Machine Learning Overview

Sargur N. Srihari

University at Buffalo, State University of New York

USA

Outline

1. AI and the Machine Learning approach
2. ML problem types
    1. Based on data:
        Supervised, Unsupervised, Semi-supervised, Reinforcement
    2. Based on output:
        Regression, Classification
    3. Based on models:
        Generative, Discriminative
3. Disruption in Software Development
    – Software 1.0 and Software 2.0

History of AI

- ## Greek Mythology

Pygmalion of Cyprus

Sculpts marble Galatea that came to life (falls in love)
GBS' Pygmalion: Higgins teaches Eliza to speak Queen's English

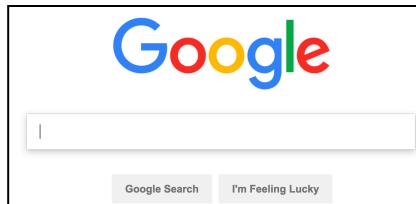Eliza is first AI program
(Weizenbaum, MIT)

- ## Indian Mythology

Ganesa (Buddhipriya)

Parvati forms sandalwood Ganesa, whose head is transplanted from an elephant

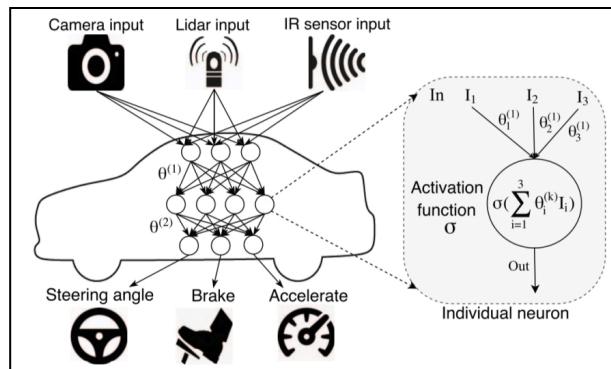# Today AI is ubiquitous

- ## Automate routine labor
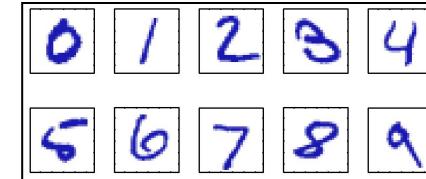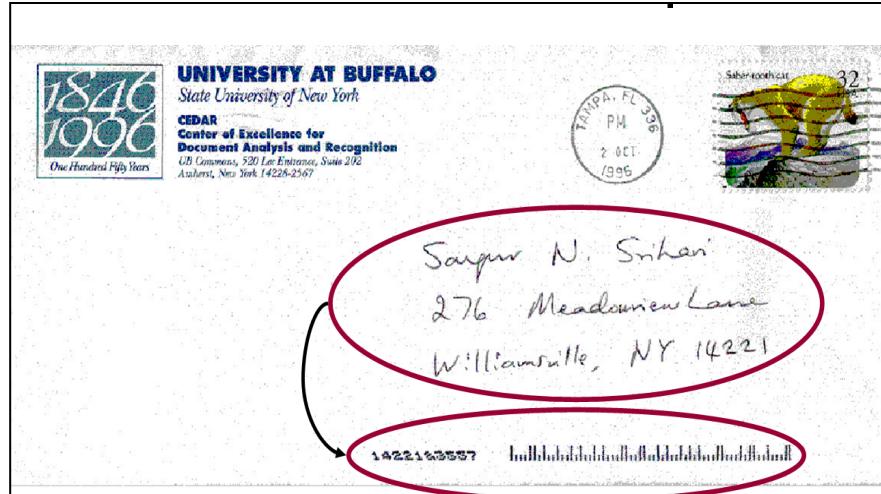 - ### Search

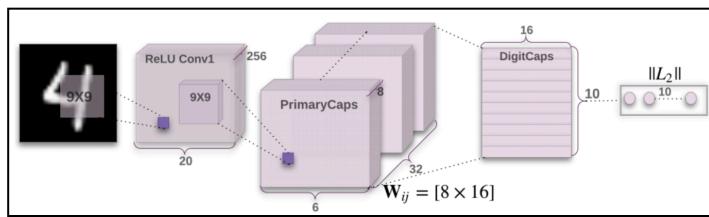- ## Understand speech
  - ### SIRI, Alexa

- ## Autonomous Vehicles

UB, AI and Fruit-fly

- **Many handcrafted rules and exceptions**
- **Better learn from training set**
- **Handwriting rec cannot be done without ML!**
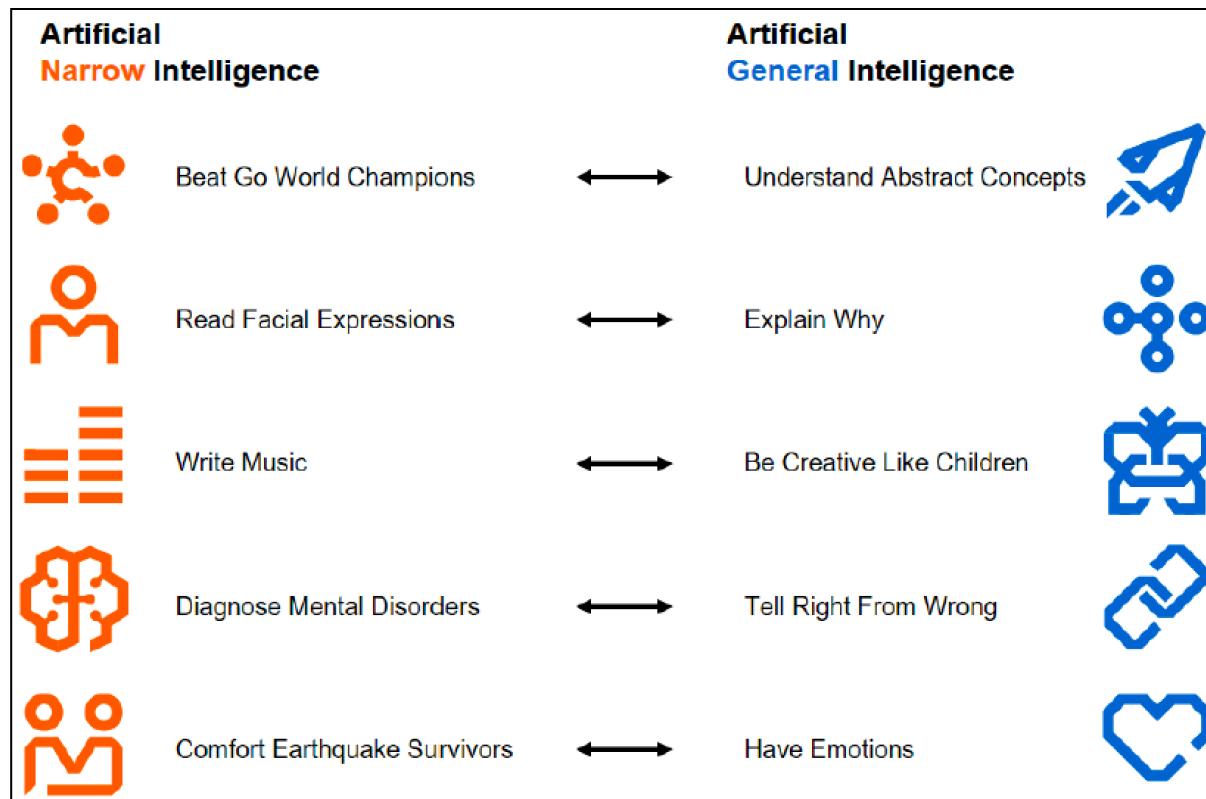

