Learning Probabilistic Automata

Borja Balle

LARCA. Laboratory for Relational Algorithmics, Complexity and Learning

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Tokyo Institute of Technology — November 7, 2010

Problem Statement

Problem

Given a bag of sequences S, obtain a generative model $\hat{D}$ that models its distribution

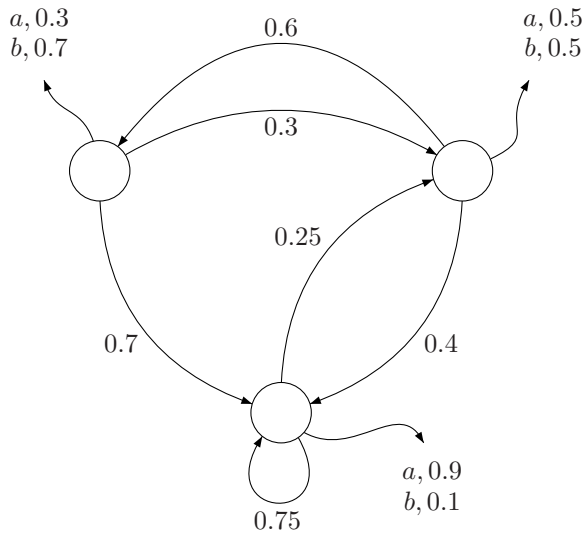
$$S = (ac, bbab, bcaaaa, acaab, bbab, caca, bacaaa, \ldots)$$

Why do we care?

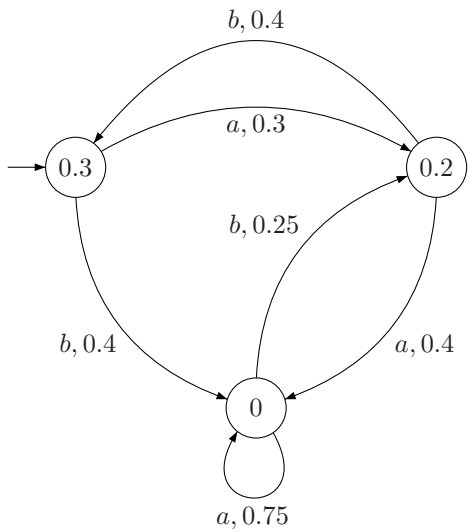
- *Generative models* can be used for: simulation, classification, prediction, clustering, ...
- *Sequential data* appears in many applications: natural language, bioinformatics, time series, ...

Two Generative Models

HMM
Hidden Markov Models



PDFA
Probabilistic Deterministic Finite Automata



- Differences in emission policy and stopping criteria
- Some tasks can be solved more efficiently in PDFA, e.g. state sequence prediction, probability computing and training

Training vs Learning

Given S, two tasks can be considered

- Training — given a structure, estimate its parameters from S
- Learning — obtain structure *and* parameters from S

Standard algorithms for these tasks

	Training	Learning
PDFA	Counting	State-Merging
HMM	Expectation-Maximization	Spectral Methods

Outline

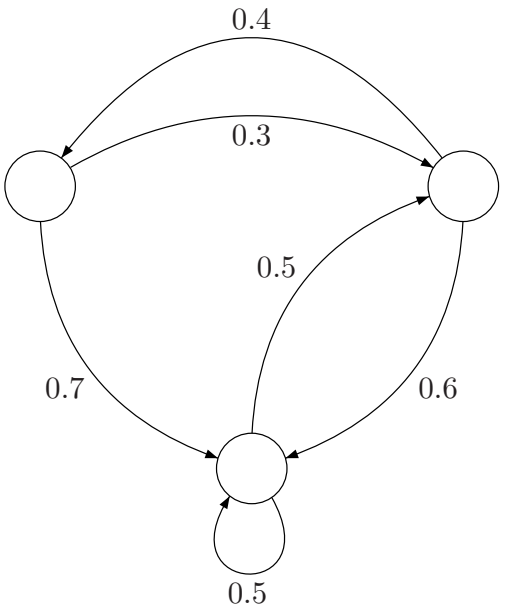
Outline

Distributions over Sequences

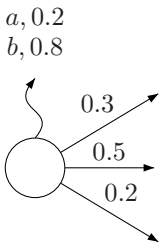
- Let Σ be a finite alphabet
- We are interested in modeling probability distributions over sets of sequences over Σ, in particular over Σ^k and Σ^*
- Several models exist, such as, in increasing complexity,
  - n-gram models
  - probabilistic automata
  - hidden Markov models
  - probabilistic context-free grammars

