

Two big challenges in machine learning

LÉON BOTTOU

FACEBOOK AI RESEARCH

ICML 2015 - LILLE

Machine Learning in 2015

- An established academic discipline
 - that attracts bright students.
- Pervasive applications
 - all over the Internet
 - and sometimes in the real world.
- Massive investments.
- Massive returns.



more logos in your ICML booklet...

Machine Learning in 2015

- An established academic discipline
 - that attracts bright students
- Pervasive applications
 - all over the world
 - an integral part of the real world.
- Massive investments.
- Massive returns.

Increased ambitions



more logos in your ICML booklet...

Machine Learning in 2015

- An established industry
 - that is growing rapidly
- Pervasive applications
 - all over the world
 - and in the real world.
- Mass adoption.
- Massive returns.

Challenges of a new kind



more logos in your ICML booklet...

Challenges of a new kind

1. Machine learning disrupts software engineering.

→ Challenging our impact on real world applications.

2. Our experimental paradigm is reaching its limits.

→ Challenging the speed of our scientific progress.

3. Elements of a solution

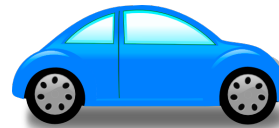
1

Machine Learning
disrupts software
engineering

Engineering complex artifacts

Counting parts down to the smallest screw

- Automobile: ~ 30,000 parts
- Airplane: ~ 3,000,000 parts



Specifications and replication

- The engineer does not design the alloy and the threads of each screw. Standard screws come with known specifications (size, strength, weight, ...)
- Many parts and many assemblies are identical.

Abstraction in engineering

Thinking with the specifications instead of the details.

- part specifications instead of part details.
- assembly specifications instead of assembly details.



**Leveraging the
work of others**

Abstractions leak!

Sometimes one needs to look more closely.

- Example: engine cooling assembly can have complicated heat inertia.
- Example: global resource allocation (cost, weight, etc.)

Abstraction leaks **limits the complexity** of what we can build.

Abstraction in mathematics and logic

Pure abstraction belongs to mathematics and logic

- When we use a theorem, we do not need to revisit its proof.
- Mathematical abstractions do not leak.

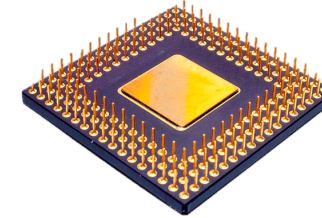
Computer programs are similar to mathematical proofs (in theory)

- **Design with contracts** – knowing the specification of a module is enough.
- In practice, there are abstraction leaks (but less.)

Computer programs

The closer to logic, the more complex the artifacts

- Automobile: ~ 30,000 parts
- Airplane: ~ 3,000,000 parts
- Processor: ~ 3,000,000,000 transistors



(note : counts include many identical parts and assemblies.)

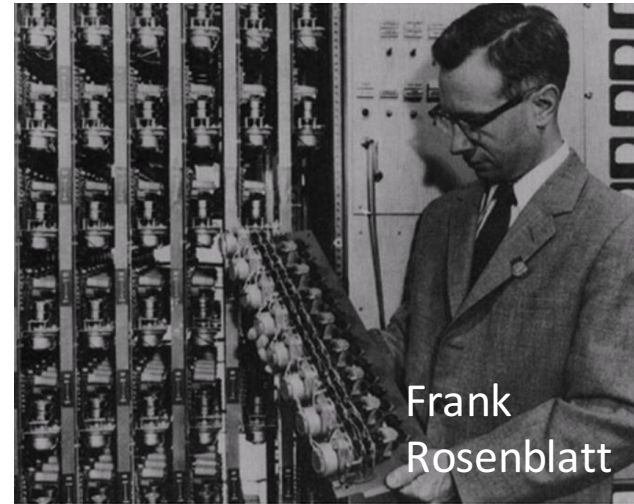
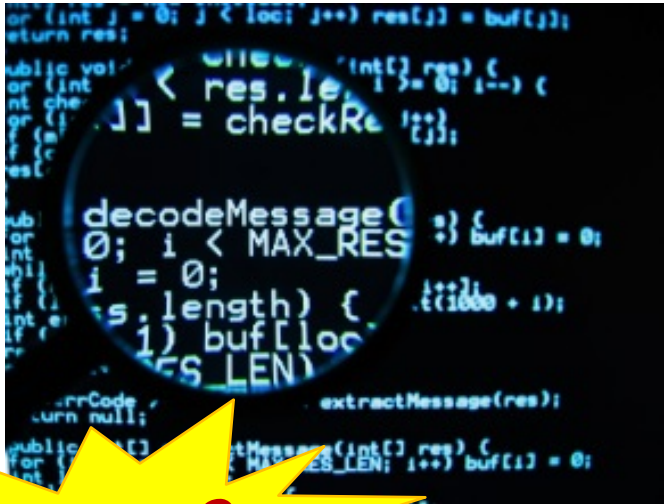
- MS Office: ~ 40,000,000 LOCs
- Facebook: ~ 60,000,000 LOCs
- Debian : ~ 400,000,000 LOCs



(note: different metric: lines of codes are rarely identical.)

Programming versus learning

Two conceptions of computing



Winner?

Digital computers (that one programs) are everywhere.

Programming makes it easy to leverage the work of others.

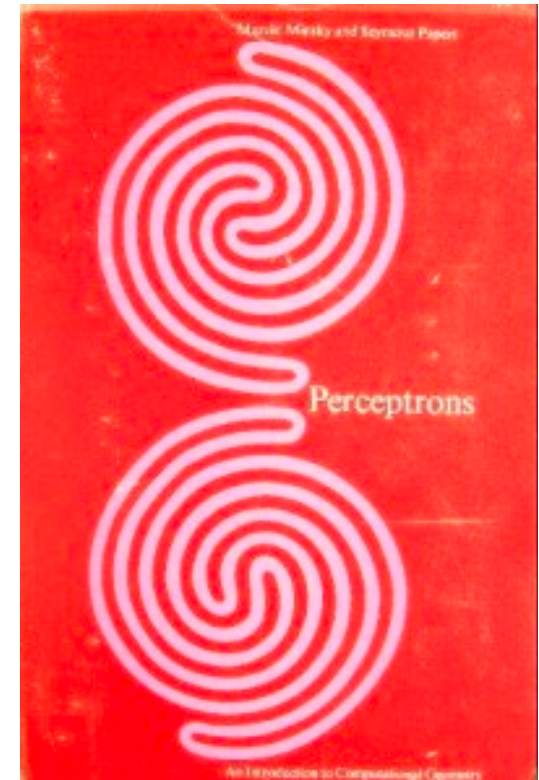
Programming versus learning

Programming has real advantages

Minsky and Papert, *Perceptrons*, 1968.

Remark

This book is not just about Rosenblatt's perceptron!
What Minsky and Papert call an “order-k perceptron” covers most machine learning models, including the convolutional neural networks that are changing computer vision.



Connectedness

Is a shape made of a single connected component?

