松本研究室 Doctor Lecture 2004

言語モデル

Daichi Mochihashi

daiti-m@is.naist.jp

daichi.mochihashi@atr.jp

NAIST 松本研究室 / ATR SLT Dept.2 (SLR)

2004.07.16

Overview

- What is the Language Model?
- N-gram models (including various smoothing)
- Variable N-grams
- N-gram Language Model evaluation
- Whole sentence maximum entropy
- Long distance models
  - (Old) cache/trigger models
  - Latent variable models

What is the Language Model?

確率的言語モデルとは?

> 単語列 $\mathbf{w} = w_1 w_2 w_3 \cdots w_n$ の結合確率 $p(\mathbf{w})$を与えるモデル.

- $\mathbf{w}$ の定義は任意
  - フレーズ，文, 文書, 動詞句, ...
- 生成モデル
  - サイコロをふることで，$p(\mathbf{w})$ に従う任意の新しいフレーズ，文, 文書, ... を生成できる.
- 自然の言語現象のモデル $\Leftrightarrow$ より適切な符号化.
(dual problem)

Why Language Models?

- 音声認識 (Speech recognition)
- 機械翻訳 (Brown et al. 1990)

$$p(J|E) \propto p(E|J)p(J) \quad (\text{翻訳モデル} \times \text{言語モデル})$$

- 情報検索 (Zhai & Lafferty 2001, Berger & Lafferty 1999)
- テキスト入力 (HCI), OCR/スペル訂正 (基礎技術)
- Shannon ゲーム, 情報理論/圧縮, テキスト生成

How to model $p(\mathbf{w})$?

$p(\mathbf{w}) = p(w_1^n)$ をどのようにモデル化するか?

- Conditional method (N-gram)

$$p(\mathbf{w}) = \prod_{t=1}^{T} p(w_t | w_1^{t-1}) \quad (1)$$

- “Whole sentence” method

$$p(\mathbf{w}) \propto p_0(\mathbf{w}) \cdot \exp(\sum_i \lambda_i f_i(\mathbf{w})) \quad (2)$$

注: (2) は (1) を含む $(p_0(\mathbf{w}) = p_{\text{cond}}(\mathbf{w}))$

N-gram approximation model

$$p(w_1^n) = \prod_{t=1}^{T} p(w_t | w_{t-1} w_{t-2} \cdots w_1) \tag{3}$$

$$\simeq \prod_{t=1}^{T} p(w_t | \underbrace{w_{t-1} \cdots w_{t-(n-1)}}_{n-1 \text{ 語}}) \tag{4}$$

$$= \begin{cases} \prod_{t=1}^{T} p(w_t | w_{t-1}, w_{t-2}) & : \text{トライグラム } (n=3) \\ \prod_{t=1}^{T} p(w_t | w_{t-1}) & : \text{バイグラム } (n=2) \\ \prod_{t=1}^{T} p(w_t) & : \text{ユニグラム } (n=1) \end{cases} \tag{5}$$

注: ベイズの公式 $p(X, Y|Z) = p(X|Y, Z)p(Y|Z)$ を式 (3) で再帰的に適用している

Language Model Smoothing

- 以下，説明のためにバイグラムの推定を考える．
- $f_{i|j}$, f_j をそれぞれ $\langle w_j \to w_i \rangle$, w_j の生起頻度とすると，

$$\textbf{最尤推定:} \quad \hat{p}(i|j) = \frac{f_{i|j}}{f_j}. \tag{6}$$

- ほとんどの n-gram が確率 0 (データスパースネス)

  ⇓

  1. n-gram のスムージング
  2. より短い文脈長での推定を使う (バックオフ)
  3. クラスベース n-gram

Katz's Backing-Off

- $f_{i|j} = 0$ (または k 以下) の場合にどうするか?

$$p(i|j) = \begin{cases} (1 - \alpha(j)) \cdot \hat{p}(i|j) & \text{when } f_{i|j} > 0 \\ \alpha(j) \cdot p(i) & \text{when } f_{i|j} = 0 \end{cases} \quad (7)$$

  - $f_{i|j} > 0$ の場合は, 最尤推定値を少し減らして, そのまま使う
  - $f_{i|j} = 0$ の場合は, 上で減らした分をもらって, ユニグラム $p(i)$ に従って分配する

- **バックオフ** (Back-off):
推定の際に, より短い文脈での推定値を用いること.
(トライグラム以上でも同じ)

