Yet More Elemental Learning Theory

Alexander G. Ororbia II

Introduction to Machine Learning

CSCI-335

2/18/2026

The Fundamental Principle of Statistical Learning

- When sample size is small → model should be simple
  - We must deliberately oversimplify our models!
  - Can use the VC/PAC to guide us
    - Occam's Razor = parsimony, choose "the simplest one"

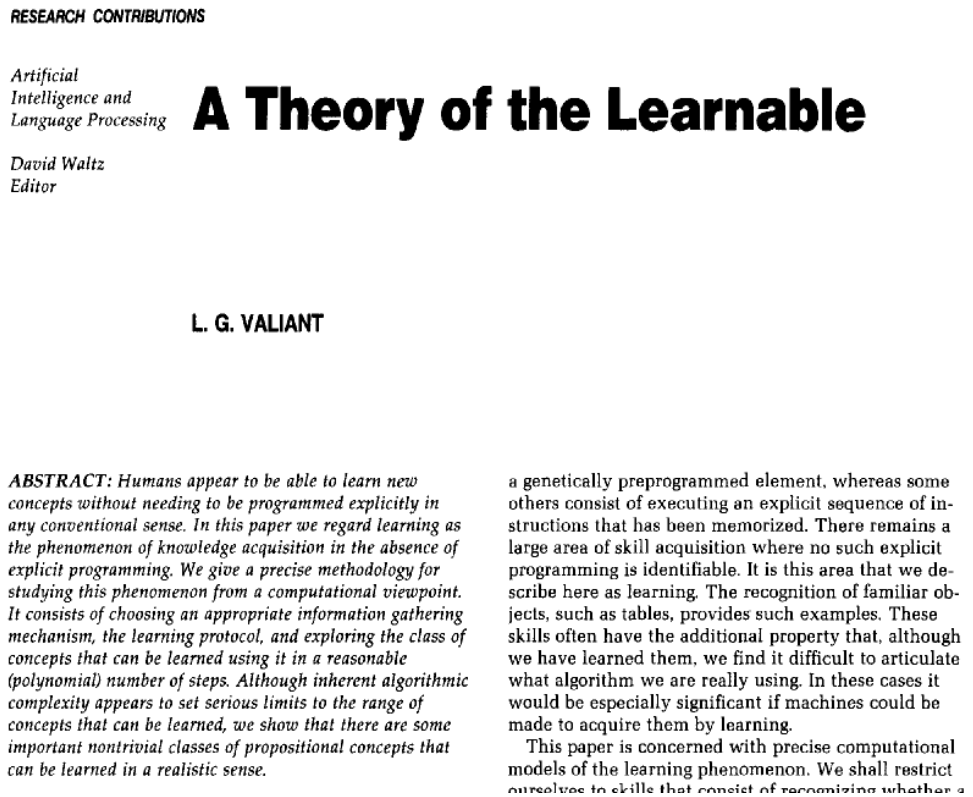
$$\text{error rate} \propto \frac{\text{model complexity}}{\text{sample size}}$$

  - What does this mean for deeper architectures & non-parametric models??

- **More sophisticated theory:**
  - Vapnik-Chervonenkis (VC) Dimension
  - Probably Approximately Correct (PAC) Framework

Probably Approximately Correct (The PAC Framework)

- Learner receives samples & must select a generalization function (hypothesis) from certain class of possible functions (hypotheses)
- ***Goal***: with high probability ("*probably*"), the selected function should have low generalization error ("*approximately correct*")
  - Learner must be able to learn a concept given arbitrary approximation ratio, probability of success, or distribution of the samples
  - Learner must be efficient (w.r.t. time-space complexity; bounded to polynomial of sample size)
  - Learner should find efficient function given sample size that is polynomially upper bounded, further accounting for approximation/likelihood adjustments/bounds



