Three Challenges for Machine Learning Research:
From Learning Algorithms to Learning Systems

Thomas G. Dietterich
School of EECS
Oregon State University
http://eecs.engr.orst.edu/~tgd
A Caricature of AI Research History


Reasoning Systems: Axioms + Inference

Learning Algorithms: Data + Algorithm

Learning Systems: Data + Algorithm + Reasoning

??
Three Challenges

- Structured Machine Learning
- Transfer Learning
- Deployed Learning Systems
Structured Machine Learning

- Emerging applications of machine learning
  - Sequence labeling (information extraction, NLP, bioinformatics, activity recognition)
  - Spatio-temporal labeling
  - Relational learning and collective classification

- Existing ideas are not scaling up well to these problems
Sequential Supervised Learning

Given: A set of training examples of the form \((X_i, Y_i)\), where

\[ X_i = \langle x_{i,1}, \ldots, x_{i,T_i} \rangle \] and

\[ Y_i = \langle y_{i,1}, \ldots, y_{i,T_i} \rangle \] are sequences of length \(T_i\)

Find: A function \(F\) for predicting new sequences: \(Y = F(X)\).
Examples of Sequential Supervised Learning

<table>
<thead>
<tr>
<th>Domain</th>
<th>Input $X_i$</th>
<th>Output $Y_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part-of-speech Tagging</td>
<td>sequence of words</td>
<td>sequence of parts of speech</td>
</tr>
<tr>
<td>Information Extraction</td>
<td>sequence of tokens</td>
<td>sequence of field labels {name, \ldots}</td>
</tr>
<tr>
<td>Text-to-speech Mapping</td>
<td>sequence of letters</td>
<td>sequence phonemes</td>
</tr>
</tbody>
</table>
How to Solve Structured ML Problems?

Existing approaches: Two main families

- **Declarative:**
  - Learn declarative knowledge
  - Feed to reasoning system to make decisions at run time

- **Procedural:**
  - Learn procedural knowledge
  - No reasoning system needed at run time

For Structured ML problems, neither approach appears to be entirely satisfactory!
Declarative vs Procedural Learning (1): Classification

Task: Classification

Declarative Approach:

- Learn Bayesian network \( P(x,y) \):
  \[
  x_2 \quad y \quad x_1 \\
  x_2 \Rightarrow y \\
  y \land x_3 \Rightarrow x_1
  \]

- Learn Association Rules:
  \[
  x_2 \Rightarrow y \\
  y \land x_3 \Rightarrow x_1
  \]

- Perform reasoning to make classification decisions:
  belief propagation
  resolution
Declarative vs Procedural Learning (1): Classification

Procedural Approach:

- Tune a black box classifier:
- No reasoning needed to make classification decisions
Declarative vs Procedural Learning (2): Sequential Decision Making

Task: Given $s_t$ choose action $a_t$ to maximize total reward $\sum_t r_t$

Declarative Approach:

- Learn transition function $P(s_{t+1} | s_t, a_t)$
- Learn reward function $R(s_{t+1} | s_t, a_t)$
- Compute policy $\pi$ via dynamic programming (MDP Planning)
- Learn STRIPS operators
- Learn goal predicates
- Compute policy via STRIPS planning
Declarative vs Procedural Learning (2): Sequential Decision Making

Procedural Approach:

- Tune policy $\pi$ incrementally
- No reasoning required to choose actions
### Experience with these methods

<table>
<thead>
<tr>
<th></th>
<th>Declarative</th>
<th>Procedural</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>learning algorithms</strong></td>
<td>simple &amp; efficient</td>
<td>complex &amp; expensive</td>
</tr>
<tr>
<td><strong>cost of inference</strong></td>
<td>expensive</td>
<td>zero</td>
</tr>
<tr>
<td><strong>easy to mix learning with hand-crafted knowledge</strong></td>
<td>easy</td>
<td>very difficult</td>
</tr>
<tr>
<td><strong>performance</strong></td>
<td>mediocre</td>
<td>excellent</td>
</tr>
</tbody>
</table>
Applying these Methods to Structured ML Problems

- **Declarative Approach:**
  - Hidden Markov models

- **Procedural Approach:**
  - Many new methods!
Declarative Approach: Hidden Markov Models

