ECE408 Lecture 11

Machine Learning and Deep Learning

ECE408 / CS483 / CSE 408

Spring 2020

(by Carl Pearson)

Objective

- To understand the application areas for machine learning.
- To learn the basic strategy for machine learning applications.
- To understand the extension to deep learning (mostly a research pitch).

Perspective is Important

Chips are cheaper than ever.

Unlike humans, digital systems offer

- **high-speed computation**,
- **low capital investment** (purchase vs. training a human), and
- **negligible operations cost** (no salary!).

If computer outperforms (or even matches) a **human, use a computer**.

Industry has done so for 40-50 years now.

What is Machine Learning?

machine learning: important method of building applications whose logic is not fully understood

Typically **by example**:

- **use labeled data** (matched input-output pairs)
- **to represent** desired **relationship**.

Iteratively adjust program logic to produce desired/approximate answers (called **training**).

Types of Learning Tasks

- classification
  - Map each input to a category
  - Ex: object recognition, chip defect detection
- regression
  - Numerical prediction from a sequence
  - Ex: predict tomorrow's temperature
- transcription
  - Unstructured data into textual form
  - Ex: optical character recognition

More Advanced Learning Tasks

- translation
  - Convert a sequence of symbols in one language to a sequence of symbols in another
- structured output
  - Convert an input to a vector with important relationships between elements
  - Ex: natural language sentence into grammatical structure
- others
  - Anomaly detection, synthesis, sampling, imputation, denoising, density estimation, genetic variant calling

Test Cycle Time is Important

You've all written code…

- code, test, code, test, code, test
- integrate, test, test, test
- and test again!

But how long is the code, test cycle?

Depends what you're building.

What's your longest?

Your Cycle Times are Probably Small

In college, **10k lines** took **½ hour** to compile on my PC.

In grad. school, **100k lines** took

- **½ hour** to compile on my workstation, or
- **2 minutes** on our cluster (research platform).

In ECE435 (networking lab), students needed

- **½ hour** to reinstall Linux after a bad bug.
- (Ever had a good bug?)

Gene sequencing / applications can take **two weeks**.

We're all a little spoiled…

Why Machine Learning Again?

In 2007, **programmable GPUs accelerated the training cycle**.

Today, **new chip designs** for learning applications **have further accelerated**.

Led to a resurgence of interest

- in Computer Vision, Speech Recognition, Document Translation, Self Driving Cars, Data Science…
- all tasks that **human brains solve regularly**, **but** for which **we** have **struggled to express solutions** systematically.

Many Problems are Still Hard

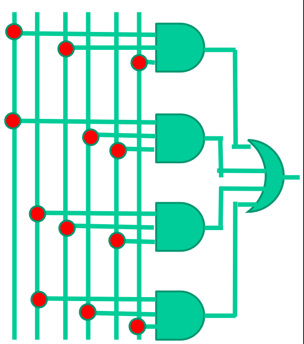
Speed is not a panacea.

- Many **tasks still require human insight**
  - for network structure and feature selection
  - for effective input and output formats, and
  - for production of high-quality labeled data.
- Other trends sometimes help: ubiquitous computing enables crowdsourcing, for example.

Many Problems Have Systematic Solutons

Example: building a Boolean function from a truth table.

Input			output
a	b	c	
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1



What if We Lack a Truth Table?

- Make enough observations to construct a rule
  - 000 → 0
  - 011 → 0
  - 100 → 1
  - 110 → 0
- If we cover all input patterns,
  we can construct a truth table!

Many Problems are Too Large

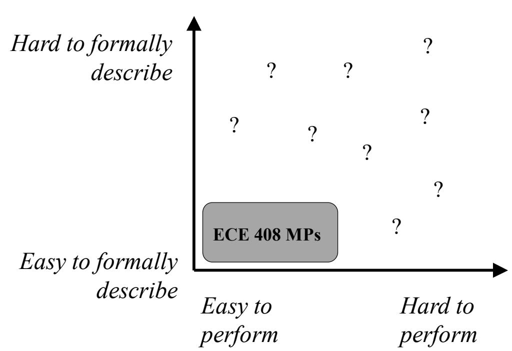
- The logic formulation of a 32x32-pixel (small) image recognition problem involves
  - 1024*8 bit input,
  - which will have a truth table of 2^{8196} entries
- If we managed to collect and label 1 billion ($\sim 2^{32}$) images as training data
  - We cover only $2^{32} / 2^{8196} = 1/ 2^{8164}$ of the truth table
  - Solution - learning processes that exploits features

Features in our logic example

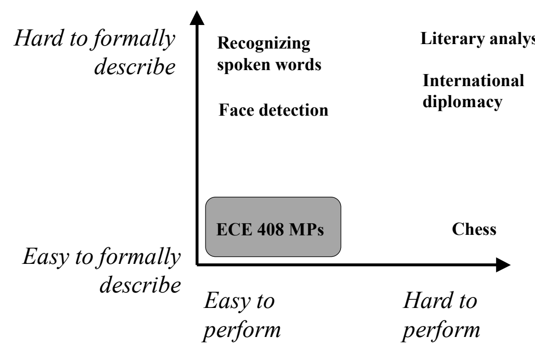
Input			output
a	b	c	
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Feature 1: bit patterns with odd number of 1's result in output 1
Feature 2: bit patterns with even number of 1's result in output 0

