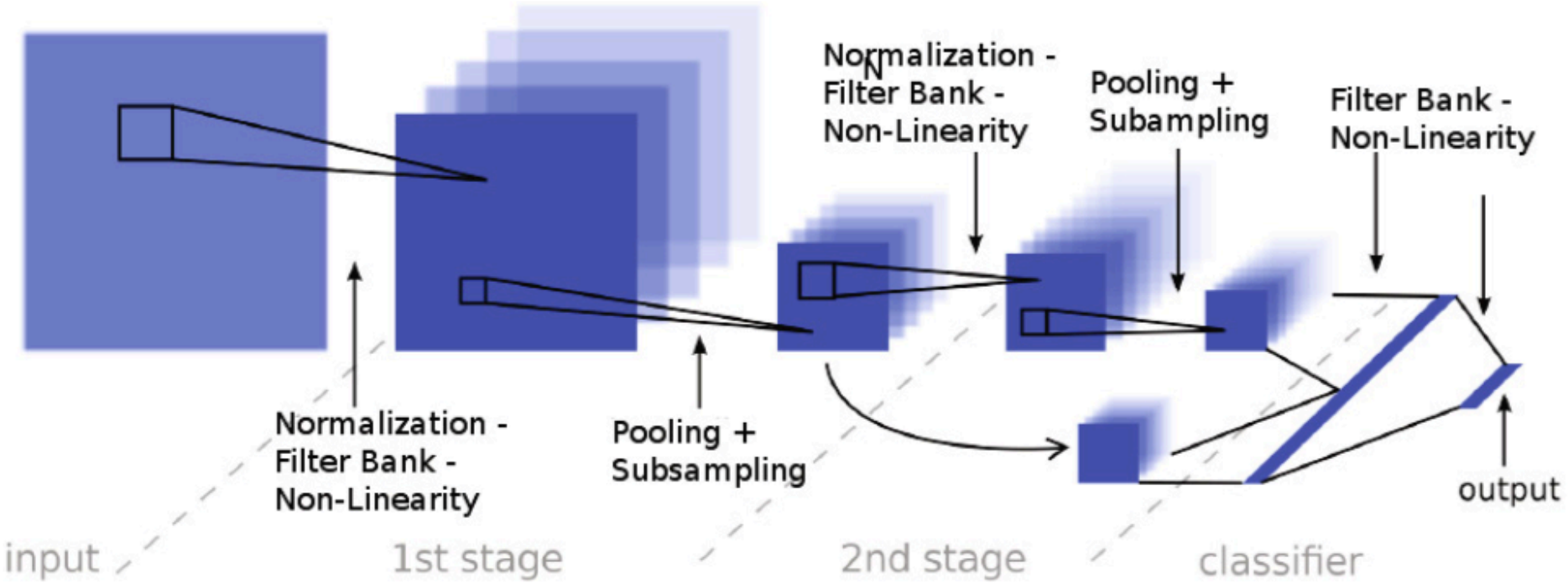


Image taken from [51]

conv nets

- Convolutional neural network is a natural generalization of taking pooled edges as a feature
  - The basic generalization is that instead of choosing a particular set of edge detecting filters, we *learn* them based on a (large) dataset
  - Via biological inspiration we stack several layers of convolution/pooling (also biologically inspired) and compose with nonlinear activations
- If a dataset for a particular problem (e.g., face detection) is large enough then these *learned* features/filters give excellent results
- Many of today's benchmarks on visual and speech related tasks currently held by conv nets (see e.g.,[1], [2], [4], [7], [8], [9], [11])
- Great free software packages (e.g., *Caffe*, *Torch*, *Tensorflow*)
- Great class on how to code up conv nets yourself http://cs231n.stanford.edu/

The resurgence of neural networks

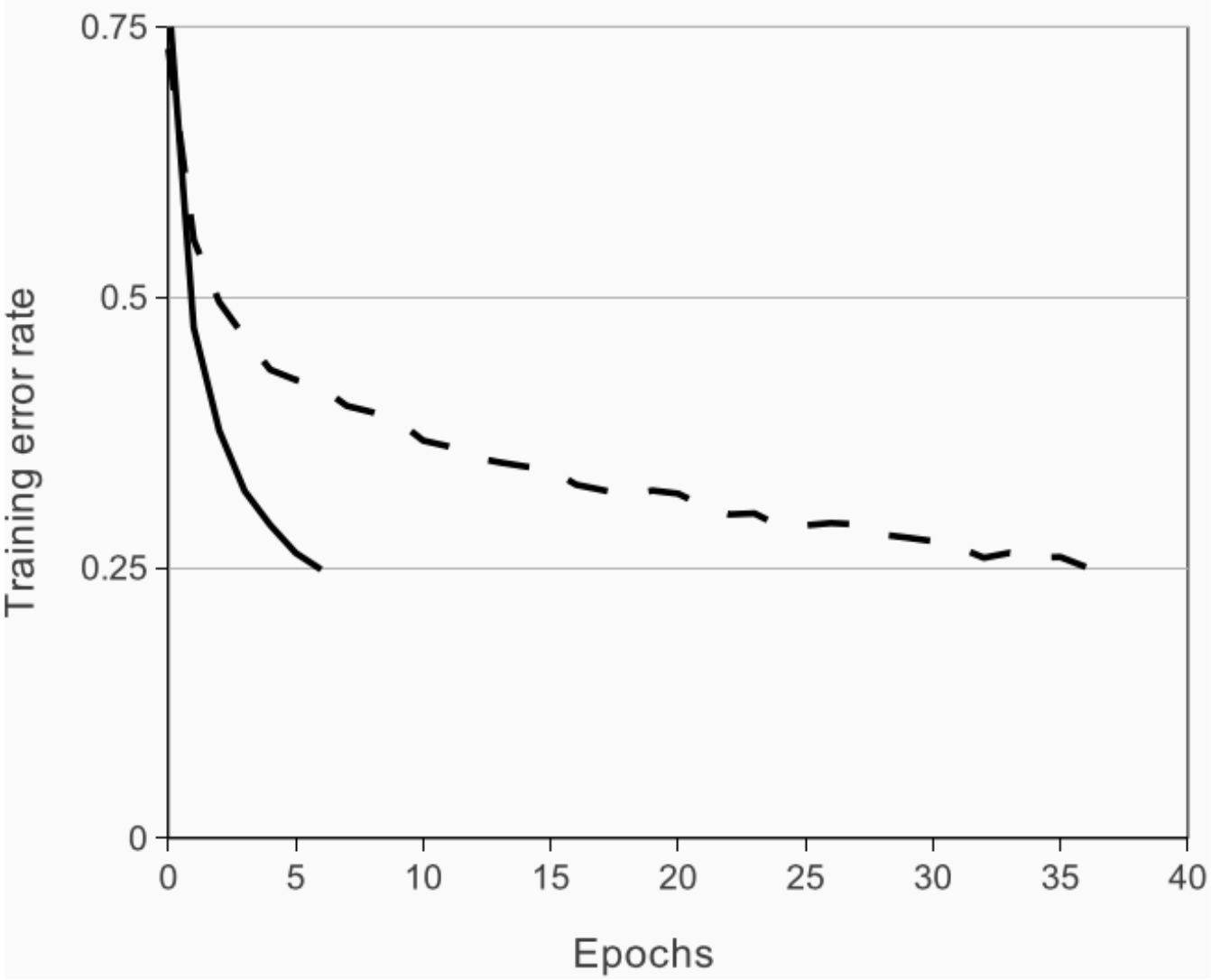
The most recent resurgence in neural nets is largely due to

1. Enormous datasets – esp. image, speech, and text
2. Huge advancement in computer speed / lowering prices
3. Neural nets scale more gracefully than fixed basis kernels
4. Neural nets naturally permit embedment of formalized knowledge
5. **Small collection of critical technical improvements**
6. (conjecture) Compositional bases may be superior to fixed ones for representing natural data

Technical innovations

- Choice of activation function – ReLU (and extensions) instead of sigmoidal seem to work better (see e.g.,[1], [2], [4], [7], [8], [9], [11])
- Initialization – special hand-made initializations determined by studying net architectures assuming specific activation functions (see e.g., [6], [16])
- Initial theoretical / experimental understanding that non-convex structure of deep nets not as bad as one might suspect [30][31]
- Greedy algorithms / unsupervised pre-training – helped bolster the resurgence in interest in neural networks in 2006 (see e.g., [12 – 15])
- Regularization and architecture - in addition to L2 regularization, dropout (see [3], [7]) and maxout networks [10])

A popular graph



The shape of a cost function using adjustable bases can differ greatly depending on the activation function used (i.e., the function you choose to compose)

Image taken from [32]

Why?

