Learning Automata with Infinite State Cardinality

fwood@robots.ox.ac.uk

Joint work with Nicholas Bartlett, David Pfau, Jonathan Huggins, Jan Gasthaus, Yee Whye Teh, Lancelot James, Cedric Archambeau and others

University of Oxford

June, 2013
Predictive Modelling

- Actors must have predictive models of worlds in order to plan
- Better predictive models can yield better plans and more rewarding sequences of actions

Ultimate goal: life-long learning algorithm that incrementally extracts a representation of the evolving world that is useful for planning from a single, continually growing sequence of observations (n.b. automata aren’t this but are instructive)
Computational Constraints Imposed By Life-Long Learning

- Incremental model estimation given a single observation sequence of growing (unbounded) length
- Estimator must have linear-time complexity
- The model must be representable in constant space
- Predictive inference must be performed in constant time
What is an automaton?

Classically

- An automaton is a finite representation of a formal language that may be an infinite set.
- Automata are often classified by the class of formal languages they are able to recognise.
- The set of all the words accepted by an automaton is called the language recognised by the automaton.

Today

- Automata with unbounded state cardinality (little existing theory).
- Learning automata from data (given a production sequence, what was the automata that produced it?)
What does this little probabilistic deterministic finite automata (PDFA) generate?

What is a state?
Automata corresponding to 1\textsuperscript{st}-order Markov model over binary alphabet with fixed emission distributions
Formally a PDFA is a quintuple

\[ M = (Q, \Sigma, \delta, \{ G[i] \}, q_0) \]

where

- \( Q \) is a finite set of states
- \( \Sigma \) is a finite set of symbols
- \( \delta : Q \times \Sigma \rightarrow Q \) is a deterministic transition function from state/symbol pairs to another state
- \( \{ G[i] : i \in Q \} \) is a set of probability vectors over symbols, one for each state
- \( q_0 \) is the initial state
Learning Problem(s)

Given a single, growing sequence \( x = x_1, x_2, \ldots, x_i, \ldots \in \Sigma^+ \) of discrete observations one might like to learn

- \( \{ G[i] : i \in Q \} \), the set of emission probability distributions with \( Q \) and \( \delta \) fixed.

or

- \( \{ G[i] : i \in Q \} \), the set of emission probability distributions and
- \( Q \), the set of states and
- \( \delta : Q \times \Sigma \rightarrow Q \), the transition function from state/symbol pairs to another state
Speculation

Automata

\[ \text{PDFA} \subset \text{PNFA} \subset \text{Probabilistic pushdown automata} \subset \text{Turing machine} \]

Corresponding Bayesian nonparametric models

\[ \text{SM} \ [30], \ \text{PDIA} \ [21] \subset \text{IHMM} \ [3, 8, 14] \subset \text{IPCFG} \ [16, 18] \subset \text{Church} \ [12] \]

- More smoothing in emissions distributions on the right, more complex latent state on the right
- Lessons learned from simpler models might inform development of more practical complex models
- Simpler models might be used to “smooth” program inference
Incrementally Learn a Model to Predict What Comes Next

<table>
<thead>
<tr>
<th>Challenge Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>01001001, 01101110, 00100000, 01110100, 01101000, 01110010, 01100101, 00100000, 01110100, 01100001, 01111001, 01110011, 00100000, 01111001, 01101111, 01110101, 01110010, 00100000, 01101000, 01100001, 01110010, 00100000, 01110010, 01100100, 00100000, 01100100, 01101000, 01101110, 01100101, 00100000, 01110111, 01101001, 01101100, 01101100, 00100000, 01100011, 01110010, 01100001, 01110011, 01101000</td>
</tr>
</tbody>
</table>
Graphical model for smoothing PDFA emission distributions

Figure: Graphical model for coupling binary Markov model (= automata with fixed $\delta$ (not shown)) emission distributions
Notation

\[ G_\varepsilon | U_\Sigma, d_0 \sim PY(d_0, 0, U_\Sigma) \]
\[ G_u | G_{\sigma(u)}, d_{|u|} \sim PY(d_{|u|}, 0, G_{\sigma(u)}) \quad \forall u \in \Sigma^+ \]
\[ x_t | x_1, \ldots, x_{t-1} = u \sim G_u \]

Here \( \sigma(x_1 x_2 x_3 \ldots x_t) = x_2 x_3 \ldots x_t \) is the suffix operator, \( \Sigma = \{0, 1\} \), \( U_\Sigma = [.5, .5] \), and \( \delta(x_1 x_2 \ldots x_{t-1}, x_t) \equiv x_1 x_2 \ldots x_{t-1} x_t \).