Types of Problems



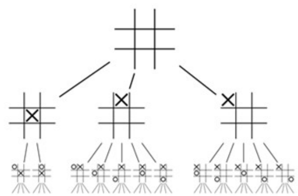
Types of Problems



Algorithm complexity, parallelism, and data bandwidth)

Chess as an AI Success (1)

- Easy to formalize
  - 64 locations, 32 pieces
  - Well-defined, allowable moves
- Score each leaf in a tree of possible board positions
- Proceed down path that results in best position



2-ply game tree for tic-tac-toe

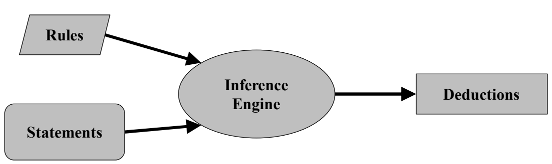
Chess as an AI Success (2)

Deep Blue defeated Gary Kasparov in 1997

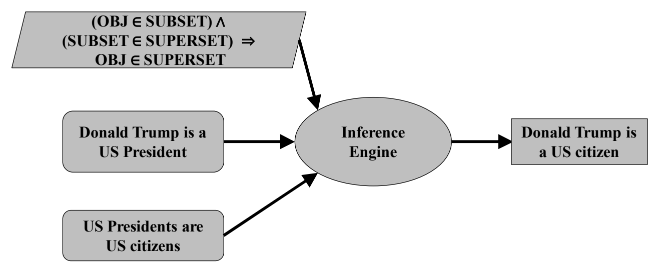
- Hard to perform
  - ~30 legal moves per position
  - 1,015 moves for 10-ply lookahead
  - 30 years of compute at 1M positions/sec
- Heuristics, pruning, parallel search, fast computers

Cyc: Extending Rule-based Systems to the Real World

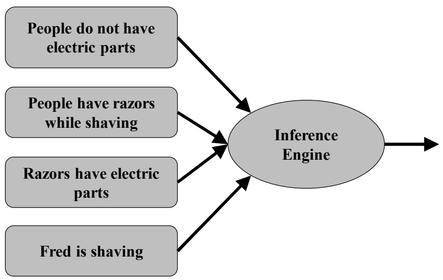
- Comprehensive ontology and knowledge base of common sense
- Cyc reasons about formal statements about the world



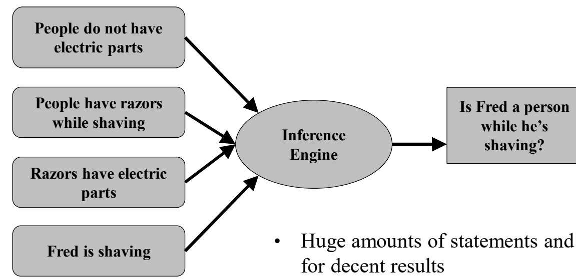
Cyc: A Simple Example



Cyc: FredWhileShaving

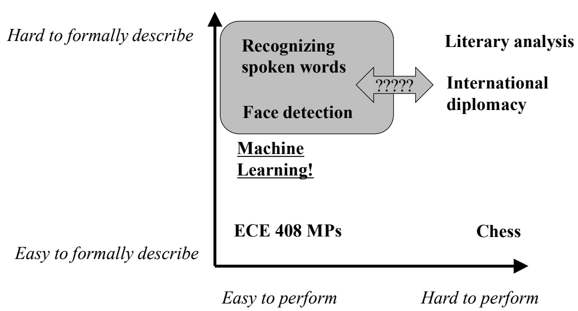


Cyc: FredWhileShaving



- Huge amounts of statements and rules for decent results
- Cannot learn new rules or statements on its own

Types of Problems



The "Machine Learning" Approach

Challenge

Hard to formalize the problem.

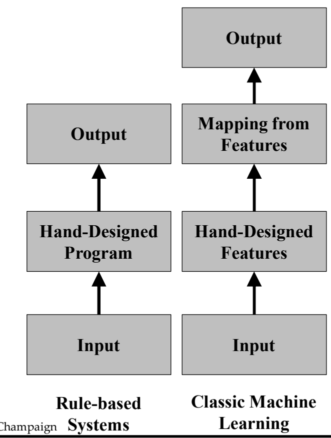
Solution

Don't formalize the problem.

Let the machine learn from experience.

