The Sequence Memoizer

Frank Wood	UCL
Cedric Archambeau	Gatsby
Jan Gasthaus	HKUST
Lancelot James	Gatsby
Yee Whye Teh	

Executive Summary

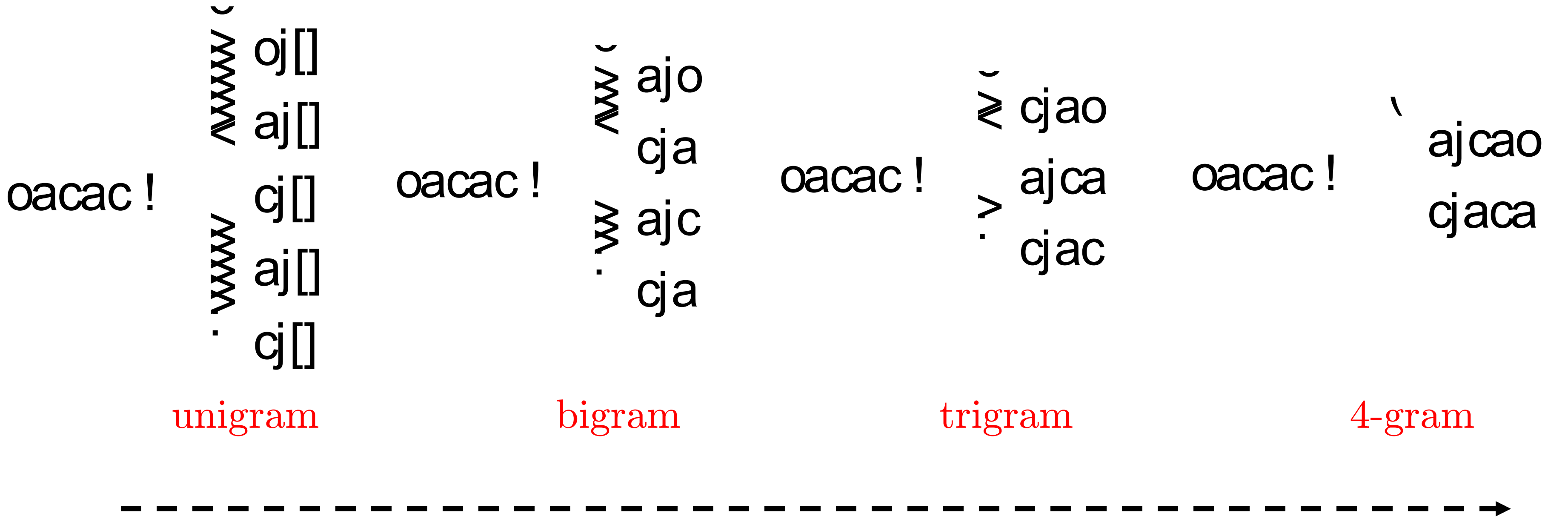
- Model
  - Smoothing Markov model of discrete sequences
  - Extension of hierarchical Pitman Yor process [Teh 2006]
    - Unbounded depth (context length)
- Algorithms and estimation
  - Linear time suffix-tree graphical model identification and construction
  - Standard Chinese restaurant franchise sampler
- Results
  - Maximum contextual information used during inference
  - Competitive language modelling results
    - Limit of n-gram language model as $n \rightarrow \infty$
  - Same computational cost as a Bayesian interpolating 5-gram language model

Executive Summary

- Uses
  - Any situation in which a low-order Markov model of discrete sequences is insufficient
  - Drop in replacement for smoothing Markov model
- Name?
  - "A Stochastic Memoizer for Sequence Data" $\rightarrow$ Sequence Memoizer (SM)
    - Describes posterior inference [Goodman et al '08]

Statistically Characterizing a Sequence

- Sequence Markov models are usually constructed by treating a sequence as a set of (exchangeable) observations in fixed-length contexts



Increasing context length / order of Markov model

Decreasing number of observations

Increasing number of conditional distributions to estimate (indexed by context)

Increasing power of model

Finite Order Markov Model

$$
\begin{aligned}
P(x_{1:N}) &= \prod_{i=1}^{N} P(x_i|x_1,\ldots,x_{i-1}) \\
&\approx \prod_{i=1}^{N} P(x_i|x_{i-n+1},\ldots,x_{i-1}); \quad n = 2 \\
&= P(x_1)P(x_2|x_1)P(x_3|x_2)P(x_4|x_3)\ldots
\end{aligned}
$$

- Example

$$
\begin{aligned}
P(oacac) &= P(o)P(a|o)P(c|a)P(a|c)P(c|a) \\
&= G_{[]}(o)G_{[o]}(a)G_{[c]}(a)G_{[a]}(c)G_{[c]}(a)
\end{aligned}
$$

Learning Discrete Conditional Distributions

- Discrete distribution $\leftrightarrow$ vector of parameters

$$G_{[u]} = [\pi_1, \ldots, \pi_K], K \in |\Sigma|$$

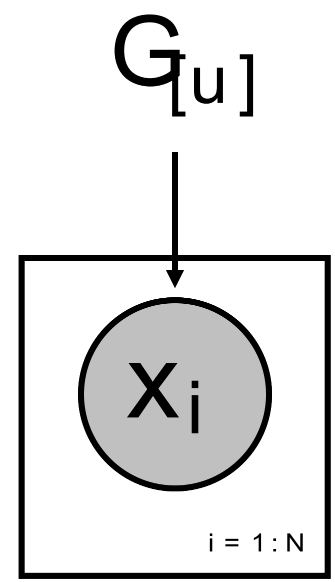
- Counting / Maximum likelihood estimation
  - Training sequence $x_{1:N}$

$$\hat{G}_{[u]}(X = k) = \hat{\pi}_k = \frac{\#\{uk\}}{\#\{u\}}$$

  - Predictive inference

$$P(X_{n+1} | x_1 \ldots x_N) = \hat{G}_{[u]}(X_{n+1})$$

- Example
  - Non-smoothed unigram model ($\mathbf{u} = \epsilon$)



Bayesian Smoothing

- Estimation

$$P(G_{[u]} \mid x_{1:n}) \propto P(x_{1:n} \mid G_{[u]}) P(G_{[u]})$$

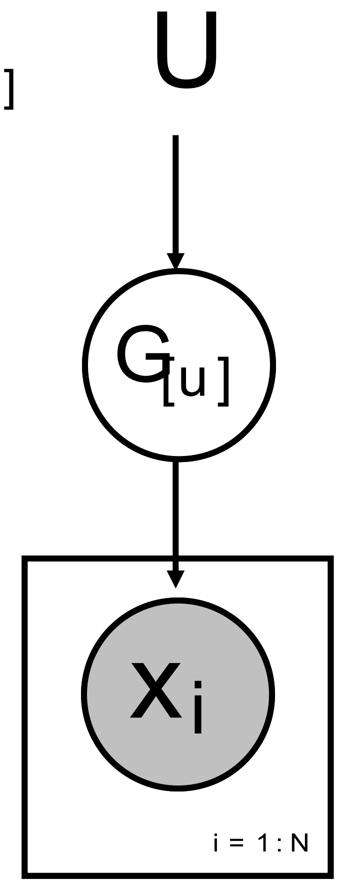
- Predictive inference

$$P(X_{n+1} \mid x_{1:n}) = \int P(X_{n+1} \mid G_{[u]}) P(G_{[u]} \mid x_{1:n}) dG_{[u]}$$

- Priors over distributions

$$G_{[u]} \sim \mathrm{Dirichlet}(U), \quad G_{[u]} \sim \mathrm{PY}(d, c, U)$$

- Net effect
  - Inference is "smoothed" w.r.t. uncertainty about unknown *distribution*
- Example
  - Smoothed unigram ($\mathbf{u} = \epsilon$)



A Way To Tie Together Distributions

$$G_{[u]} \quad \text{»} \quad PY(\overbrace{d}^{\text{discount}}; \overbrace{c}^{\text{concentration}}; \underbrace{G_{[\text{¾}(u)]}}_{\text{base distribution}})$$

$$x_i \quad \text{»} \quad G_{[u]}$$

- Tool for tying together related distributions in hierarchical models
- Measure over measures
- Base measure is the "mean" measure

$$E[G_{[u]}(dx)] = G_{[\text{¾}(u)]}(dx)$$

- A distribution drawn from a Pitman Yor process is related to its base distribution
  - (equal when $c = \infty$ or $d = 1$)