- Learning (in fully-observed case) does not require inference
- Classification inference procedure: Viterbi algorithm
  \[
  \arg\max_{y_1 \ldots y_T} \prod_t P(y_t|y_{t-1}) \cdot P(x_t|y_t)
  \]
- Performance often mediocre
Procedural Approach

A completely procedural approach is not feasible because there are $K^T$ possible output sequences $Y$ (for $K$ classes and sequence length $T$)
Procedural Approach (2)

- Learn a scoring function $\Psi$ such that the Viterbi algorithm gives the right answer
  $$\arg\max_{\langle y_1...y_T \rangle} \sum_t \psi(y_{t-1}, y_t, x_t)$$

- Algorithms:
  - voted perceptron
  - conditional random fields (CRF)
  - extensions of Support Vector Machines
  - graph transformer networks
Learning “through” Inference

All of these algorithms perform inference at learning time

- Given $X$ and current $\Psi$
- Perform inference to compute $\langle \hat{y}_1 \ldots \hat{y}_T \rangle$
- Compare $\langle \hat{y}_1, \ldots, \hat{y}_T \rangle$ to $\langle y_1, \ldots, y_T \rangle$ and update $\Psi$

![Diagram showing the flow of information from input $\langle x_1, \ldots, x_T \rangle$ through Viterbi inference, comparing the inferred $\langle \hat{y}_1, \ldots, \hat{y}_T \rangle$ with the actual $\langle y_1, \ldots, y_T \rangle$, and updating $\Psi$.](image-url)
Signs of Trouble

- **Learning cost is dominated by cost of inference**
  - Nobody publishes results on large data sets
  - Text-to-Speech Mapping
    - Conditional random fields are too expensive to apply

- **Learned classifier often performs worse than simple “sliding window”**
  - Sliding window treats \((x_t, y_t)\) as independent examples (possibly considering a “window”, e.g., \(\langle x_{t-2}, x_{t-1}, x_t, x_{t+1}, x_{t+2}\rangle\))
  - Protein secondary structure prediction:
    - Neural network sliding window: 76-78% correct
    - Conditional random fields: 66% correct
  - Semantic Role Labeling
    - Boosted tree sliding window F1 = 71.41 (75 labels)
    - Conditional random fields: F1 = 60.43 (15 labels)
What to Try Next?
Hints from Cognitive Psychology

Intelligence (as measured by IQ) is required to carry out this process on novel tasks.

Initial performance based on declarative knowledge is mediocre.

Skill is acquired slowly and becomes opaque.
Skill is not simply declarative knowledge combined with inference
- Skill acquisition is more than just acquiring declarative knowledge
- Skill acquisition is not just compiling declarative knowledge into more efficient form (a la EBL and SOAR)

Somehow, declarative knowledge and inference guide the acquisition of procedural skill
- initialize the policy $\pi$?
- constrain the tuning of $\pi$?

In structured ML tasks, we should explore methods that do not require extensive run-time inference
- learn declarative knowledge first
- then apply it to guide procedural learning?
Three Challenges

- Structured Machine Learning
- Transfer Learning
- Deployed Learning Systems
Transfer Learning

- System learns to perform Task A on new Task B, system either
  - immediately performs better on it (Type I), or
  - learns to perform well with less experience than would otherwise have been required (Type II)
Measuring Type I Transfer

When B is not a learning task

amount of task A training data

task B performance

task B
Measuring Type II Transfer

- When B is a learning task
- After a fixed amount of training on A

Task B performance vs. amount of task B training data

- Task B after training on A
- Task B no training on A
Dimensions of Transfer Learning

- Amount of sharing possible
- Depth of sharing
- Direct sharing versus mapped sharing
- Engineered transfer versus “transfer in the wild”
### Amount of Sharing Possible