- Largely an empiracle finding, but some theoretical work has begun to explain why ReLU activations work better (see e.g., [31])
- Some technical facts
  - ReLU gradients are trivial / involve fewer calculations than sigmoid
  - ReLU gradients only saturate on one end

The resurgence of neural networks

The most recent resurgence in neural nets is largely due to

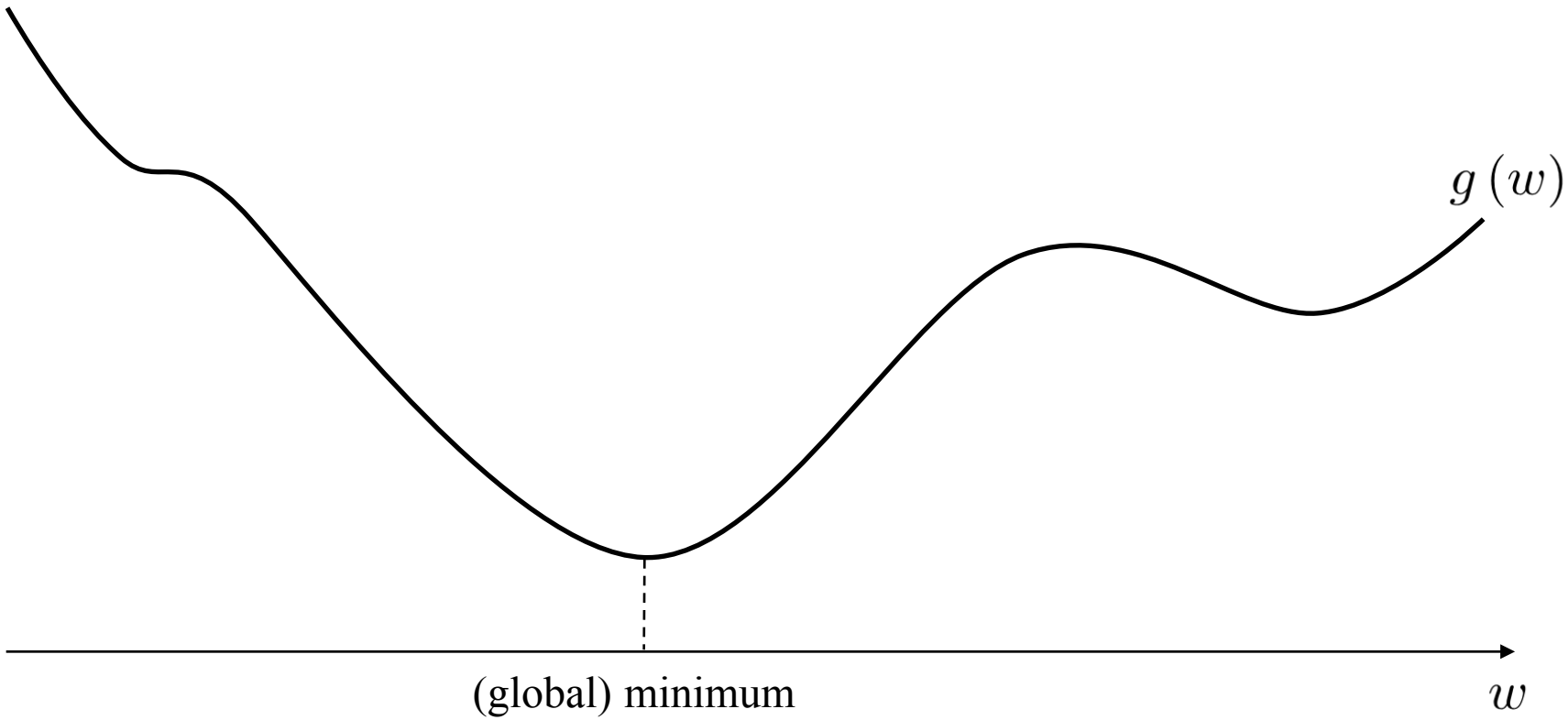
1. Enormous datasets previously unavailable – particularly images/videos, speech, and text – hundreds of millions of data points (compared to thousands or hundreds of thousands)
2. Huge advancement in computer speed / lowering prices for computers
3. Neural nets scale more gracefully than kernels with huge sets of high dimensional data
4. Small collection of critical technical improvements
5. Neural nets naturally permit embedment of formalized knowledge of a phenomenon (e.g., conv nets)
6. Compositional bases may be superior to fixed ones for natural data [33]

very big picture view on numerical optimization

Why learn numerical optimization?

- In virtually all machine learning applications we look to find the (global) minimum of an associated cost function

- This (global) minimum corresponds to the *optimal* parameters/weights for the model at hand

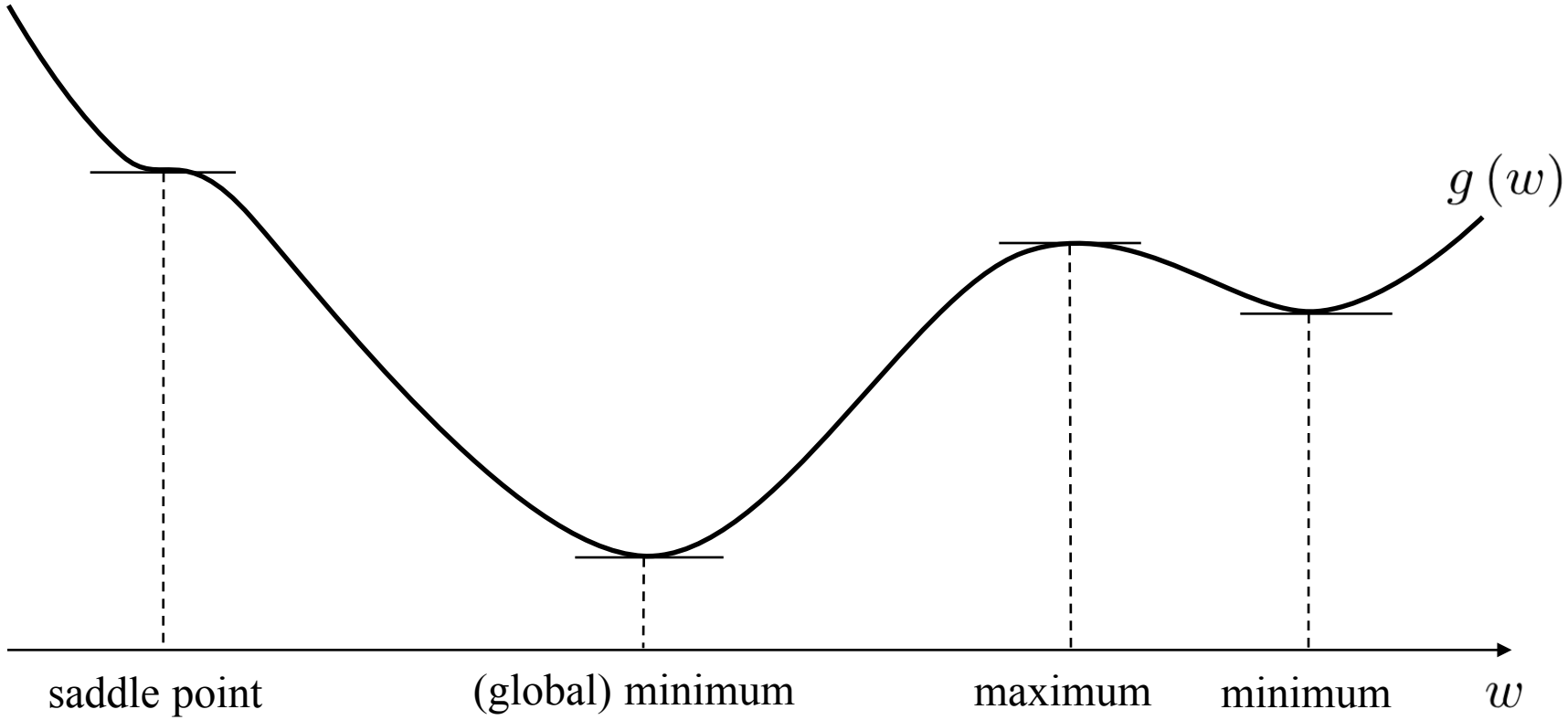


A useful tool from Calculus

- **The first order condition for optimality** gives a nice criterion for finding all *stationary* points:

$$w \text{ is a stationary point of } g \text{ if } g'(w) = 0$$

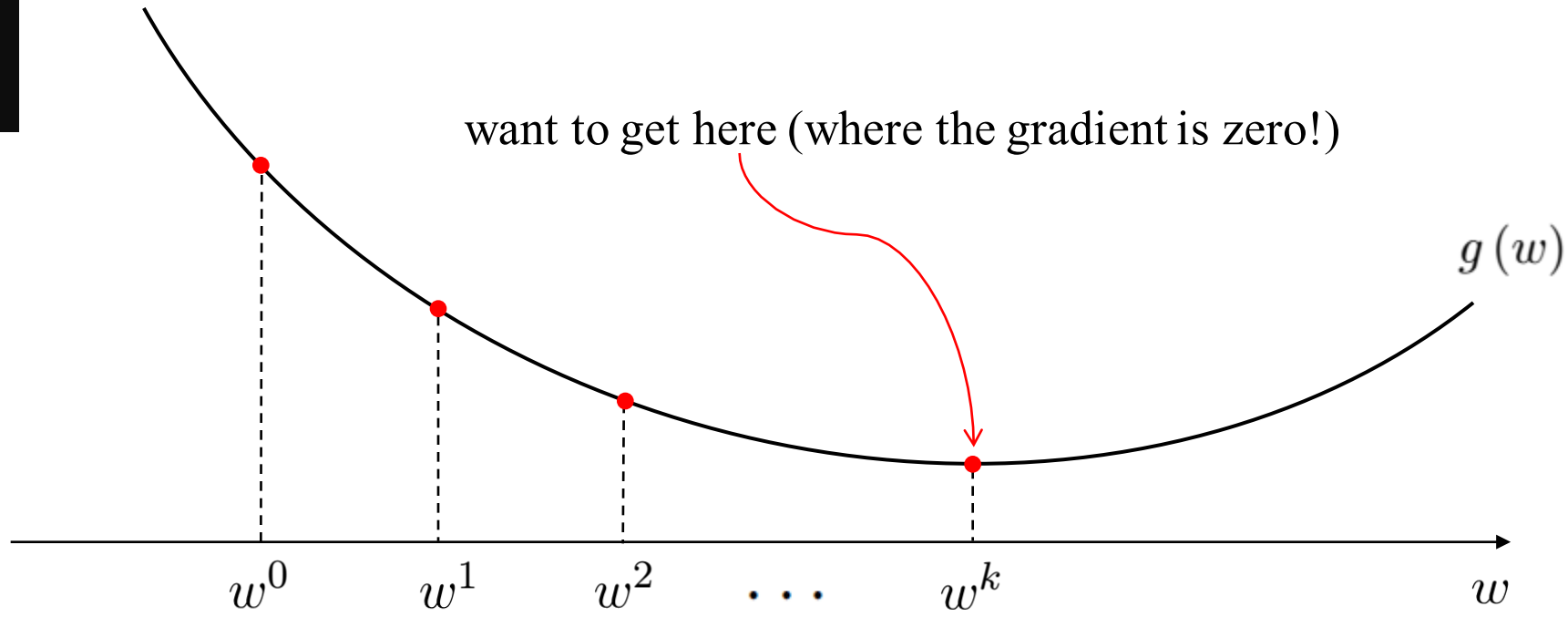
For a general *N*-d input $\mathbf{w}_{N\times 1}$ we have the analogous condition $\nabla g(\mathbf{w}) = \mathbf{0}_{N\times 1}$