Class-based n-gram

- 単語の連接確率をそのまま用いるとスパースネスの問題
  → クラスの連接確率で置き換える (like HMM)

- $w_1, w_2, \ldots$ に対応するクラスを (確率 1 で) $c_1, c_2, \ldots$ とおくと,

$$p(w_1^T) = \prod_{t=1}^{T} p(w_t|c_t)p(c_t|c_{t-1}) \tag{8}$$

ここで,

$$\langle \log p(w_1^T) \rangle = \frac{1}{T}[\log p(w_t|c_t) + \log p(c_t|c_{t-1})] \tag{9}$$

$$\underset{T\to\infty}{\to} \sum_{c_1,c_2} p(c_1,c_2) \log \frac{p(c_1,c_2)}{p(c_1)p(c_2)} + \sum_{w} p(w) \log p(w)$$

$$= \sum_{c_1,c_2} I(c_1,c_2) - H(w). \quad \text{(相互情報量最大化)}$$

Langage Model Smoothing

- Simple Smoothing Methods
  - Laplace smoothing, Lidstone's law
  - 古典的な統計的手法
- Extended Smoothing Methods
  - Good-Turing smoothing, Kneser-Ney Smoothing, Bayes smoothing
  - NLP 向けに考案されたスムージング手法
  - 他にも色々あるが，最初の 2 つは基礎知識として重要

Simple Smoothing Methods

- Laplace smoothing

$$p(i|j) = \frac{f_{i|j} + 1}{\sum_i^W (f_{i|j} + 1)} = \frac{f_{i|j} + 1}{f_j + W} \tag{10}$$

- Lidstone's law パラメータ: λ (特に, $\lambda = 1/2$)

$$p(i|j) = \frac{f_{i|j} + \lambda}{\sum_i^W (f_{i|j} + \lambda)} = \frac{f_{i|j} + \lambda}{f_j + W\lambda} \tag{11}$$

$$= \mu \cdot \frac{f_{i|j}}{f_j} + (1 - \mu) \cdot \frac{1}{W} \quad (\mu = \frac{f_j}{f_j + W\lambda}) \tag{12}$$

- 均一なユニグラム確率との線形補間になっている (不適切!)

Extended Smoothing methods

- Good-Turing smoothing
  - よく使われる
- (Modified) Kneser-Ney smooothing
  - Chen and Goodman (1998) の有名な比較によれば, (下のものを除いて) 精度が現在最良
- Hierarchical Bayes optimal smoothing (MacKay 1994)

Good-Turing smoothing

$$p(i|j) = \frac{(f_{i|j}+1)\cdot\frac{N(f_{i|j}+1)}{N(f_{i|j})}}{f_j} \quad \text{if } f_{i|j} < \theta. \tag{13}$$

$N(x) : x$ 回現れた n-gram の総数

- cf. 最尤推定値 $\hat{p}(i|j) = \frac{f_{i|j}}{f_j}$
- 閾値 θ の最適値は一意に決まらない
- 出なかった n-gram には均等の確率 (バックオフが必要)
  - $p(\text{well}|\text{quite}) = p(\text{epistemological}|\text{quite})$?
- 単語全体にわたって正規化する必要がある

(Modified) Kneser-Ney smoothing

- Absolute discounting (頻度から一定数を引く)

$$p(i|j) = \frac{f_{i|j} - D(f_{i|j})}{f_j} + \gamma(j)p(i) \qquad (14)$$

- $D(n)$: ディスカウント関数
  - $n = 1$ の場合: $D(1) = 1 - 2 \cdot \frac{N(1)}{N(1)+2N(2)} \cdot \frac{N(2)}{N(1)}$
  - $n = 2$ の場合: $D(2) = 2 - 3 \cdot \frac{N(1)}{N(1)+2N(2)} \cdot \frac{N(3)}{N(2)}$
  - $n \geq 3$ の場合: $D(3+) = 3 - 4 \cdot \frac{N(1)}{N(1)+2N(2)} \cdot \frac{N(4)}{N(3)}$
  - $\gamma(j)$ は単語 j に後続する単語の種類数から決まる

Hierarchical Bayes Optimal Smoothing (MacKay 1994)

$$E[p(i|j)] = \frac{f_{i|j} + \alpha_i}{\sum_i (f_{i|j} + \alpha_i)} \quad (15)$$

$$= \frac{f_j}{f_j + \alpha_0} \cdot \hat{p}(i|j) + \frac{\alpha_0}{f_j + \alpha_0} \cdot \bar{\alpha}_i \quad (16)$$

where $\alpha_0 = \sum_k \alpha_k$ and $\bar{\alpha}_i = \frac{\alpha_i}{\alpha_0}$

- $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_W)$：ハイパーパラメータ.
- バックオフと線形補間の両方の性質を持つ適応的補間
  - ただし，$\bar{\alpha}_i \neq p(i)$ (Unigram)
- ($\boldsymbol{\alpha}$ はどうやって計算?)