RESEARCH CONTRIBUTIONS

Artificial Intelligence and Language Processing

David Waltz Editor

A Theory of the Learnable

L. G. VALIANT

ABSTRACT: Humans appear to be able to learn new concepts without needing to be programmed explicitly in any conventional sense. In this paper we regard learning as the phenomenon of knowledge acquisition in the absence of explicit programming. We give a precise methodology for studying this phenomenon from a computational viewpoint. It consists of choosing an appropriate information gathering mechanism, the learning protocol, and exploring the class of concepts that can be learned using it in a reasonable (polynomial) number of steps. Although inherent algorithmic complexity appears to set serious limits to the range of concepts that can be learned, we show that there are some important nontrivial classes of propositional concepts that can be learned in a realistic sense.

a genetically preprogrammed element, whereas some others consist of executing an explicit sequence of instructions that has been memorized. There remains a large area of skill acquisition where no such explicit programming is identifiable. It is this area that we describe here as learning. The recognition of familiar objects, such as tables, provides such examples. These skills often have the additional property that, although we have learned them, we find it difficult to articulate what algorithm we are really using. In these cases it would be especially significant if machines could be made to acquire them by learning.

This paper is concerned with precise computational models of the learning phenomenon. We shall restrict ourselves to skills that consist of recognizing whether a

No Free Lunches

Finding the best model (one model to rule them all):

- Overly complex family does not necessarily include target function, true data generating process, or even an approximation
- Best fitting model obtained not by finding the right number of parameters
- Instead, best fitting model is a large model that has been regularized appropriately

We will develop this concept in many contexts!

The No Free Lunch Theorem:

You can only get generalization through assumptions. No one algorithm will solve all problems (some will work better than others in some instances).

Arbitrary Capacity and Nonparametric Models

- When do we reach most extreme case of arbitrarily high capacity?
- Parametric models such as linear regression:
  - learn a function described by a parameter whose size is finite and fixed before data is observed
- Nonparametric models have no such limitation
- Nearest-neighbor regression is an example
  - Their complexity is a function of training set size

Nearest Neighbor Regression and Classification

K-NN = K nearest neighbors

- Simply store the $\boldsymbol{X}$ and $\boldsymbol{t}$ from the training set
- When asked to classify a test point $\boldsymbol{x}$ the model looks up the nearest entry in the training set and returns the associated target, i.e.,

$$\hat{t} = t_i \quad \text{where} \quad i = \arg\min || \boldsymbol{X}_{i,:} - \boldsymbol{x} ||_2^2$$

- Algorithm can be generalized to distance metrics other than L^2 norm such as learned distance metrics

Usually used w/ feature scaling:

$$a_i = \frac{v_i - \min v_i}{\max v_i - \min v_i}$$

Can also employ feature-weighted NN:

$$D(a,b) = \sqrt{\sum_k w_k (a_k - b_k)^2}$$

NN = Nearest Neighbor (1-NN is K=1 Nearest Neighbor(s) Model)

The K-NN Model

Algorithm: $k = K = 1$

1. Find example $(\mathbf{x}^*, t^*)$ (from the stored training set) closest to the test instance $\mathbf{x}$. That is:

$$\mathbf{x}^* = \underset{\mathbf{x}^{(i)} \in \text{train. set}}{\text{argmin}} \ \text{distance}(\mathbf{x}^{(i)}, \mathbf{x})$$