Solving the first order system

- $\nabla g(\mathbf{w}) = \mathbf{0}_{N\times 1}$ is a system of N equations $\begin{cases} \frac{\partial}{\partial w_1} g = 0 \\ \frac{\partial}{\partial w_2} g = 0 \\ \vdots \\ \frac{\partial}{\partial w_N} g = 0 \end{cases}$
- Easy to solve when linear (e.g., linear regression)
- But in most cases nonlinear in $\mathbf{w}$ with no closed form solution
- Iterative methods (e.g., gradient descent or Newton's method) are used to *approximately* solve this system

Iterative methods

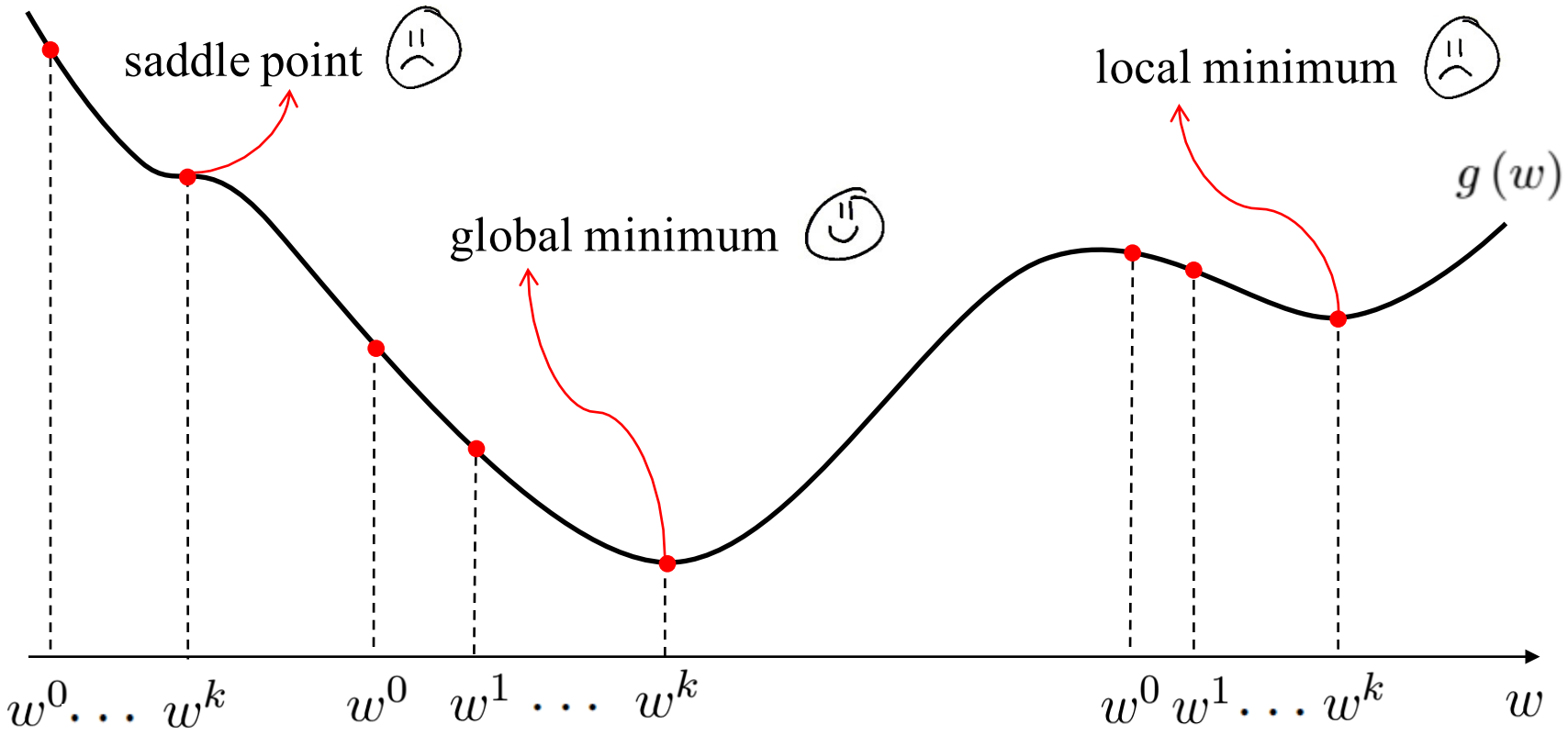


(1) Start the minimization process from some *initial point* $\mathbf{w}^0$.

(2) Take *iterative* steps denoted by $\mathbf{w}^1$, $\mathbf{w}^2$, ..., going "downhill" towards a stationary point of g.

(3) Repeat step (2) until the sequence of points converges to a stationary point of g.

Initial point



(1) Start the minimization process from some *initial point* $\mathbf{w}^0$.

(2) Take *iterative* steps denoted by $\mathbf{w}^1$, $\mathbf{w}^2$, ..., going "downhill" towards a stationary point of g.

(3) Repeat step (2) until the sequence of points converges to a stationary point of g.

Initial point

- With non-convex cost functions it is possible to end up at a saddle point or a local minimum depending on the initialization
- In such cases run the iterative method multiple times with different initializations and take the lowest result
- nonconvex ≠ bad, simply worth being aware of

(1) Start the minimization process from some *initial point* $\mathbf{w}^0$.

(2) Take *iterative* steps denoted by $\mathbf{w}^1$, $\mathbf{w}^2$, ..., going "downhill" towards a stationary point of g.

(3) Repeat step (2) until the sequence of points converges to a stationary point of g.

A primer on backpropogation

The iterative steps

- The iterative steps are taken based on the *Taylor series approximation* of the cost function

- Two popular methods: **gradient descent** (based on 1st order Taylor approximation), and Newton's method (based on 2nd order Taylor approximation)

(1) Start the minimization process from some *initial point* $\mathbf{w}^0$.

(2) Take *iterative* steps denoted by $\mathbf{w}^1$, $\mathbf{w}^2$, ..., going "downhill" towards a stationary point of g.

(3) Repeat step (2) until the sequence of points converges to a stationary point of g.

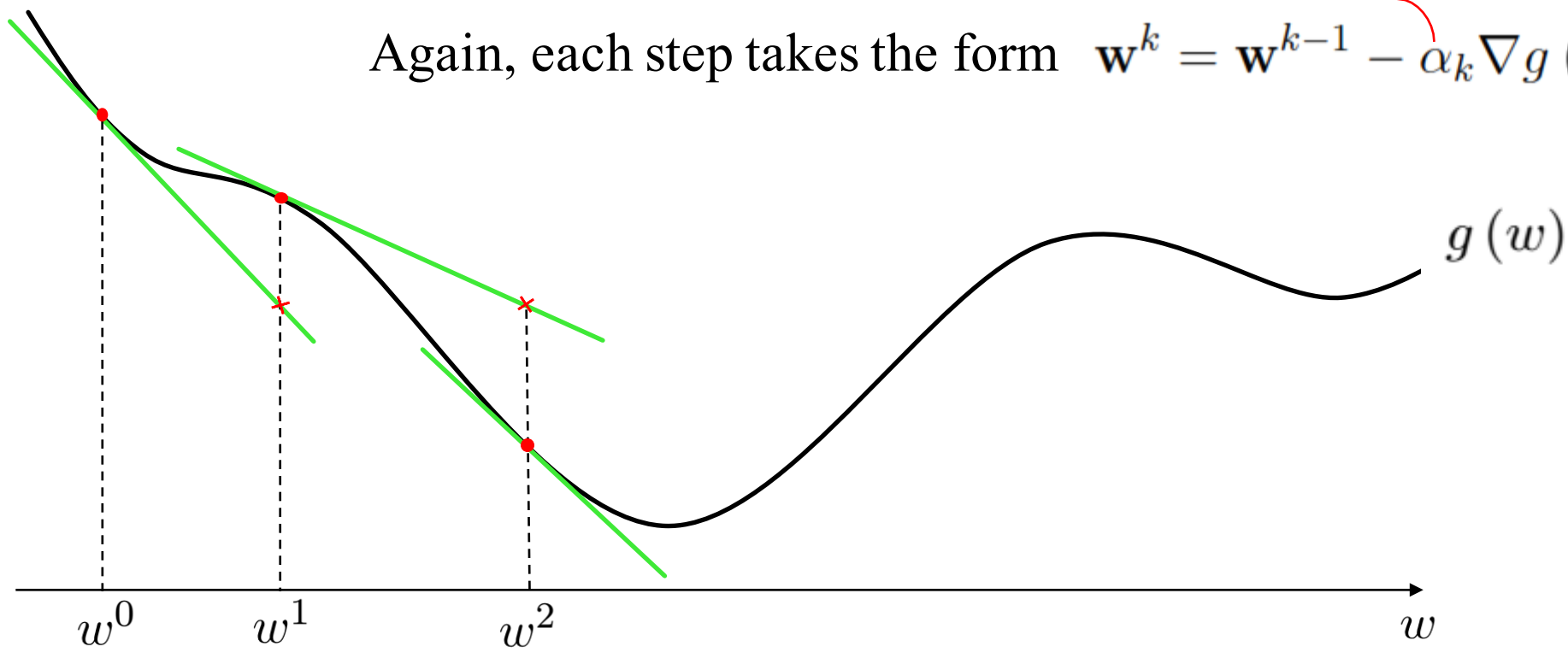
Gradient descent

- Begin at a point $\mathbf{w}^0$
- Travel downward on tangent hyperplane to g at $\mathbf{w}^0$ in the direction of negative gradient
- (Hop back onto the function)
- Repeat until a stationary point is reached

step length (aka learning rate)

