Hierarchical Bayesian Models of Language and Text

Yee Whye Teh
Gatsby Computational Neuroscience Unit, UCL

Joint work with Frank Wood*, Jan Gasthaus*, Cedric Archambeau, Lancelot James
Overview

• Probabilistic Models for Language and Text Sequences

• The Sequence Memoizer
  • Hierarchical Bayesian Modelling on Context Trees
  • Modelling Power Laws with Pitman-Yor Processes
  • Non-Markov Models
  • Efficient Computation

• Conclusions
Overview

• Probabilistic Models for Language and Text Sequences

• The Sequence Memoizer
  • Hierarchical Bayesian Modelling on Context Trees
  • Modelling Power Laws with Pitman-Yor Processes
  • Non-Markov Models
  • Efficient Computation

• Conclusions
Sequence Models for Language and Text

- Probabilistic models for sequences of words and characters, e.g.
  - statistical, machine, learning
  - s, t, a, t, i, s, t, i, c, a, l, _, m, a, c, h, i, n, e, _, l, e, a, r, n, i, n, g

- Uses:
  - Natural language processing: speech recognition, OCR, machine translation.
  - Compression.
  - Cognitive models of language acquisition.
  - Sequence data arises in many other domains.
Probabilistic Modelling

• Set of potential outcomes/observations X.
• Set of unobserved latent variables Y.

• Joint distribution over X and Y:
  \[ P(x \in X, y \in Y | \theta) \]
  \( \theta \) parameters of the model.

• Inference:
  \[ P(y \in Y | x \in X, \theta) = \frac{P(y, x | \theta)}{P(x | \theta)} \]

• Learning:
  \[ P(\text{training data} | \theta) \]

• Bayesian learning:
  \[ P(\theta | \text{training data}) = \frac{P(\text{training data} | \theta)P(\theta)}{Z} \]
Communication via Noisy Channel

Mary likes little Sam

Mary has a little lamb

Sentence

$P(s)$

Utterance

$P(u|s)$

Reconstructed sentence

$P(s|u) = \frac{P(s)P(u|s)}{P(u)}$
Communication via Noisy Channel

Mary has a little lamb

María tiene un pequeño cordero

Mary has a little lamb

Sentence: $P(s)$

Foreign sentence: $P(u|s)$

Reconstructed sentence:

$$P(s|u) = \frac{P(s)P(u|s)}{P(u)}$$
Markov Models for Language and Text

• Probabilistic models for sequences of words and characters.

\[
P(\text{statistical machine learning}) = \\
P(\text{statistical})* \\
P(\text{machine} \mid \text{statistical})* \\
P(\text{learning} \mid \text{statistical machine})
\]

• Usually makes a Markov assumption:

\[
P(\text{statistical machine learning}) = \\
P(\text{statistical})* \\
P(\text{machine} \mid \text{statistical})* \\
P(\text{learning} \mid \text{machine})
\]

• Order of Markov model typically ranges from \(~3\) to \(> 10\).
Sparsity in Markov Models

- Consider a high order Markov models:

\[ P(\text{sentence}) = \prod_i P(\text{word}_i|\text{word}_{i-N+1} \ldots \text{word}_{i-1}) \]

- Large vocabulary size means naively estimating parameters of this model from data counts is problematic for \( N>2 \).

\[ P^{ML}(\text{word}_i|\text{word}_{i-N+1} \ldots \text{word}_{i-1}) = \frac{C(\text{word}_{i-N+1} \ldots \text{word}_i)}{C(\text{word}_{i-N+1} \ldots \text{word}_{i-1})} \]

- Naive priors/regularization fail as well: most parameters have no associated data.
  - Smoothing.
  - Hierarchical Bayesian models.
Smoothing in Language Models

• Smoothing is a way of dealing with data sparsity by combining large and small models together.

\[ P^\text{smooth}(\text{word}_i|\text{word}_{i-N+1}) = \sum_{n=1}^{N} \lambda(n)Q_n(\text{word}_i|\text{word}_{i-n+1}) \]

• Combines expressive power of large models with better estimation of small models (cf bias-variance trade-off).