Encodes natural assumptions:

- Recency assumption encoded by form of coupling related conditional distributions ("back-off")
- Power-law effects encoded by the choice of hierarchical modeling glue: the Pitman-Yor process (PY).
A Pitman-Yor process $\text{PY}(c, d, H)$ is a distribution over distributions with three parameters

- A discount $0 \leq d < 1$ that controls power-law behavior
  - $d = 0$ is Dirichlet process (DP)
- A concentration $c > -d$ like that of the Dirichlet process (DP)
- A base distribution $H$ also like that of the DP

Key properties:

If $G \sim \text{PY}(c, d, H)$ then a priori

- $\mathbb{E}[G(s)] = H(s)$
- $\text{Var}[G(s)] = (1 - d)H(s)(1 - H(s))$
Power-law scaling of word frequencies in English text. Relative word frequency is plotted against each words’ rank when ordered according to frequency. There are a few very common words and a large number of relatively rare words.

Figure from [31].
\[ G_u | G_{\sigma(u)}, d_u | \sim PY(d_u | u, 0, G_{\sigma(u)}) \quad \forall u \in \Sigma^+ \]

Space. And time.
PY properties used to achieve linear space sequence memoizer.

### Coagulation

If

\[ G_2 | G_1 \sim \text{PY}(d_1, 0, G_1) \]

and

\[ G_3 | G_2 \sim \text{PY}(d_2, 0, G_2) \]

then

\[ G_3 | G_1 \sim \text{PY}(d_1d_2, 0, G_1) \]

with \( G_2 \) marginalized out.

---

### Fragmentation

Suppose

\[ G_3 | G_1 \sim \text{PY}(d_1d_2, 0, G_1) \]

Fragmentation operations exist to reinstantiate \( G_2 \) and \( G_3 \).

---

\(^1[23, 13, 30]\)
$O(T)$ Sequence Memoizer\textsuperscript{2} Graphical Model for $x = 110100$

\[G[\_]

\text{[Wood et al, 2009]}

\text{Wood (Oxford, Engineering)}
SM versus $n^{th}$ order Markov models with hierarchical PYP priors as $n$ varies. In red is computational complexity.
Nonparametric $\rightarrow O(T)$ space $\implies$ doesn’t work for life-long learning
Towards Lifelong Learning: Incremental Estimation

Sequential Monte Carlo inference [Gasthaus, Wood, and Teh, 2010]
- Wikipedia 100MB → 20.8MB (1.66 bits/byte\(^3\))
- \(\approx\) PAQ, special-purpose, non-incremental, Hutter-prize leader

Still \(O(T)\) storage complexity.

\[
-\frac{1}{T} \sum_{t=1}^{T} \log_2 P(x_t|x_{1:t-1})
\]

\(^3\)average log-loss

is average # of bits to transmit one byte under optimal entropy encoding (lower is better)
Towards Lifelong Learning: Constant Space Approximation

“Forgetting Counts” [Bartlett, Pfau, and Wood, 2010]

Constant-state-count SM approximation
Data: Calgary corpus, States = number of states used in 3-10 grams

“Improvements to the SM” [9]
Constant-size representation of predictive distributions
SM with life-long learning computational characteristics.

Wikipedia: 26.8GB $\rightarrow$ 4GB ($\approx 1.2$) bits/byte

$L$ is number of states in SM, data Wikipedia dump 2010 head, depth 16,
the fastest way to make money is . . .

being player. bucketti wish i made it end in the reducers and assemblance smart practices to allow city is something to walk in most of the work of agriculture. i ’d be able to compete with an earlier goals: words the danger of conduct in southern california, has set of juice of the fights lack of audience that the eclasm beverly hills. ” she companies or running back down that the book, ” grass. and the coynes, exaggerate between 1972. the pad, a debate on emissions on air political capital crashing that the new obviously program ” – irock price, ” coach began refugees, much and

---

⁴constant-space byte-SM, NY Times Corpus (50MB), log-loss 1.49 bits/byte, 2 million rest.’s
SM = powerful Bayesian nonparametric sequence model

- Intuition: \( \lim_{n \to \infty} \) smoothing \( n \)-gram
- Byte predictive performance \textit{in human range}
- Bayesian-nonparametric \( \implies \) incompatible with lifelong learning
- Non-trivial approximations preserve performance
- Resulting model compatible with lifelong learning

Widely useful

- Language-modeling
- Lossless compression
- Plug-in replacement for Markov model component of any complex graphical model
- Direct plug-in replacement for \( n \)-gram language model

Software: \url{http://www.sequencememoizer.com/}
A general purpose streaming lossless compressor ("deplump") built on the sequence memoizer is available for demonstration at

- http://www.deplump.com/
  - real time performance

Source code for the sequence memoizer is downloadable from

- http://www.sequencememoizer.com/
Can We Do Better?