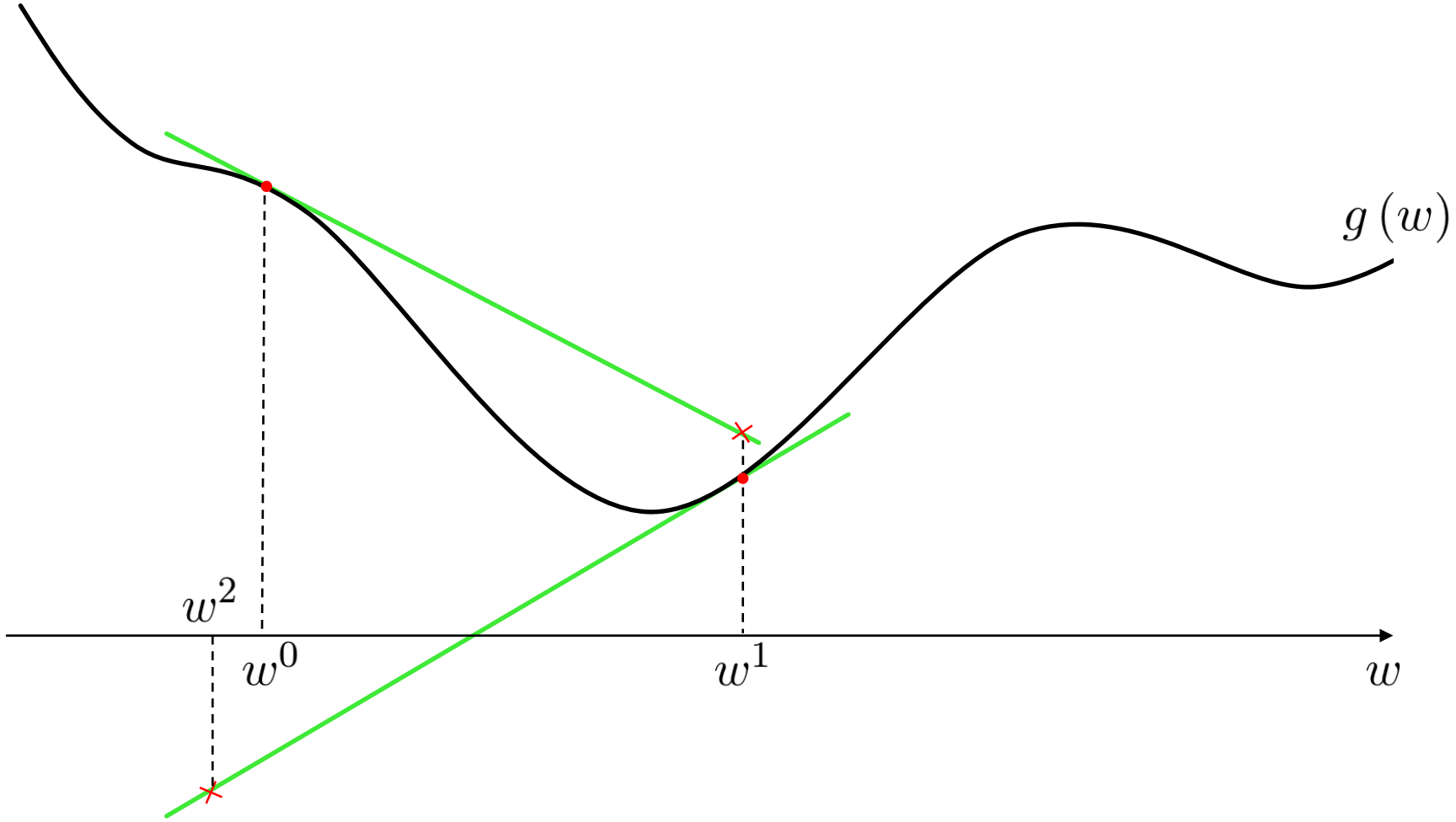
determines how far to travel down the negative gradient direction

Again, each step takes the form $\mathbf{w}^k = \mathbf{w}^{k-1} - \alpha_k \nabla g\left(\mathbf{w}^{k-1}\right)$



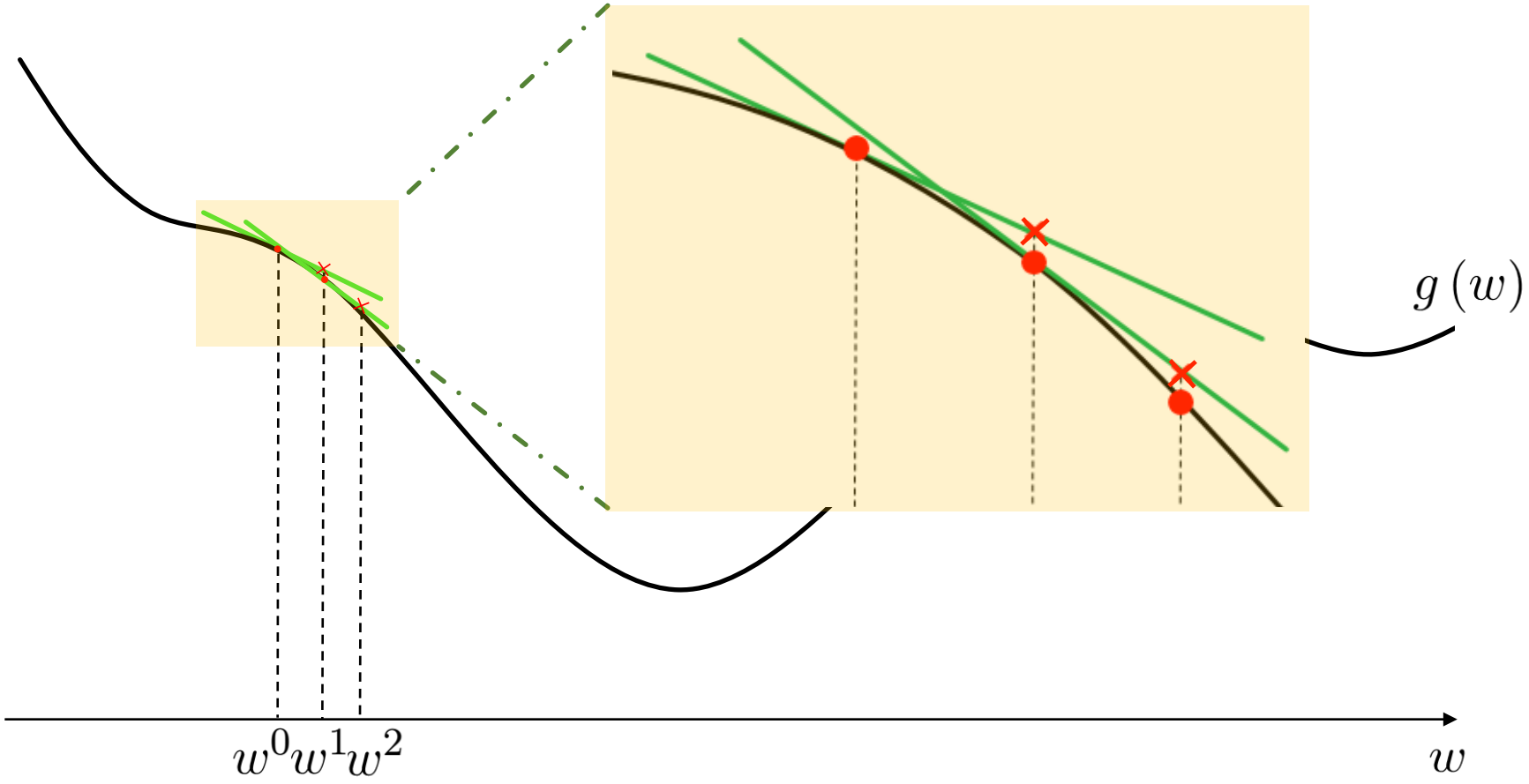
Gradient descent step length

- What happens when the step length is too large?



Gradient descent step length

- What happens when the step length is too small?



Gradient descent step length

- How to select a "good" step length?

1. **Naïve trial and error**: Start with a fixed large step size for all iterations, if it does not produce decreasing value of g in then decrease and try again!

