Programmation: l’ère des spécifications, l’ère de l’apprentissage, l’ère du feedback

Michèle Sebag

TAO

AFIA – AFIHM, 2015
Revisiting the art of programming

1970s Specifications

1990s Programming by Examples

2010s Interactive Learning and Optimization

- Optimizing coffee taste  Herdy, 96
- Visual rendering  Brochu et al., 10
- Choice query  Viappiani et al., 10
- Information retrieval  Joachims et al., 12
- Robotics
  Akrouk et al., 12; Wilson et al., 12; Knox et al., 13; Saxena et al., 13

Programming with the Human in the Loop

Interaction, Learning, Optimization
Computing Machinery and Intelligence

... the problem is mainly one of programming.

brain estimates: \(10^{10}\) to \(10^{15}\) bits

I can produce about a thousand digits of program lines a day

[Therefore] more expenditious method seems desirable.

⇒ Machine Learning
Overview

Preamble

Machine Learning: All you need is...
  ...logic
  ...data
  ...optimization

All you need is expert’s feedback
  Reinforcement learning
  Programming by Feedback

Programming, An AI Frontier
ML: All you need is logic

Perception → Symbols → Reasoning → Symbols → Actions

*Let’s forget about perception and actions for a while...*

Symbols → Reasoning → Symbols

Requisite

- Strong representation
- Strong background knowledge
- Strong optimization tool
**Adam**: generate hypotheses from background knowledge and experimental data, design experiments to confirm/infirm hypotheses

**Eve**: drug screening, hit conformation, and cycles of QSAR hypothesis learning and testing.
ML: The logic era

So efficient

► Search: Reuse constraint solving, graph pruning, ...

Requirement / Limitations

► Initial conditions: critical mass of high-order knowledge
► ... and unified search space
► Symbol grounding, noise

Of primary value: intelligibility

► (A means: for debugging)
► An end: to keep the expert involved.
Overview

Preamble

Machine Learning: All you need is...
  ...logic
  ...data
  ...optimization

All you need is expert’s feedback
  Reinforcement learning
  Programming by Feedback

Programming, An AI Frontier
All you needed was data

Ast. series

Pierre de Rosette

Maths.

Natural
phenomenons

World

Human–related
phenomenons

Data / Principles

Maths.
Modelling

Common
Sense

You are here
All you need is big data

Scientific data

Natural phenomena

Mathematics

Data / Principles

Common sense

Human-related phenomena

You are here
Big data

IBM Watson defeats human champions at the quiz game Jeopardy

1  2  3  4  5  6  7  8
1000i  kilo  mega  giga  tera  peta  exa  zetta  yotta  bytes

- Google: 24 petabytes/day
- Facebook: 10 terabytes/day; Twitter: 7 terabytes/day
- Large Hadron Collider: 40 terabytes/seconds
The Higgs boson ML Challenge

Balazs Kégl, Cécile Germain et al.

https://www.kaggle.com/c/higgs-boson September 2014, 15th
Overview

Preamble

Machine Learning: All you need is...
  ...logic
  ...data
  ...optimization

All you need is expert’s feedback
  Reinforcement learning
  Programming by Feedback

Programming, An AI Frontier
ML: All you need is optimization

Old times

- Find the best hypothesis
- Find the best optimization criterion
  - statistically sound
  - a well-posed optimization problem
  - tractable
Episode 1

- NNs are universal approximators,…
- … but their training yields non-convex optimization problems
- … and some cannot reproduce the results of some others…

Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.
SVMs and Deep Learning

Episode 2

- At last, SVMs arrive!  
  Vapnik 92; Cortes & Vapnik 95

- Principle
  - Min $||h||^2$
  - subject to Constraint on $h(x)$
    $$h(x_i).y_i > 1, |h(x_i) - y_i| < \epsilon, h(x_i) < h(x'_i), h(x_i) > 1...$$

- Convex optimization! (well, except for hyper-parameters)

- More sophisticated optimization (alternate, upper bounds)...

Hastie 04; Bach 04; Srebro 11; ...
Episode 3

- Did you forget our AI goal? (learning ↔ learning representation)
- At last Deep learning arrives!

Principle

- We always knew that many-layered NNs offered compact representations

![Diagram](image-url)
SVMs and Deep Learning

Episode 3
- Did you forget our AI goal?
  (learning $\leftrightarrow$ learning representation)
- At last Deep learning arrives!