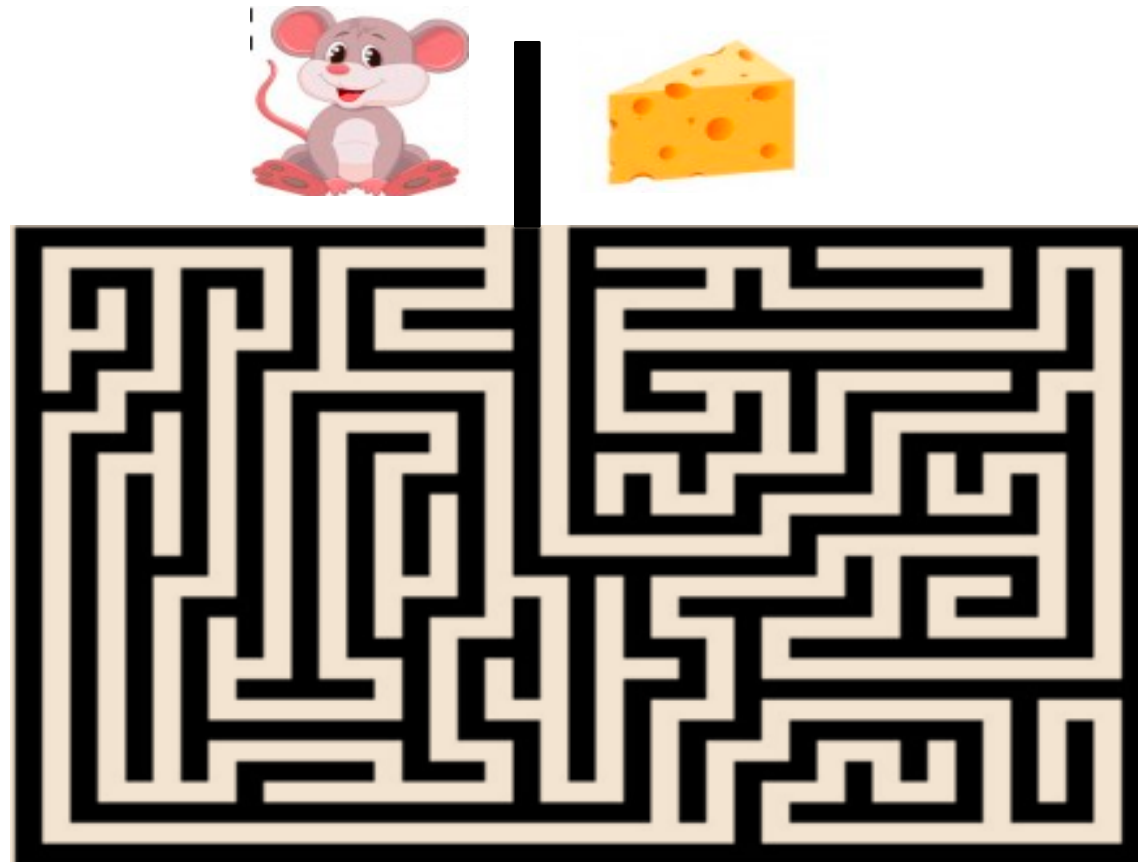
M&P prove that a small-order perceptron cannot say, but a **simple algorithm can**.

Theorem 9.2: For any ϵ there is a 2-symbol Turing machine that can verify the connectedness of a figure X on any rectangular array R , using less than $(2 + \epsilon) \log_2 |R|$ squares of tape.

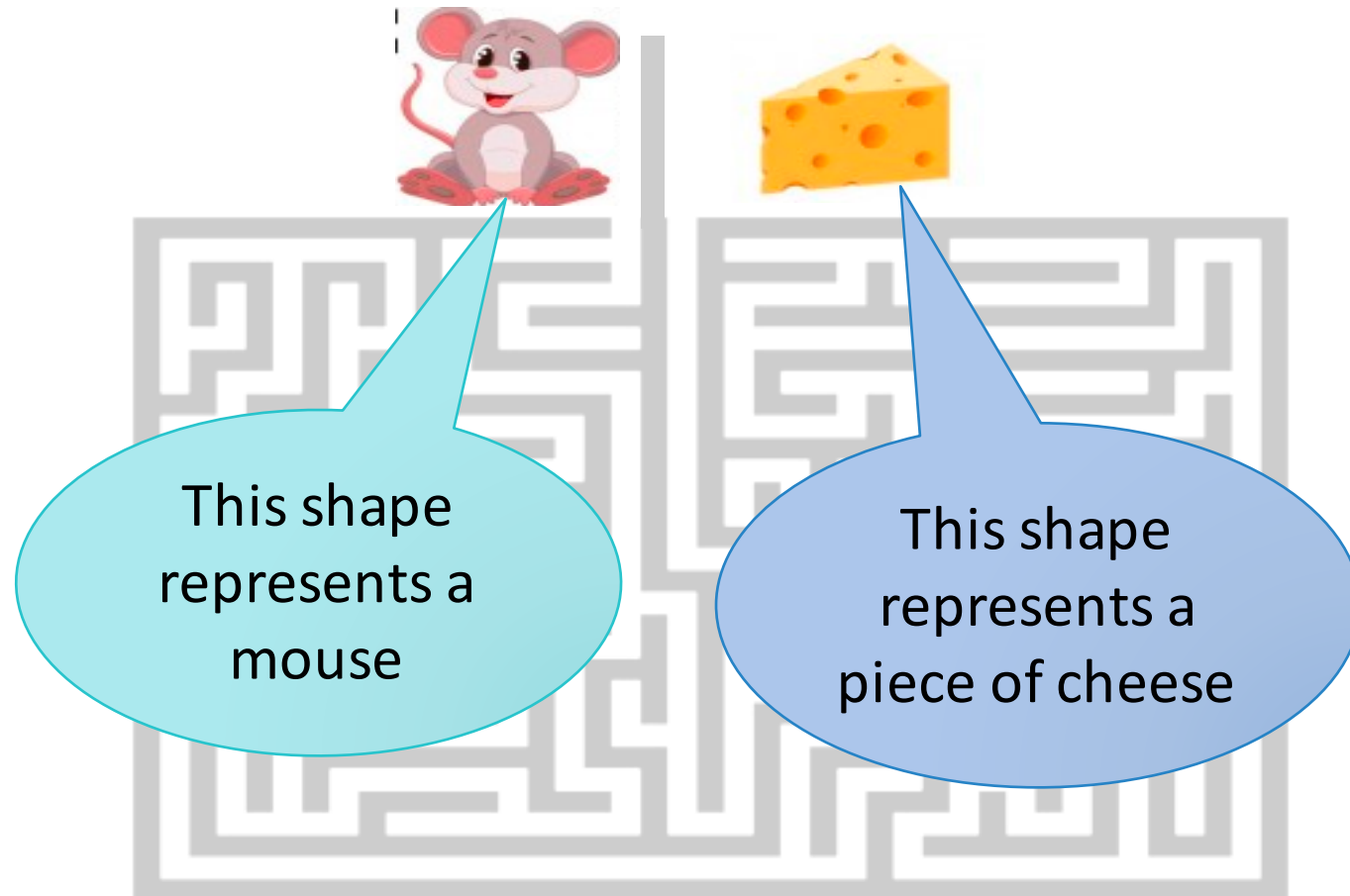


Such an algorithm fulfils a clear contract: verifying connectedness.

Is connectedness easy for us?



What is easy for us?



Why is learning gaining momentum?

- Since “connectedness” has a **clear mathematical specification**, we **can envision provable algorithms**.
- Since “mousiness” and “cheesiness” **do not have such a specification**, we cannot envision provable algorithms.

We must rely on

- **heuristic specifications**
- or **learning techniques**.



“Connectedness?”



“Mousiness?”



“Cheesiness?”

Why is learning gaining momentum?

- We must rely on
 - heuristic specifications (e.g. rules)
 - or learning techniques.



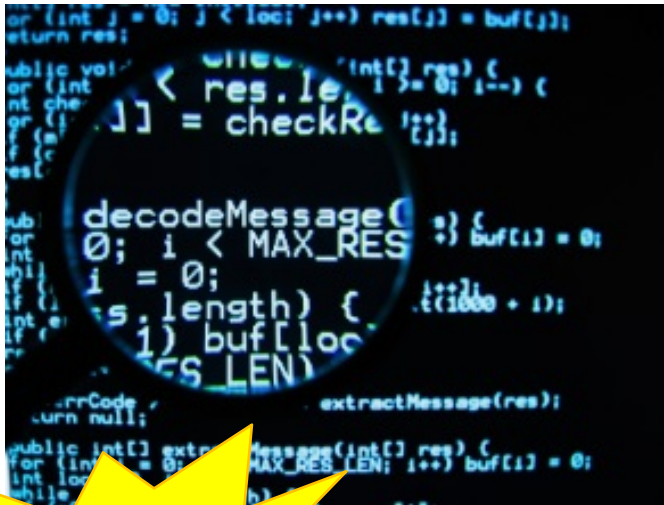
Big data and big computing power

When the volume of data increases

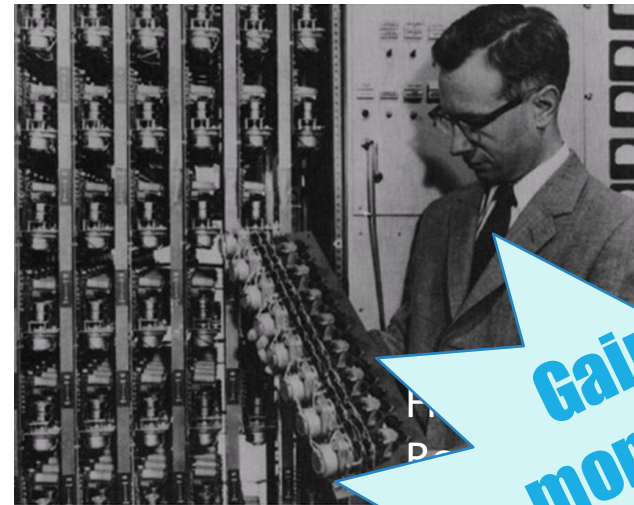
- defining heuristic specifications **becomes harder**.
- training learning systems **becomes more effective**.