Markovian Hidden State Models

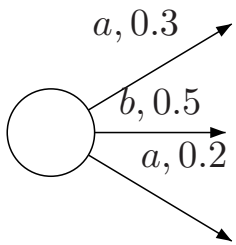
- Generative models with a hidden state transition structure given by a directed graph with probabilities
- Depending on emission details we obtain different models
- With stopping probabilities, distributions over Σ^*, without stopping probabilities, distributions over Σ^k



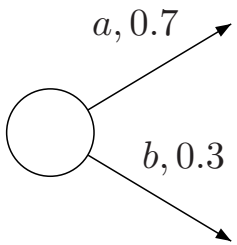
HMM



PNFA



PDFA



Relation Between Models

Regardless of stopping probabilities

- HMM and PNFA are equivalent
  - HMM n states $\longrightarrow$ PNFA n states
  - PNFA n states $\longrightarrow$ HMM n^2 states
- PDFA are strictly weaker — some PNFA cannot be expressed by any PDFA

With stopping probabilities

- Any PNFA can be arbitrarily approximated by a large enough PDFA

Two Particular Models

For the rest of the talk...

HMM

- Emission on states
- Without stopping probabilities
- Notation: vector π of initial probabilities, matrices T, O and $P = O\,T\,\mathrm{diag}(\pi)\,O^{\mathrm{T}}$

$$T_{ij} = \mathbb{P}[q_{t+1} = i | q_t = j] \quad O_{ij} = \mathbb{P}[x_t = i | q_t = j] \quad P_{ij} = \mathbb{P}[x_{t+1} = i, x_t = j]$$

PDFA

- Emission on transitions
- With stopping probabilities
- Notation: for state $q \in Q$ and symbol $\sigma \in \Sigma$, probability of emission and transition to $\tau(q, \sigma)$ is $\gamma(q, \sigma)$. Stopping probability is $\gamma(q, \xi)$

Distinguishability of a PDFA

- A PDFA defines a probability distribution D over Σ^*
- For each state q we define D^q as the probability distribution obtained by setting q as the initial state of the PDFA
- An important parameter related to the learnability of a PDFA is its distinguishability defined as

$$\mu = \min_{q \neq q'} \mathsf{L}_\infty(D^q, D^{q'}) = \min_{q \neq q'} \max_{x \in \Sigma^*} \left| D^q(x) - D^{q'}(x) \right| \quad .$$

- Intuitively, μ quantifies how similar are the different states in the PDFA
- There exist PDFA with n states and $1/\mu = 2^{\Theta(n)}$ — worst case, very similar states

Outline

Training vs Learning

Training

An algorithm *trains* the parameters θ in a parametric family D_θ over X if for any ϵ and δ, given samples from an unknown distribution D, it outputs $D_{\hat{\theta}}$ such that $\mathrm{KL}(D|D_{\hat{\theta}}) - \inf_\theta \mathrm{KL}(D|D_\theta) \leqslant \epsilon$, with probability larger than $1 - \delta$ using time and sample size in $\mathrm{poly}(1/\epsilon, 1/\delta, \mathrm{size}(D_\theta))$

Probably Approximately Correct Learning

An algorithm *PAC learns* a family of distributions $\mathcal{D}$ over X if for any ϵ and δ, and for every $D \in \mathcal{D}$, given samples from D, it outputs a distribution $\hat{D} \in \mathcal{D}$ such that $\mathrm{KL}(D|\hat{D}) \leqslant \epsilon$, with probability larger than $1 - \delta$ using time and sample size in $\mathrm{poly}(1/\epsilon, 1/\delta, \mathrm{size}(D))$