\[ P^\text{smooth}(\text{learning}|\text{statistical machine}) = \lambda(3)Q_3(\text{learning}|\text{statistical machine}) + \lambda(2)Q_2(\text{learning}|\text{machine}) + \lambda(1)Q_1(\text{learning}|\emptyset) \]
• [Chen and Goodman 1998] found that Interpolated and modified Kneser-Ney are best under virtually all circumstances.
Overview

• Probabilistic Models for Language and Text Sequences

• The Sequence Memoizer
  • **Hierarchical Bayesian Modelling on Context Trees**
  • Modelling Power Laws with Pitman-Yor Processes
  • Non-Markov Models
  • Efficient Computation

• Conclusions
Hierarchical Bayesian Models


- In machine learning, have been used for multitask learning, transfer learning, learning-to-learn and domain adaptation.
• Context of conditional probabilities naturally organized using a tree.

• Smoothing makes conditional probabilities of neighbouring contexts more similar.

• Later words in context more important in predicting next word.

\[ P_{\text{smooth}}(\text{learning}|\text{statistical machine}) = \lambda(3)Q_3(\text{learning}|\text{statistical machine}) + \lambda(2)Q_2(\text{learning}|\text{machine}) + \lambda(1)Q_1(\text{learning}|\emptyset) \]
Hierarchical Bayesian Models on Context Tree

• Parametrize the conditional probabilities of Markov model:

\[ P(\text{word}_i = w | \text{word}_{i-N+1}^{i-1} = u) = G_u(w) \]

\[ G_u = [G_u(w)]_{w \in \text{vocabulary}} \]

• \( G_u \) is a probability vector associated with context \( u \).

• [MacKay and Peto 1994].

\[ G_{\text{machine}} \]

\[ G_{\text{statistical machine}} \]

\[ G_{\text{Bayesian machine}} \]

\[ G_{\in \text{statistical machine}} \]

\[ G_{\text{is statistical machine}} \]
Hierarchical Dirichlet Language Models

- What is $P(G_u|G_{pa(u)})$? [MacKay and Peto 1994] proposed using the standard Dirichlet distribution over probability vectors.

\[
P(G_u|G_{pa(u)})
\]

<table>
<thead>
<tr>
<th>T</th>
<th>N-1</th>
<th>IKN</th>
<th>MKN</th>
<th>HDLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2 \times 10^6$</td>
<td>2</td>
<td>148.8</td>
<td>144.1</td>
<td>191.2</td>
</tr>
<tr>
<td>$4 \times 10^6$</td>
<td>2</td>
<td>137.1</td>
<td>132.7</td>
<td>172.7</td>
</tr>
<tr>
<td>$6 \times 10^6$</td>
<td>2</td>
<td>130.6</td>
<td>126.7</td>
<td>162.3</td>
</tr>
<tr>
<td>$8 \times 10^6$</td>
<td>2</td>
<td>125.9</td>
<td>122.3</td>
<td>154.7</td>
</tr>
<tr>
<td>$10 \times 10^6$</td>
<td>2</td>
<td>122.0</td>
<td>118.6</td>
<td>148.7</td>
</tr>
<tr>
<td>$12 \times 10^6$</td>
<td>2</td>
<td>119.0</td>
<td>115.8</td>
<td>144.0</td>
</tr>
<tr>
<td>$14 \times 10^6$</td>
<td>2</td>
<td>116.7</td>
<td>113.6</td>
<td>140.5</td>
</tr>
<tr>
<td>$14 \times 10^6$</td>
<td>1</td>
<td>169.9</td>
<td>169.2</td>
<td>180.6</td>
</tr>
<tr>
<td>$14 \times 10^6$</td>
<td>3</td>
<td>106.1</td>
<td>102.4</td>
<td>136.6</td>
</tr>
</tbody>
</table>

- We will use Pitman-Yor processes instead [Perman, Pitman and Yor 1992], [Pitman and Yor 1997], [Ishwaran and James 2001].
Overview

- Probabilistic Models for Language and Text Sequences
- The Sequence Memoizer
  - Hierarchical Bayesian Modelling on Context Trees
  - Modelling Power Laws with Pitman-Yor Processes
- Non-Markov Models
- Efficient Computation
- Conclusions
Pitman-Yor Processes

Word frequency vs. Rank (according to frequency)

- Red dashed line: Pitman-Yor
- Black solid line: English text
- Blue dotted line: Dirichlet
What does $\text{PY}(\theta, \phi)$ look like?