[Pitman and Yor '97]

Pitman-Yor Process Continued

- Generalization of the Dirichlet process ($d = 0$)
  - Different (power-law) properties
  - Better for text [Teh, 2006] and images [Sudderth and Jordan, 2009]
- Posterior predictive distribution

Can't actually do this integral this way

$$
\begin{aligned}
P(X_{N+1} | x_{1:N}; c, d) &\approx \int P(x_{N+1} | G_{[u]}) P(G_{[u]} | x_{1:N}; c, d) dG_{[u]} \\
&= E\left[\frac{\sum_{k=1}^{K} (m_k - d) I(\phi_k = X_{N+1})}{c + N} + \frac{c + dK}{c + N} G_{[\sigma(u)]}(X_{N+1}) \right]
\end{aligned}
$$

- Forms the basis for straightforward, simple samplers
- Rule for stochastic memoization

Hierarchical Bayesian Smoothing

- Estimation

$$\Theta = \{G_{[u]}, G_{[v]}, G_{[w]}\}, \quad w = \sigma(u) = \sigma(v)$$

$$P(\Theta | x_{1:N}) \propto P(x_{1:N} | \Theta) P(\Theta)$$

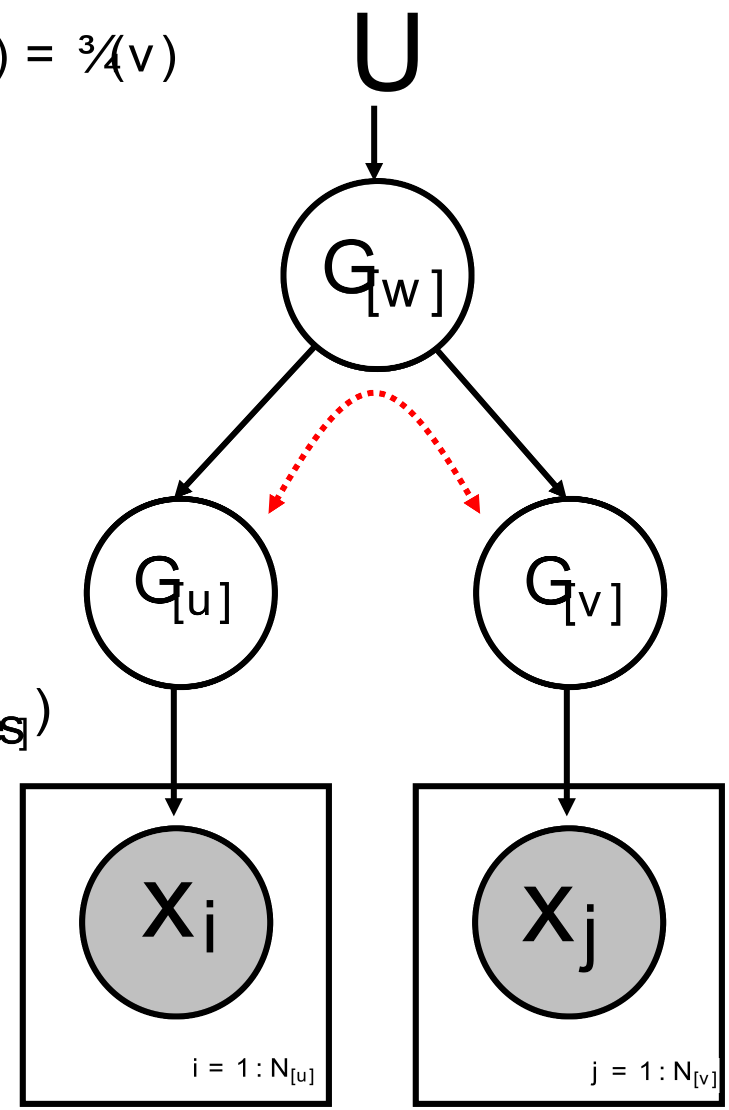
- Predictive inference

$$P(X_{N+1} | x_{1:N}) = \int P(X_{N+1} | \Theta) P(\Theta | x_{1:N}) d\Theta$$

- Naturally related distributions tied together

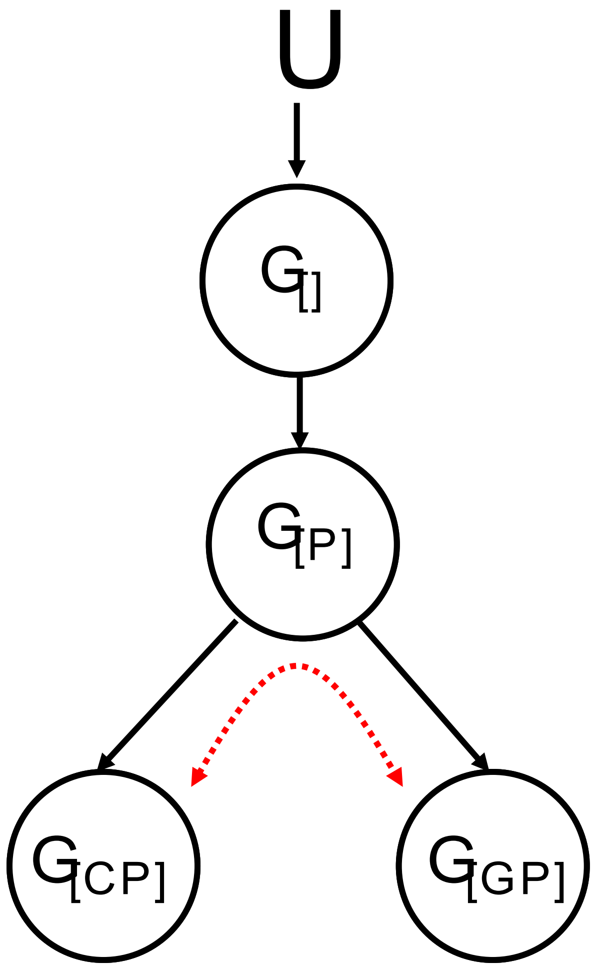
$$G_{[\text{the United States}]} \sim PY(d, c, G_{[\text{United States}]})$$

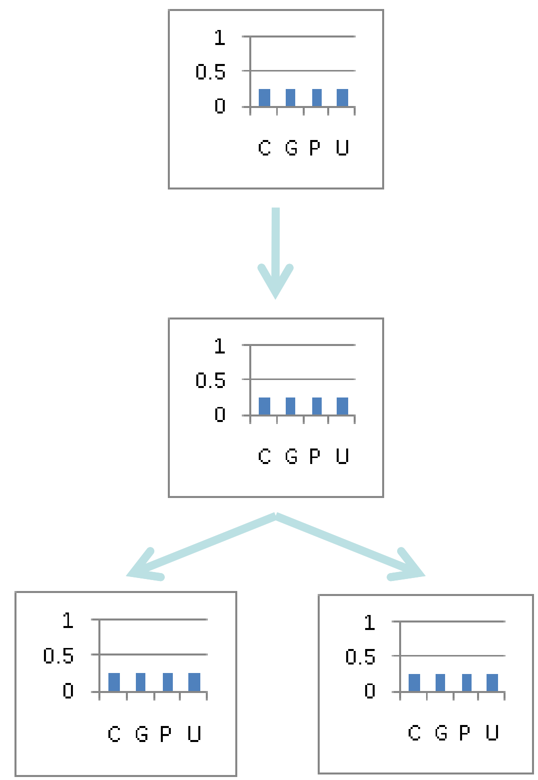
- Net effect
  - Observations in one context affect inference in other context.
  - Statistical strength is shared between similar contexts
- Example
  - Smoothing bi-gram ($\mathbf{w} = \epsilon$, $\mathbf{u},\mathbf{v} \in \Sigma$)