Handwriting is the fruit fly of AI
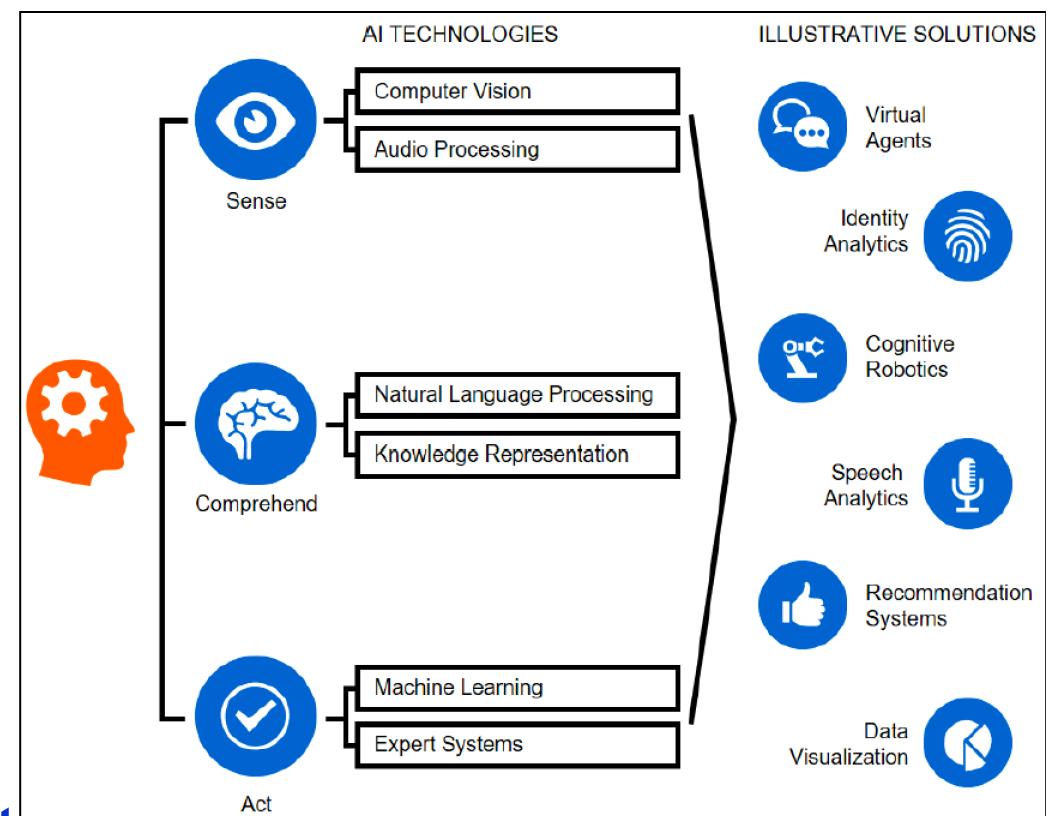

NYU, Toronto, Montreal, Baidu   5

AI Paradox

- Hard problems for people are easy for AI
- Easy problems are hard for AI
  - Narrow Intelligence   General Intelligence

People easy tasks:

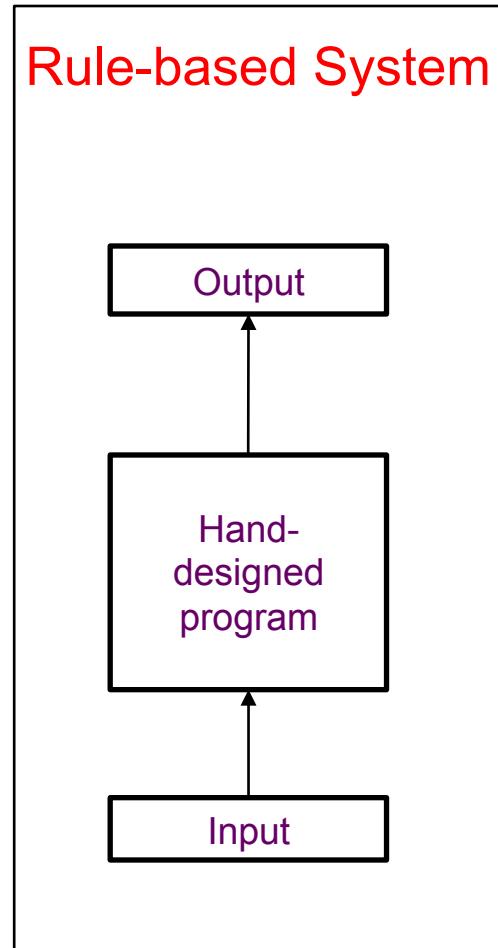| **Artificial Narrow Intelligence**		**Artificial General Intelligence**
Beat Go World Champions	⟷	Understand Abstract Concepts
Read Facial Expressions	⟷	Explain Why
Write Music	⟷	Be Creative Like Children
Diagnose Mental Disorders	⟷	Tell Right From Wrong
Comfort Earthquake Survivors	⟷	Have Emotions

6

What tasks require intelligence?

- **Reasoning**
  - Puzzles, Judgments

- **Planning**
  - Action sequences

- **Learning**
  - Improve with data

- **Natural language**

- **Integrating skills**

- **Abilities to sense, act**

Everyday life needs knowledge

- **Knowledge is intuitive and subjective**
  - Key challenge of AI is how to get this informal knowledge into a computer

- **Knowledge-based Approach**
  - Hard-code knowledge in a formal language
  - Computer can reason about statements in these languages using inference rules
  - Examples:
    - Expert systems for diagnosis (MYCIN, CADUCEUS)
    - Design (VAX)

-

Knowledge-Based AI

Rule-based System

Output

Hand-
designed
program

Input

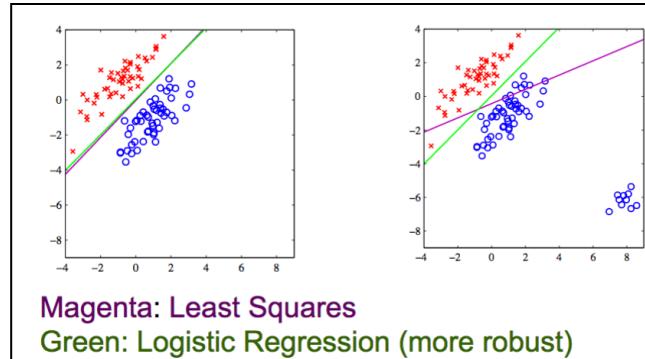Disadvantage: Unwieldy process
Time of human experts
People struggle to formalize rules with enough complexity to describe the world
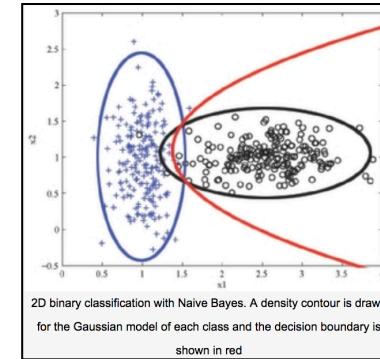
The Machine Learning approach

- Difficulties of hard-coded approach suggests:
  - Allow computers to learn from experience
- First determine what features to use
- Learn to map the features to outputs

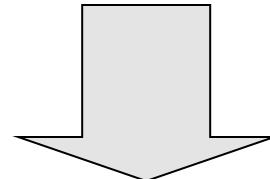Linear classifier   Quadratic classifier

Magenta: Least Squares
Green: Logistic Regression (more robust)