Principle
- We always knew that many-layered NNs offered compact representations
- But, so many local optima! (poor optima)
Episode 3

- Did you forget our AI goal? (learning ↔ learning representation)
- At last Deep learning arrives!

Principle

- We always knew that many-layered NNs offered compact representations
  
  Hasted 87

- But, so many local optima! (poor optima)

- Breakthrough: unsupervised layer-wise learning
  
  Hinton 06; Bengio 06
SVMs and Deep Learning

From prototypes to features

- $n$ prototypes $\rightarrow$ $n$ regions
- $n$ features $\rightarrow$ $2^n$ regions
SVMs and Deep Learning

Last Deep news

- Supervised training works, after all
  Glorot Bengio 10
- Does not need to be deep, after all
  Ciresan et al. 13, Caruana 13
**SVMs and Deep Learning**

**Last Deep news**
- Supervised training works, after all
  - Glorot Bengio 10
- Does not need to be deep, after all
  - Ciresan et al. 13, Caruana 13
  - Ciresan et al: use prior knowledge (non linear invariance operators) to generate new examples
  - Caruana: use deep NN to label hosts of examples; use them to train a shallow NN.
SVMs and Deep Learning

Last Deep news

- Supervised training works, after all  
  Glorot Bengio 10
- Does not need to be deep, after all 
  Ciresan et al. 13, Caruana 13
- SVMers’ view: the main thing is linear learning complexity

Take home message

- It works
- But why?
- Intelligibility?
SVMs and Deep Learning

Last Deep news
- Supervised training works, after all
- Does not need to be deep, after all
- SVMers’ view: the main thing is linear learning complexity

Take home message
- It works
- But why?
- Intelligibility?
Overview

Preamble

Machine Learning: All you need is...
  ...logic
  ...data
  ...optimization

All you need is expert’s feedback
  Reinforcement learning
  Programming by Feedback

Programming, An AI Frontier
Interactive optimization

Optimizing the coffee taste
Black box optimization:

\[ \mathcal{F} : \Omega \rightarrow \mathbb{R} \quad \text{Find } \arg \max \mathcal{F} \]

The user in the loop replaces \( \mathcal{F} \)

Herdy et al., 96
Interactive optimization

Optimizing the coffee taste
Black box optimization:

\[ \mathcal{F} : \Omega \rightarrow \mathbb{R} \quad \text{Find} \ \arg \max \mathcal{F} \]

The user in the loop replaces \( \mathcal{F} \)

Optimizing visual rendering

Optimal recommendation sets

Information retrieval

Herdy et al., 96

Brochu et al., 07

Viappiani & Boutilier, 10

Shivaswamy & Joachims, 12
Interactive optimization

Features

- Search space \( X \subset \mathbb{R}^d \) (recipe \( x \): 33% arabica, 25% robusta, etc)
- A non-computable objective
- Expert can (by tasting) emit preferences \( x \prec x' \).

Scheme

1. Alg. generates candidates \( x, x', x'', .. \)
2. Expert emits preferences
3. goto 1.

Issues

- Asking as few questions as possible \( \neq \) active ranking
- Modelling the expert’s taste surrogate model
- Enforce the exploration vs exploitation trade-off
Overview

Preamble

Machine Learning: All you need is...
  ...logic
  ...data
  ...optimization

All you need is expert’s feedback
  Reinforcement learning
  Programming by Feedback

Programming, An AI Frontier
Reinforcement Learning

Generalities

- An agent, spatially and temporally situated
- Stochastic and uncertain environment
- Goal: select an action in each time step,
  ... in order maximize expected cumulative reward over a time horizon

What is learned?

A policy = strategy = \{ state \mapsto action \}
Reinforcement Learning, formal background

**Notations**
- **State space** $S$
- **Action space** $A$
- **Transition** $p(s, a, s') \mapsto [0, 1]$
- **Reward** $r(s)$
- **Discount** $0 < \gamma < 1$

**Goal:** a policy $\pi$ mapping states onto actions

$$\pi : S \mapsto A$$

s.t.