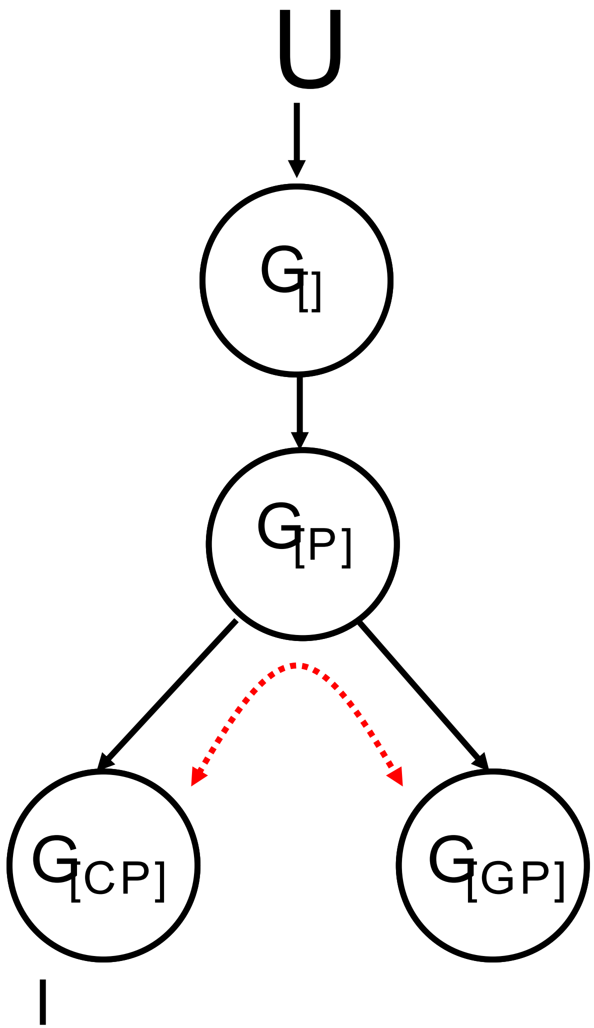
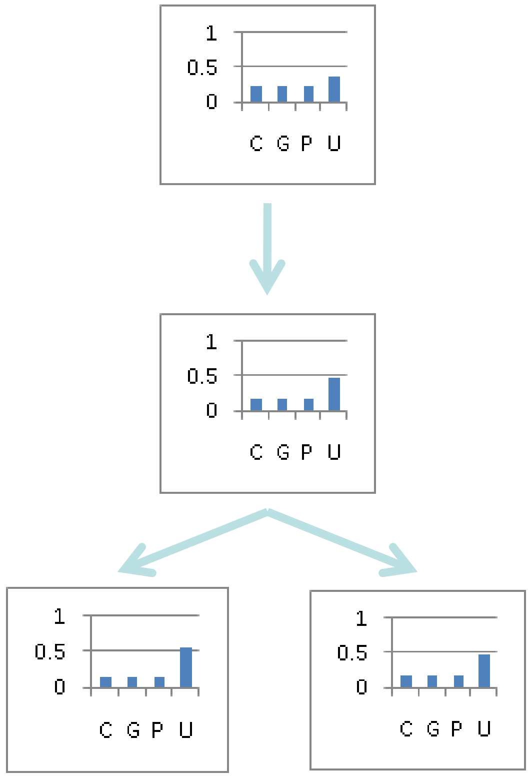
SM/HPYP Sharing in Action

Observations	Conditional Distributions	Posterior Predictive Probabilities





CRF Particle Filter Posterior Update

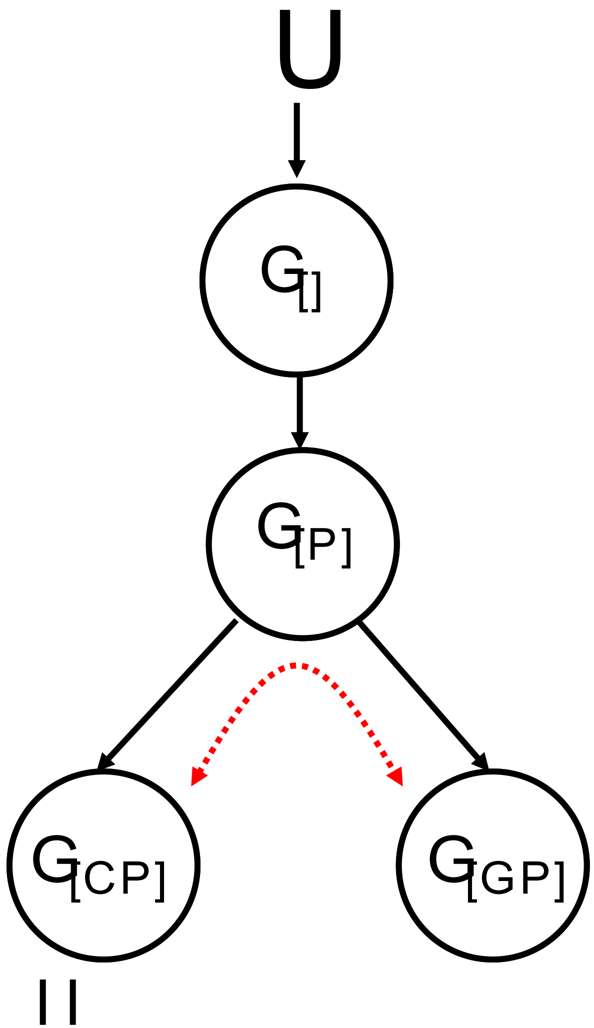
Observations	Conditional Distributions	Posterior Predictive Probabilities
CPU		

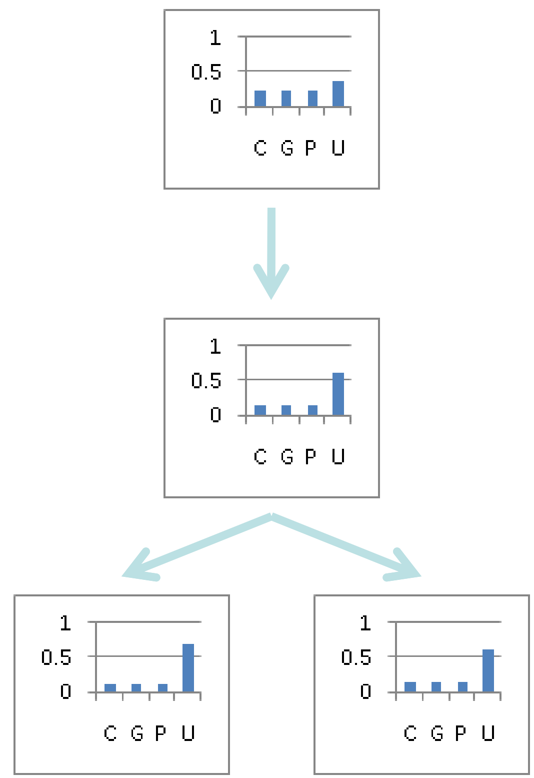
CRF Particle Filter Posterior Update

Observations	Conditional Distributions	Posterior Predictive Probabilities

CPU

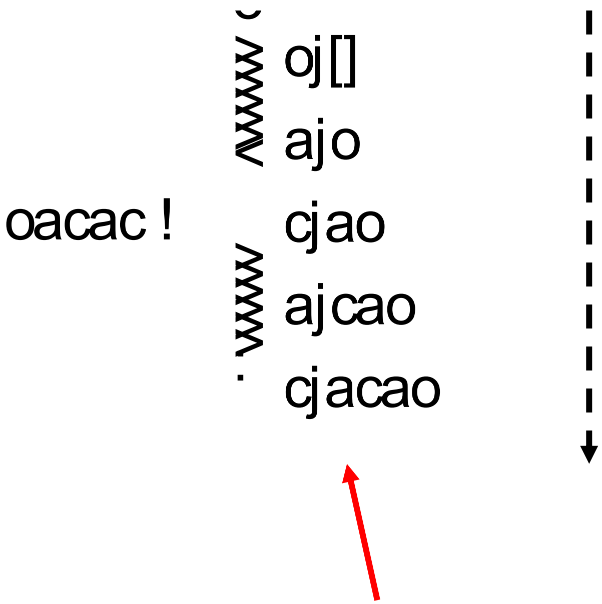
CPU





Alternative Sequence Characterization

- A sequence can be characterized by a set of *single* observations in unique contexts of growing length



Increasing context length

Always a single observation

Foreshadowing: all suffixes of the string “cacao”