2. **Does g have bounded curvature?** If so, a fixed step length can be calculated with provable convergence (details ahead)

3. **Adaptive step length selection:** Use an adaptive step length procedure that essentially does "trial and error" at each step (see [29] for all the details)

Pseudo-code

Algorithm Gradient descent (with fixed step length)

Input: differentiable function g, fixed step length α, and initial point $\mathbf{w}^0$

$k = 1$

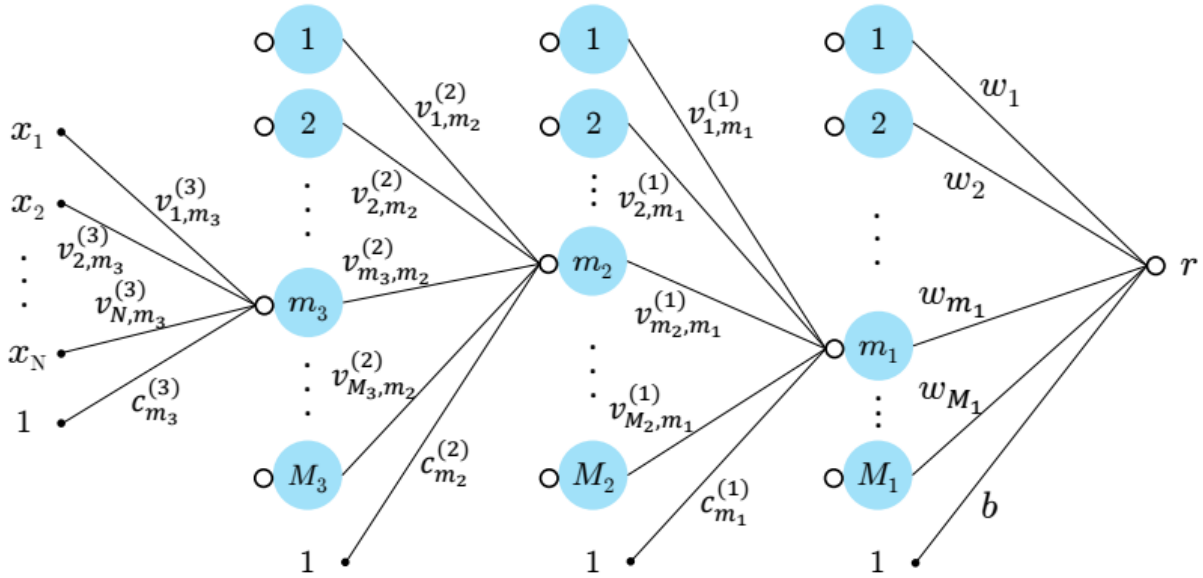
Repeat until stopping condition is met:

$$\mathbf{w}^k = \mathbf{w}^{k-1} - \alpha \nabla g\left(\mathbf{w}^{k-1}\right)$$

$$k \leftarrow k + 1$$

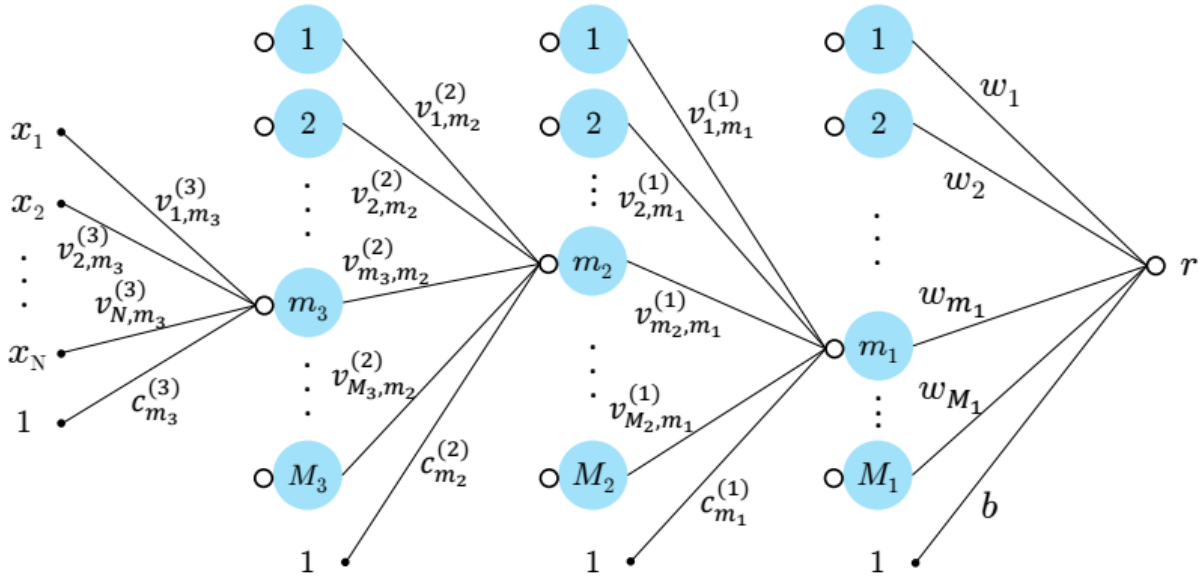
Gradient descent

- When applied to a model in which neural networks are used as a feature basis gradient desent is known as the **backpropagation algorithm**
- This is due to an interpretation of neural networks as a graph, and because one must use the **chain rule** extensively in taking a neural net derivative
- These partials are computed from outer → inner layers, and the chain rule always carries the error (or objective value) with it 'backwards'



Gradient descent

- Easier to think about as simply ‘gradient descent’ where, due to compositional nature of neural net, the chain rule must be heavily used
- Often in practice ‘backpropagation algorithm’ also means ***stochastic gradient descent***



Stochastic gradient descent

Standard gradient descent

- For datasets containing hundreds of thousands or millions of data points, taking each gradient step requires evaluating the gradient over the entire dataset

$$\tilde{\mathbf{w}}^k = \tilde{\mathbf{w}}^{k-1} - \alpha_k \nabla g\left(\tilde{\mathbf{w}}^{k-1}\right)$$

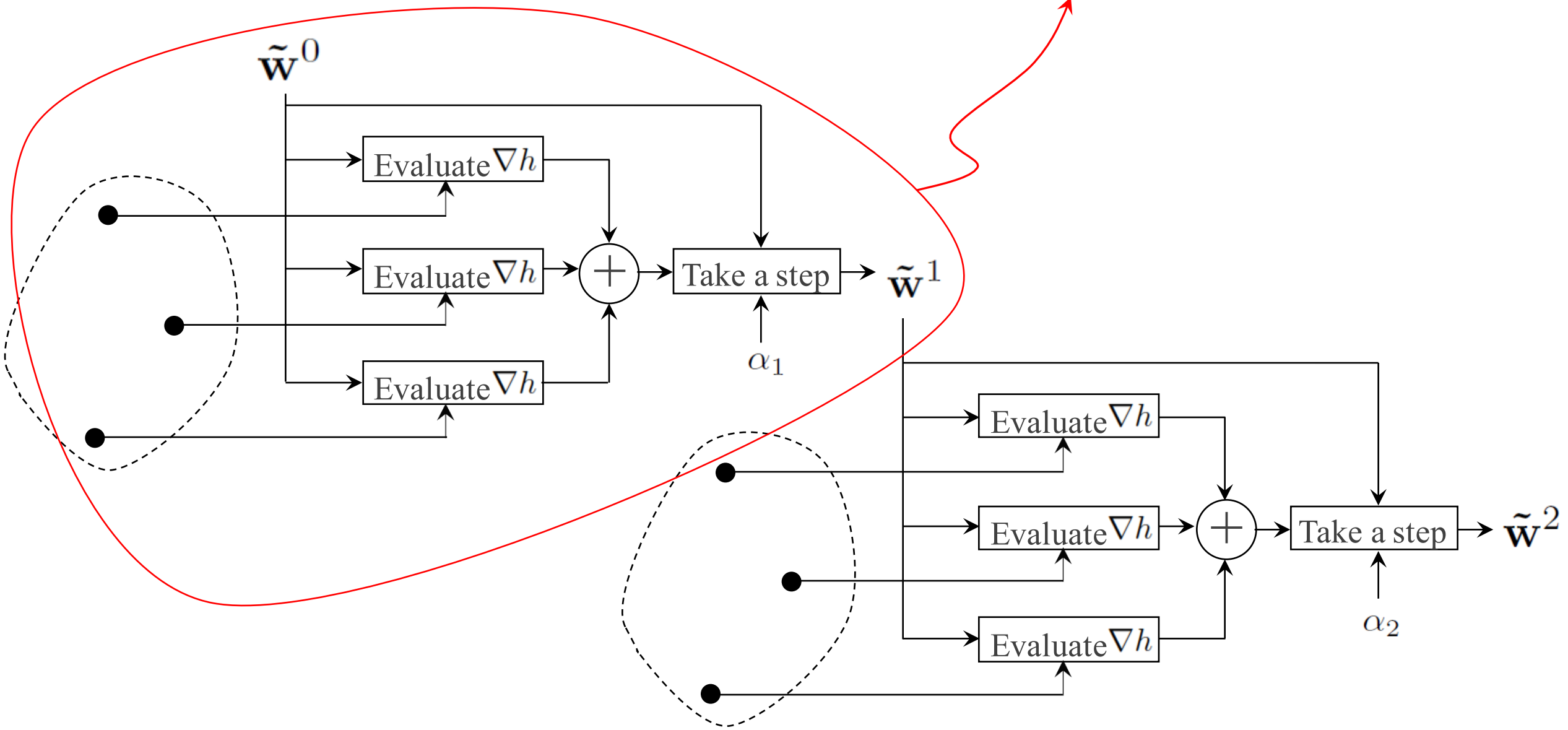
- **Observation:** many machine learning cost functions can be written as a sum of individual costs over each data point

$$g\left(\tilde{\mathbf{w}}\right) = \sum_{p=1}^{P} h\left(\tilde{\mathbf{w}}, \tilde{\mathbf{x}}_p\right)$$