Programming versus learning

Two conceptions of computing



Everywhere



**Gaining
momentum
as we speak**

Integrating machine learning in large software systems

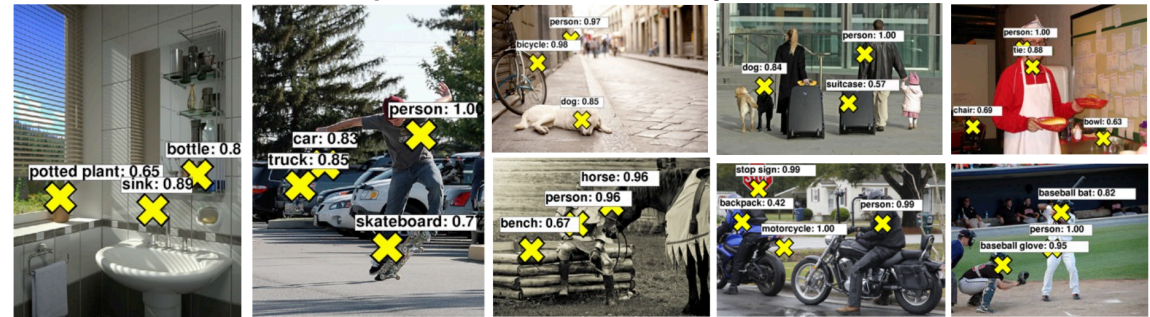
Machine learning systems must* work with digital software because digital computers are everywhere.

- Use trained models as software modules.
- Use learning algorithms as software modules.

* at least until we solve AI...

Trained models as software modules

“DeepVisotron™, SM, ® detects 1000 object categories with less than 1% errors.”



What is the nature of the contract?

- This does not mean that one rolls a dice for each picture.
- This statement refers to a specific testing set.
The error guarantee is lost if the image distribution changes.

Weak contracts

- A smart programmer makes an inventive use of a **sorting routine**.
- The sorting routine **fulfils** the contract by **sorting** the data (however strange.)
- The code of the smart programmer **does what was intended**.



- A smart programmer makes an inventive use of a **trained object recognizer**.
- The object recognizer receives data that **does not resemble the testing data** and outputs nonsense.
- The code of the smart programmer **does not work**.

Weak contracts

- A smart programmer makes an inventive use of a **sorting routine**.
- The sorting routine **fulfils** the contract by **sorting** the data (however strange.)
- The code of the smart programmer **does what was intended**.



- A smart programmer makes an inventive use of a **trained object recognizer**.
- The object recognizer receives data that **does not resemble the testing data** and outputs nonsense.
- The code of the smart programmer **does not work**.

Collecting new data and retraining may help ... or may not!

Learning algorithms as software modules

Machine Learning: The High-Interest Credit Card of Technical Debt

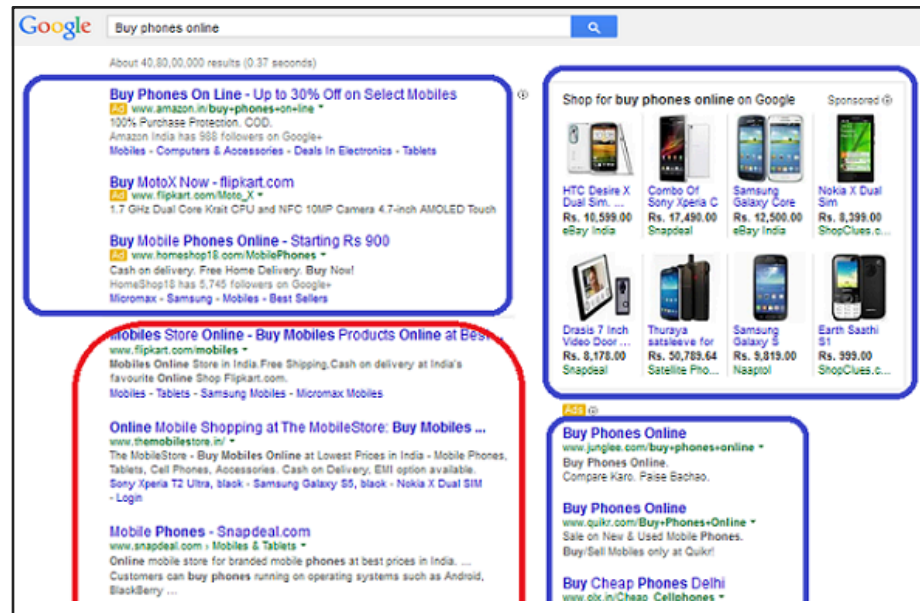
D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov,
Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young
{dsculley, gholt, dgg, edavydov}@google.com
{toddphillips, ebner, vchaudhary, mwyong}@google.com
Google, Inc

2.1 Entanglement

From a high level perspective, a machine learning package is a tool for mixing data sources together. That is, machine learning models are machines for creating entanglement and making the isolation of improvements effectively impossible.

Example 1

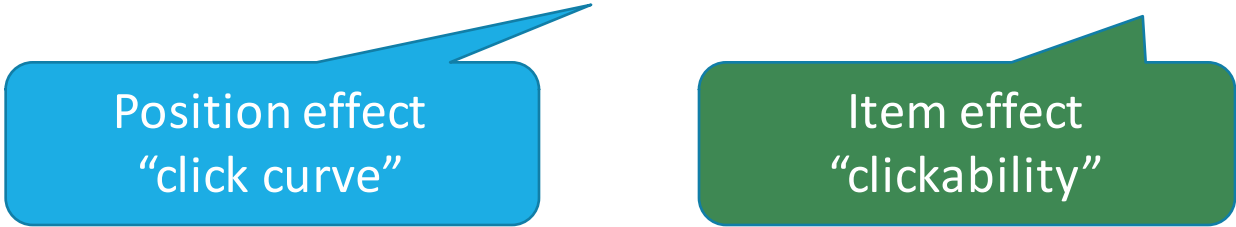
Click curves



Many web sites offer lists of items
(search results, recommendations, ads ...)
and select them by modeling click probabilities.

A common click model

$$P(\text{click}|\text{context}, \text{item}, \text{pos}) = F(\text{pos}) \times G(\text{context}, \text{item})$$



Position effect
“click curve”

Item effect
“clickability”

Why separating position effect and item effect?

- Click probabilities of this form allow a greedy placement algorithm:
 1. Sort items by clickability $G(\text{context}, \text{item})$.
 2. Allocate them greedily to the best positions.

A common click model

Why separating position effect and item effect?

- Click probabilities of this form work well in auction theory results.
- There are easy ways to estimate $F(position)$ and $G(context, item)$.

How incorrect is this model?

- All ML models are incorrect.
- Model misspecification should only degrade the performance gracefully.
- How bad can things go?

How bad things can go?

- Assume there are only two contexts C1 and C2 and two positions P1,P2
- Assume the true click probability is

$$P^*(click|context,item,pos) = F^*(context,pos) \times G^*(context,item)$$

F^*	P1	P2
C1	0.8	0.2
C2	0.6	0.4

G^*	R1	R2	R3	R4
C1	0.12	0.10	0	0
C2	0	0	0.12	0.10

- Our model estimates some intermediate click curve $F(pos)$

How bad can things go in context C1 ?

True position effect steeper than estimated.

- $F(P1) < F^*(C1, P1) \Rightarrow$ overestimate clickability of items shown in P1
- $F(P2) > F^*(C1, P2) \Rightarrow$ underestimate clickability of items shown in P2

Combined with the greedy placement algorithm.

- After retraining, whatever we placed in position P1 looks better than it really is. Therefore we **keep placing it in position P1**.
- After retraining, whatever we placed in position P2 looks worse than it really is. Therefore we **keep placing it in position P2** (or stop showing it.)