"Non-Markov" Model

$$P(x_{1:N}) = \prod_{i=1}^{N} P(x_i | x_1, \ldots, x_{i-1})$$
$$= P(x_1)P(x_2|x_1)P(x_3|x_2, x_1)P(x_4|x_3, \ldots, x_1)\ldots$$

- Example

$$P(oacac) = P(o)P(a|o)P(c|oa)P(a|oac)P(c|oaca)$$

- Smoothing essential
  - Only one observation in each context!
- Solution
  - Hierarchical sharing ala HPYP

HPYP LM Sharing Architecture

- Share statistical strength between sequentially related predictive conditional distributions
  - Estimates of highly specific conditional distributions

$$G_{[\text{was on the}]}$$

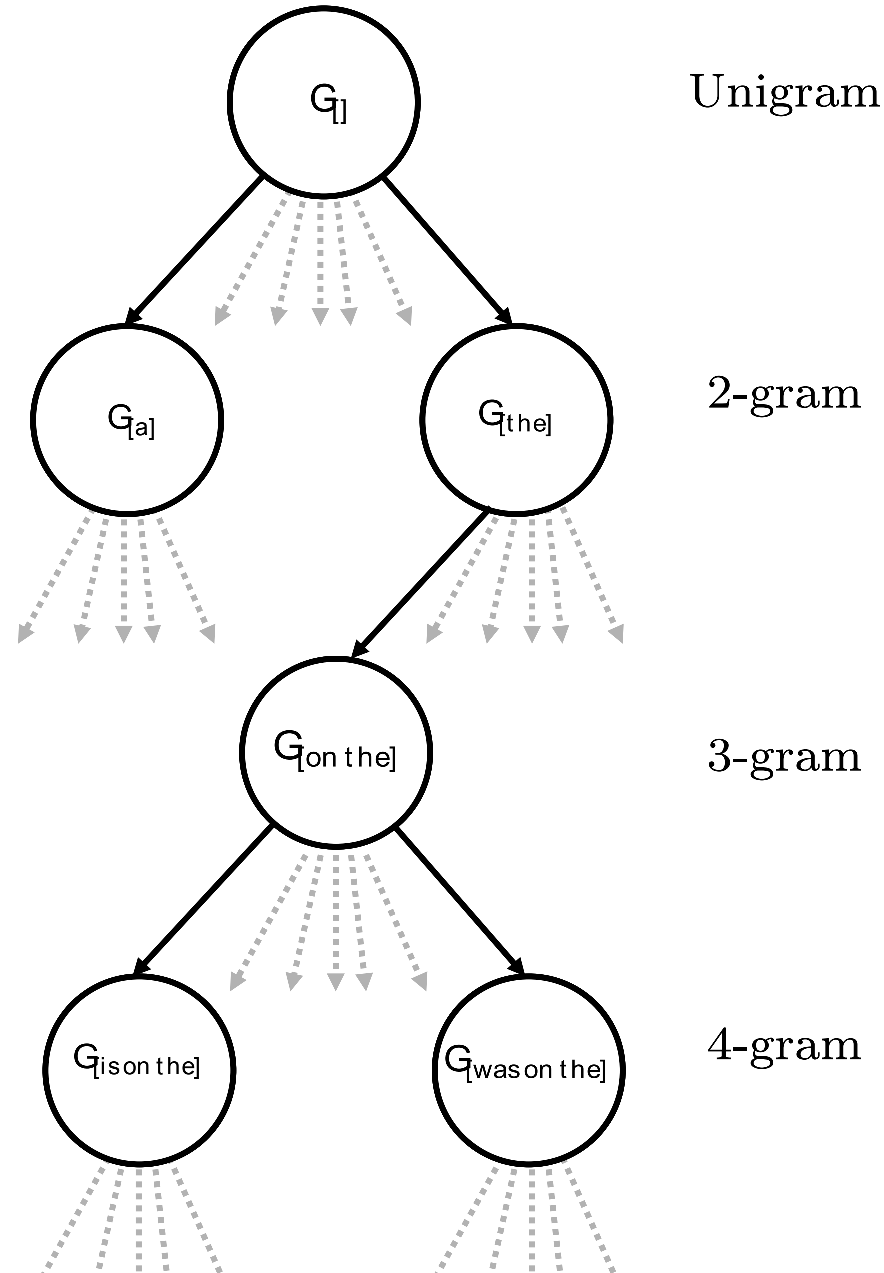
  - Are coupled with others that are related

$$G_{[\text{is on the}]}$$

  - Through a single common, more-general shared ancestor

$$G_{[\text{on the}]}$$

- Corresponds intuitively to back-off



Hierarchical Pitman Yor Process

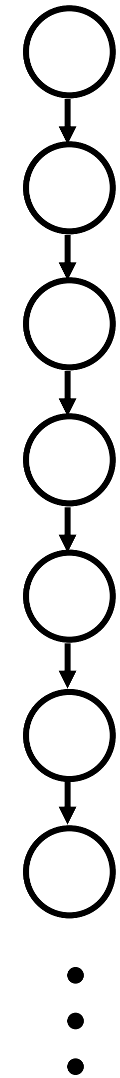
$$
\begin{array}{rcl}
G_{[]} \; \text{j} \; d_0; U & \text{»} & PY(d_0; 0; U) \\
G_{[u]} \; \text{j} \; d_{\text{j}u\text{j}}; G_{[\text{¾}(u)]} & \text{»} & PY(d_{\text{j}u\text{j}}; 0; G_{[\text{¾}(u)]}) \\
x_{\text{i}} \; \text{j} \; x_{1:\text{i}\,\text{¡}\,1} = u & \text{»} & G_{[u]} \\
i = 1; :::; T & & \\
8u \; 2 \; \text{§}^{n\,\text{¡}\,1} & &
\end{array}
$$

- Bayesian generalization of smoothing n-gram Markov model
- Language model : outperforms interpolated Kneser-Ney (KN) smoothing
- Efficient inference algorithms exist
  - [Teh, '06; Teh, Kurihara, Welling, '08]
- Sharing between contexts that differ in most distant symbol only
- Finite depth

[Teh '06]