No closed form expression, but can draw $\text{G} \sim \text{PY}(\theta, \phi)$. 

Jan Gasthaus's Gatsby Unit, UCL. 

Sequence Memoizer, DCC 2010.
Chinese Restaurant Processes

- Easiest to understand them using Chinese restaurant processes.

\[
p(\text{sit at table } k) = \frac{c_k - d}{\theta + \sum_{j=1}^K c_j}
\]

\[
p(\text{sit at new table}) = \frac{\theta + dK}{\theta + \sum_{j=1}^K c_j}
\]

- Defines an exchangeable stochastic process over sequences \(x_1, x_2, \ldots\)

- The de Finetti measure is the Pitman-Yor process,

\[
G \sim \text{PY}(\theta, d, H)
\]

\[
x_i \sim G \quad i = 1, 2, \ldots
\]

- [Perman, Pitman & Yor 1992, Pitman & Yor 1997]
Power Law Properties of Pitman-Yor Processes

• Chinese restaurant process:

\[ p(\text{sit at table } k) \propto c_k - d \]
\[ p(\text{sit at new table}) \propto \theta + dK \]

• Pitman-Yor processes produce distributions over words given by a power-law distribution with index \( \frac{1}{2} + d \)

• Customers = word instances, tables = dictionary look-up;
• Small number of common word types;
• Large number of rare word types.

• This is more suitable for languages than Dirichlet distributions.

• [Goldwater, Griffiths and Johnson 2005] investigated the Pitman-Yor process from this perspective.
Power Law Properties of Pitman-Yor Processes

![Graph showing word frequency vs rank for Pitman-Yor, English text, and Dirichlet processes.](image)

- **Pitman-Yor**
- **English text**
- **Dirichlet**
Power Law Properties of Pitman-Yor Processes
Hierarchical Pitman-Yor Language Models

- Parametrize the conditional probabilities of Markov model:

\[ P(\text{word}_i = w|\text{word}_{i-N+1}^{i-1} = u) = G_u(w) \]

\[ G_u = [G_u(w)]_{w \in \text{vocabulary}} \]

- \( G_u \) is a probability vector associated with context \( u \).

- Place Pitman-Yor process prior on each \( G_u \).
Hierarchical Pitman-Yor Language Models

- Significantly improved on the hierarchical Dirichlet language model.
- Results better Kneser-Ney smoothing, state-of-the-art language models.

<table>
<thead>
<tr>
<th>T</th>
<th>N-1</th>
<th>IKN</th>
<th>MKN</th>
<th>HDLM</th>
<th>HPYLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2 \times 10^6$</td>
<td>2</td>
<td>148.8</td>
<td><strong>144.1</strong></td>
<td>191.2</td>
<td>144.3</td>
</tr>
<tr>
<td>$4 \times 10^6$</td>
<td>2</td>
<td>137.1</td>
<td><strong>132.7</strong></td>
<td>172.7</td>
<td>132.7</td>
</tr>
<tr>
<td>$6 \times 10^6$</td>
<td>2</td>
<td>130.6</td>
<td>126.7</td>
<td>162.3</td>
<td><strong>126.4</strong></td>
</tr>
<tr>
<td>$8 \times 10^6$</td>
<td>2</td>
<td>125.9</td>
<td>122.3</td>
<td>154.7</td>
<td><strong>121.9</strong></td>
</tr>
<tr>
<td>$10 \times 10^6$</td>
<td>2</td>
<td>122.0</td>
<td>118.6</td>
<td>148.7</td>
<td><strong>118.2</strong></td>
</tr>
<tr>
<td>$12 \times 10^6$</td>
<td>2</td>
<td>119.0</td>
<td>115.8</td>
<td>144.0</td>
<td><strong>115.4</strong></td>
</tr>
<tr>
<td>$14 \times 10^6$</td>
<td>2</td>
<td>116.7</td>
<td>113.6</td>
<td>140.5</td>
<td><strong>113.2</strong></td>
</tr>
<tr>
<td>$14 \times 10^6$</td>
<td>1</td>
<td>169.9</td>
<td><strong>169.2</strong></td>
<td>180.6</td>
<td>169.3</td>
</tr>
<tr>
<td>$14 \times 10^6$</td>
<td>3</td>
<td>106.1</td>
<td>102.4</td>
<td>136.6</td>
<td><strong>101.9</strong></td>
</tr>
</tbody>
</table>