How bad can things go in context C1 ?

True position effect steeper than estimated.

- $F(P1) < F^*(C1, P1) \Rightarrow$ overestimated position shown in P1
- $F(P2) > F^*(C1, P2) \Rightarrow$ underestimated position shown in P2

Combined with

- After retraining, whatever we placed in position P1 looks better than it really is. Therefore we **keep placing it in position P1**.
- After retraining, whatever we placed in position P2 looks worse than it really is. Therefore we **keep placing it in position P2** (or stop showing it.)

Exploration issue.

- New items eligible for context C1 may have a hard time competing with the current winner.

How bad can things go in context C2 ?

True position effect less steep than estimated.

- $F(P1) > F^*(C1, P1) \Rightarrow$ underestimate clickability of items shown in P1
- $F(P2) < F^*(C1, P2) \Rightarrow$ overestimate clickability of items shown in P2

Combined with the greedy placement algorithm.

- After retraining, whatever we placed in position P1 looks worse than it really is. Therefore we might **move it down to position P2**.
- After retraining, whatever we placed in position P2 looks better than it really is. Therefore we might **move it up position P1**.

How bad can things go in context C2 ?

True position effect less steep than estimated.

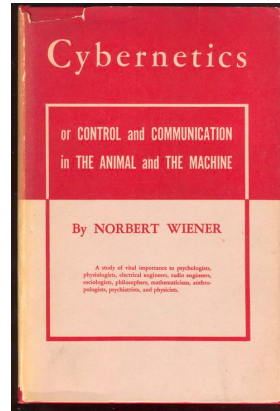
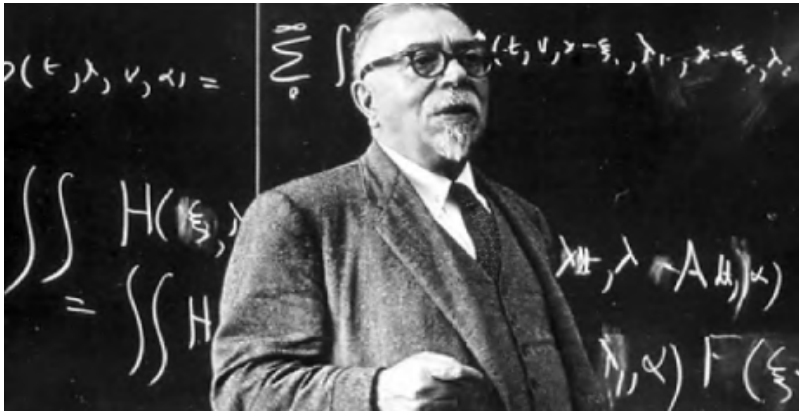
- $F(P1) > F^*(C1, P1) \Rightarrow$ underestimation shown in P1
- $F(P2) < F^*(C1, P2) \Rightarrow$ overestimation shown in P2

Combined with

- After retraining, whatever we placed in position P2 looks better than it really is. Therefore we might move it up position P1.
- After retraining, whatever we placed in position P1 looks worse than it really is. Therefore we might move it down position P2.

This is an expensive oscillator!
■ Items alternate positions whenever we retrain the click probability model.
■ Damping the oscillation leads to the same problem as in context C1.

Feedback loops in machine learning

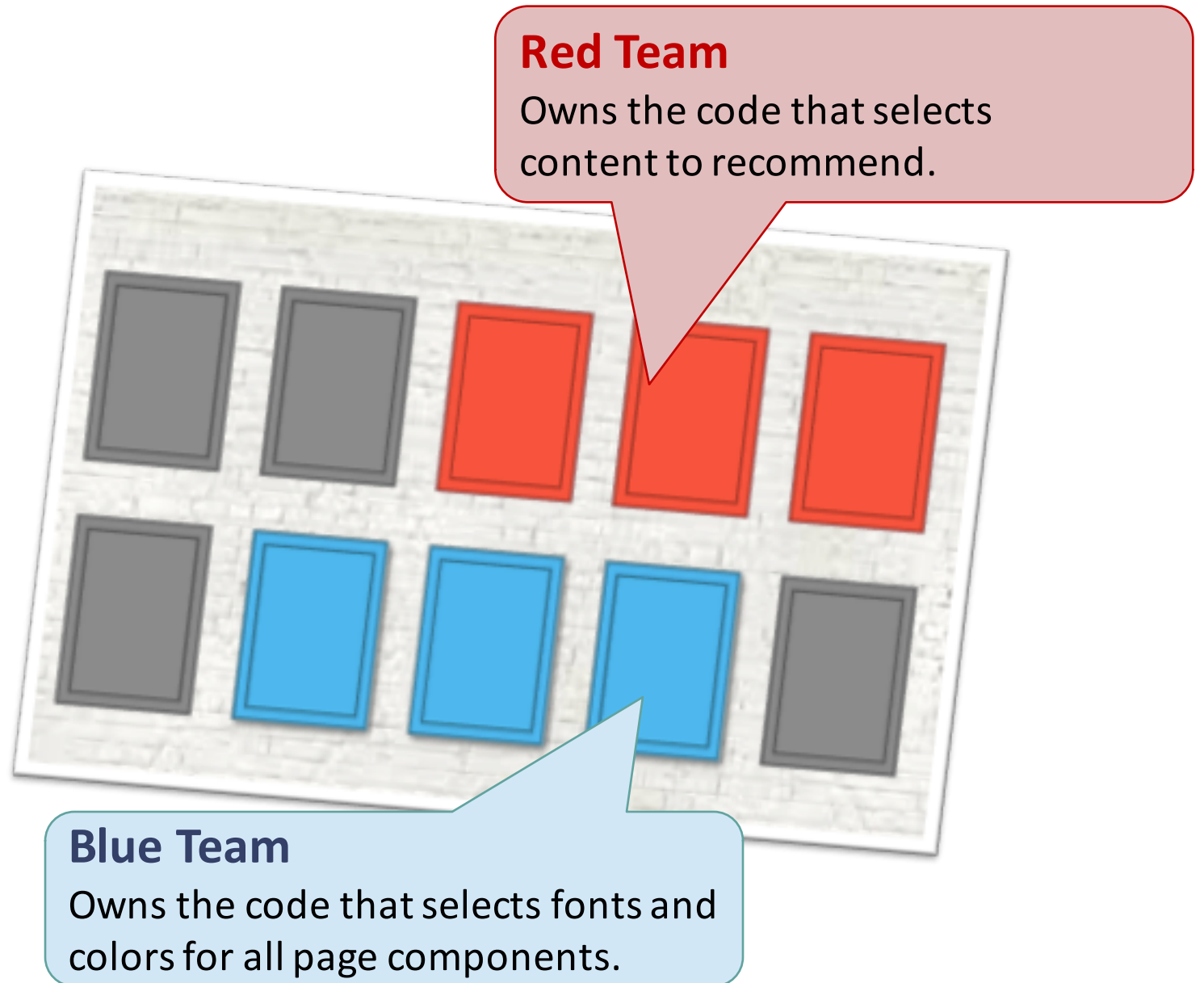


Norbert Wiener,
Cybernetics, 1948

- “Information (signal) feedback loops are everywhere.”
“They are central to adaptation and learning...”
- See (Bottou et al., JMLR 2013) for a possible treatment of causal loops.

Example 2

Team work



Red team

The red team knows the bandit literature

- There are lots of potential recommendations to select from.
- One can only model the click probabilities of the recommendations that have been shown often enough in each context.
- Therefore the red team code performs ϵ -greedy exploration, showing a small proportion of random recommendations.
- Although exploration has a cost, such the system cannot discover the most relevant recommendations without it.

Blue team

The blue team also knows machine learning.

- The job of the blue team is to emphasize things that the user will like.
- The blue team code runs a click probability model to spot what users like.
- The click probability model easily discovers that some of the recommendations generated by the red team are not very good.
- Therefore these recommendations will be shown with a smaller font.

Blue team

The blue team also knows machine learning.

- The job of the blue team is to emphasize the items that the user will like.
- The blue team could also emphasize items that are not what users like.
- The click probability of the recommended items is very good.
- Therefore the blue team displays exploration items in a smaller font.

All the exploration items are displayed with a very small font.
→ They receive less clicks.

Red team

The red team is very content!