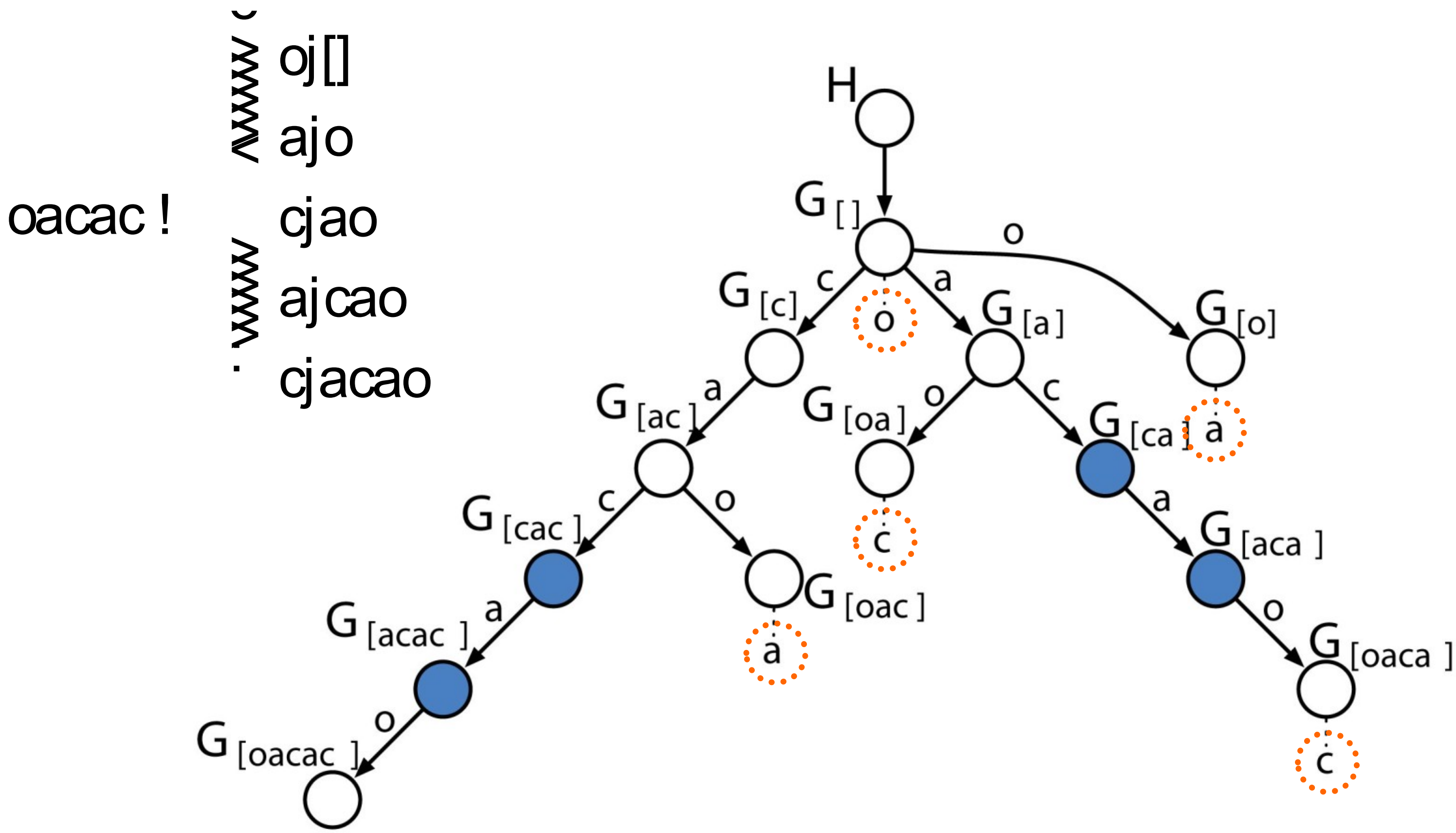
Sequence Memoizer

$$
\begin{aligned}
G_{[]} \mid d_0, U &\sim \mathrm{PY}(d_0, 0, U) \\
G_{[u]} \mid d_{|u|}, G_{[\sigma(u)]} &\sim \mathrm{PY}(d_{|u|}, 0, G_{[\sigma(u)]}) \\
x_i \mid x_{1:i-1} = u &\sim G_{[u]} \\
i &= 1, \ldots, T \\
\forall u &\in \Sigma^{+}
\end{aligned}
$$



- Eliminates Markov order selection
- Always uses full context when making predictions
- Linear time, linear space (in length of observation sequence) graphical model identification
- Performance is limit of n-gram as $n \to \infty$
- Same or less overall cost as 5-gram interpolating Kneser Ney

Graphical Model Trie



Observations

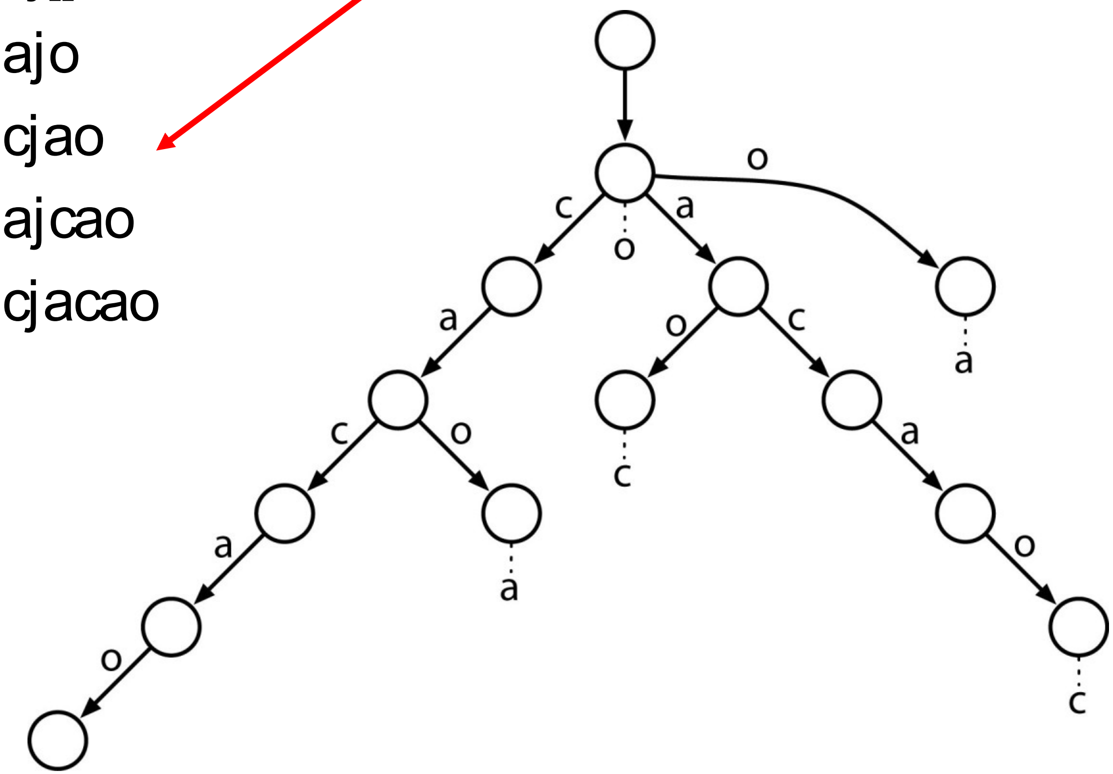
Latent conditional distributions with Pitman Yor priors / stochastic memoizers

Suffix Trie Datastructure

All suffixes of the string “cacao”

oacac !

oj[]
ajo
cjao
ajcao
cjacao



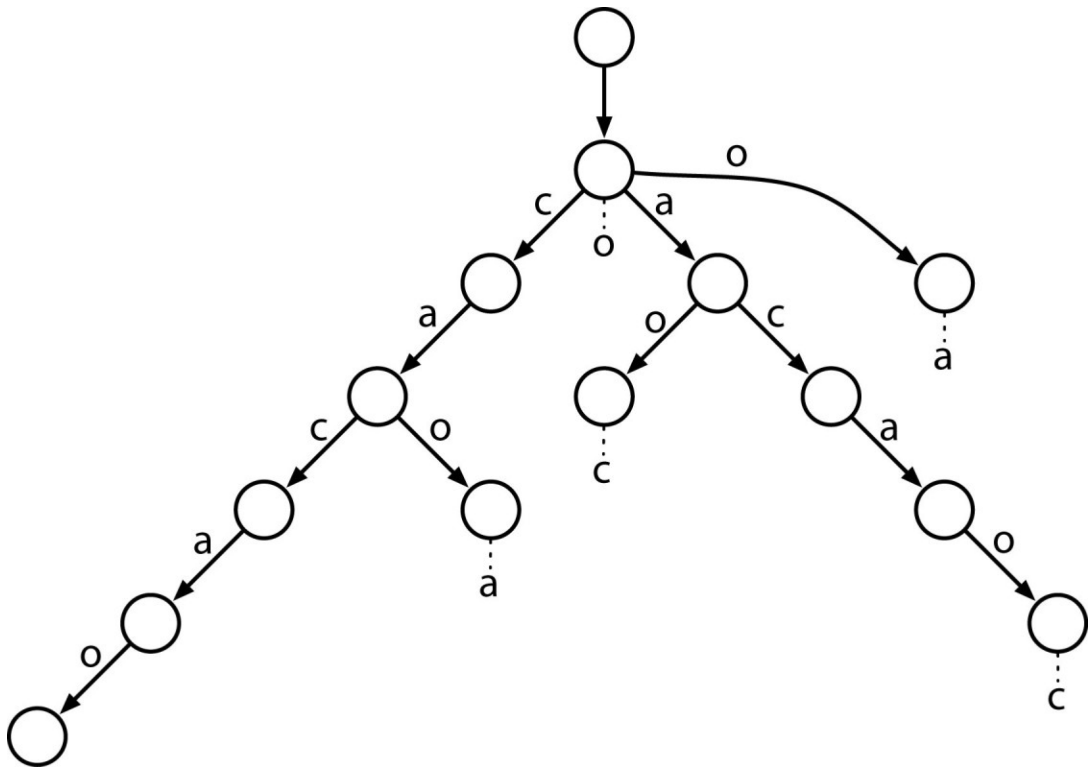
Suffix Trie Datastructure

- Deterministic finite automata that recognizes all suffixes of an input string.
- Requires $O(N^2)$ time and space to build and store [Ukkonen, 95]
- Too intensive for any practical sequence modelling application.