- Similarity of perplexities not a surprise---Kneser-Ney can be derived as a particular approximate inference method.
Overview

• Probabilistic Models for Language and Text Sequences

• The Sequence Memoizer
  • Hierarchical Bayesian Modelling on Context Trees
  • Modelling Power Laws with Pitman-Yor Processes

• Non-Markov Models
  • Efficient Computation

• Conclusions
Markov Models for Language and Text

- Usually makes a Markov assumption to simplify model:
  \[ P(\text{south parks road}) \sim P(\text{south}) \times P(\text{parks} \mid \text{south}) \times P(\text{road} \mid \text{south parks}) \]

- Language models: usually Markov models of order 2-4 (3-5-grams).
- How do we determine the order of our Markov models?
- Is the Markov assumption a reasonable assumption?
  - Be nonparametric about Markov order...
Non-Markov Models for Language and Text

- Model the conditional probabilities of each possible word occurring after each possible context (of unbounded length).

- Use hierarchical Pitman-Yor process prior to share information across all contexts.

- Hierarchy is infinitely deep.

- *Sequence memoizer.*
Overview

• Probabilistic Models for Language and Text Sequences

• The Sequence Memoizer
  • Hierarchical Bayesian Modelling on Context Trees
  • Modelling Power Laws with Pitman-Yor Processes
  • Non-Markov Models
  • Efficient Computation

• Conclusions
The sequence memoizer model is very large (actually, infinite).

Given a training sequence (e.g.: o,a,c,a,c), most of the model can be ignored (integrated out), leaving a finite number of nodes in context tree.

But there are still $O(T^2)$ number of nodes in the context tree...
Model Size: Infinite -> $O(T^2) \rightarrow 2T$

- **Idea:** Integrate out non-branching, non-leaf nodes of the context tree.

- Resulting tree is related to a suffix tree data structure, and has at most $2T$ nodes.

- There are linear time construction algorithms [Ukkonen 1995].
Closure under Marginalization

• In marginalizing out non-branching interior nodes, need to ensure that resulting conditional distributions are still tractable.

  \[ G_{[a]} \]
  \[ \rightarrow \]
  \[ G_{[ca]} \]
  \[ \rightarrow \]
  \[ G_{[aca]} \]

  \[ G_{[a]} \]
  \[ \rightarrow \]
  \[ G_{[ca]} \]
  \[ \rightarrow \]
  \[ G_{[aca]} \]

• E.g.: If each conditional is Dirichlet, resulting conditional is not of known analytic form.
Closure under Marginalization

- In marginalizing out non-branching interior nodes, need to ensure that resulting conditional distributions are still tractable.

- For certain parameter settings, Pitman-Yor processes are closed under marginalization!

- [Pitman 1999, Ho, James & Lau 2006]
Coagulation and Fragmentation Operators

\[ \Pi_1 \]

\[ \Pi_2 \]
Coagulation and Fragmentation Operators

\[ \Pi_2 \]

A
3
6

B
2
7

C
4
5
8

D
9

\[ \Pi_1 \]

A
B

C
D

Coagulate
Coagulation and Fragmentation Operators

\[\Pi_1\]

\[
\begin{array}{c}
A \\
B \\
C \\
D \\
\end{array}
\]

\[\Pi_2\]

\[
\begin{array}{c}
A \\
B \\
C \\
D \\
\end{array}
\]

Coagulate

\[
\begin{array}{c}
1 \\
2 \\
3 \\
4 \\
\end{array}
\]

\[
\begin{array}{c}
5 \\
6 \\
7 \\
8 \\
9 \\
\end{array}
\]

\[
\begin{array}{c}
1 \\
2 \\
3 \\
4 \\
\end{array}
\]

\[
\begin{array}{c}
5 \\
6 \\
7 \\
8 \\
9 \\
\end{array}
\]
Coagulation and Fragmentation Operators

Coagulate

Fragment
Coagulation and Fragmentation Operators

• The following statements are equivalent:

(I) \( \pi_2 \sim \text{CRP}_n(\alpha d_2, d_2) \) and \( \pi_1|\pi_2 \sim \text{CRP}_{\pi_2}(\alpha, d_1) \)