2. Output $y = t^*$

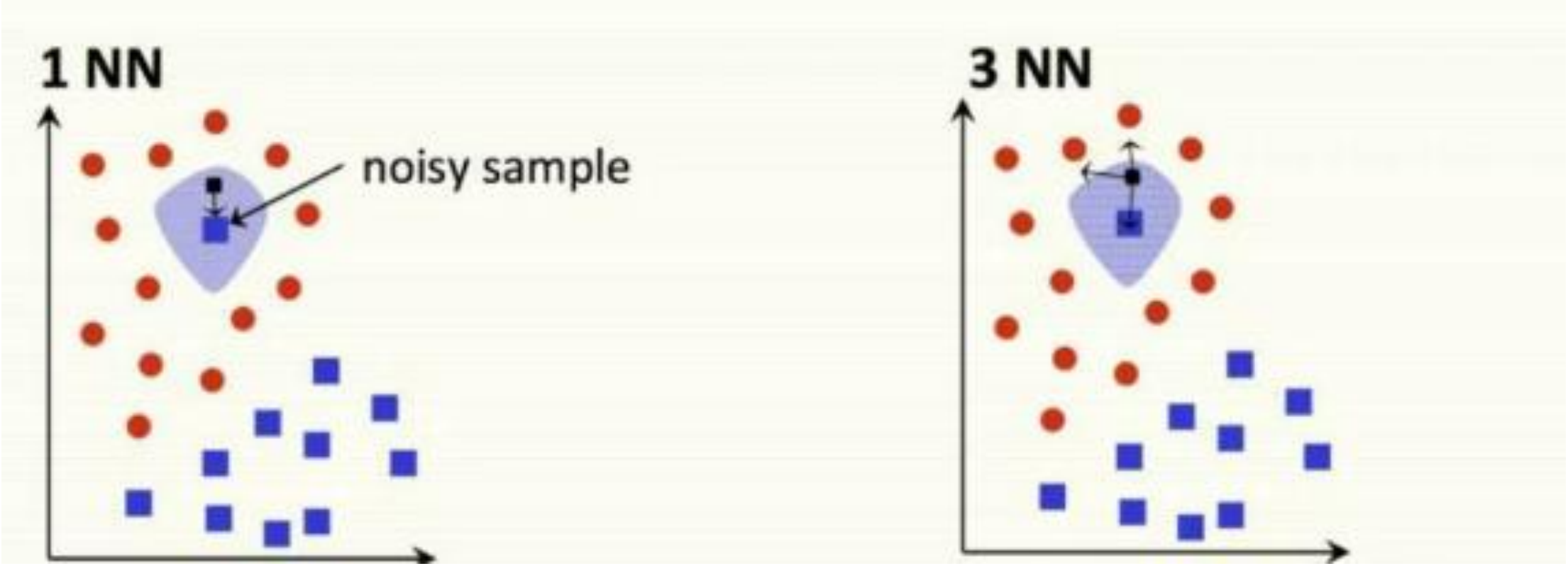
Algorithm (kNN):

1. Find k examples $\{\mathbf{x}^{(i)}, t^{(i)}\}$ closest to the test instance $\mathbf{x}$
2. Classification output is majority class