- Analysis of the exploration traffic shows that there isn't much to improve.
→ they can safely reduce the exploration level!
- Whenever they try new features in the click model, randomized testing shows a reduction in performance
→ it is amazing, but it seems they got their model right the first time!

Red team

The red team is very content!

- Analysis of the exploration traffic shows that there isn't much to improve.
→ they can safely reduce the exploration level

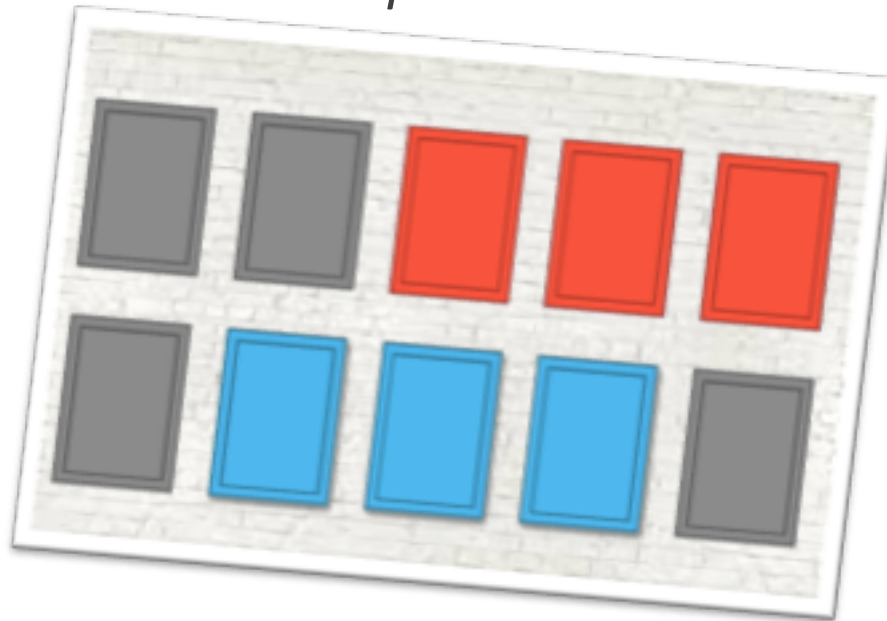
- Whenever they try new features, they get a randomized set of users.
→ it is not possible to know if they got their model right the first time!

None of this is true.

Paying attention

“A manager of the two teams should have paid more attention.”

“The software architect should have paid more attention.”



Micro-management does not work because it does not scale...

beyond two teams



Conclusion of part 1

Machine learning in large software systems

- Use trained models as software modules.
 - problematic because trained models offer weak contracts.
- Use learning algorithms as software modules.
 - problematic because the output of the learning algorithm depends on the training data which itself depends on every other module.

Working around these difficulties could save billions...

2

Our experimental
paradigm is reaching
its limits

Minsky and Papert 1968, F.A.Q.

13.5 Why Prove Theorems?

Why did you prove all these complicated theorems? Couldn't you just take a perceptron and see if it can recognize $\psi_{\text{CONNECTED}}$?

No.

Are we dealing with
an exact science (like mathematics) or
an empirical science (like physics)?



Essential epistemology

	Exact sciences	Experimental sciences	Engineering
Deals with	Axioms & Theorems	Facts & Theories	Artifacts
Truth is	Forever	Temporary	It works.
Examples	Mathematics C.S. theory	Physics Biology	Lots...

Essential epistemology

– Exact sciences

	Exact sciences	Experimental sciences	Engineering
Deals with	Axioms & Theorems	Facts & Theories	Artifacts
Truth is	Forever	Temporary	It works.
Examples	Mathematics C.S. theory	Physics Biology	C.S.

Exact sciences

- Deal with axioms and theorems.
- Axioms are neither true or false.
Theorems are true once proven.
They remain true forever.
- Examples
 - Mathematics
 - Theoretical C.S. (e.g., algorithms, type theory, ...)
 - Theoretical Physics (e.g., string theory)

Essential epistemology

– Experimental sciences

	Exact sciences	Experimental sciences	Engineering
Deals with	Axioms & Theorems	Facts & Theories	Artifacts
Truth is	Forever	Temporary	It works.
Examples	Mathematics C.S. theory	Physics Biology	C.S.

*The word “true”
has a different
meaning.*

Experimental sciences

- Deal with facts and theories.
- Facts are true when they can be replicated.
Theories are true when they predict the facts.
New facts can invalidate theories
that were previously considered true (K. Popper)
- Examples
 - Physics, Biology,...
 - Social sciences, Experimental psychology,...

Essential epistemology

– Engineering

	Exact sciences	Experimental sciences	Engineering
Deals with	Axioms & Theorems	Facts & Theories	Artifacts
Truth is	Forever	Temporary	It works.
Examples	Mathematics C.S. theory	Physics Biology	C.S.

Engineering

- Deals with artifacts.
- A claim is true when the artifact works, (the artifact fulfils a contract.)
- Examples
 - Mechanical engineering, ...
 - Nuclear engineering, ...
 - Computer science.

Machine learning?

What is machine learning?

- An engineering discipline?
- An exact science?
- An experimental science?

Yes: applications abound.

Yes: statistical/algorithmic learning theory.

Yes: papers often report on experiments.

This is all good

Machine learning?

What is machine learning?

- | | |
|------------------------------|---|
| ■ An engineering discipline? | Yes: applications abound. |
| ■ An exact science? | Yes: statistical/algorithmic learning theory. |
| ■ An experimental science? | Yes: most papers include experiments. |

This is all good ... as long as we don't mix the genres.

- *"My software works really well, therefore my theory is true."*
- *"This is NP-complete, therefore this experiment is wrong."*



ML as an experimental science

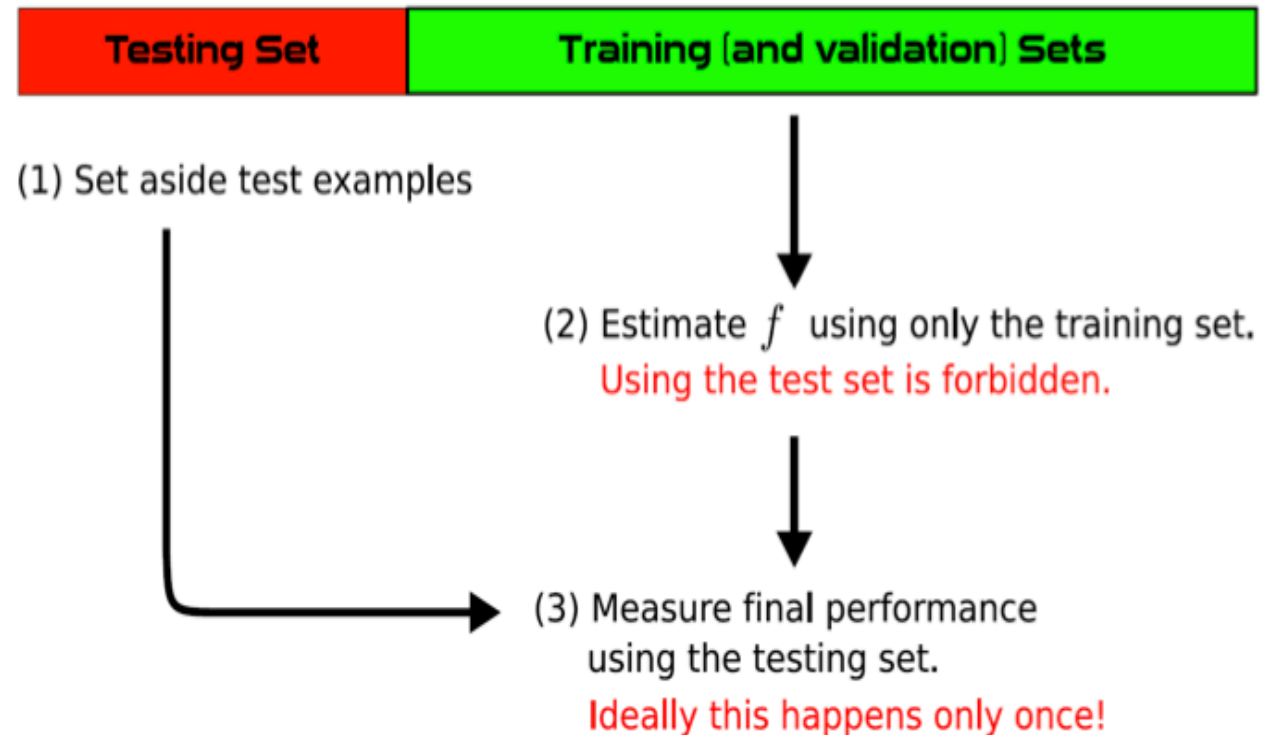
- Some problems of interest do not have a clear specification. They are not amenable to provable algorithms.