- Therefore the gradient itself is decomposable, and the gradient descent step can be written as

$$\tilde{\mathbf{w}}^k = \tilde{\mathbf{w}}^{k-1} - \alpha_k \sum_{p=1}^{P} \nabla h\left(\tilde{\mathbf{w}}^{k-1}, \tilde{\mathbf{x}}_p\right)$$

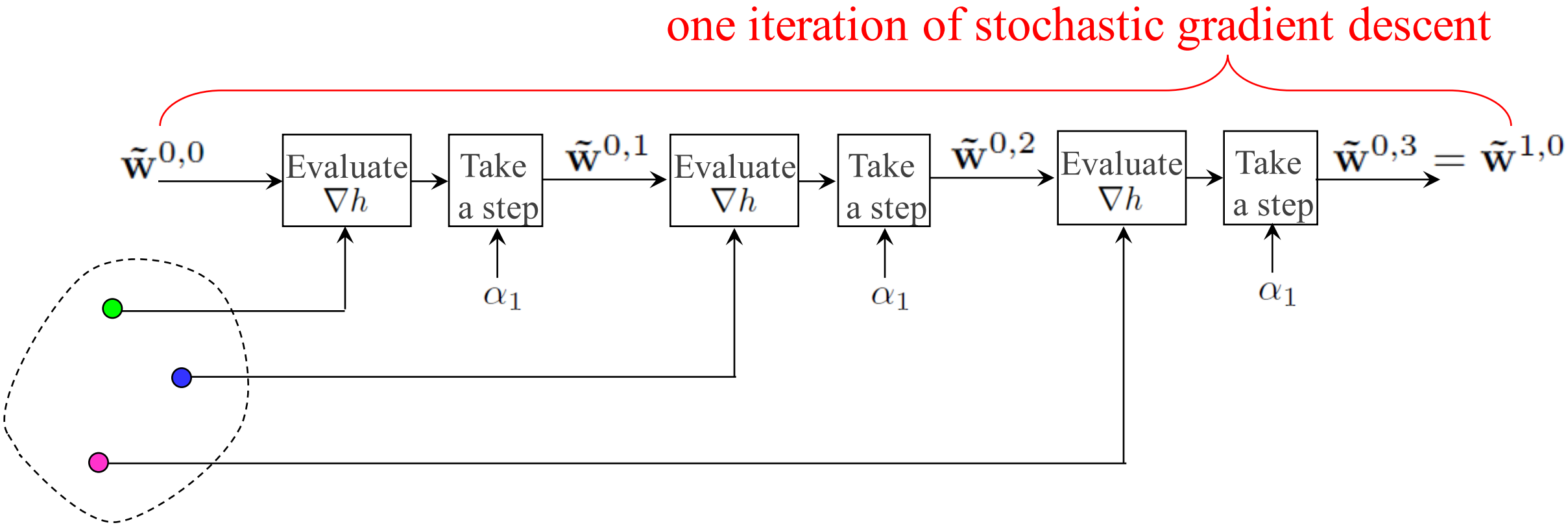
Standard gradient descent



Stochastic gradient descent

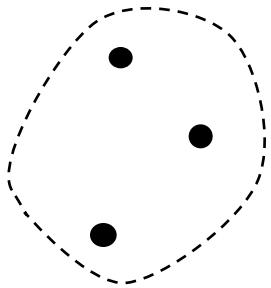
- With stochastic gradient descent, instead of taking a single gradient step over the entire dataset we take a sequence of P shorter steps in each data point individually

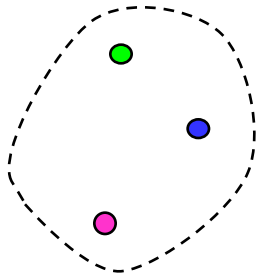
$$\tilde{\mathbf{w}}^{k,p} = \tilde{\mathbf{w}}^{k,p-1} - \alpha_k \nabla h\left(\tilde{\mathbf{w}}^{k,p-1}, \tilde{\mathbf{x}}_p\right) \qquad p = 1 \ldots P$$

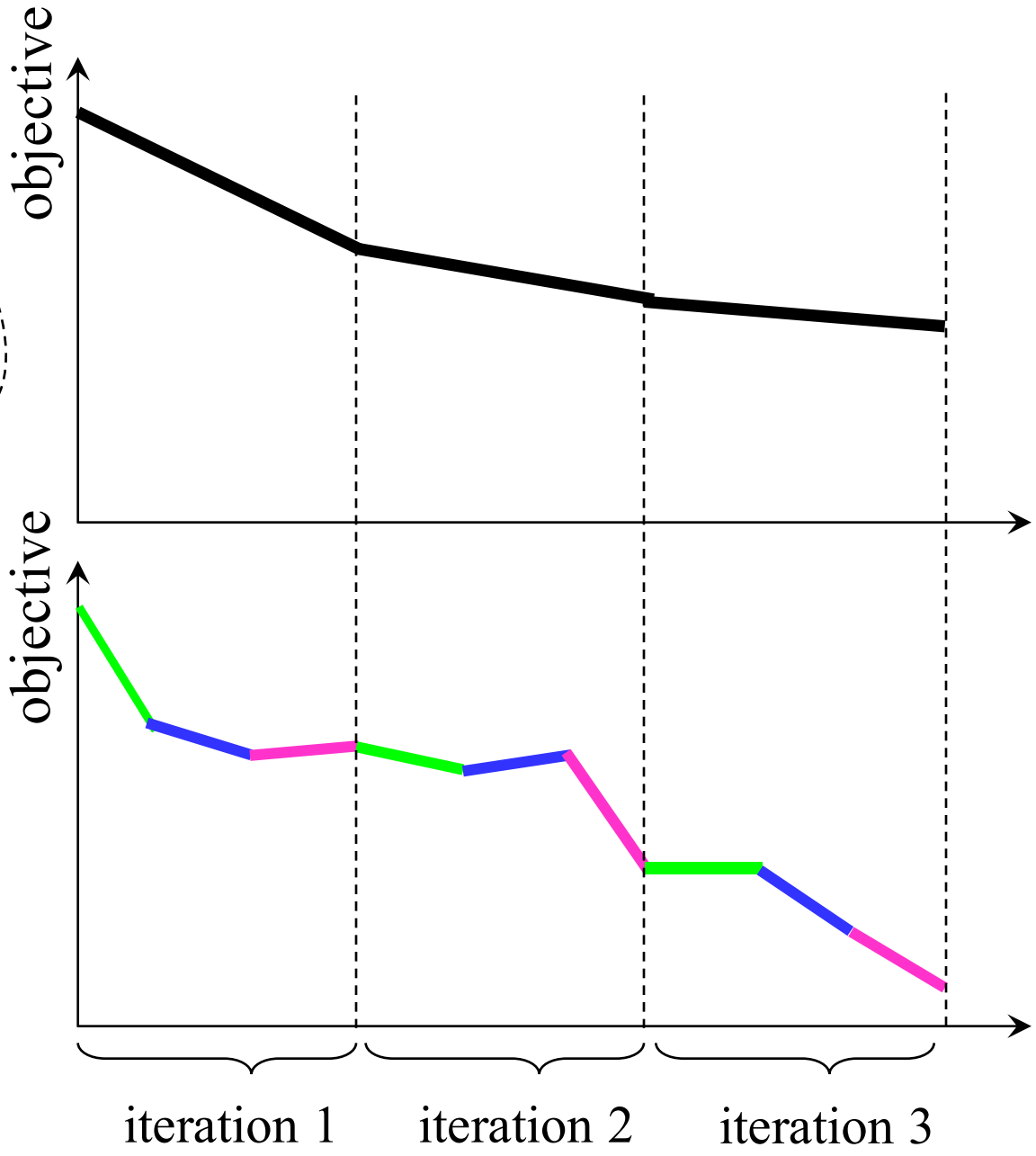


Standard vs. stochastic gradient descent

When far from a point of convergence, substantial empirical evidence [7,10] suggests that the stochastic method tends to converge to progress much faster towards a solution compared to standard gradient descent