2D binary classification with Naive Bayes. A density contour is drawn
for the Gaussian model of each class and the decision boundary is
shown in red

The ML Approach

Data Collection
  Samples
Model Selection
  Probability distribution to model process
Parameter Estimation
  Values/distributions

Generalization

(Training)

Inference
  Find responses to queries

Decision
(Inference
OR
Testing)

Learning Problem Definition

- Improving some measure of performance P when executing some task T through some type of training experience E

- Example: Learning to detect credit card fraud

- **Task** T
  - Assign label of fraud or not fraud to credit card transaction

- **Performance measure** P
  - Accuracy of fraud classifier

  With higher penalty when fraud is labeled as not fraud

- **Training experience** E
  - Historical credit card transactions labeled as fraud or not

12

ML Problem Types

1. Based on Type of Data

    1. Supervised, Unsupervised, Semi-supervised

    2.   Reinforcement Learning
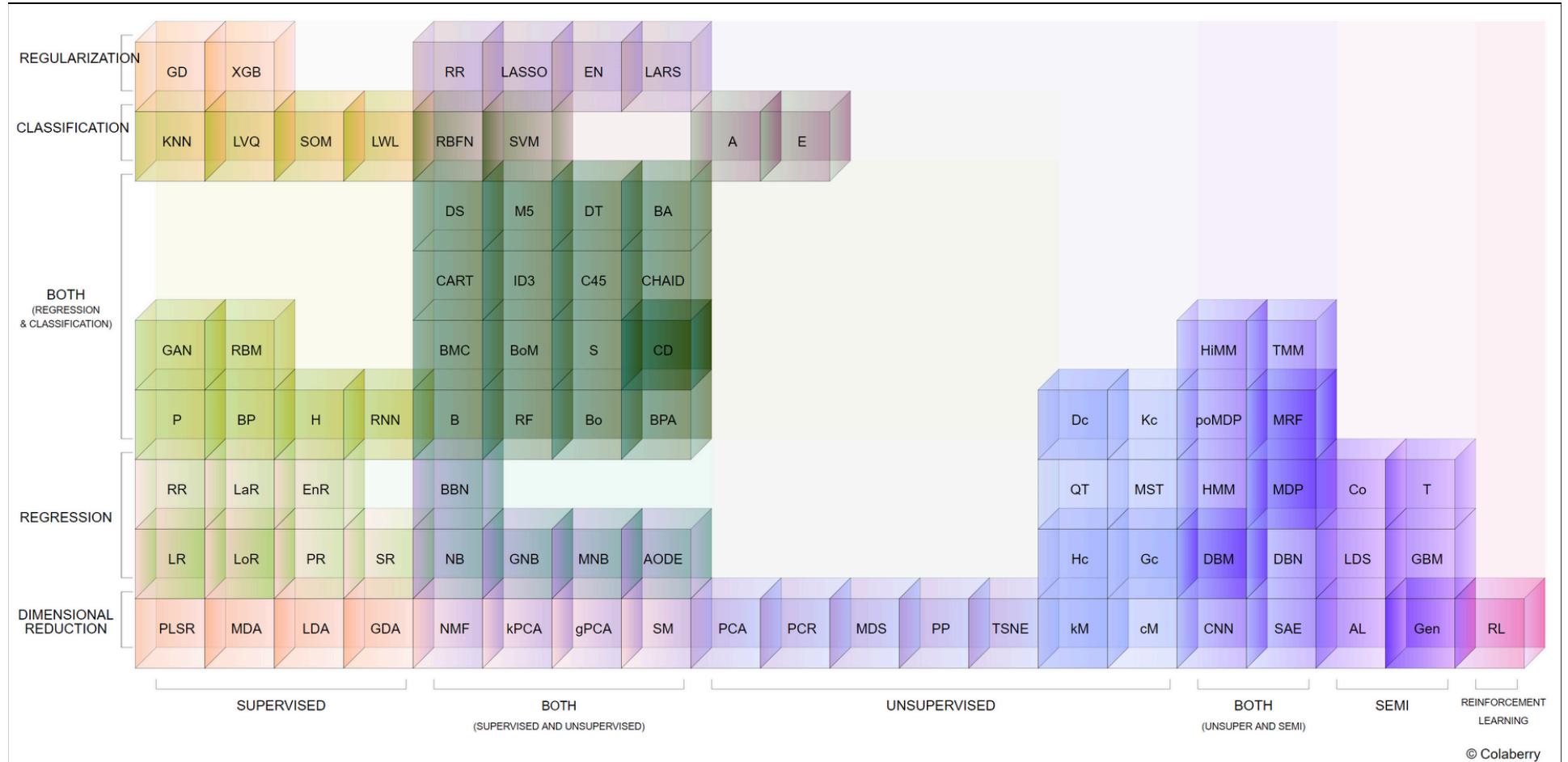
2. Based on Type of Output

    –   Regression, Classification

3. Based on Type of Model

    –   Generative, Discriminative
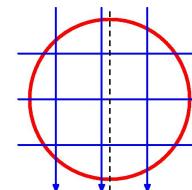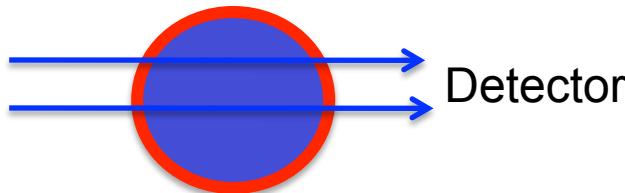
Periodic Table of ML algorithms

We will look at examples first proceeding along the horizontal axis and then along the vertical

Supervised Learning

- Most widely used methods of ML, e.g.,
  - Spam classification of email
  - Face recognizers over images
  - Medical diagnosis systems

- Inputs x are vectors or more complex objects
  - documents, DNA sequences or graphs

- Outputs are binary, multiclass(K),
  - Multi-label (more than one class), ranking,
  - Structured:
    - y is a graph satisfying constraints, e.g., POS tagging
  - Real-valued or mixture of discrete and real-valued 15

Supervised Classification Example

- ## Off-shore oil transfer pipelines

  - Non-invasive measurement of *proportion* of oil,water, gas

    - Called Three-phase Oil/Water/Gas Flow

- ## Input data: Dual-energy gamma densitometry

  - Beam of gamma rays passed through pipe

  - Attenuation in intensity indicates density of material

  - Single beam insufficient

    - Two degrees of freedom: fraction of oil, fraction of water
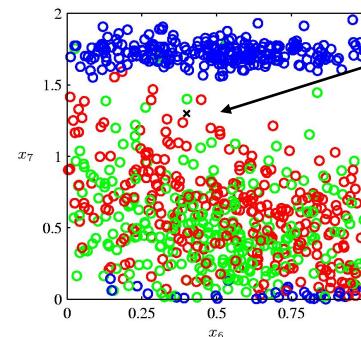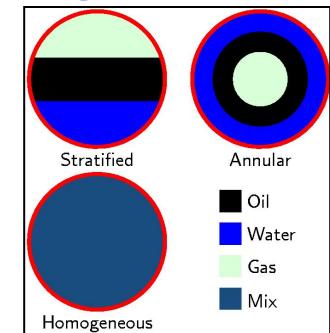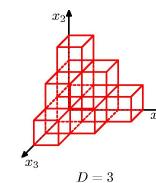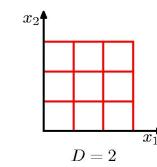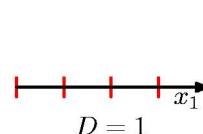   - One beam of Gamma rays of two energies (frequencies)