- Machine learning replaces the missing specification by lots of data.
- Important aspects of the data cannot be described by a compact mathematical statement (otherwise we have a specification!)
- Therefore experimentation is necessary!

ML as an experimental science

Progress during the last few decades has been driven by a single experimental paradigm!



ML as an experimental science

Progress during the last few decades has been driven by a single experimental paradigm!

This is unusual for an experimental science!

Physicists, biologists, experimental psychologists, ..., must work a lot harder to design experiments and interpret their results.

Where does the data come from?

How to improve the performance of a ML application?

- Get better algorithm? +
- Get better features? +++
- Get better data? ++++++

Data collection is hard work

- The data distribution must match **the operational conditions**.
- Rely on **manual data curation** to avoid dataset bias.

The impact of dataset bias

Training/testing on biased datasets gives unrealistic results.

- E.g. : Torralba and Efros, *Unbiased look at dataset bias*, CVPR 2011.

task	Test on:		SUN09	LabelMe	PASCAL	ImageNet	Caltech101	MSRC
	Train on:							
“car” classification	SUN09		28.2	29.5	16.3	14.6	16.9	21.9
	LabelMe		14.7	34.0	16.7	22.9	43.6	24.5
	PASCAL		10.1	25.5	35.2	43.9	44.2	39.4
	ImageNet		11.4	29.6	36.0	57.4	52.3	42.7
	Caltech101		7.5	31.1	19.5	33.1	96.9	42.1
	MSRC		9.3	27.0	24.9	32.6	40.3	68.4
	Mean others		10.6	28.5	22.7	29.4	39.4	34.1

Increased ambitions: Big Data

Too much data for manual curation

- Big data exists because **data collection is automated**.
- Nobody checks that the data distribution matches the operational conditions of the system.

→ All large scale datasets are **biased**.

Training/testing on a big dataset can give unrealistic results.

Increased ambitions: A.I.

Statistical machine learning

- Building a classifier that works well under a certain distribution.

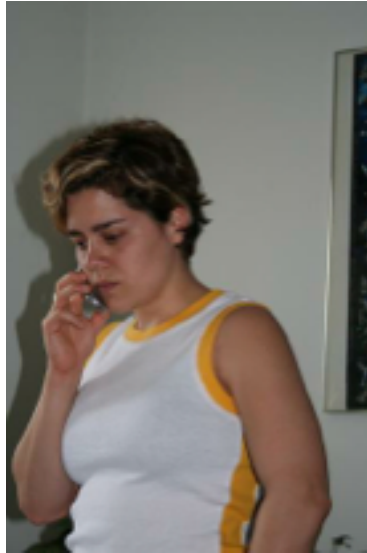
Machine learning for artificial intelligence

- Building a classifier that recognizes a “concept”.
- Concepts exist independently of the pattern distribution.
- Training data will never cover the whole range of possible inputs.

In fact, a system that recognizes a “concept” fulfils a stronger contract than a classifier that works well under a certain distribution.

Concepts \neq Statistics

Example: detection of action “giving a phone call”



(Oquab et al., CVPR 2014)

Concepts \neq Statistics

Computer vision is not a statistical problem



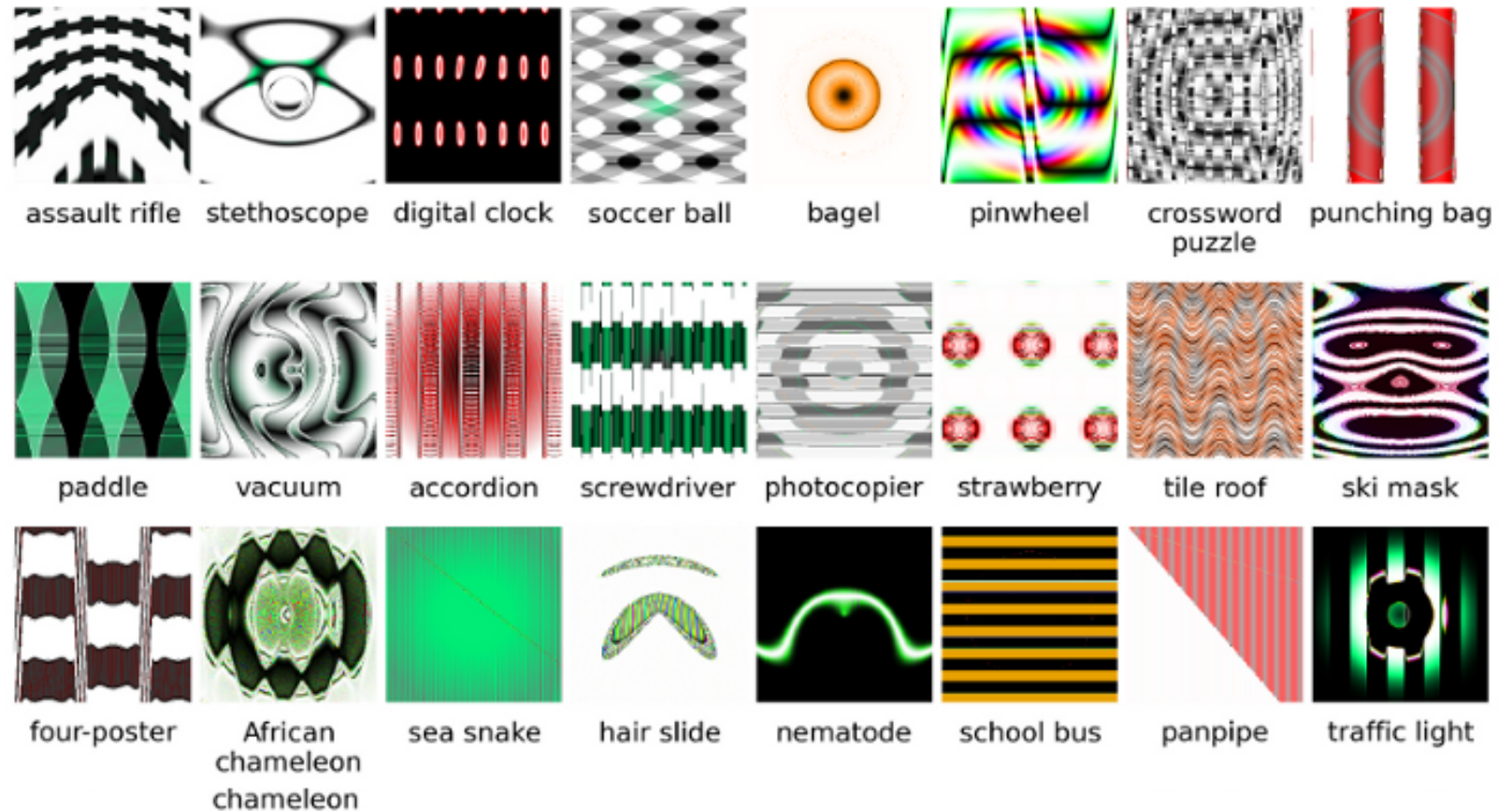
Car examples in ImageNet



Is this less of a car
because the context is wrong?

Concepts \neq Statistics

Convolutional
networks
can be
fooled.



(Nguyen et al, CVPR 2015)

Concepts \neq Statistics

A herd of giraffes walk down the street in the middle of some trees.



From a talk of (Zitnick 14)

Caption generation

- Eight independent papers in 2014. According to the BLEU score, some of these systems perform better than humans...
- They met in Berkeley in January 2015.
- Evaluation is very difficult.
- Example of difficulty : caption generation vs retrieval

Conclusion of part 2

Training/testing only goes so far...

Training/testing is an unusually convenient experimental paradigm...

- it makes experimentation easier than in other sciences
- it has played a role in the fast progress of ML.

...but may not fulfil our increased ambition (big data, AI.)

- the end of an anomaly?

Working around this could save decades.