Problem:
- No long-range coherence

Solution:
- Develop models that more efficiently encode long-range dependencies
  - Probabilistic deterministic infinite automata
  - Structured infinite hidden semi-Markov model
  - PCFG?
  - Probabilistic program?
Example probabilistic deterministic finite automata (PDFA) with no finite-order Markov model equivalent

This *Even process* assigns positive probability to ABBA, AAABBBBBABB, BBBBBBBBBB, but not to ABA
Probabilistic Deterministic Infinite Automata (PDIA)

A PDIA [Pfau, Bartlett, and Wood, 2011] is a Bayesian nonparametric PDFA with an unbounded number of states.

Key Characteristics:
- Incrementally learnable
- Prior that biases automata complexity
- Structure learned from data
- $O(T)$ storage

Approximations for life-long learning:
- Future work

---

Markov models are in the support of PDIA, can we exploit that?
Formally a PDIA is a quintuple

\[ M = (Q, \Sigma, \delta, \{G_i\}, q_0) \]

where
- \( Q \) is a countable set of states
- \( \Sigma \) is a finite set of symbols
- \( \delta : Q \times \Sigma \rightarrow Q \) is a deterministic transition function from state/symbol pairs to another state
- \( \{G_i : i \in Q\} \) is a set of probability vectors over symbols, one for each state
- \( q_0 \) is the initial state
A PDIA Prior

\[\begin{align*}
H_0 & \sim \ PY(\gamma, d_0, H) \\
H_\sigma & \sim \ PY(\alpha, d, H_0) \\
\delta(q_i, \sigma) & \sim \ H_\sigma \\
G_i & \sim \ Dir(\beta/|\Sigma|, \ldots, \beta/|\Sigma|)
\end{align*}\]

given a sequence \(x\) a posterior over

\[H = \{\delta, \{G_i\}\}\]

can be learned from data and used for prediction

\[P(x_T|x) = \int P(x_T|H)P(H|x)dP(H)\]
With state $\xi_t \in Q$ and $\xi_t = \delta(\xi_{t-1}, x_{t-1})$ the PDIA likelihood

$$P(x|\{G_i\}, \delta) = G_{\xi_0}(x_0) \prod_{t=1}^{T} G_{\xi_t}(x_t)$$

- doesn’t require marginalizing over paths (vs. HMM/PNFA) (fast)
- is of a form that $\{G_i\}$ can be marginalized out

Sampling $\delta$ requires “integrating” over paths diverging from the “current state” of the machine
Posterior expectations “average over PDFA’s”.

- World states are *learned* from data.
- Prior biases towards a state re-use (other alternatives?)
- State-to-state transitions learned from data.
- States not identifiable by context (vs. Markov model)

**Lessons**

- Smoothness: in order to “move around” space of PDIA’s *a priori* every state must be able to reach every other state and every emission distribution must be able to emit everything.
- Smoothing: because relationships between states are unknown, hierarchical smoothing of emission distributions is not possible.
We ourselves are more interested in establishing new ways to produce smoothed predictive conditional distributions. Inference in the PDIA presents a completely new approach to smoothing, smoothing by averaging over PDFA model structure rather than hierarchically smoothing related emission distribution estimates. Our PDIA approach gives us an attractive ability to trade-off between model simplicity in terms of number of states, computational complexity in terms of asymptotic cost of prediction, and predictive perplexity. While our PDIA approach may not yet outperform the best smoothing Markov model approaches in terms of predictive perplexity alone, it does outperform them in terms of model complexity required to achieve the same predictive perplexity, and outperforms HMMs in terms of asymptotic time complexity of prediction. This suggests that a future combination of smoothing over model structure and smoothing over emission distributions could produce excellent results. PDIA inference gives researchers another tool to choose from when building models. If very fast prediction is desirable and the predictive perplexity difference between the PDIA and, for instance, the most competitive n-gram is insignificant from an application perspective, then doing finite sample inference in the PDIA offers a significant computational advantage in terms of memory.