Detector

- Six Beams
- 12 measurements
  - attenuation
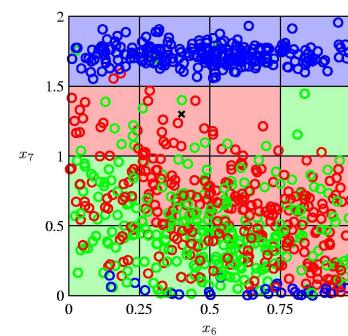
16

Prediction Problems

1. Predict Volume Fractions of oil/water/gas
2. Predict configuration (one of three)
   – Twelve Features
     – Three classes
     – Two variables, 100 points shown

- Naïve cell based voting fails
  – exponential growth of cells with dimensionality
  – 12 dimensions discretized into 6 gives 3 million cells
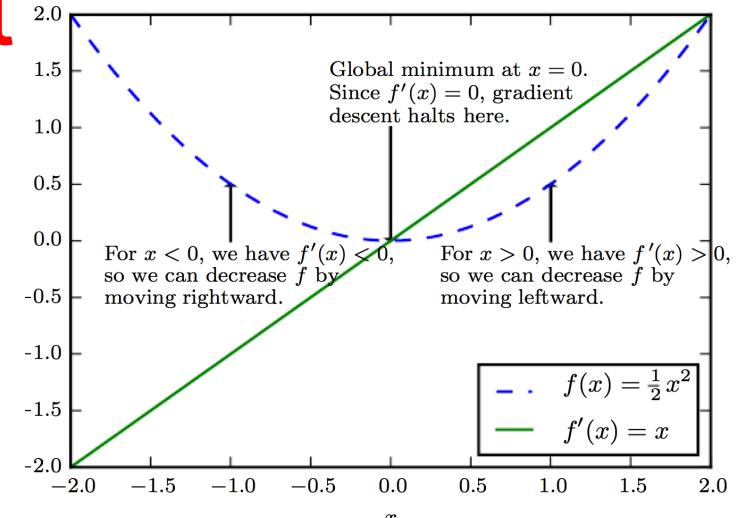- Hardly any points in each cell

Which class should x belong to?
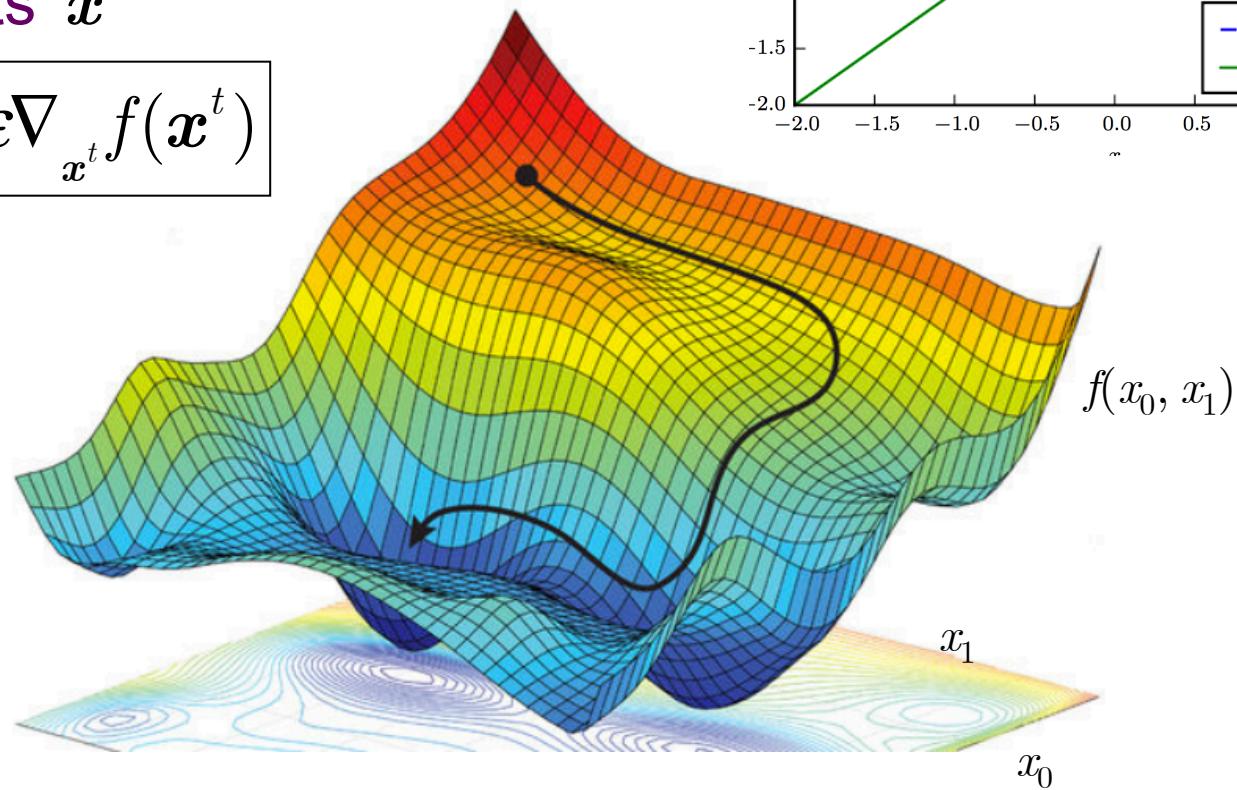
Supervised Learning by Gradient Descent

Classification Task:

Loss function $f(x)$,
e.g., sum of squared errors,
given weights x

$$x^{t+1} = x^t - \varepsilon \nabla_{x^t} f(x^t)$$

Global minimum at $x = 0$.
Since $f'(x) = 0$, gradient
descent halts here.

For $x < 0$, we have $f'(x) < 0$,
so we can decrease f by
moving rightward.

For $x > 0$, we have $f'(x) > 0$,
so we can decrease f by
moving leftward.

$f(x) = \frac{1}{2}x^2$

$f'(x) = x$

$f(x_0, x_1)$

x_1

x_0

18

Ability to Generalize
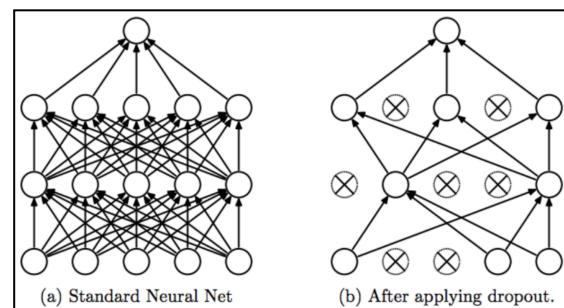
- ## ML algorithms need to perform well not just on training data but on new inputs as well

  1. Parameter Norm Penalties (L^2- and L^1- regularization)
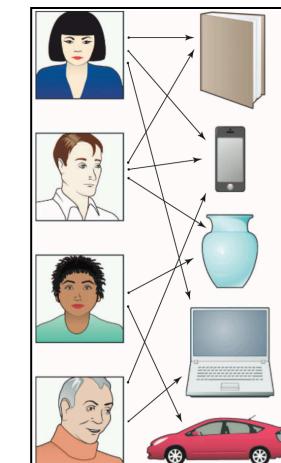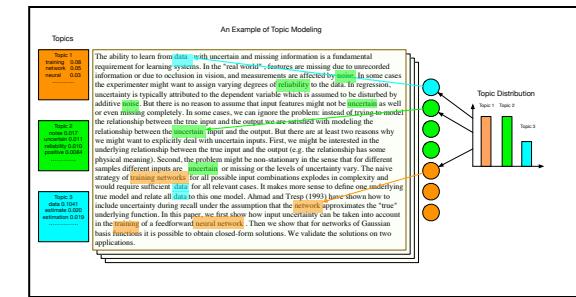 2. Data Set Augmentation
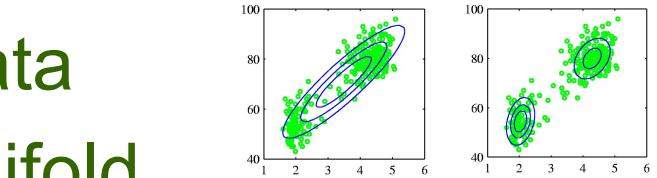
Translation
Rotation
Scaling