3

Elements
of a solution

1. Process challenges

Challenges of a new kind

- We are used to face technical challenges.
- These are “*process challenges*.”

1. Process challenges

Engineering process

- There is a rich literature about the software engineering process.
- What should be the machine learning engineering process?

1. Process challenges

Scientific process

- We cannot **solely** rely on training/testing experiments.
- We can understand more things with ad hoc experiments.

Example in question answering: the “BABI” tasks (Bordes et al., 2015)

- questions that require combining one/two/more factoids.
 - questions that require spatial/temporal reasoning.
 - etc.
- What are the best practices in other experimental sciences?

1. Process challenges

A constructive proposition

- Machine learning papers rarely **show the limits of their approach**.
 - For fear that reviewers will use them against their work.
- Reviewers should instead **demand** that authors discuss these limits.
 - Additional credits if they illustrate these limits with experiments.
 - This would make the papers more informative...
 - and would therefore facilitate scientific progress.

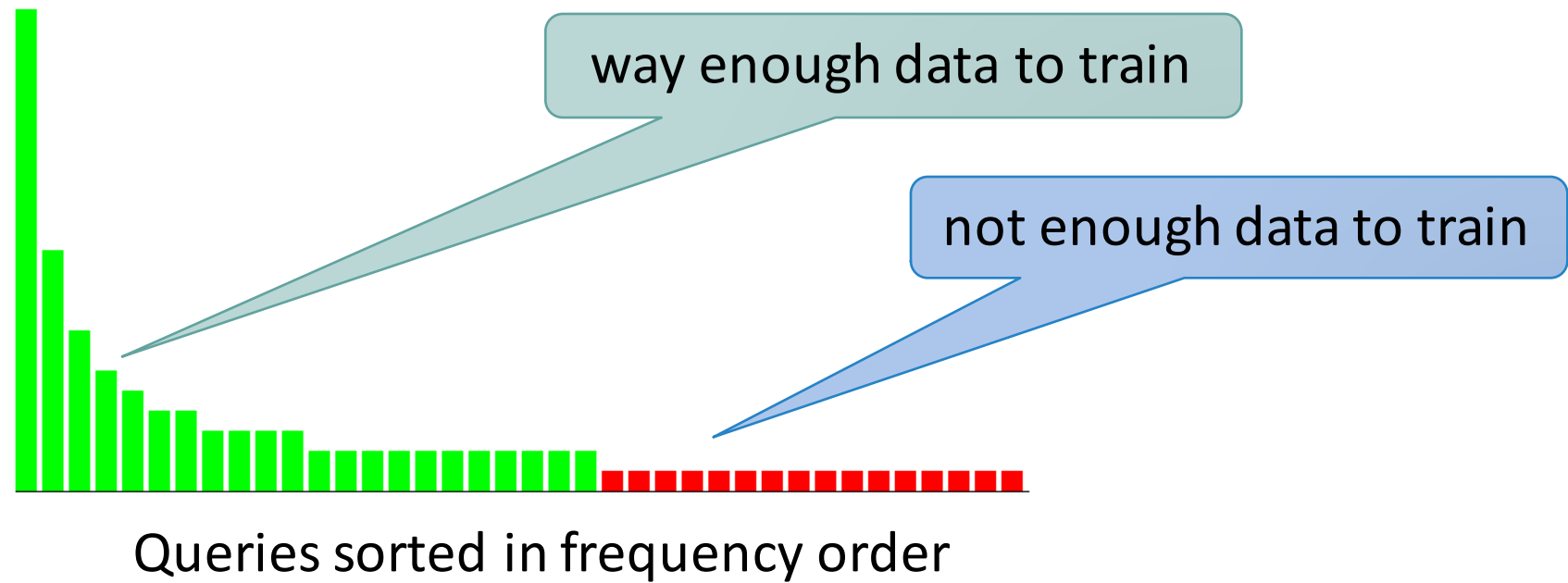
2. ML with stronger contracts

How to better resist distribution shifts?

- Optimize measure of **coverage** instead of **average accuracy**.
 - Selective classification (El Yaniv's, ...)
 - KWIK (Li & Littman)
 - Conformal learning (Vovk, ...)
- Very interesting properties when the data is Zipf distributed...

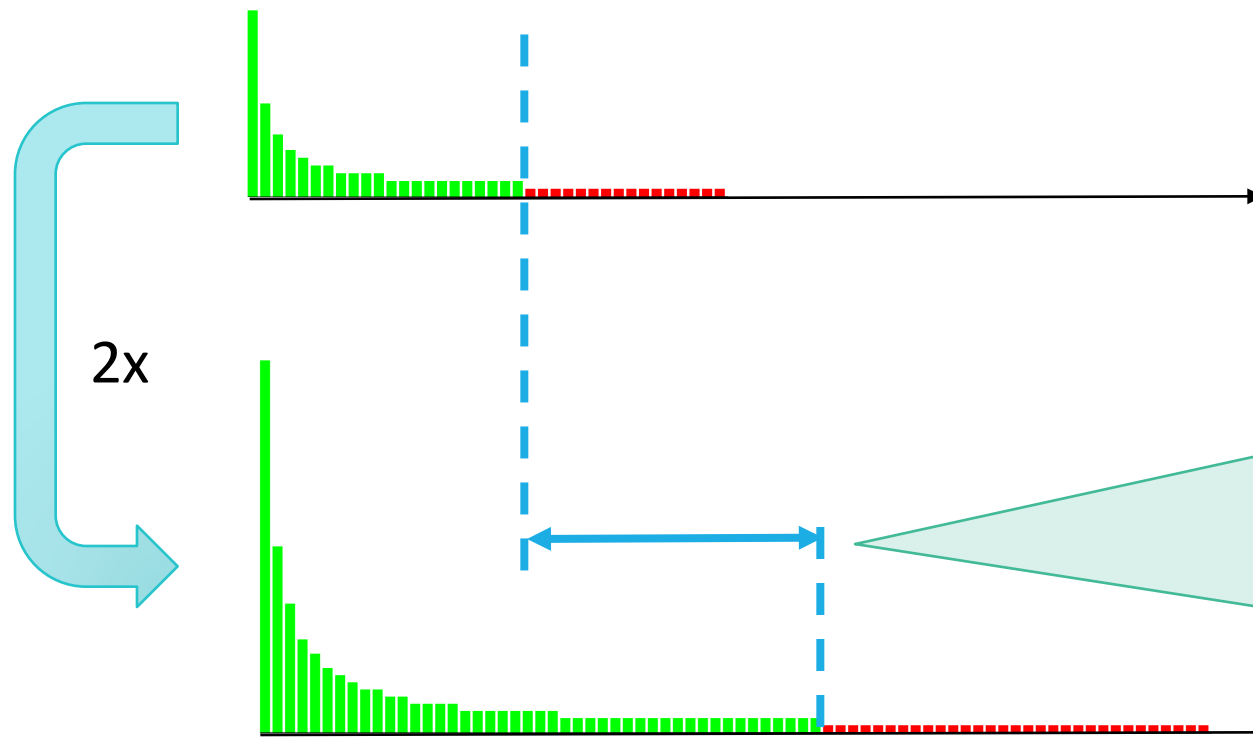
2. ML with stronger contracts

Zipf distributed patterns



2. ML with stronger contracts

Doubling the size of a Zipf distributed dataset.



Diminishing returns
for average accuracy
improvements.

**No diminishing
returns**
on number of
queries for which we
can learn correct
answers.

3 How to package the work of others

- Digital computers : “software”
- Learning machines :
 - Trained module as a software component?
 - ✗ Trained components only offer “weak contracts”.
 - Training software?
 - ✓ If you can get the training data and replicate the rig.
Recent example : AlexNet.
 - Task specific trained features
 - ✓ Nearly as good.