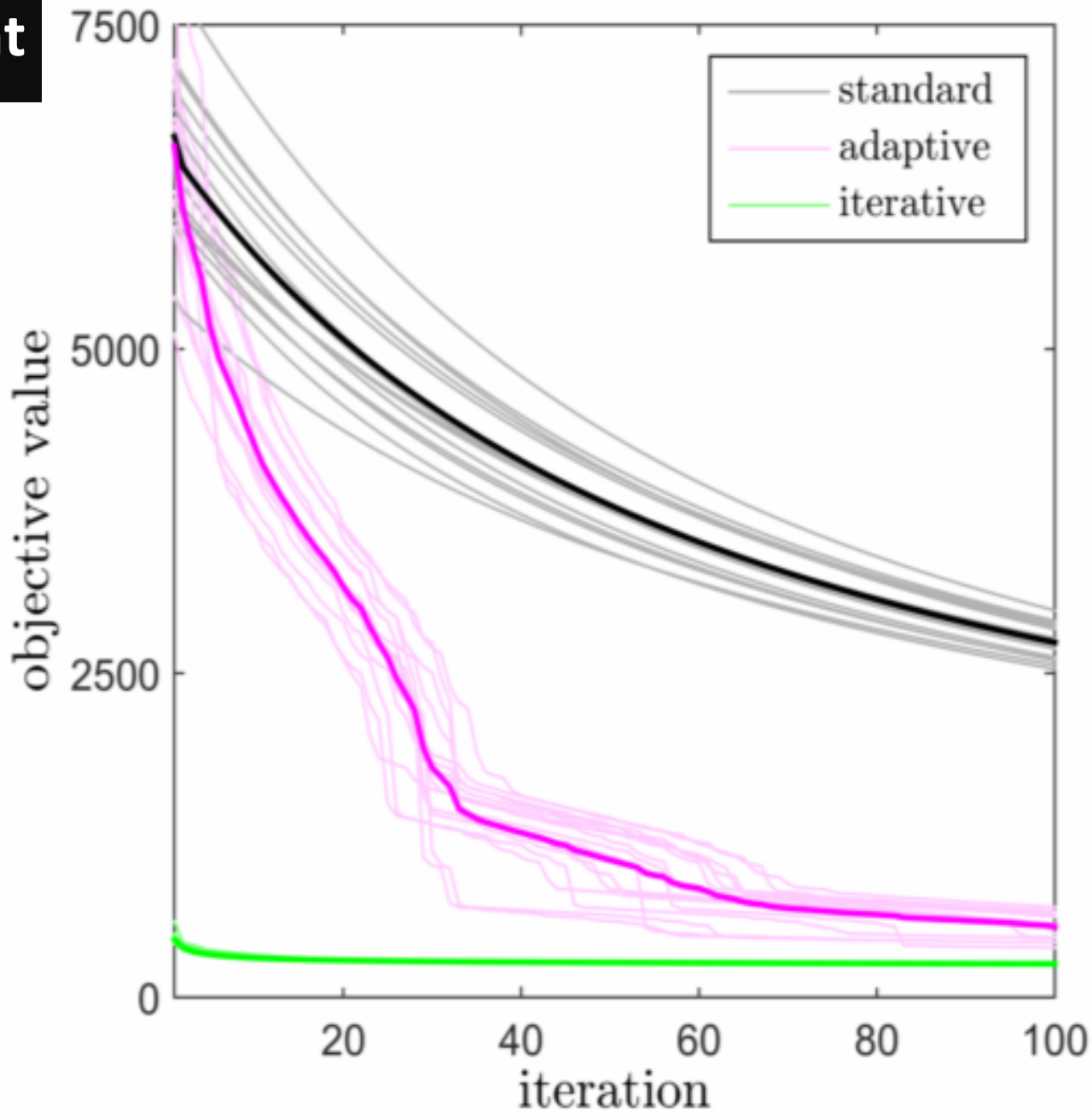


[7] Dimitri P Bertsekas. Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. *Optimization for Machine Learning*, 2010:1–38, 2011.

[10] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

Standard vs. stochastic gradient descent

- **Application:** face detection
- **Size of data:** $P = 10{,}000$
- **Cost:** Softmax/logistic regression
- **Number of runs:** 15
  (with shared random initializations)



Stochastic gradient descent: implementation

- Often in practice data points in real datasets are sorted. It has been found that randomizing the order of the data prior to running the algorithm improve its convergence

- The following two conditions, if met, guarantee the stochastic gradient descent algorithm to converge to a stationary point

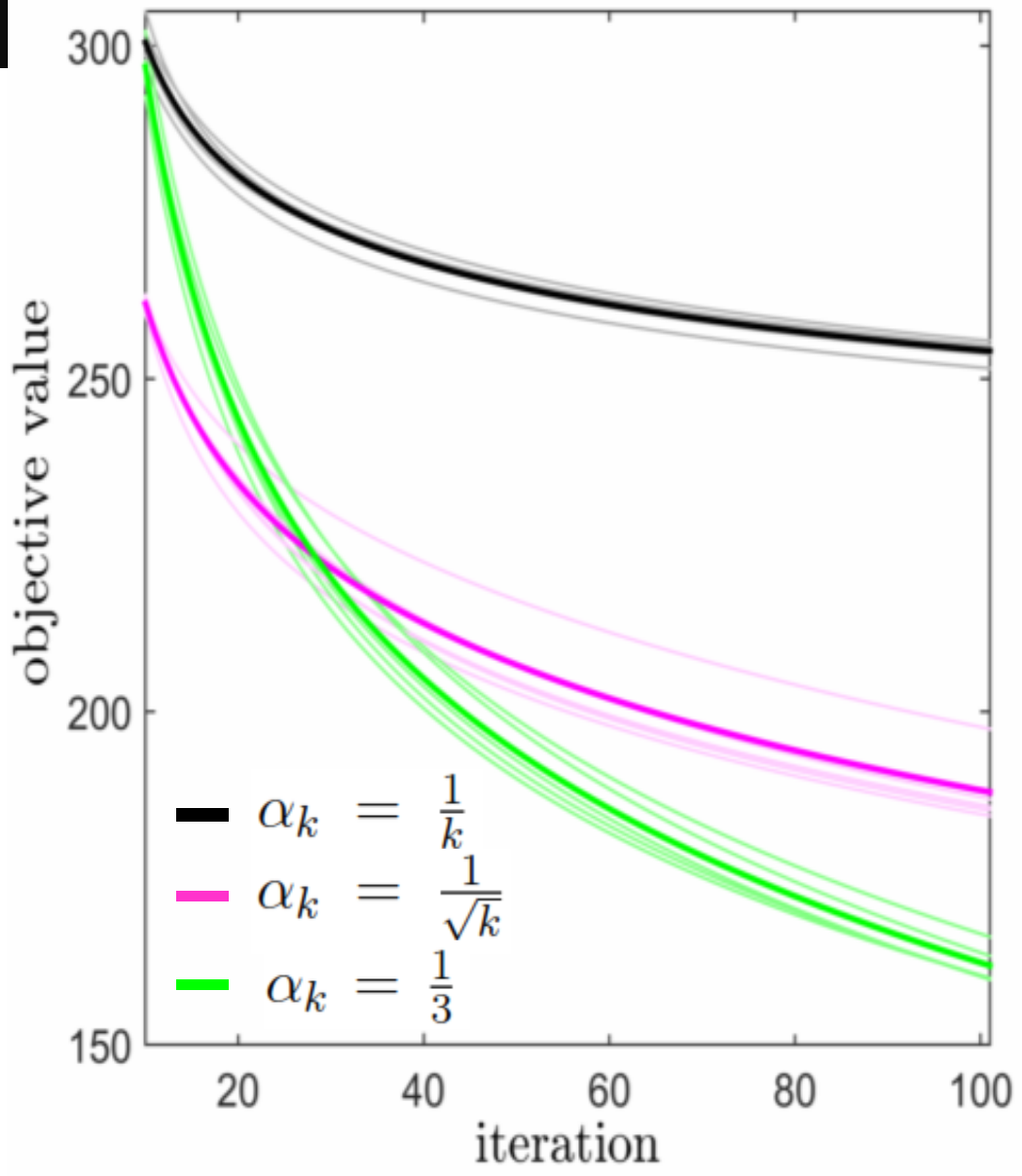
(1) The step size must diminish as the number of iterations increases: $\alpha_k \longrightarrow 0$ as $k \longrightarrow \infty$

(2) The sum of the step sizes is not finite: i.e., $\sum_{k=1}^{\infty} \alpha_k = \infty$

- For instance: $\alpha_k = \frac{1}{k}$ or $\alpha_k = \frac{1}{\sqrt{k}}$

Stochastic gradient descent: implementation

- **Application:** face detection
- **Size of data:** $P = 10{,}000$
- **Cost:** Softmax/logistic regression
- **Number of runs:** 5
  (with shared random initializations)



Summary

Summary

The most recent resurgence in neural nets is largely due to

1. Enormous datasets – esp. image, speech, and text
2. Huge advancement in computer speed / lowering prices
3. Neural nets scale more gracefully than fixed basis kernels
4. Neural nets naturally permit embedment of formalized knowledge
5. Small collection of critical technical improvements made by a cadre of persistent researchers

“Backpropagation = gradient descent (or stochastic gradient descent)

Data-Knowledge spectrum

- ‘By hand’ and ‘learned’ define two ends of a spectrum for feature design – not two isolated approaches
  - Convolutional neural networks are a prime example of something in between a purely ‘knowledge-driven’ and purely ‘data-driven’ approach to feature design (more on this in part III of talk!)

- No silver bullet here: both “ends of the spectrum” have limitations
  - **By hand:** it is often difficult to understand a phenomenon well enough to create meaningful features that generalize well
  - **Learning:** it can be difficult to collect large enough amounts of quality data for particular tasks to make learning features effective