  3. Early Stopping
  4. Dropout

(a) Standard Neural Net   (b) After applying dropout.
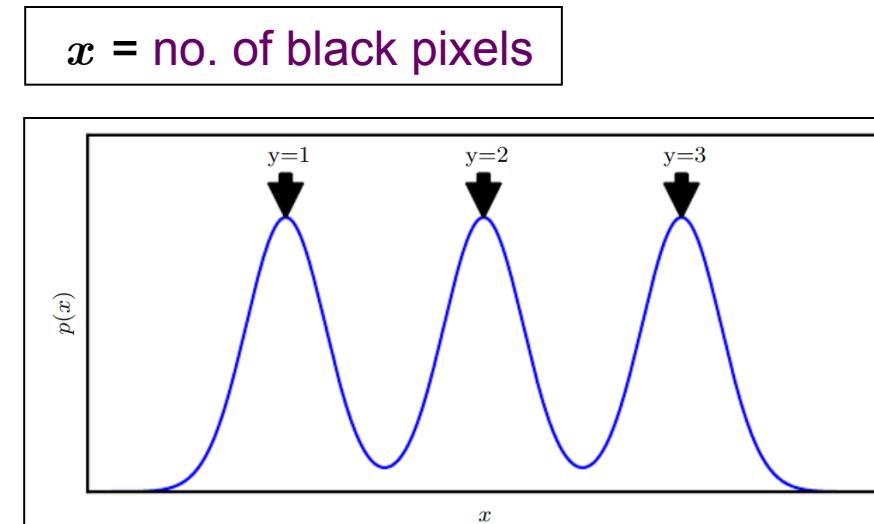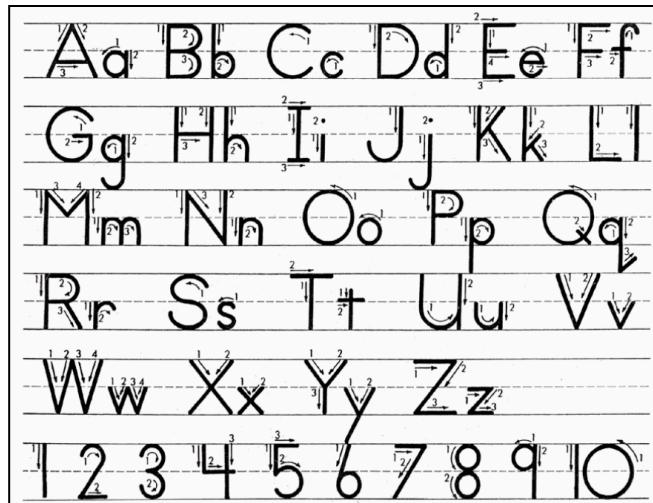
Unsupervised Learning

- Unlabeled data assuming underlying structure

  1. Clustering to find partition of data

  2. Identify a low-dimensional manifold

      - PCA, Autoencoder

  3. Topic modeling

      - Topics are distributions over words

      - Document: a distribution across topics

          – Methods: SVD, Collaborative Filtering

  4. Recommendation Systems

  – Data links between users and items

  – Suggest other items to user

  – Solution: SVD, Collaborative Filtering

20

How semi-supervised can succeed

- Ex: density over x is a mixture over three components, one per value of $\mathbf{y}=$ cap/small/digit

- If components well-separated:
  - modeling $p(\mathbf{x})$ reveals where each component is
    - A single labeled example per class enough to learn $p(\mathbf{y}|\mathbf{x})$

x = no. of black pixels
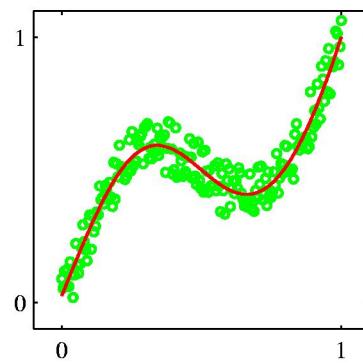
In this case $p(\mathbf{y}|\mathbf{x})$ is a univariate Gaussian for $\mathbf{y}=1,2,3$

21

Reinforcement Learning

- Training data inbetween supervised/unsupervised
  - Indication of whether action is correct or not
  - Reward signal may refer to entire input sequence
  - Dog is given a reward/punishment for an action
    - Policies: what actions to take in a particular situation
    - Utility estimation: how good is state ($\rightarrow$used by policy)

- No supervised output but delayed reward

- Credit assignment
  - what was responsible for outcome

- Applications:
  - Game playing, Robot in a maze, Multiple agents, partial observability, …

22

Regression

Problem data set

Corresponding inverse problem by reversing x and t
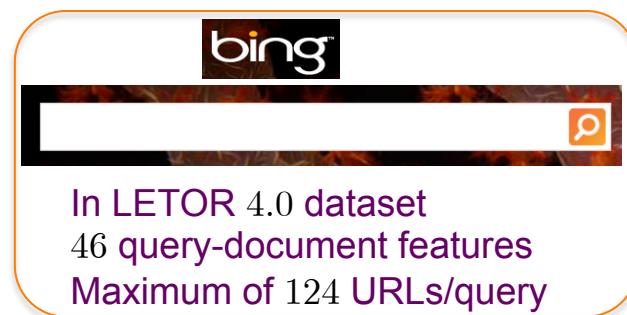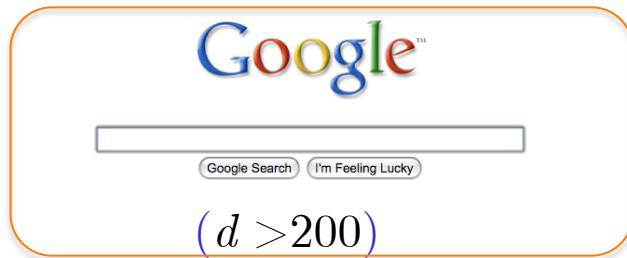
Red curve is result of fitting a two-layer neural network by minimizing squared error
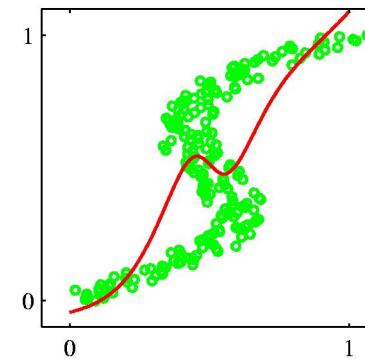

Very poor fit
to data:
GMMs used here

Regression: Learning to Rank

$(d > 200)$

In LETOR 4.0 dataset
46 query-document features
Maximum of 124 URLs/query

Input (x_i):
(d Features of Query-URL pair)