Note: sometimes definitions use L_1 instead of KL

Measuring Model Size

- n — number of states
- $|\Sigma|$ — alphabet size
- k — length of strings (if distribution over Σ^k)
- L — expected length (if distribution over Σ^*)
- μ — distinguishability between pairs of states (for PDFA)
- σ_O, σ_P — nth singular values of matrices O and P (for HMM)

Statistical and Computational Complexity

HMM training (Abe-Warmuth '92) — a single HMM over Σ^k with n states

- HMM can be trained with $\tilde{O}(n^2|\Sigma|k^2\epsilon^{-2}\log 1/\delta)$ *samples*
- HMM cannot be trained in *time* polynomial in $|\Sigma|$ unless RP $=$ NP

PDFA learning (Kearns et al. '94, Clark-Thollard '04, B et al. '10)

- PDFA cannot be learned in *time* polynomial in n unless noisy parities can be learned in polynomial time
- PDFA cannot be learned with a number of *samples* independent of L
- PDFA cannot be learned in *time* independent of μ using L_∞-based tests

Algorithms

Training HMM

- Expectation-Maximization — widely used, but no convergence guarantees
- Input: structure to be trained and S

Learn PDFA over Σ^*

- State-Merging algorithm — learns with PAC guarantees wrt KL
- Input: n, δ and S
- Works with $\tilde{O}(n^3|\Sigma|^3 L^9 \mu^{-2} \epsilon^{-6} \log 1/\delta)$ samples

Learn HMM over Σ^k

- Spectral algorithm — learns with PAC guarantees wrt L_1
- Input: n and S
- Works with $O(n|\Sigma|k^2 \sigma_O^{-2} \sigma_P^{-4} \epsilon^{-2} \log 1/\delta)$ samples
- Requires some assumptions on the target

How to Read Bounds

Lower bounds

- Worst case — there is something in the class which is hard to train/learn
- But, maybe in practice you never find that targets
- Indicate that complexity of target does no only depend on n and $|\Sigma|$, but on the particular structure

Upper bounds

- Possibly unreliable *quantitative* information — may be loose due to prove techniques
- But, provide useful *qualitative* insight into what makes a target difficult to train/learn
- Algorithms might work well in practice with fewer examples
- Proves algorithm performance improves with larger samples

Outline

A Long Story Made Short

- State-merging algorithms originated in DFA learning
  - Empirically — Trakhtenbrot-Barzdin '73
  - With queries — Angluin '87
  - In the limit — Oncina-García '92
- Later adapted for PDFA learning
  - In the limit — Carrasco-Oncina '99
  - PAC for Acyclic PDFA wrt KL — Ron et al. '98
  - PAC for all PDFA wrt KL — Clark-Thollard '04
- Further developments improved on Clark-Thollard's algorithm
  - PAC wrt L_1 — Palmer-Goldberg '05
  - Smaller sample and less parameters — Castro-Gavaldà '08

Basic Ideas of the State-Merging Method

- In a deterministic automaton each prefix leads to a single state
- Thus, cluster prefixes in the sample using information from the distribution over suffixes
- A statistical test between empirical suffix distributions is all you need
- Build a transition structure using these clusters
- Organize computations cleverly for efficiency