(II) \( C \sim \text{CRP}_n(\alpha d_2, d_1 d_2) \) and \( F_a|C \sim \text{CRP}_a(-d_1 d_2, d_2) \) \( \forall a \in C \)
Final Model Specification

Probability of sequence:

\[ P(x_{1:T}) = \prod_{i=1}^{T} P(x_i|x_{1:i-1}) = \prod_{i=1}^{T} G_{x_{1:i-1}}(x_i) \]

Prior over conditional probabilities:

\[ G_{\emptyset} \sim PY(\theta_{\emptyset}, d_{\emptyset}, H) \]
\[ G_u|G_{\sigma(u)} \sim PY(\theta_u, d_u, G_{\sigma(u)}), \text{ for } u \in \Sigma^* \setminus \{\emptyset\}, \]

Constraint on parameters:

\[ \theta_u = \theta_{\emptyset} \prod_{v \neq \emptyset, \text{ suffix of } u} d_v \]
Comparison to Finite Order HPYLM

![Graph showing the comparison between Perplexity and Number of nodes with context length (n). The graph illustrates the decline in perplexity with increasing context length and the corresponding increase in the number of nodes.]
Inference using Gibbs Sampling
Inference using Gibbs Sampling

![Convergence Speed Graph]

- **Cross entropy on test set**
- **Number of iterations**

The graph illustrates the convergence speed of cross entropy on the test set as a function of the number of iterations.
Entropic Coding for Compression

- Encoder:

  \[ x_i \rightarrow \begin{array}{c}
  \text{Model} \\
  P(x_i|x_1...x_{i-1},\theta_i)
  \end{array} \rightarrow -\log_2 P(x_i|x_1...x_{i-1},\theta_i) \]

- Decoder:

  \[ x_i \leftarrow \begin{array}{c}
  \text{Model} \\
  P(x_i|x_1...x_{i-1},\theta_i)
  \end{array} \leftarrow -\log_2 P(x_i|x_1...x_{i-1},\theta_i) \]

- \( \theta_i \) parameter value estimated from \( x_1...x_{i-1} \).

- A good probabilistic model = good compressor.

Claude Shannon
## Compression Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Average bits/byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>gzip</td>
<td>2.61</td>
</tr>
<tr>
<td>bzip2</td>
<td>2.11</td>
</tr>
<tr>
<td>CTW</td>
<td>1.99</td>
</tr>
<tr>
<td>PPM</td>
<td>1.93</td>
</tr>
<tr>
<td>Sequence Memoizer</td>
<td>1.89</td>
</tr>
</tbody>
</table>

Calgary corpus
SM inference: particle filter
PPM: Prediction by Partial Matching
CTW: Context Tree Weigting
Online inference, entropic coding.
Related Works

- Infinite Markov models [Mochihashi & Sumita 2008]
- Bayesian nonparametric grammars (Goldwater, Johnson, Blunsom, Cohn etc).
- Language model smoothing algorithms [Chen & Goodman 1998, Kneser & Ney 1995].
- Hierarchical Bayesian nonparametric models [Teh & Jordan 2010].
Conclusions

• Probabilistic models of sequence models without making Markov assumptions with efficient construction and inference algorithms.
• State-of-the-art text compression and language modelling results.
• Hierarchical Bayesian modelling leads to improved performance.
• Pitman-Yor processes allow us to encode prior knowledge about power-law properties, leading to improved performance.
• Hierarchical Pitman-Yor processes have been used successfully for various more linguistically motivated models.

• www.sequencememoizer.com (Java implementation)
• www.deplump.com (text compression demo)
• Jan Gasthaus’ webpage (C++ implementation)
Publications

- A Hierarchical Bayesian Language Model based on Pitman-Yor Processes.

- A Bayesian Interpretation of Interpolated Kneser-Ney.

- A Stochastic Memoizer for Sequence Data.

- Text Compression Using a Hierarchical Pitman-Yor Process Prior.

- Forgetting Counts: Constant Memory Inference for a Dependent Hierarchical Pitman-Yor Process.

- Some Improvements to the Sequence Memoizer.

- The Sequence Memoizer.

- Hierarchical Bayesian Nonparametric Models with Applications.
Thank You!

Acknowledgements:
Frank Wood, Jan Gasthaus, Cedric Archambeau, Lancelot James

Lee Kuan Yew Foundation
Gatsby Charitable Foundation