$$y = arg\max_{t^{(z)}} \sum_{r=1}^{k} \delta(t^{(z)}, t^{(r)})$$

Over $z \in Z$ classes

(Lazy) Instance-Based Learning with K-NN

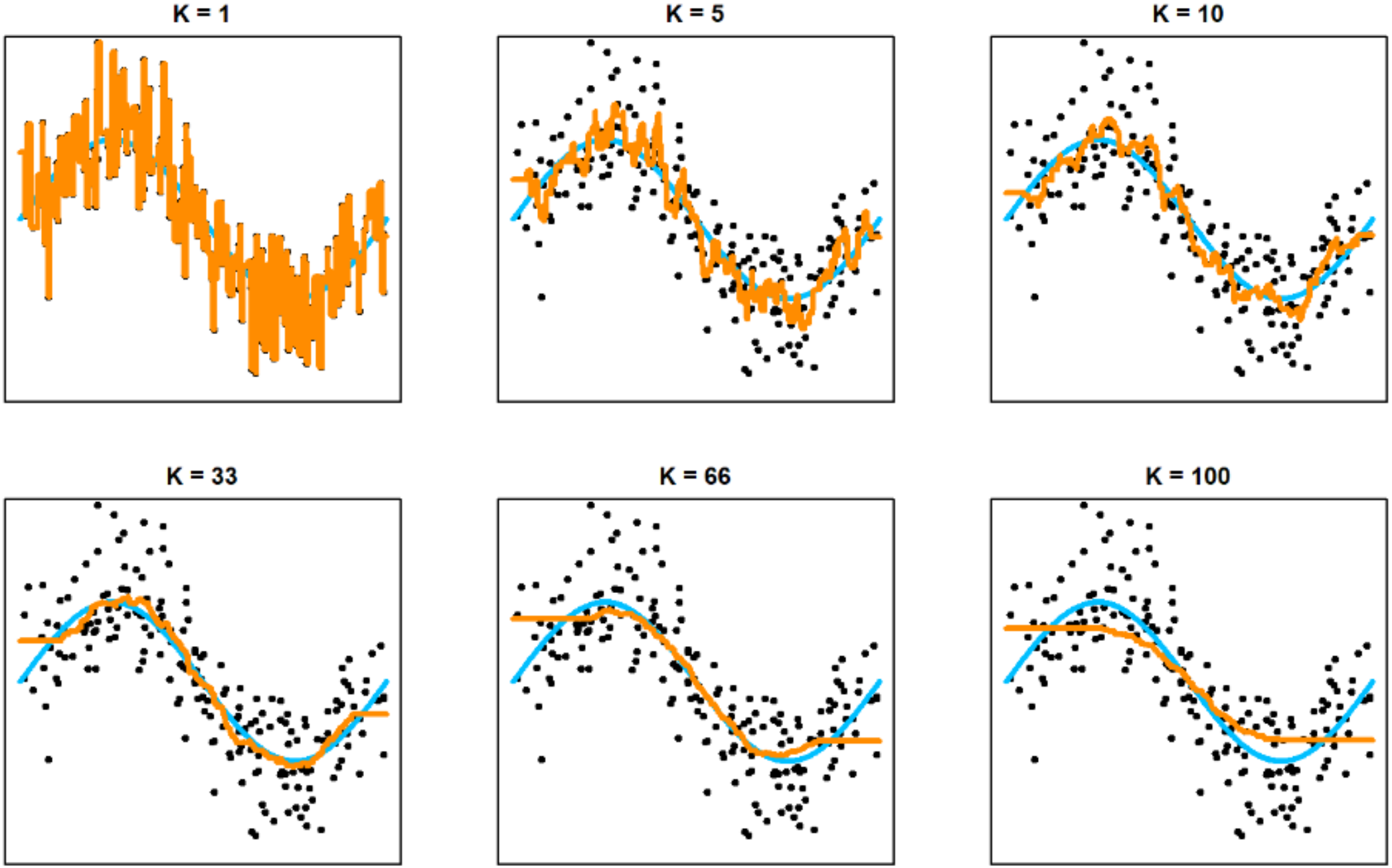


Every sample in blue shaded zone will be misclassified as blue class

Every sample in blue shaded zone will be misclassified as blue class

A Note on *K*-NN Complexity/Capacity

- For *K-NN* & non-parametrics, data samples are “parameters”
- For *K-NN*, *K* controls model “complexity”/capacity, under *N* data samples
  - *K* is inversely proportional to capacity
  - Higher *K* lowers complexity, lower *K* raises complexity
- *K* = 1 (most complex)
  - Every data sample has a “role” in model prediction (“learns” N different patterns)
  - Overfitting (zero training error, high test error; low bias, high variance)
- *K* = N (least complex)
  - Model can only make one prediction (an average; line); “globs” data
  - Underfitting (high training error, high test error; high bias, low variance)