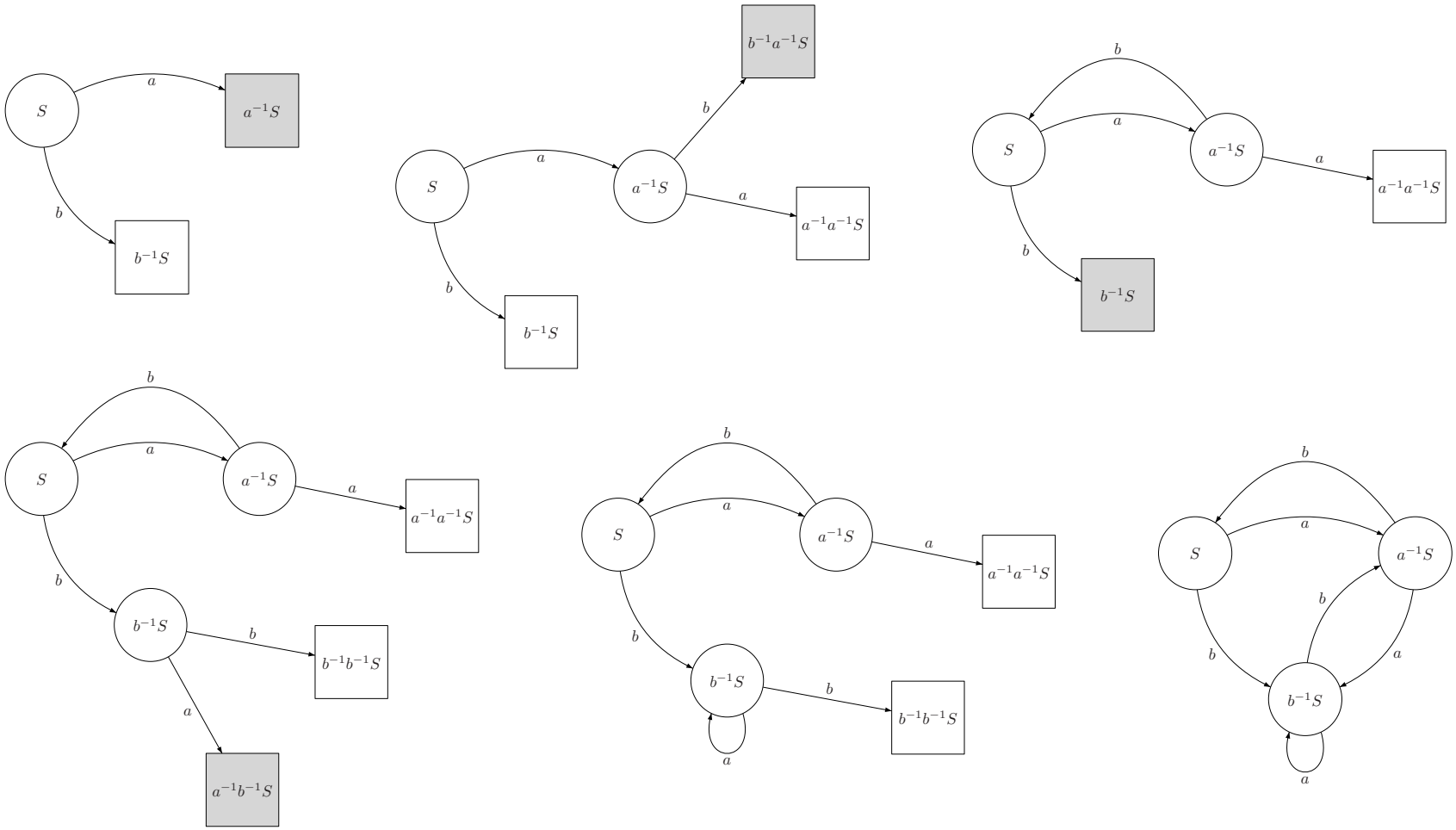
Pseudocode for Graph Identification

Input: S, n, Σ, δ
$\texttt{Safe}(S)$; $\forall \sigma \in \Sigma$ $\texttt{Candidate}(\sigma^{-1}S)$;
while $|\texttt{Safe}| < n \wedge |\texttt{Candidate}| > 0$ **do**
  $c \leftarrow \texttt{Select}(\texttt{Candidate})$;
  if $\exists s \in \texttt{Safe}$ $\texttt{Test}_{n,\delta}(s \approx c)$ **then** $\texttt{Merge}(s, c)$;
  else $\texttt{Safe}(c)$; $\forall \sigma \in \Sigma$ $\texttt{Candidate}(\sigma^{-1}c)$;