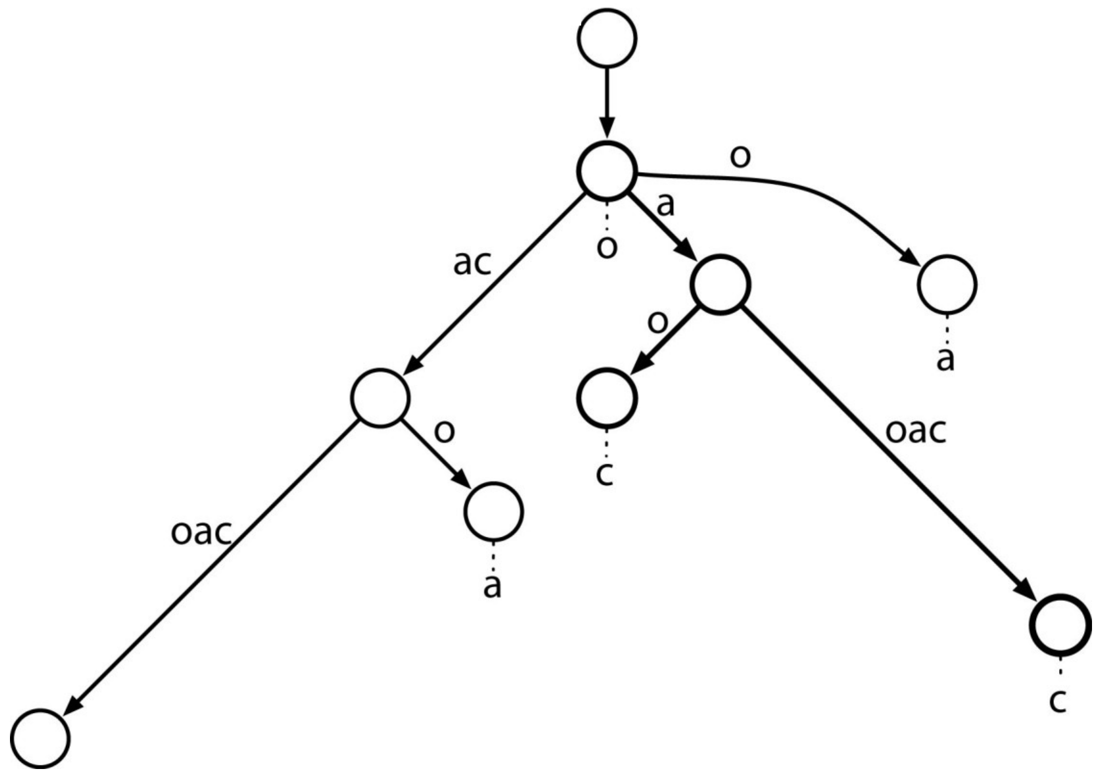
Suffix Tree

- Deterministic finite automata that recognizes all suffixes of an input string
- Uses path compression to reduce storage and construction computational complexity.
- Requires only O(N) time and space to build and store [Ukkonen, 95]
- Practical for large scale sequence modelling applications

Suffix Trie Datastructure



Suffix Tree Datastructure



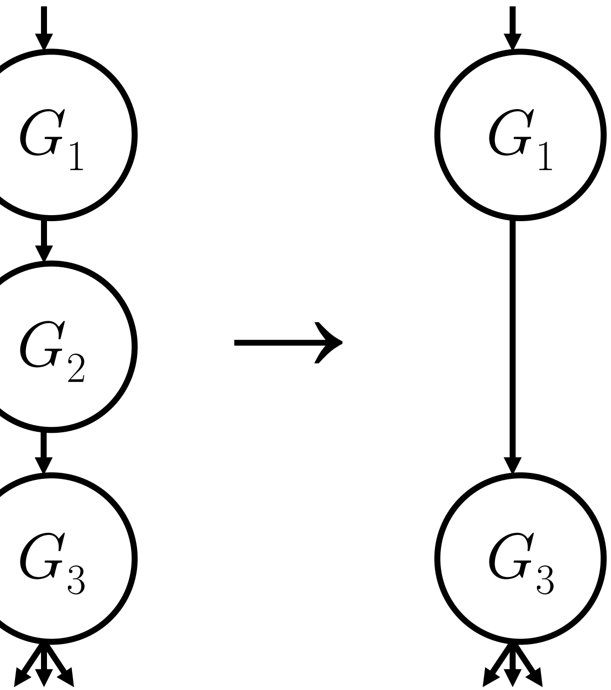
Graphical Model Identification

- This is a graphical model transformation under the covers.
- These compressed paths require being able to analytically marginalize out nodes from the graphical model
- The result of this marginalization can be thought of as providing a different set of caching rules to memoizers on the path-compressed edges

Marginalization

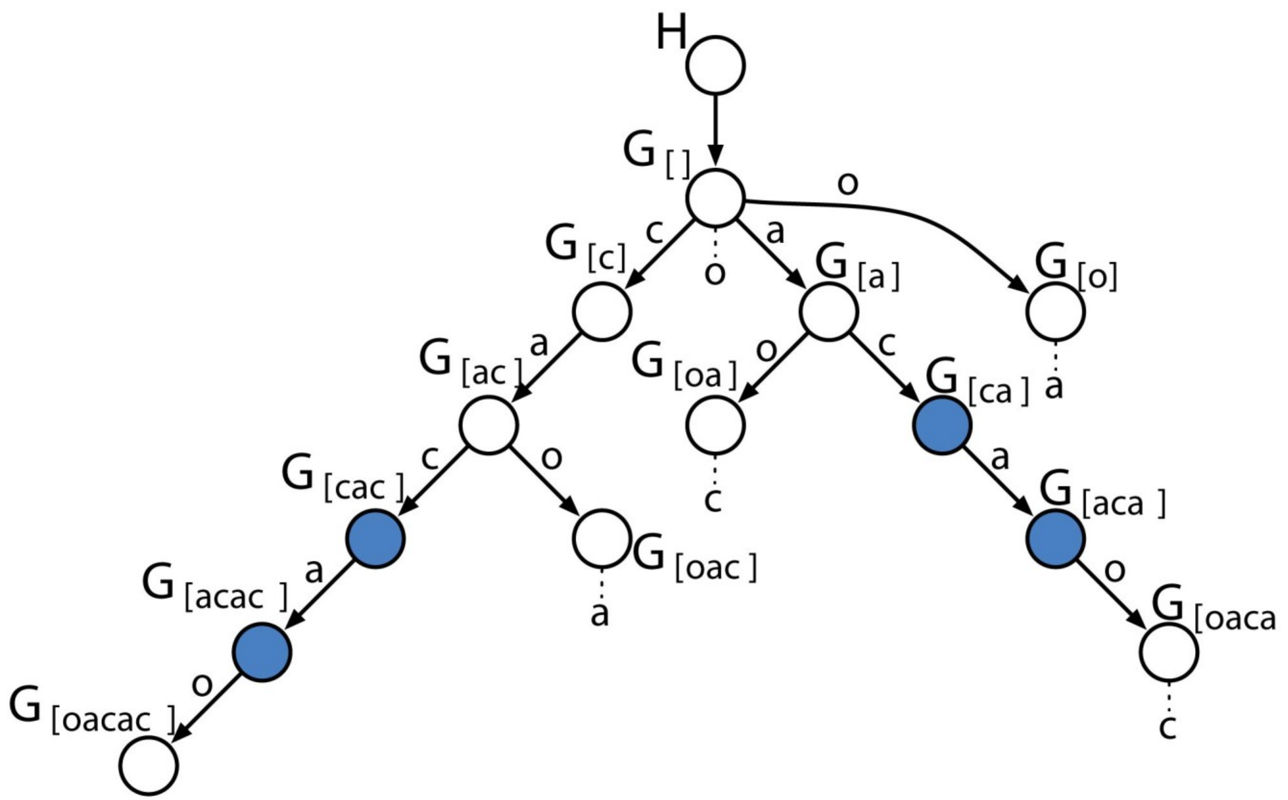
- Theorem 1: Coagulation

If $G_2 j G_1$ » $PY(d_1; 0; G_1)$ and $G_3 j G_2$ » $PY(d_2; 0; G_2)$
then $G_3 j G_1$ » $PY(d_1 d_2; 0; G_1)$ with G_2 marginalized out.