- Log frequency of query in
                     anchor text
- Query word in color on page
- # of images on page
- # of (out) links on page
- PageRank of page
- URL length
- URL contains "~"
- Page length
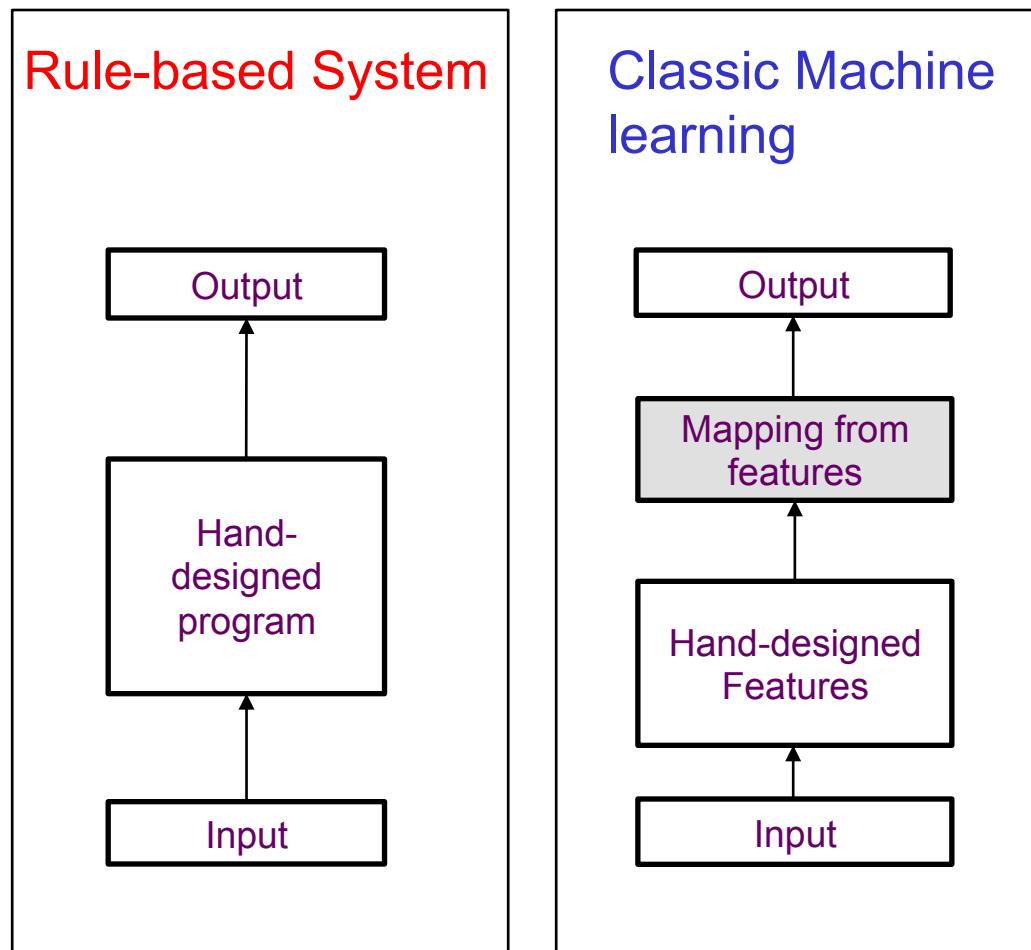
Traditional IR uses TF/IDF

Output (y):
Relevance Value

Target Variable
- Point-wise ($0,1,2,3$)
- Regression returns continuous value
  -Allows fine-grained ranking of URLs

24

Two paradigms in AI

Rule-based System

Output

↑

Hand-designed program

↑

Input

Classic Machine learning

Output

↑

Mapping from features

↑

Hand-designed Features

↑

Input

☐ Shaded boxes indicate components that can learn from data

Designing right set of features

- Simple Machine Learning depends heavily on *representation* of given data

- For detecting a car in photographs
  - Tire shape difficult in terms of pixel values
    - Shadows, glare, occlusion
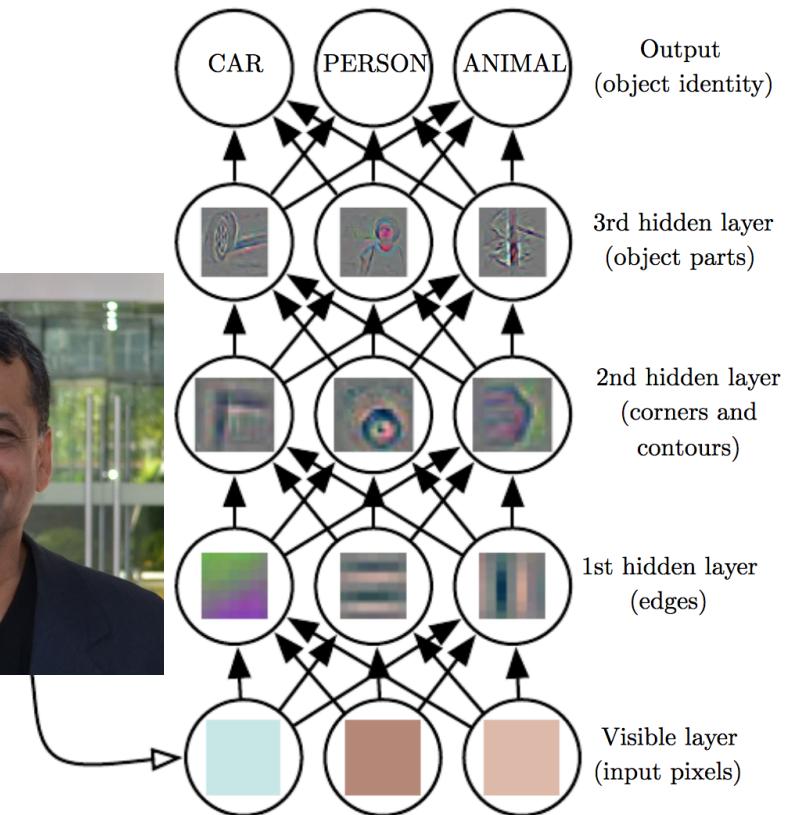
Representation Learning

- Solution: use ML to not only learn mapping from representation to output but representation itself
  - Better results than hand-coded representations

- Allows AI systems to rapidly adapt to new tasks
  - Designing features can take great human effort
    - Can take decades for a community of researchers