We indeed believe the most promising approach to improving PDIA predictive performance is to construct a smoothing hierarchy over the state specific emission distributions, as is done in the smoothing n-gram models. For an n-gram, where every state corresponds to a suffix of the sequence, the predictive distributions for a suffix is smoothed by the predictive distribution for a shorter suffix, for which there are more observations. This makes it possible to increase the size of the model indefinitely without generalization performance suffering. In the PDIA, by contrast, the predictive probabilities for states are not tied together. Since states of the PDIA are not uniquely identified by suffixes, it is no longer clear what the natural smoothing hierarchy is. It is somewhat surprising that PDIA learning works nearly as well as n-gram modeling even without a smoothing hierarchy for its emission distributions. Imposing a hierarchical smoothing of the PDIA emission distributions remains an open problem.
### Comparative Byte Prediction Performance

<table>
<thead>
<tr>
<th></th>
<th>PDIA</th>
<th>PDIA-MAP</th>
<th>HMM-EM</th>
<th>bigram</th>
<th>trigram</th>
<th>4-gram</th>
<th>5-gram</th>
<th>6-gram</th>
<th>SM (PF-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AIW</strong></td>
<td>5.13</td>
<td>5.46</td>
<td>7.89</td>
<td>9.71</td>
<td>6.45</td>
<td>5.13</td>
<td>4.80</td>
<td>4.69</td>
<td>4.78</td>
</tr>
<tr>
<td></td>
<td>365.6</td>
<td>379</td>
<td>52</td>
<td>28</td>
<td>382</td>
<td>2,023</td>
<td>5,592</td>
<td>10,838</td>
<td>19,358</td>
</tr>
<tr>
<td><strong>DNA</strong></td>
<td>3.72</td>
<td>3.72</td>
<td>3.76</td>
<td>3.77</td>
<td>3.75</td>
<td>3.74</td>
<td>3.73</td>
<td>3.72</td>
<td>3.56</td>
</tr>
<tr>
<td></td>
<td>64.7</td>
<td>54</td>
<td>19</td>
<td>5</td>
<td>21</td>
<td>85</td>
<td>341</td>
<td>1,365</td>
<td>314,166</td>
</tr>
</tbody>
</table>

**Table**: PDIA inference performance (perplexity) relative to HMM and fixed order Markov models. Top rows: perplexity. Bottom rows: number of states in each model. For the PDIA this is an average number.

**Results**

- **Expected**: PDIA is more compact than the SM
- **Unexpected**: SM predicts best
- **Expected**: single HMM even more compact
- **More Unexpected**: single HMM predicts worst

**Questions**

- More data? Convergence rates? Wrong objective? Smoothing vs. compactness?

AIW - $\approx 10k$ char. train, $\approx 4k$ char. test; DNA $\approx 150k$ b.p. train, $\approx 50k$ b.p. test. Perplexity $= 2^{\log_{-}\text{loss}}$
Figure: Two PNFA’s outside the class of PDFAs. (a) can be represented by a mixture of two PDFAs, one following the right branch from state 0, the other following the left branch. (b), in contrast, cannot be represented by any finite mixture of PDFAs.

PNFA’s can be transformed into HMM’s and vice versa [6, 11].

HMM with infinite number of states + key additional characteristics:

- Explicitly parameterised duration distributions (like [15])
  - Allows for direct inference about long-range dependencies
- Prior transition constraints $\Rightarrow$ a posteriori structured HMM
  - e.g. left-to-right infinite HMM

Approximations for life-long learning:

- Future work

---

Structured, dependent, infinite-dimensional transition distributions?
ISHSMM: Recipe

- Poisson process [17]
- Gamma process [17]
- SNΓPs [24]
- Structured, dependent, infinite-dimensional transition distributions [Huggins and Wood, to appear 2013.]
Gamma process [17] as Poisson process over $\Theta \otimes V \otimes [0, \infty)$ with rate / mean measure

$$\mu(\tilde{\Theta}, \tilde{V}, \tilde{S}) = \alpha(\tilde{\Theta}, \tilde{V}) \int_{\tilde{S}} \gamma^{-1} e^{-\gamma}$$

A draw from a Gamma process with

$$\alpha(\tilde{\Theta}, \tilde{V}) = c_0 H_{\theta}(\tilde{\Theta}) H_{v}(\tilde{V}).$$

[17] has the form

$$G = \sum_{m=1}^{\infty} \gamma_m \delta(\theta_m, v_m)$$

where $(\theta_m, v_m) \sim H_{\theta} \times H_{v}$. 
Non-disjoint “restricted projections” of Gamma processes are dependent Gamma processes (SNΓPs) [24]

\[ H_v = \text{Geom}(p) \]

\[ \begin{align*}
R_0 & \quad R_1 & \quad R_2 & \quad R_3 & \quad R_4 & \quad R_5 & \quad R_6 & \quad R_+ & \quad \cdots \\
0 & \quad v_0 & \quad v_1 & \quad v_2 & \quad v_3 & \quad v_4 & \quad v_5 & \quad v_6 & \quad v_7 & \quad v_8 & \quad v_9 & \quad \cdots
\end{align*} \]

\[ G_0 = \sum_{m \neq 0} \cdots, \quad \cdots, \quad G_4 = \sum_{m \neq 4} \gamma_m \delta_{\theta_m}, \quad \cdots \]
Normalized dependent GP draws are dependent Dirichlet process draws.\(^7\) In the ISEDHMM, DP draws are the dependent, structured, infinite-dimensional transition distributions.