<table>
<thead>
<tr>
<th>Distance</th>
<th>feature relevance</th>
<th>ontology</th>
<th>task decomposition</th>
<th>declarative facts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Near</td>
<td>Shared</td>
<td>Shared</td>
<td>Shared</td>
<td>Shared</td>
</tr>
<tr>
<td>Medium</td>
<td>Shared</td>
<td>Shared</td>
<td>Shared</td>
<td>Not Shared</td>
</tr>
<tr>
<td>Medium</td>
<td>Shared</td>
<td>Shared</td>
<td>Not Shared</td>
<td>Not Shared</td>
</tr>
<tr>
<td>Far</td>
<td>Shared</td>
<td>Not Shared</td>
<td>Not Shared</td>
<td>Not Shared</td>
</tr>
<tr>
<td>Infinite</td>
<td>Not Shared</td>
<td>Not Shared</td>
<td>Not Shared</td>
<td>Not Shared</td>
</tr>
</tbody>
</table>
Example 1: Transfer of Learned Facts

Task A: Meeting Planning
Who should attend budget meeting for Project X?

Task B: Purchasing
Who can approve purchases on Project X?

Learned on Task A
- Financial officers should attend budget meetings
- Stephen Q. is financial officer for Project X

Learned on Task B
- Financial officers can approve purchases

Stephen Q. should attend budget meeting
Stephen Q. can approve purchases
Example 2: Transfer of Learned Subprocedures

Task A: Purchasing Computers

- Tradeoff Specs, Price, Availability
- Computer Meets Specs
- Availability
  - Shipping Cost

Computer Specs:
- CPU speed
- Memory size
- Disk size

Availability:
- Discontinued
- Back ordered
- Delivery date

Task B: Purchasing Books

- Tradeoff Specs, Price, Availability
- Book Meets Specs
- Availability
  - Shipping Cost

Book Specs:
- Title
- Author
- Binding

Availability:
- Out of print
- Back ordered
- Delivery date
Example 3: Transfer of Learned Ontology

Task A: Tenure review in university

Task B: Command and control in police force

Organization is a hierarchy of groups
Each group has a team leader and team members
The members of all groups except the lowest are the team leaders of subgroups

Note: Domain facts and procedures do NOT transfer:

Tenure dossier flows up hierarchy

Orders flow down hierarchy
Example 4: Transfer of Learned Feature Relevance

Task A: Routing Complaints

- Carpenter: framing, installing cabinets
- Drywaller: taping, sealing, texturing
- Painter: masking, painting
- Contractor: scheduling, project planning

Task B: Meeting Scheduling

- “Chief Evangelist” might be able to substitute for “Evangelist” in meeting

Job title determines job responsibilities

These inferences can be made without even knowing what “sealing” or “Evangelist” mean
## Amount of Sharing Possible

<table>
<thead>
<tr>
<th>Distance</th>
<th>feature relevance</th>
<th>ontology</th>
<th>task decomposition</th>
<th>declarative facts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Near</td>
<td>Shared</td>
<td>Shared</td>
<td>Shared</td>
<td>Shared</td>
</tr>
<tr>
<td>Medium</td>
<td>Shared</td>
<td>Shared</td>
<td>Shared</td>
<td>Not Shared</td>
</tr>
<tr>
<td>Medium</td>
<td>Shared</td>
<td>Shared</td>
<td>Not Shared</td>
<td>Not Shared</td>
</tr>
<tr>
<td>Far</td>
<td>Shared</td>
<td>Not Shared</td>
<td>Not Shared</td>
<td>Not Shared</td>
</tr>
<tr>
<td>Infinite</td>
<td>Not Shared</td>
<td>Not Shared</td>
<td>Not Shared</td>
<td>Not Shared</td>
</tr>
</tbody>
</table>

**Notes:**
- Stephen Q is financial officer
- Book specifications must match
- Hierarchical Organization
- Job title determines responsibilities

---

**Details:**
- **Declarative facts**
  - **Fact 1:** Book specifications must match
  - **Fact 2:** Hierarchical Organization
  - **Fact 3:** Job title determines responsibilities

---

**Notes:**
- Task decomposition
  - **Task 1:** Determines responsibilities
  - **Task 2:** Categorization of features
  - **Task 3:** Integration of ontology

---

**Relevance:**
- **Relevance 1:** Feature complexity
- **Relevance 2:** Task complexity
- **Relevance 3:** Ontology integration

---

**Distance:**
- **Near:** Immediate relevance
- **Medium:** Moderate relevance
- **Far:** Remote relevance
- **Infinite:** Indeterminate relevance
Transfer Learning Summary