- Does not need programmer to have deep knowledge of the problem domain

27
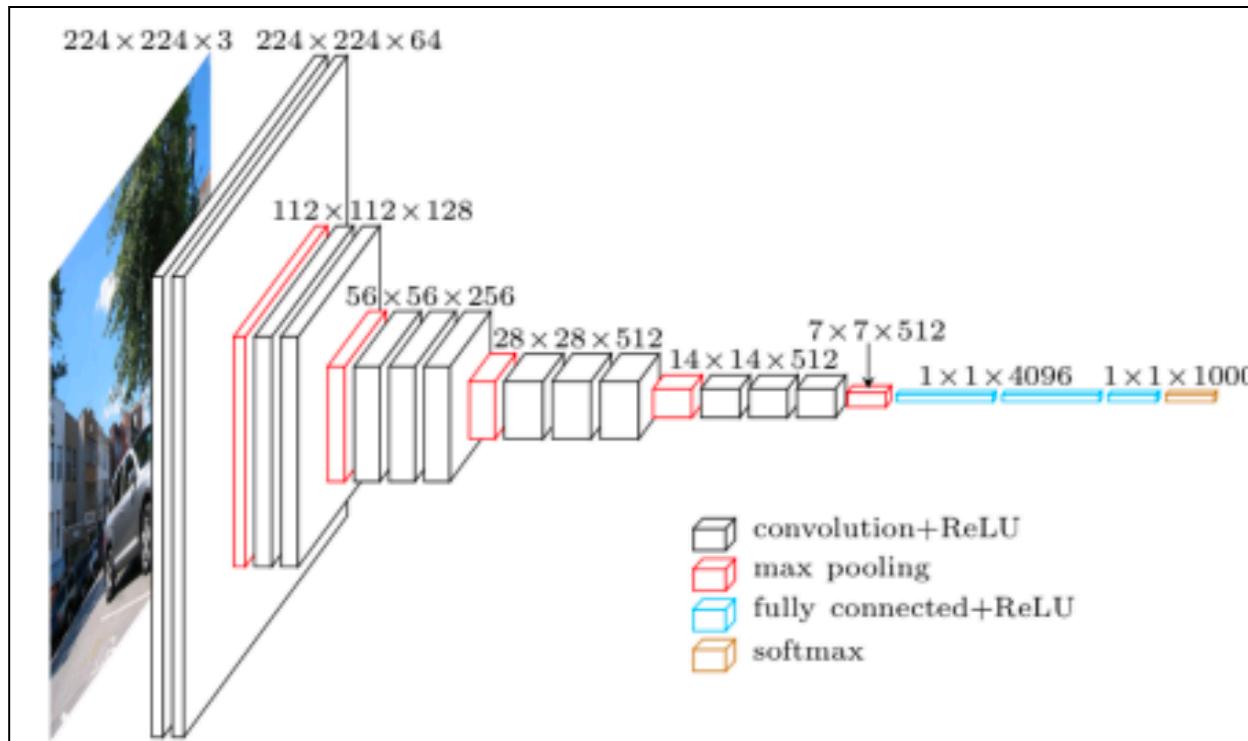
Feature Learning for Classification

- Function to map pixels to object identity is complicated

- Series of hidden layers extract increasingly abstract features

- Final decision made by a simple classifier
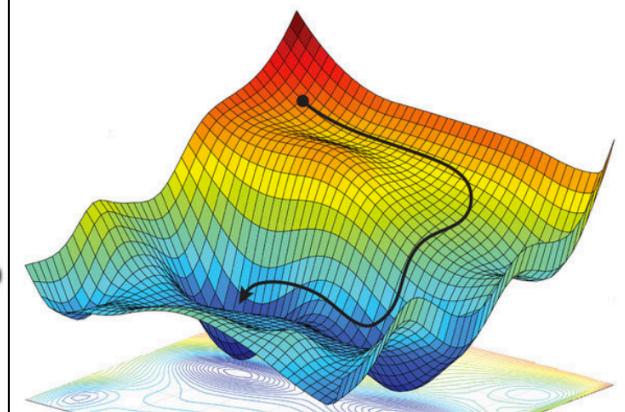
Deep Learning
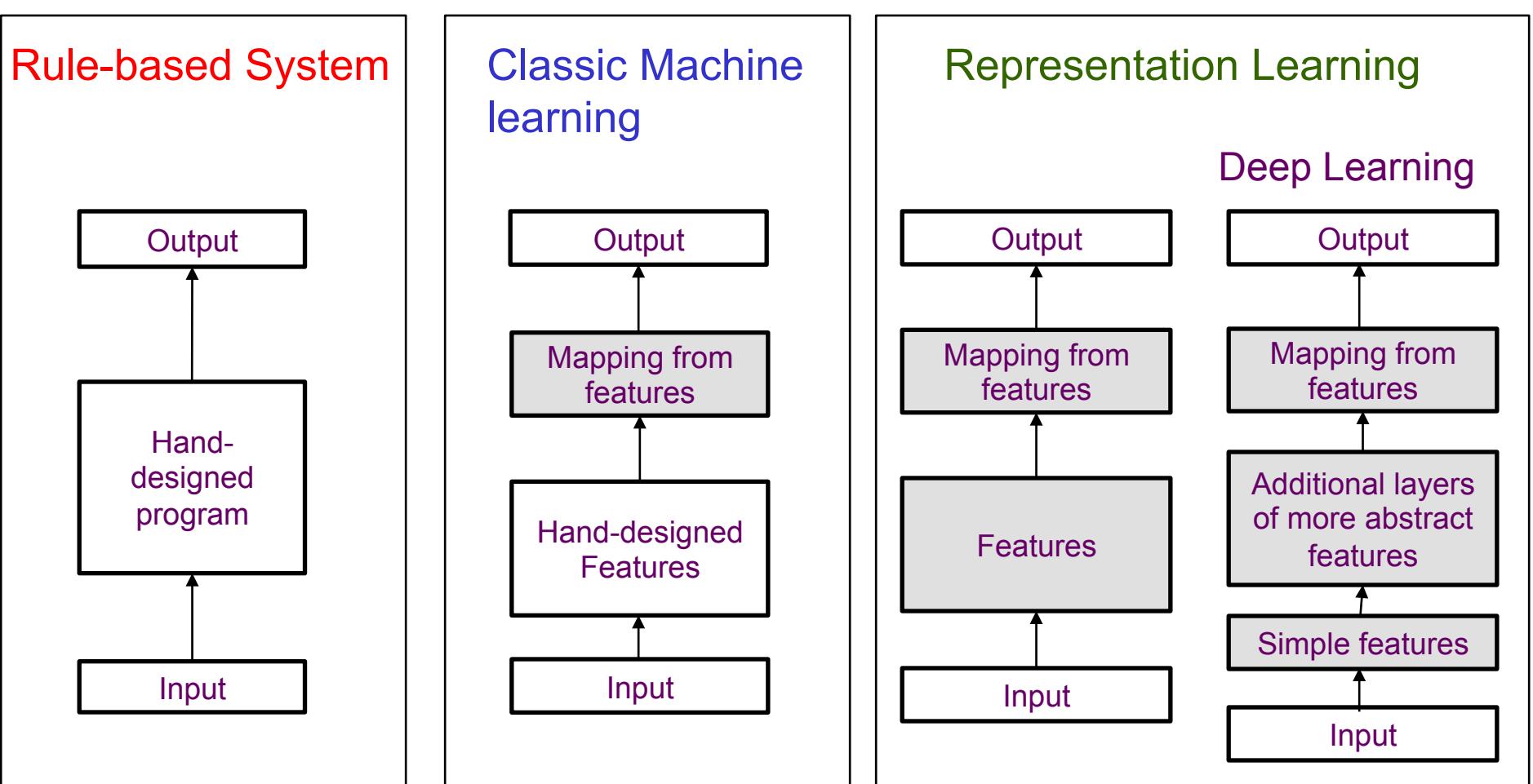
- ## Understand the world as hierarchy of concepts

  - How these concepts are built on top of each other is deep, with many layers

  - Weights learnt by gradient descent

$$x^{t+1} = x^t - \varepsilon \nabla_{x^t} f(x^t)$$

convolution+ReLU
max pooling
fully connected+ReLU
softmax

29

Summary of AI Models

Rule-based System	Classic Machine learning	Representation Learning

Rule-based System

Output

↑

Hand-designed program

↑

Input

Classic Machine learning

Output

↑

Mapping from features

↑

Hand-designed Features

↑

Input

Representation Learning

Output

↑

Mapping from features

↑

Features

↑

Input

Deep Learning

Output

↑

Mapping from features

↑

Additional layers of more abstract features

↑

Simple features

↑

Input

Shaded boxes indicate components that can learn from data

AI Paradigm Shift

- **Physics paradigm shift**
  - Newtonian Physics
    - Cannot explain black-body radiation
  - Quantum Mechanics