\[
D_4 = \frac{G_4}{G_4(\Theta)}
\]

\[
= \frac{\sum_{m \neq 4} \gamma_m \delta_{\theta_m}}{\sum_{\theta \in \Theta} \sum_{m' \neq 4} \gamma_{m'} \delta_{\theta_{m'}}}
\]

\[
= \frac{\sum_{m \neq 4} \gamma_m \delta_{\theta_m}}{\sum_{m' \neq 4} \gamma_{m'}}
\]

\[
= \sum_{m \neq 4} \sum_{m' \neq 4} \frac{\gamma_m}{\gamma_{m'}} \delta_{\theta_m}
\]

\(^7\) a draw from a Dirichlet process [7] is an infinite sum of weighted atoms [26] where the weights sum to one.
Structured, dependent, infinite dimensional transition distributions $\pi_m$ can be formed from draws from DDPs [Huggins and Wood, to appear 2013.]
Inference: Beam Sampling

We employ the forward-filtering, backward slice-sampling approach for the IHMM of [29] and EDHMM of [Dewar, Wiggins and Wood, 2012], in which the state and duration variables $s$ and $r$ are sampled conditioned on auxiliary slice variables $u$.

Net result: efficient, always finite forward backward procedure for sampling latent states.
To illustrate SIHSMM learning on synthetic data, five hundred datapoints were generated using a 4 state EDHMM with Poisson duration distributions

$$\lambda = (10, 20, 3, 7)$$

and Gaussian emission distributions with means

$$\mu = (-6, -2, 2, 6)$$

all unit variance.
SIHSMM: Synthetic Data Results

[Graph showing time-series data with emission/mean values and number of states counts over time.]

Wood (Oxford, Engineering)
SIHSMM: Synthetic Data, State Duration Parameter Posterior

Wood (Oxford, Engineering)
SIHSM/MM: Synthetic Data, State Mean Posterior

![Graph showing emission mean distribution with count on the y-axis and emission mean on the x-axis. The graph displays multiple peaks representing different states or categories.](image-url)
Application from [15]
SIHSMM → Left-to-Right Infinite HSMM

Wood (Oxford, Engineering)

June, 2013 50 / 66
Lessons and Questions

- Improving prediction must require latent variable models, right?
  - Do we bias towards the right ones?
  - Do existing priors probably penalize model complexity too harshly?
  - Existing latent variable models generally don’t smooth like the SM.
    - How do we build models like this?
    - Would introducing such smoothing help prediction?

- Do we need more data/time?
  - What is the regime in which complex models will surpass simpler models for in-painting-like tasks?
  - Is there a convergence rate theory we’re missing?

- Can we “smooth” downward? How?

- Early observation: award maximization and predictive loss minimization seem to result in opposite orderings
Comparative Predictive Performance for Reinforcement Learning

Rewards for Gridworld

From [Doshi-Velez, Pfau, Wood and Roy, in sub.]
The End

Thank you.


Thoughts and Questions

- Bayesian nonparametric estimators fundamentally have the wrong storage asymptotics.
  - Should we abandon them?
  - Is the class of parametric estimators derived from approximate nonparametric estimators interesting?
  - Is there a provable advantage to or optimal strategy for forgetting in Bayesian nonparametric models?

- How do we distinguish between the expressivity of models of infinite complexity?

- How do we learn $\sigma$, or, more generally, how to we compute in a model with countable back-offs?

- What happens when the input vocabulary gets very big and the typical input vector is high-dimensional?
Musings

Models like the SM are tough to beat in in-painting-type applications because only local specificity matters.

- Text in-painting

  She put it ___ the bank.

- Image in-painting
Hypothesis: The Right Latent Variable World State Representation Should Matter

- Text in-painting
  - She just pulled a fish from the river.
  - She just received her allowance.

  *She put it ___ the bank.*

- Image in-painting

For planning, arguably, only the latent state matters.
Figure: Multiple target tracking (ANTS) [Neiswanger and Wood, in sub.]
Figure: Multiple target tracking (PETS2000) [Neiswanger and Wood, in sub.]
Virtual Perspective Inference

Figure: icareti [Brooks, Zhang, Forde, and Wood, in sub.]