- People exhibit transfer learning
- Transfer learning requires identifying shared components (feature relevance, ontology, subprocedures, domain facts)
- Transfer learning could involve a wide variety of mechanisms
  - Current learning systems can transfer if the input feature space encompasses both tasks and there is enough training data
  - This may not be statistically feasible in more complex problems
Three Challenges

- Structured Machine Learning
- Transfer Learning
- Deployed Learning Systems
Deployed Learning Systems

Two views of machine learning

– as a data-driven software development methodology
  - tablet PC
  - US mail address reader

– as the foundation for adaptive software systems
  - collaborative filtering
  - spam filtering
  - TaskTracer
Challenges for Deployed Learning Systems

- Autonomous Learning ("learning in the wild")
  - Can’t afford to deploy a machine learning expert with each software system
- Non-stationarity
  - space of classes is changing
  - space of features is changing
  - underlying probability distribution is changing
  - “don’t call it ‘drift’”
Example Application: TaskTracer

- TaskTracer: make the Windows desktop task-sensitive

- Hypothesis:
  - user’s time at the keyboard can be divided into episodes each devoted to a general activity
    - working on CS534
    - working on sequential supervised learning
    - preparing for Iberamia conference
  - these general activities provide a key way to organize the user’s documents, web pages, contacts, appointments, folders, phone calls, etc.
TaskTracer 1.0.0

- User defines a hierarchy of activities
- User tells TaskTracer what activity he is working on right now
- TaskTracer monitors all file visits in IE, Office, etc.
Task-Based Access to Documents

Task Explorer provides easy access to user’s documents based on current activity.
Activity Predictor

Users forget to update the current activity

Task predictor learns to predict the current activity from
  – title of window
  – path name of file
  – URL of web page
  – etc.
Task Predictor

- Tasks: 
  - 430
  - TaskTracer
  - grant nsf kp
  - 539
  - home
  - admissions
  - grant darpa kp seed
  - grant admin darpa KIlearn
  - (none)
  - admin
  - service autonomic computing conference

- Predicted Probabilities:
  - 0.972353410074475
  - 0.0123189871235458
  - 0.00098971379574165
  - 0.0004197893914135
  - 0.0001164232904463
  - 9.8718243161531E-05
  - 5.08976855338363E-05
  - 5.0814820551174986E-06
  - 5.0814820551174986E-06
  - 2.1114422227537E-08

- Top 3 Most Likely tasks:
  - 430
  - TaskTracer
  - grant nsf kp

- Relevant Documents:
  - Home - OSU Online Catalog
  - Schedules By Subject - OSU Online Catalog
  - Schedule of Classes - OSU Online Catalog
  - Quick-Jump Help - OSU Online Catalog
  - Course detail - OSU Online Catalog
  - Course list - OSU Online Catalog
  - CS430: Introduction to AI - Syllabus
  - CS430: Introduction to Artificial Intelligence
  - CS430 Class Project: Spam Filter
Machine Learning Challenges

Set of activities changes over time
- projects and classes are finished
- new projects and classes begin

Set of input features changes over time
- new attributes are added with each release
- old attributes are removed or become redundant

User behavior changes over time
- user relies more on predictions and provides less feedback
How can ordinary software engineers learn to design deployed adaptive systems?
- What design tools and methodologies do they need?

How can we verify and validate adaptive systems?
- What measures of system behavior can we verify before deployment and check after deployment?

How can we support easy maintenance of deployed adaptive systems?
- Changing the definitions of input features, the number of input features
- Avoiding having to retrain the system after each upgrade

How can system staff install new releases and repair problems without access to user’s private data?
Concluding Remarks

Maturation of the machine learning field gives rise to three challenges

– Structured Machine Learning
– Transfer Learning
– Design and Software Engineering of Deployed Adaptive Systems
Acknowledgements

- **Structured Machine Learning**
  - Conversations with Pat Langley, William Cohen, Alan Fern, Yaroslav Bulatov, Lluís Márquez

- **Transfer Learning**
  - Leslie Kaelbling and other members of the CALO team

- **Task Tracer**
  - Jon Herlocker, fellow PI
  - Kevin Johnsrude, software developer
  - Phuoc Do, undergraduate

- **Research support from NSF (MKIDS program) and DARPA (real-world learning and CALO)**