Data-Knowledge spectrum

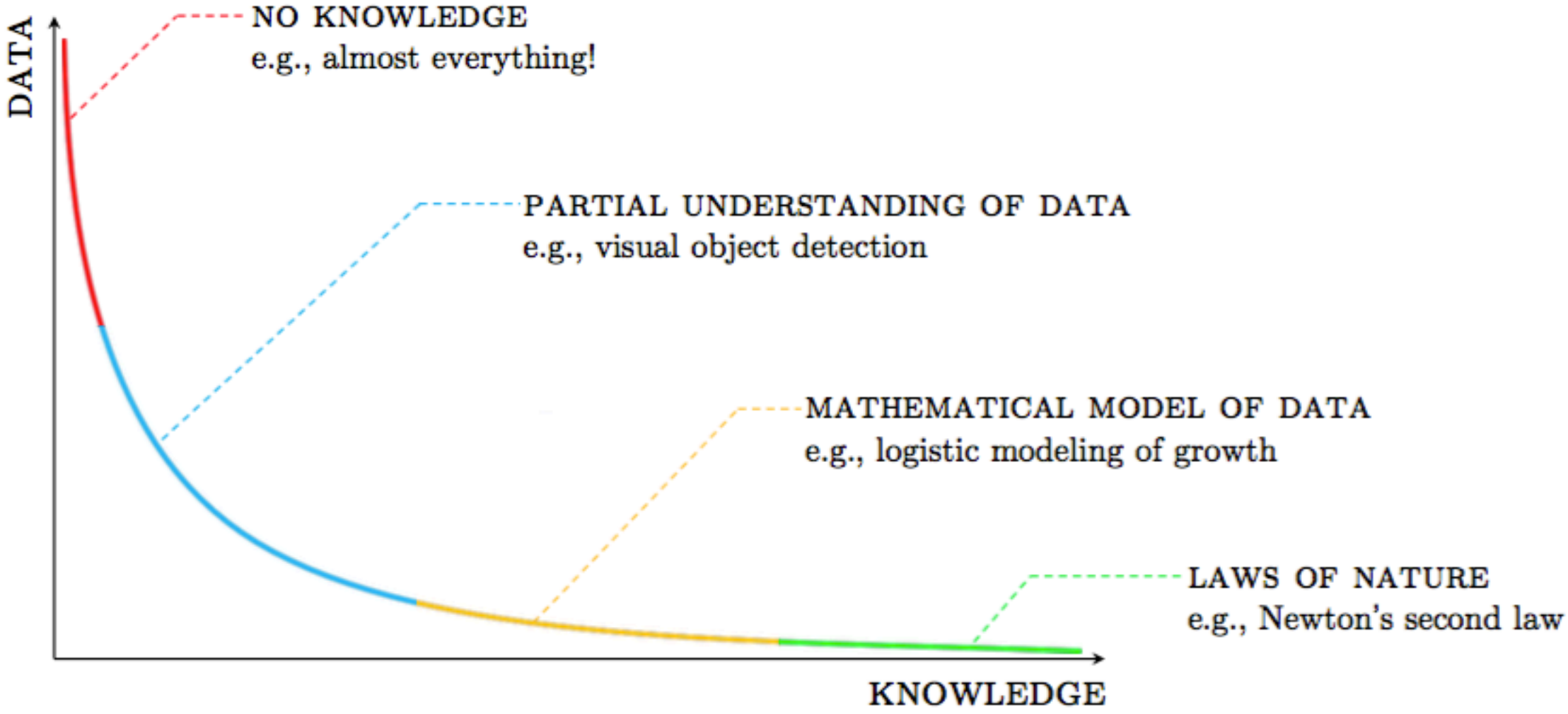


Image taken from [29]

References

[1] Nair, Vinod, and Geoffrey E. Hinton. "Rectified linear units improve restricted boltzmann machines." *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*. 2010.

[2] Glorot, Xavier, Antoine Bordes, and Yoshua Bengio. "Deep sparse rectifier neural networks." *International Conference on Artificial Intelligence and Statistics*. 2011.

[3] Hinton, Geoffrey E., et al. "Improving neural networks by preventing co-adaptation of feature detectors." *arXiv preprint arXiv:1207.0580* (2012).

[4] Zeiler, Matthew D., et al. "On rectified linear units for speech processing."*Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013.

[5] Martens, James. "Deep learning via Hessian-free optimization." *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*. 2010.

[6] Glorot, Xavier, and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." *International conference on artificial intelligence and statistics*. 2010.

References

[7] Dahl, George E., Tara N. Sainath, and Geoffrey E. Hinton. "Improving deep neural networks for LVCSR using rectified linear units and dropout."*Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013.

[8] Zeiler, Matthew D., et al. "On rectified linear units for speech processing."*Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013.

[9] He, Kaiming, et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." *arXiv preprint arXiv:1502.01852*(2015).

[10] Goodfellow, Ian J., et al. "Maxout networks." *arXiv preprint arXiv:1302.4389*(2013).

[11] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." *Nature* 521.7553 (2015): 436-444.

References

[12] Hinton, Osindero & Teh “A Fast Learning Algorithm for Deep Belief Nets”, Neural Computation, 2006

[13] Bengio, Lamblin, Popovici, Larochelle “Greedy Layer-Wise Training of Deep Networks”, NIPS’2006

[14] Ranzato, Poultney, Chopra, LeCun “Efficient Learning of Sparse Representations with an Energy-Based Model”, NIPS’2006

[15] Bengio, Yoshua, et al. "Greedy layer-wise training of deep networks."*Advances in neural information processing systems* 19 (2007): 153.

[16] Sutskever, Ilya, et al. "On the importance of initialization and momentum in deep learning." *Proceedings of the 30th international conference on machine learning (ICML-13)*. 2013.

References

[16] Choromanska, Anna, et al. "The loss surface of multilayer networks." *arXiv preprint arXiv:1412.0233* (2014).

[17] Goodfellow, Ian J., and Oriol Vinyals. "Qualitatively characterizing neural network optimization problems." *arXiv preprint arXiv:1412.6544* (2014).

[18] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556*(2014).

[19] Bengio, Yoshua, Ian Goodfellow, and Aaron Courville. "Deep learning." *An MIT Press book in preparation. Draft chapters available at http://www. iro. umontreal. ca/~ bengioy/dlbook* (2014).

[20] Gomes, Lee. "Machine-learning maestro michael jordan on the delusions of big data and other huge engineering efforts." *IEEE Spectrum, Oct* 20 (2014).

References

[21] Yann LeCun post to Google+ on 28 Oct 2013
https://plus.google.com/+YannLeCunPhD/posts/Qwj9EEkUJXY

[22] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE, 2005.

[23] Benenson, Rodrigo, et al. "Ten years of pedestrian detection, what have we learned?." *Computer Vision-ECCV 2014 Workshops*. Springer International Publishing, 2014.

[24] Viola, Paul, and Michael J. Jones. "Robust real-time face detection."*International journal of computer vision* 57.2 (2004): 137-154.

[25] Taigman, Yaniv, et al. "Deepface: Closing the gap to human-level performance in face verification." *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014.

References

[26] Paul Sastrasinh CS 143 Project 4 webpage accessed on 1/10/16 http://cs.brown.edu/courses/cs143/2011/results/proj4/psastras/

[27] Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. "Thumbs up?: sentiment classification using machine learning techniques." *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Association for Computational Linguistics, 2002.

[28] Maas, Andrew L., et al. "Learning word vectors for sentiment analysis."*Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011.

[29] Watt, Jeremy, Borhani, Reza, Katsaggelos, Aggelos “Machine Learning Refined” ." *A Cambridge University Press book in preparation. Draft chapters available at www.mlrefined.com*

References

[30] Goodfellow, Ian J., and Oriol Vinyals. "Qualitatively characterizing neural network optimization problems." *arXiv preprint arXiv:1412.6544* (2014).

[31] Choromanska, Anna, et al. "The loss surface of multilayer networks." *arXiv preprint arXiv:1412.0233* (2014).

[32] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems*. 2012.

[33] Bengio, Yoshua, and Yann LeCun. "Scaling learning algorithms towards AI."*Large-scale kernel machines* 34.5 (2007).

[34] Hubel, David H., and Torsten N. Wiesel. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex." *The Journal of physiology* 160.1 (1962): 106.

References

[35] Hyvärinen, Aapo, Jarmo Hurri, and Patrick O. Hoyer. *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision*. Vol. 39. Springer Science & Business Media, 2009.

[36] Marčelja, S. "Mathematical description of the responses of simple cortical cells*." *JOSA* 70.11 (1980): 1297-1300.

[37] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013).

[38] Csurka, Gabriella, et al. "Visual categorization with bags of keypoints."*Workshop on statistical learning in computer vision, ECCV*. Vol. 1. No. 1-22. 2004.

References

[39] Zhang, Jianguo, et al. "Local features and kernels for classification of texture and object categories: A comprehensive study." *International journal of computer vision* 73.2 (2007): 213-238.

[40] Lazebnik, Svetlana, Cordelia Schmid, and Jean Ponce. "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories."*Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. Vol. 2. IEEE, 2006.

[41] Yang, Jianchao, et al. "Linear spatial pyramid matching using sparse coding for image classification." *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009.

[42] Tang, Yichuan. "Deep learning using linear support vector machines." *arXiv preprint arXiv:1306.0239* (2013).

References

[43] Lowe, David G. "Distinctive image features from scale-invariant keypoints."*International journal of computer vision* 60.2 (2004): 91-110.

[44] Bottou, Léon. "Stochastic gradient descent tricks." *Neural Networks: Tricks of the Trade*. Springer Berlin Heidelberg, 2012. 421-436.

[45] Hinton, Geoffrey, et al. "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups." *Signal Processing Magazine, IEEE* 29.6 (2012): 82-97.

[46] Ford, Henry. *My life and work*. Cosimo, Inc., 2007.

References

[47] Rahimi, Ali, and Benjamin Recht. "Random features for large-scale kernel machines." *Advances in neural information processing systems*. 2007.

[48] Pennington, Jeffrey, Felix Yu, and Sanjiv Kumar. "Spherical random features for polynomial kernels." *Advances in Neural Information Processing Systems*. 2015.

[49] Schölkopf, Bernhard, and Alexander J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[50] Oliva, Aude, and Antonio Torralba. "Modeling the shape of the scene: A holistic representation of the spatial envelope." *International journal of computer vision* 42.3 (2001): 145-175.

[51] LeCun, Yann. "Learning invariant feature hierarchies." *Computer vision–ECCV 2012. Workshops and demonstrations*. Springer Berlin Heidelberg, 2012.

[52] Olshausen, Bruno A., and David J. Field. "Sparse coding with an overcomplete basis set: A strategy employed by V1?." *Vision research* 37.23 (1997): 3311-3325.