Watch out for details on

- What statistical test to use
- Candidate selection policy
- With which safe to merge in case of multiple choices
- To update or not the bags in safes after a merge
- What to do with remaining candidates

Example Run with $\Sigma = \{a, b\}$ and $n = 4$



Test sequence: $S \not\approx a^{-1}S$, $S \approx b^{-1}a^{-1}S$, $S \not\approx b^{-1}S$, $a^{-1}S \not\approx b^{-1}S$, $b^{-1}S \approx a^{-1}a^{-1}S$, $b^{-1}S \approx b^{-1}b^{-1}S$

Why Does It Work?

- Use a test based on L_∞ with theoretical guarantees over false positive and false negative rate
- It is enough to find a subgraph of the target graph where all *important* states and transitions are correctly represented, thanks to (Carrasco '97)

$$\mathrm{KL}(A|A') = \sum_{q\in Q}\sum_{q'\in Q'} W(q,q') \sum_{\sigma\in\Sigma\cup\{\xi\}} \gamma(q,\sigma)\log\frac{\gamma(q,\sigma)}{\gamma'(q',\sigma)}$$

- Train the structure by counting
- To account for rare unseen events, apply a smoothing procedure — unnecessary if using L_1 instead of KL
- Altogether, $\tilde{O}(n^3|\Sigma|^3L^9\epsilon^{-6}\mu^{-2}\log 1/\delta)$ samples and similar time (Clark-Thollard '04, Castro-Gavaldà '08)

Outline

Recent Developments

Beyond the basic state-merging algorithm we can:

- Extend state-merging algorithms to learn more general distributions with a deterministic structure
- Use spectral techniques to learn non-deterministic structures under certain assumptions

Learning Timed Structures with State-Merging Methods

- An Asynchronous PDFA is a PDFA where each transition is assigned an exponential random variable $\mathrm{Exp}(\lambda_{q,\sigma})$ that models the *time* taken by the transition
- AsPDFA define distributions over *timed strings* in $(\Sigma \times \mathbb{R})^*$, e.g.

$$x = (b, 1.24)(a, 0.5)(a, 6.2)(b, 1.01)(b, 3.1)$$

Theorem (B et al. '10)

The class of AsPDFA can be PAC learned with samples of size at most

$$\tilde{O}\left(\frac{n^5 L^9 |\Sigma|^3}{\epsilon^6 \mu^2} \cdot \ln\frac{1}{\delta} \cdot \left(1 + \ln^3 \frac{\lambda_{\max}}{\lambda_{\min}}\right)\right) \quad .$$

Learning HMM with Spectral Techniques

- Spectral techniques originate in system identification for *continuous* state spaces
- PAC algorithm for learning HMM wrt L_1 (Hsu et al. '09)
- Uses SVD of matrices computed from samples — beware, analysis ignores possible numerical errors
- Matrices have $|\Sigma|^2$ entries — independent of n, k and $|S|$
- Requires assumptions
  - π is stationary distribution
  - $n \leqslant |\Sigma|$
- Works with $O(n|\Sigma|k^2\sigma_O^{-2}\sigma_P^{-4}\epsilon^{-2}\log 1/\delta)$ samples and similar time

Outline

Take Home Message

Got models?

- Details matter: stopping policy, initial distribution, emission policy
- Model variations likely to yield interesting new problems
- PDFA can easily incorporate more information — useful for applications, e.g. time

Got samples?

- Let an algorithm find structure for you
- Sample from a process that begins and ends $\Rightarrow$ learn PDFA
- Sample from a running process $\Rightarrow$ learn HMM

Got time?

- Missing good open source implementation of state-merging
- Lots of implementation issues — need special data structures for efficiency

Learning Probabilistic Automata

Borja Balle

LARCA. Laboratory for Relational Algorithmics, Complexity and Learning

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Tokyo Institute of Technology — November 7, 2010