Maximize $E[\pi | s_0] = \text{Expected discounted cumulative reward}$

$$= r(s_0) + \sum_t \gamma^{t+1} p(s_t, a = \pi(s_t), s_{t+1})r(s_{t+1})$$
Find the treasure

Single reward: on the treasure.
Wandering robot

Nothing happens...
The robot finds it
Robot updates its value function

\[ V(s, a) = \text{“distance“ to the treasure on the trajectory.} \]
Reinforcement learning

* Robot most often selects \( a = \arg \max V(s, a) \)
* and sometimes explores (selects another action).
* Lucky exploration: finds the treasure again
Updates the value function

* Value function tells how far you are from the treasure given the known trajectories.
Finally

* Value function tells how far you are from the treasure
Finally

Let’s be greedy: selects the action maximizing the value function
Reinforcement learning

Three tasks

▶ Learn values
▶ Learn transition model
▶ Explore

Issues

▶ Exploration / Exploitation dilemma
▶ Representation, approximation, scaling up
▶ REWARDS

designer’s duty
Relaxing Expertise Requirements
Relaxing Expertise Requirements in RL

Expert
- Associates a reward to each state  
- Demonstrates a (nearly) optimal behavior
- Compares and revises agent demonstrations
- Compares demonstrations

Agent
- Computes optimal policy based on rewards
- Imitates verbatim expert’s demonstration
- Imitates and modifies
- Learns the expert’s utility
- Learns, and selects demonstrations
- Accounts for the expert’s mistakes
Overview

Preamble

Machine Learning: All you need is...
  ...logic
  ...data
  ...optimization

All you need is expert’s feedback
  Reinforcement learning
  Programming by Feedback

Programming, An AI Frontier
Programming by feedback

Loop

1. Computer presents the expert with a pair of behaviors $y_1, y_2$
2. Expert emits preferences $y_1 \succ y_2$
3. Computer learns expert’s utility function $\langle w, y \rangle$
4. Computer searches for behaviors with best utility

Critical issues

- Asks few questions
- Be robust wrt noise (expert makes mistakes & changes his mind)
Programming by Feedback

Ingredients

- Modelling the expert’s competence
- Learning the expert’s utility
- Selecting the next best behaviors
  - Which optimization criterion
  - How to optimize it
Modelling the expert’s competence

Noise model

Given preference margin $z = \langle w^*, y - y' \rangle$

$$P(y \prec y' \mid w^*, \delta) = \begin{cases} 
0 & \text{if } z < -\delta \\
1 & \text{if } z > \delta \\
\frac{1+z}{2} & \text{otherwise}
\end{cases}$$

$\delta \sim U[0, M]$
Experimental validation

- Sensitivity to expert competence
  Simulated expert, grid world

- Continuous case, no generative model
  The cartpole

- Continuous case, generative model
  The bicycle

- Training in-situ
  The Nao robot
Sensitivity to (simulated) expert incompetence

Grid world: discrete case, no generative model
25 states, 5 actions, horizon 300, 50% transition noise

$M_E$  Expert incompetence
$M_A > M_E$  Computer estimate of expert’s incompetence

<table>
<thead>
<tr>
<th></th>
<th>$\ldots$</th>
<th>$1/4$</th>
<th>$1/2$</th>
<th>$1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1/64$</td>
<td>$1/4$</td>
<td>$1/2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$1/128$</td>
<td>$1/64$</td>
<td></td>
<td>$1/4$</td>
<td></td>
</tr>
<tr>
<td>$1/256$</td>
<td>$1/128$</td>
<td>$1/64$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

True $w^*$ on gridworld
Two notions
- The true human’s competence
- The learner’s confidence in the human competence

What is best: trusting a (mildly) competent human, or (mildly) distrusting a competent human?
A cumulative (dis)advantage phenomenon:

The number of expert’s mistakes *increases* as the computer underestimates the expert’s competence.

For low $M_A$, the computer learns faster, submits more relevant demonstrations to the expert, thus priming a virtuous educational process.
Overview

Preamble

Machine Learning: All you need is...
  ...logic
  ...data
  ...optimization

All you need is expert’s feedback
  Reinforcement learning
  Programming by Feedback

Programming, An AI Frontier
Feasibility of Programming by Feedback for simple tasks

Back on track:

One could carry through the organization of an intelligent machine with only two interfering inputs, one for pleasure or reward, and the other for pain or punishment.
Programming by Feedback

About interaction: as designer; as user

- No need to debug if you can just say: No! and the computer reacts (appropriately).

- I had a dream: a world where I don’t need to read the fucking manual...
Future: Tackling the Under-Specified

Knowledge-constrained

Computation, memory-constrained
Acknowledgments

Riad Akrour
Marc Schoenauer
Alexandre Constantin
Jean-Christophe Souplet
Related

- Sumit Gulwani, Automating string processing in spreadsheets using input-output examples, ACM SIGPLAN Notices 2011
- Dianhuan Lin & al., Bias reformulation for one-shot function induction, ECAI 2014.