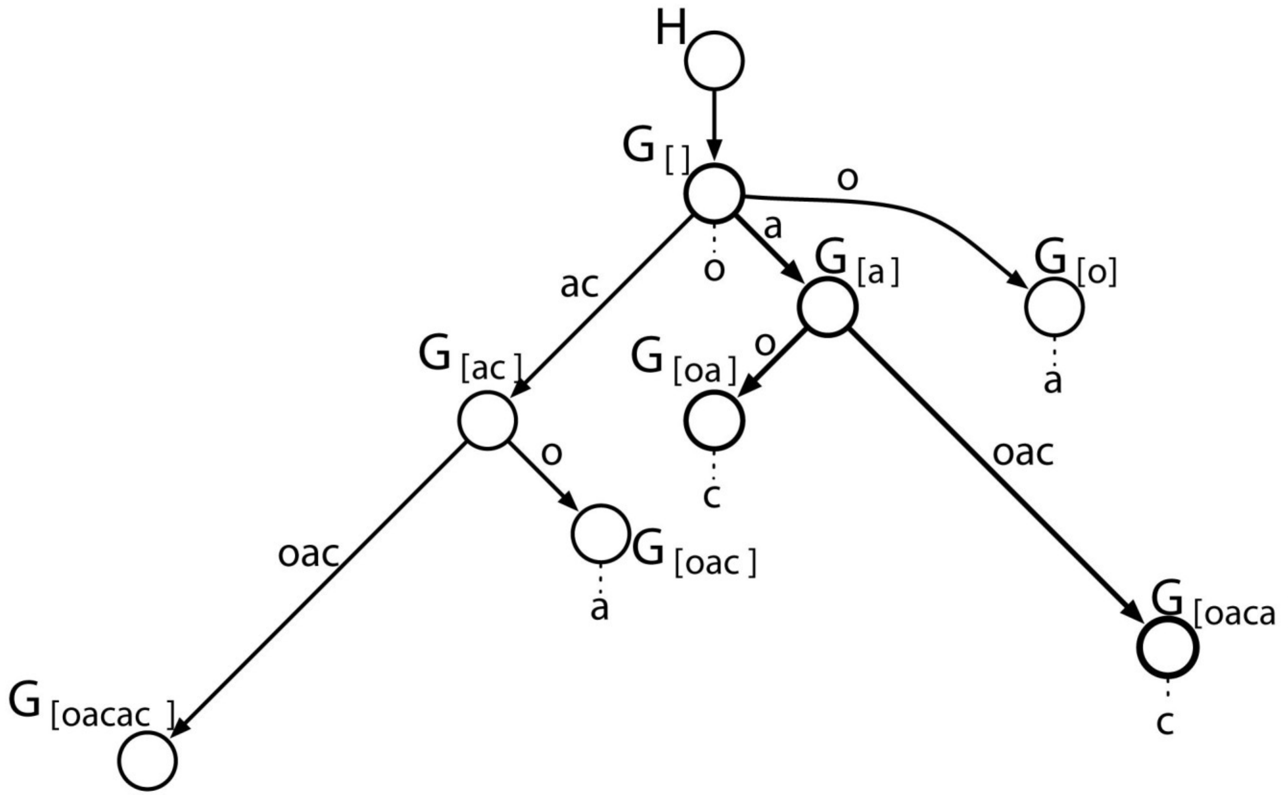


[Pitman ’99; Ho, James, Lau ’06; W., Archambeau, Gasthaus, James, Teh ‘09]