K = 1
K = 5
K = 10
K = 33
K = 66
K = 100

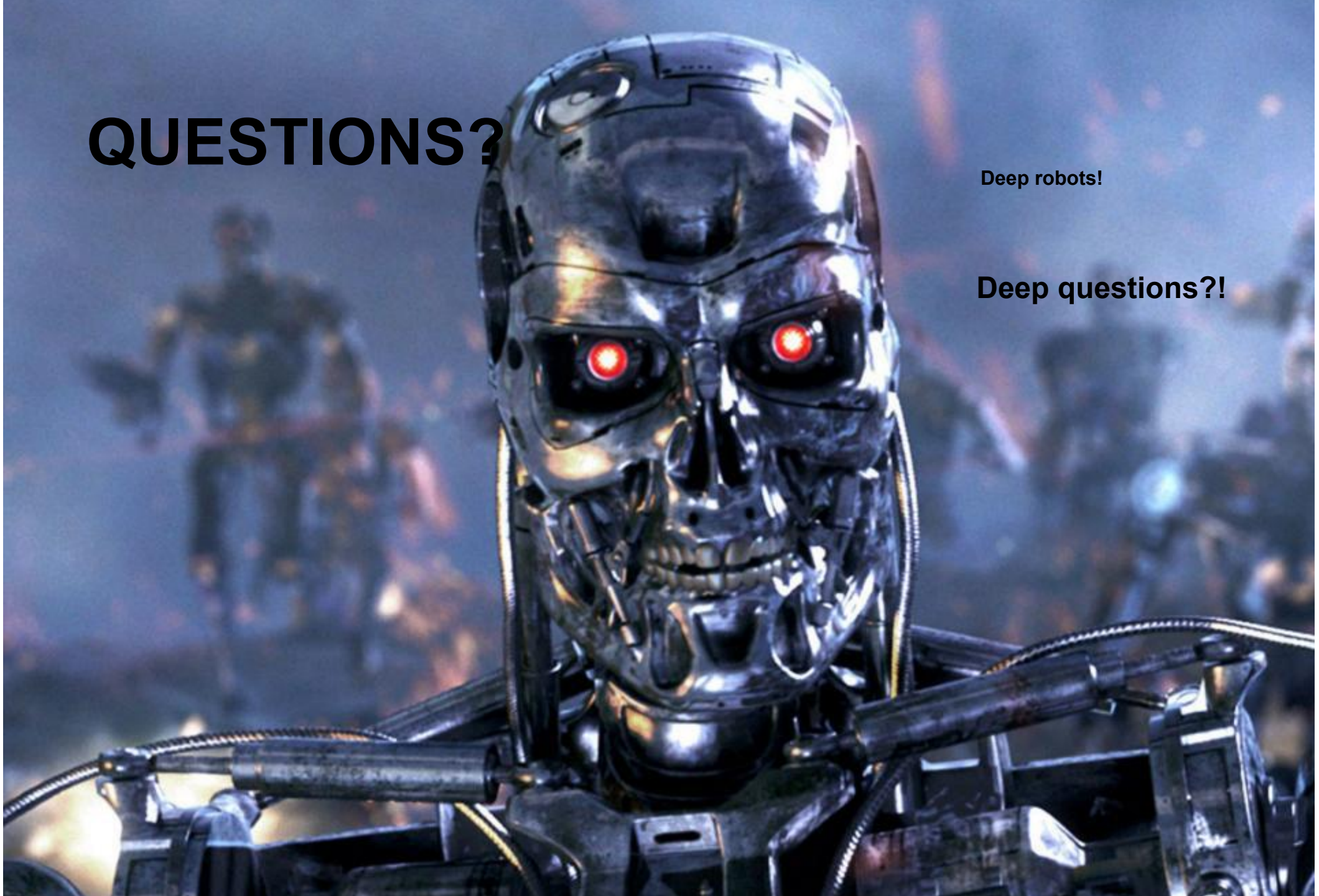
Effect of Training Set Size

- Expected generalization error can never increase as the no of training examples increases
- For nonparametric models, more data yields better generalization until best possible error is achieved

How to Reduce Variance?

- Choose a simpler classifier
  - Occam's Razor: *Among competing hypotheses, the one with the fewest assumptions should be selected.*
- Regularize the parameters
  - Apply penalties / constraints
  - Go Bayesian – apply prior distributions over parameters
- Get more training data
  - ***BIG*** data (some theory justifies this!)

Slide credit: D. Hoiem


QUESTIONS?
Deep robots!
Deep questions?!