Classic Machine Learning

- Humans choose features
- Learn how features are associated with outputs



25

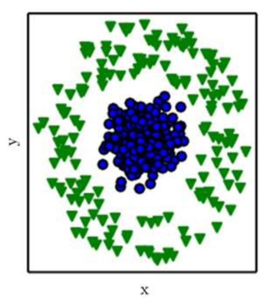
You may have heard of…

- Naïve Bayes:
  features as independent contributors to output
- Logistic Regression:
  – learn how to weight each feature's contribution to output,
  – usually through gradient descent*

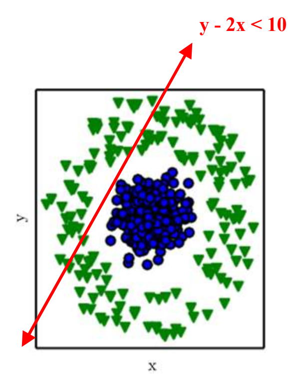
*more on this topic later in these slides

26

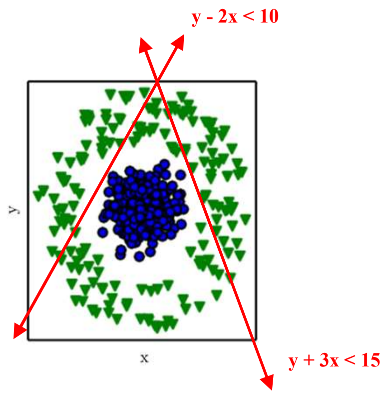
Data Representation is important!



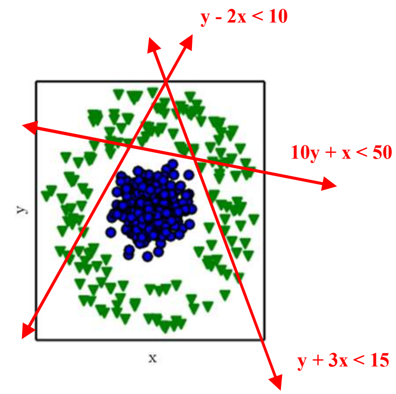
27



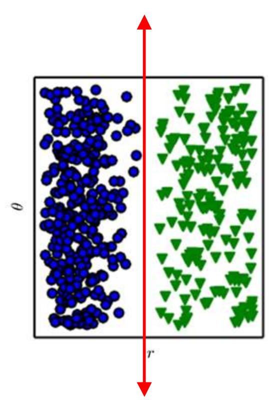
28



29



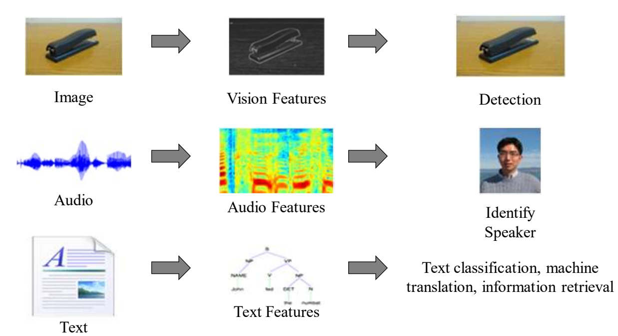
30



$\Theta = \mathrm{arctan}(\ y\ /\ x\ )$
$r = \mathrm{sqrt}(\ x^2 + y^2\ )$

31

Different Features for Different Tasks



32

Which Data Features are Relevant

- Detecting a car in an image
- Cars have wheels ➡ presence of a wheel?
- Can we describe pixel values that make up a wheel?
  - Circle-shaped?
  - Dark around perimeter?

Which Data Features are Relevant

- Detecting a car in an image
- Cars have wheels ➡ presence of a wheel?
- Can we describe pixel values that make up a wheel?
  - Circle-shaped?
  - Dark around perimeter?
- But what about?
  - Occlusion, perspective, shadows, white-walled tires, …

Identify Factors of Variation that Explain Data

- Unobserved objects or forces that affect observed quantities
- Mental constructs that provide simplifying explanations or inferred causes
- Ex: speech
  - Age, sex, accent, words being spoken
- Ex: car
  - Position, color, angle of sun
- Many factors influence each piece of observed data

Representation Learning Approach

Challenge

Which data features are relevant?

Solution

Learn the features too!

(Looking ahead)

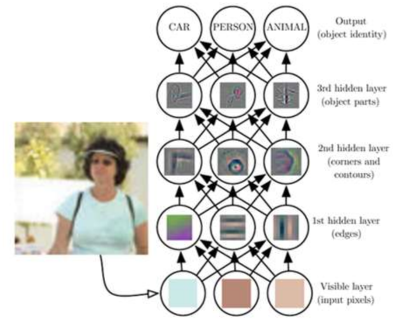
Deep Learning: a deep hierarchy of features

Machine Learning

- Ability to acquire knowledge by extracting patterns from data

Deep Learning

- A type of representation learning
- Representations expressed in terms of other representations



Deep Learning Approach

Challenge	**Solution**
Hard to formalize the problem?	Don't formalize the problem Let the machine learn from experience
Which data features are relevant?	Hierarchy of concepts to capture simple and complicated features Learn the hierarchy too!

Summary