Graphical Model Trie



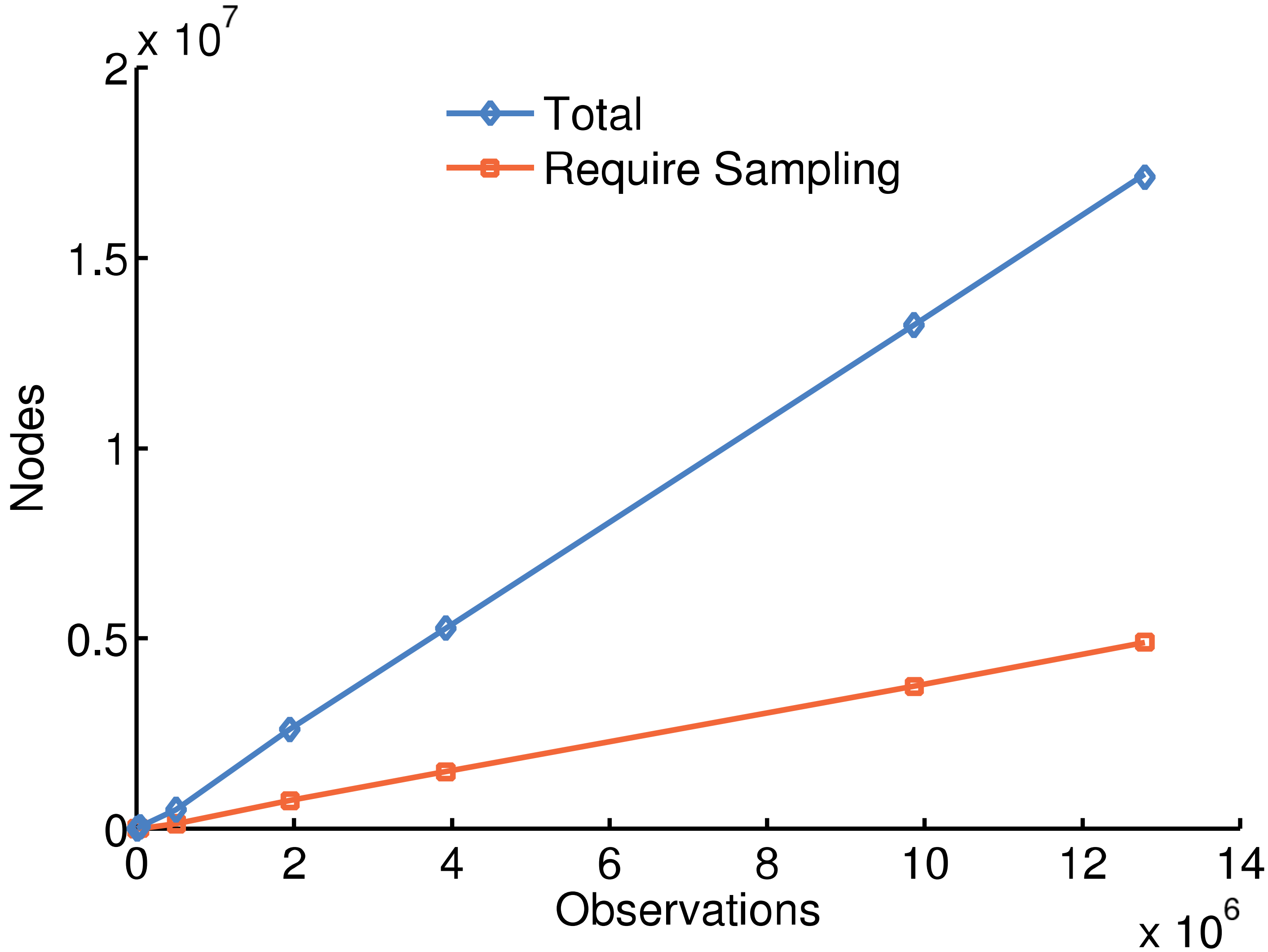
Graphical Model Tree



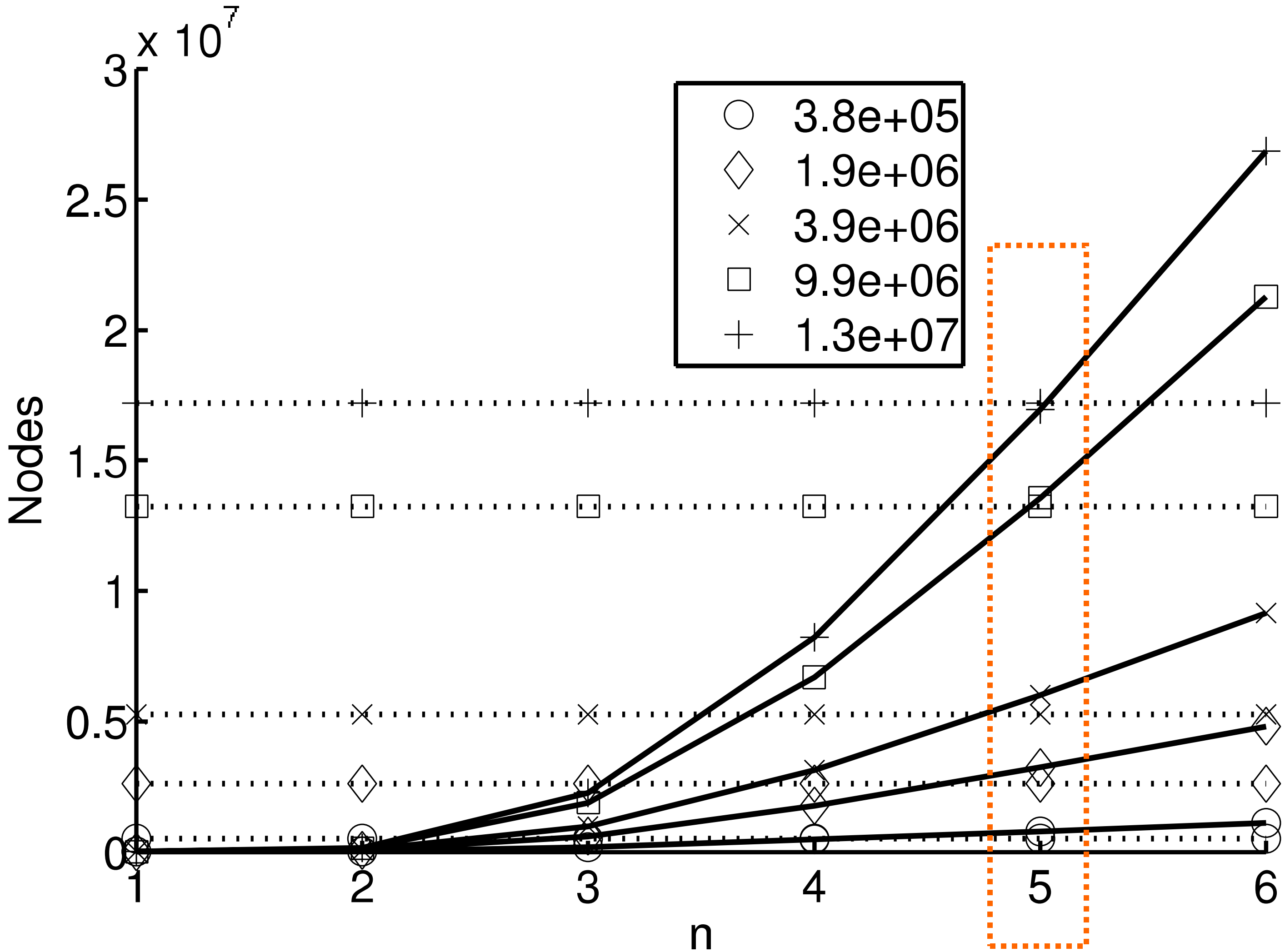
Graphical Model Initialization

- Given a single input sequence
  - Ukkonen's linear time suffix tree construction algorithm is run on its reverse to produce a prefix tree
  - This identifies the nodes in the graphical model we need to represent
  - The tree is traversed and path compressed parameters for the Pitman Yor processes are assigned to each remaining Pitman Yor process

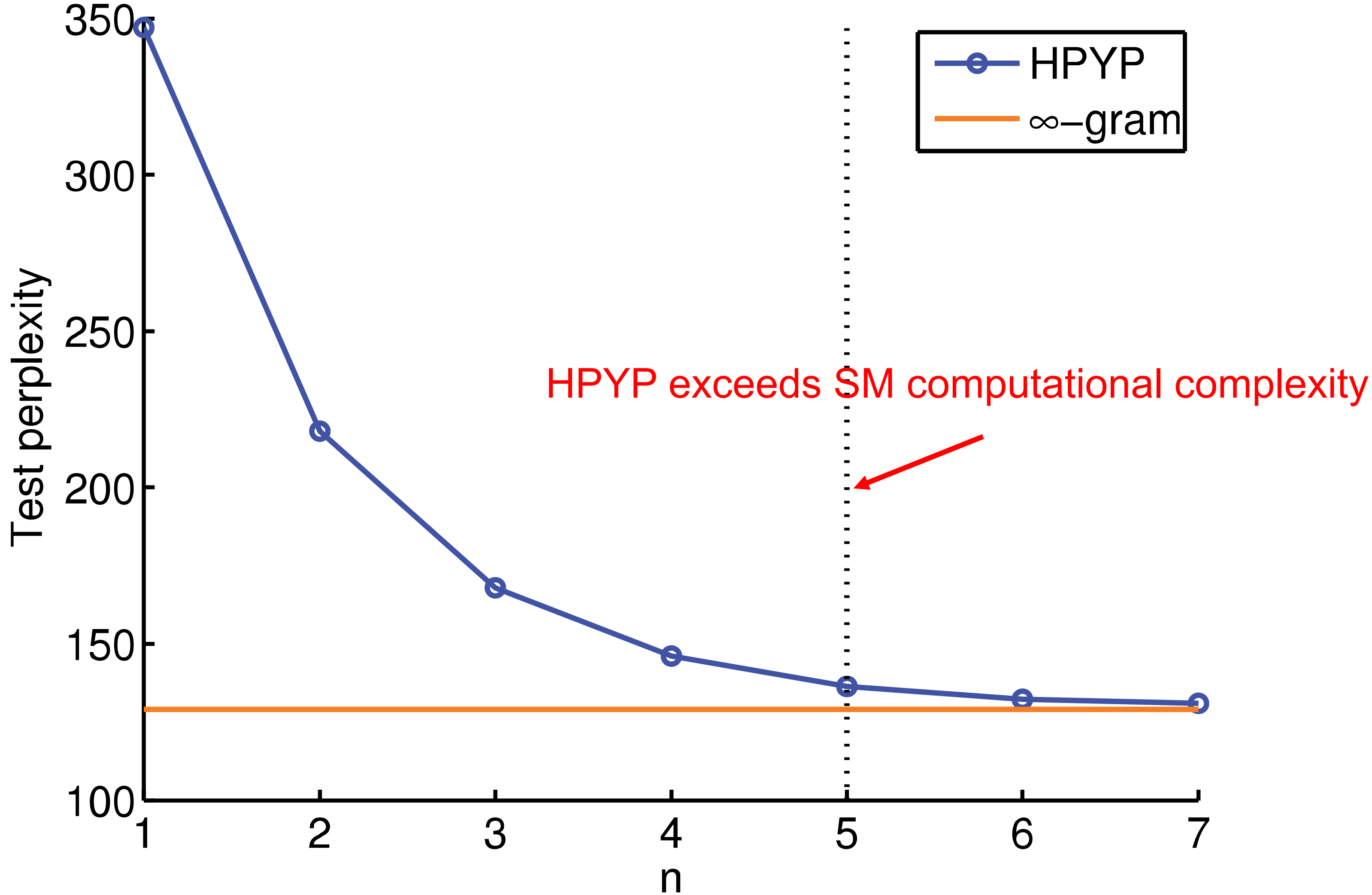
Nodes In The Graphical Model



Never build more than a 5-gram



Sequence Memoizer Bounds N-Gram Performance



Language Modelling Results

AP News Test Perplexity	
[Mnih & Hinton, 2009]	112.1
[Bengio et al., 2003]	109.0
4-gram Modified Kneser-Ney [Teh, 2006]	102.4
4-gram HPYP [Teh, 2006]	101.9
Sequence Memoizer (SM)	**96.9**

The Sequence Memoizer

- The Sequence Memoizer is a deep (unbounded) smoothing Markov model
- It can be used to learn a joint distribution over discrete sequences in time and space linear in the length of a single observation sequence
- It is equivalent to a smoothing ∞-gram but costs no more to compute than a 5-gram