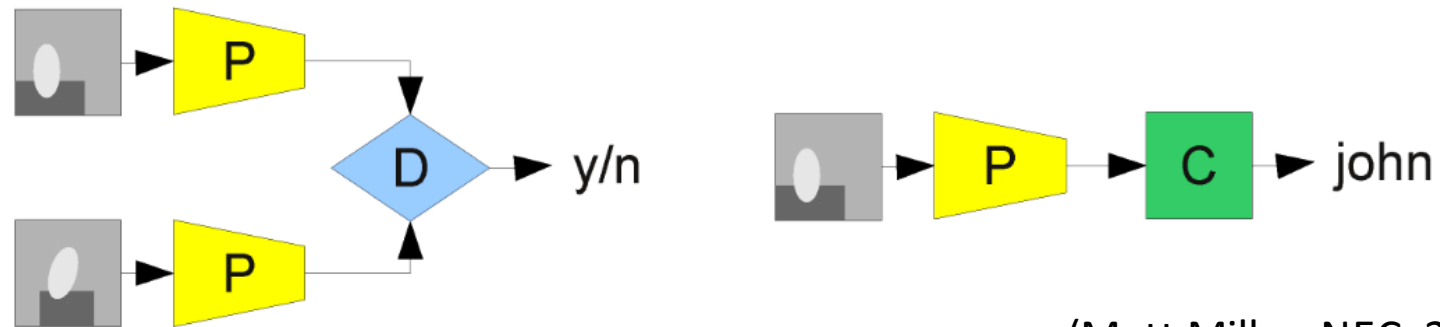
Example: face recognition

Interesting task: “Recognizing the faces of 10^6 persons.”

- How many labeled images per person can we obtain?

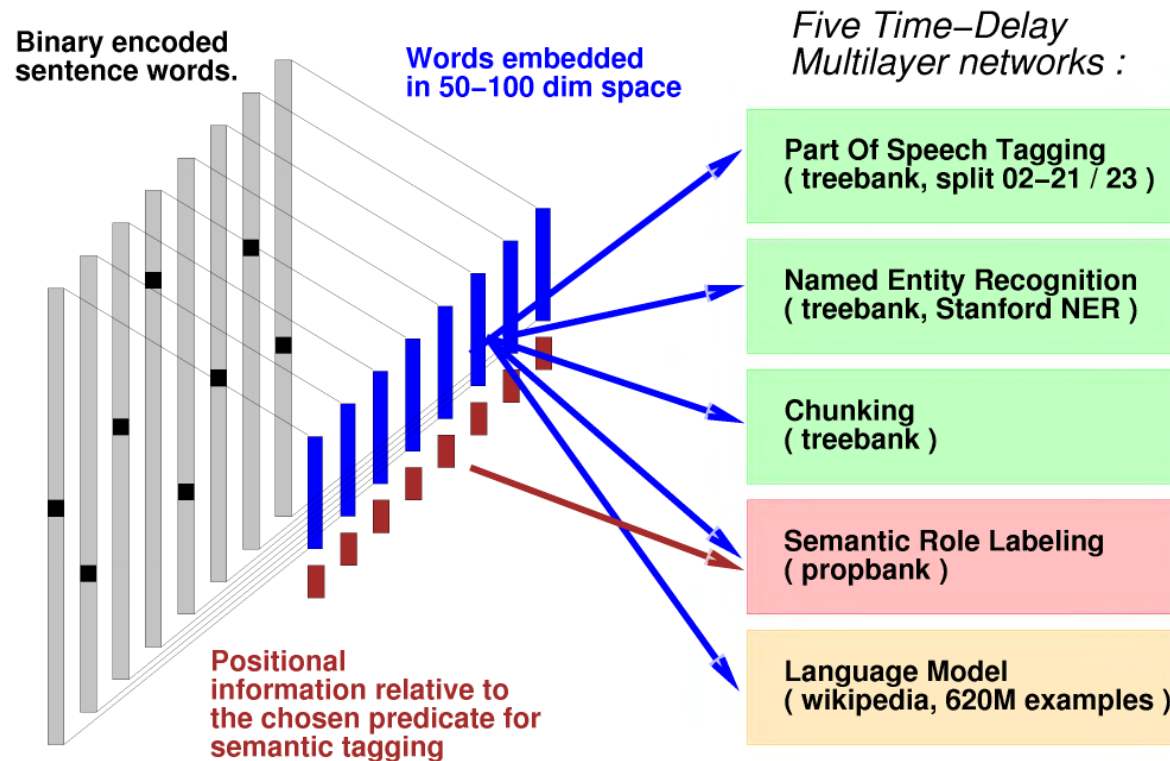
Auxiliary task: “Do these faces belong to the same person?”

- Two faces in the same picture usually are different persons.
- Two faces in successive frames are often the same person.



(Matt Miller, NEC, 2006)

Example: NLP tagging



(Collobert, Weston, et al., 2008-2011)

Example: object recognition

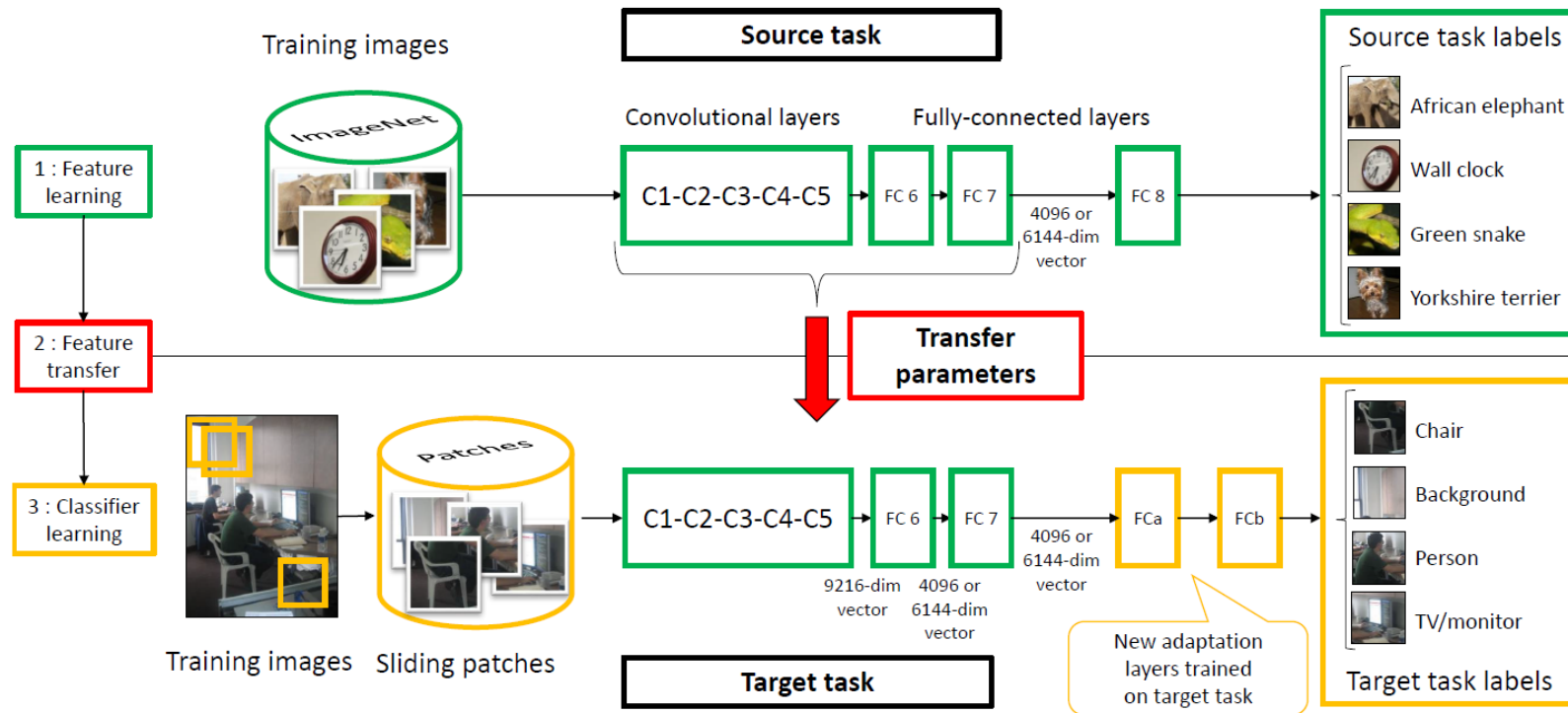
Dogs in ImageNet
($\sim 10^6$ dogs)



Dogs in Pascal VOC
(only $\sim 10^4$ images)



Example: object recognition



Several CVPR 2014 papers – figure from (Oquab et al., 2014)

Conclusion

Two process challenges

- Machine learning disrupts software engineering.
→ Challenging our impact on real world applications.
- Our experimental paradigm is reaching its limits.
→ Challenging the speed of our scientific progress.