- **AI paradigm shift**
  - Knowledge-based systems
    - Cannot perform simple recognition tasks
  - Simple machine learning methods
    - Cannot perform complex recognition tasks
  - Deep Learning methods
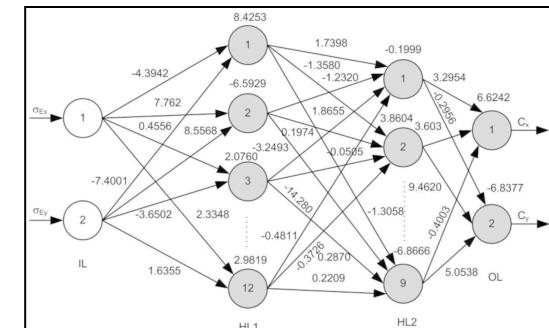
ML as a Software Area

1. ML is programming computers for AI:
   – Perform tasks that humans perform well but difficult to specify algorithmically

2. ML is principled way to build IT systems
   – Probabilistic responses to queries—IR
   – Adaptive user interfaces, personalized assistants (information systems)
   – Scientific/engineering applications

Disruption in Software Development

- **Machine Learning is not just another tool**
  - It is a fundamental shift in how software is written
- **Software** 1.0 **(Classical "stack")**
  - It is code we write
    - e.g., LAMP(Linux, Apache, MySQL, Python/Perl)
- **Software** 2.0 **(Code written by Optimizer)**
  - It is in a user unfriendly language
    - There are millions of weights
  - No human involved in coding

Software 1.0 and 2.0: Fizzbuzz
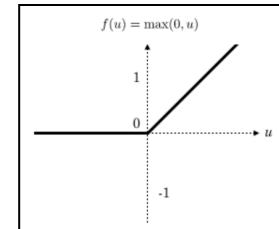
- Print $i = 1$ to 100, except:
  - if divisible by 3 print fizz,
  - if divisible by 5 print buzz,
  - if divisible by both 3 and 5 print fizzbuzz

- Two approaches:
  - Software 1.0: C++   Software 2.0: Python/Tensorflow

```cpp
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      for(int i=1;i<=100;i++){
6          if((i%3 == 0) && (i%5==0))
7              cout<<"FizzBuzz\n";
8          else if(i%3 == 0)
9              cout<<"Fizz\n";
10         else if(i%5 == 0)
11             cout<<"Buzz\n";
12         else
13             cout<<i<<"\n";
14     }
15     return 0;
16 }
```

$$\text{def model}(X, w\_h, w\_o):$$
$$h = tf.nn.relu(tf.matmul(X, w\_h))$$
$$\text{return } tf.matmul(h, w\_o)$$

$f(u) = \max(0, u)$
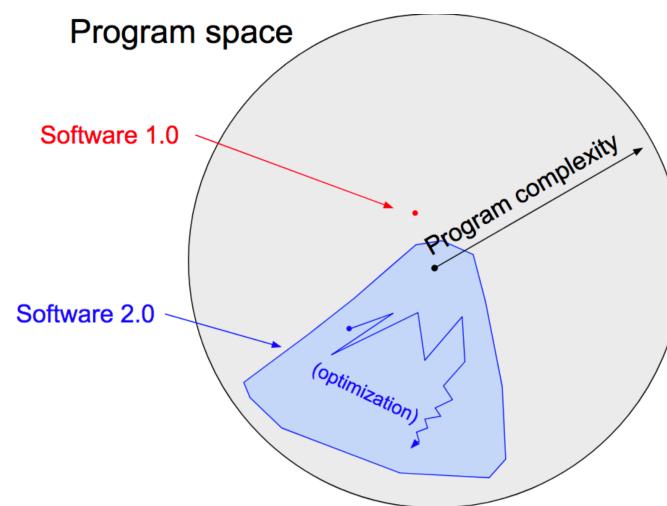
Program Space: Software 1.0 vs 2.0

Software 1.0

By writing each line of code, programmer identifies a point in program space with some desirable behavior

Software 2.0

Restrict search space to continuous subset of program space
Search is made efficient by using stochastic gradient descent

Program space

Software 1.0

Program complexity

Software 2.0

(optimization)

*A. Karpathy, Tesla

35

Benefits of Software 2.0

- Computationally homogeneous
  - Sandwich of two operations: matrix multiply, RELU

- Simple to bake into silicon
  - Small instruction set

- Constant run time
  - Every iteration of forward pass has same FLOPS
  - Constant memory use

- Highly portable: sequence of matrix multiplies is easier

- Very agile
  - C++ is hard to speed-up, instead remove half of channels

- Can meld into optimal whole
  - Software often has modules, can jointly optimize

- It is better than you

36

Limitations of Software 2.0 stack

- After training we have large networks that work very well, but hard to tell how
  - 90% accurate model we understand
  - 99% accurate model we don't
- Can fail unintuitively (Adversarial examples)

Correct steering   With darker image

y ="panda" with 58% confidence

x $\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$ $\boldsymbol{x} + \epsilon\,\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

$+\ .007 \times$

y ="gibbon" with 99% confidence

37

How ML disrupts CS

- ## Software Developer
  - Present: Write and maintain layers of tangled code
  - Future: A teacher– curate training data & analyze results

- ## Mathematics
  - Present: Logic, Discrete mathematics
  - Future: Probability Theory, Calculus, Linear Algebra

- ## Programming Environments
  - Present: C++, Java
  - Future: Tensorflow/Pytorch/Gluon/…
    - In ten years most software jobs won't involve programming

- ## Hardware
  - Present: CPUs
  - Future: GPUs

The transition

- Conventional software can be replaced with a ML solution with improvement, e.g.*,
  - Upgrading search ranking
  - Data center energy usage
  - Language translation   *Projects at Alphabet:
  - Solving GO

- Repositories
  - Github is home for Software 1.0 code
  - Software 2.0 repositories are datasets
    - Commits are additions and edits of the labels

ML Models rely on Probability Theory

- **Sum Rule** for Marginalization

$$p(X = x_i) = \sum_{j=1}^{L} p(X = x_i, Y = y_j)$$

- **Product Rule: for combining**

$$p(X,Y) = \frac{n_{ij}}{N} = p(Y \mid X)p(X)$$

- **Bayes Rule** $\quad p(Y \mid X) = \dfrac{p(X \mid Y)p(Y)}{p(X)} \quad$ where $\quad p(X) = \sum_Y p(X \mid Y)p(Y)$

  Viewed as Posterior α likelihood x prior

- **Fully Bayesian approach**

  - Conjugate distributions
  - Feasible with increased computational power
  - Intractable posterior handled using either
    - Variational Bayes or
    - Stochastic sampling
    - e.g., Markov Chain Monte Carlo, Gibbs

40

Generative/Discriminative Models

- **Generative**
  - Naïve Bayes
  - Mixtures of multinomials
  - Mixtures of Gaussians
  - Hidden Markov Models (HMM)
  - Bayesian networks
  - Markov random fields

- **Discriminative**
  - Logistic regression
  - SVMs
  - Traditional neural networks
  - Nearest neighbor
  - Conditional Random Fields (CRF)

Summary

- **Machine Learning as an AI approach**
  - Overcomes limitations of knowledge-based systems
- **Types of AI tasks**
  - Data: Supervised, Unsupervised, Semi-supervised, Reinforcement
  - Output: Classification, Regression, Ranking
  - Model: Generative, Discriminative
- **Disruption of computer science**
  - Principled approach to develop IT systems
  - Drivers are mobile systems (big data), personalization