Hierarchical Bayes Optimal Smoothing (2)

$$p(\boldsymbol{F}|\boldsymbol{\alpha}) = \prod_{j=1}^{W} \left[\frac{\Gamma(\alpha_0)}{\prod_{i=1}^{W} \Gamma(\alpha_i)} \cdot \frac{\prod_{i=1}^{W} \Gamma(f_{i|j} + \alpha_i)}{\Gamma(f_j + \alpha_0)} \right] \tag{17}$$

これは $\boldsymbol{\alpha}$ に関して上に凸で, 大域的最適解が存在.
次の iteration で解くことができる (Minka 2003).

$$\alpha_i^{(t+1)} = \alpha_i^{(t)} \cdot \frac{\sum_j \Psi(f_{i|j} + \alpha_j) - \Psi(\alpha_j)}{\sum_j \Psi(f_j + \sum_k \alpha_k) - \Psi(\sum_k \alpha_k)} \tag{18}$$

- MATLAB で 45 行
- 詳細は SVM2004 で

Variable Order n-grams (1)

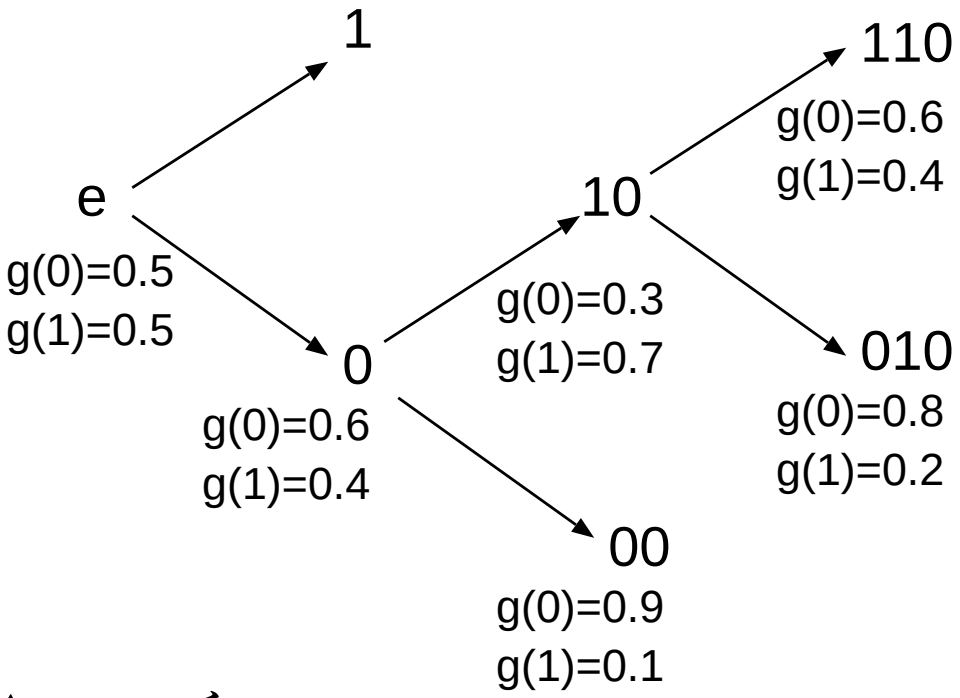
- いつも同じ文脈長 (ex. 2 words) を用いる → 不適.
  - 助詞/助動詞の連続, typical なフレーズ
    (牛鮭定食 で も 食っ て なさいっ て こっ た)
  - 単語の切り方に依存する
  - 多くの単語からなる Named Entity.
- 文脈長を適応的に変えられないか?

Variable Order n-grams (2)

- Pereira et al. (1995) “Beyond Word N-Grams”
- 基礎となる研究:
  - Ron, Singer, Tishby (1994) “The Power of Amnesia”
    - Prediction Suffix Tree (PST)
    - 確率オートマトンと等価.
  - Willems at al. (1995) (情報理論/情報圧縮)
    - Context Tree Weighting method (CTW)
    - 情報理論，圧縮の分野で注目されている圧縮方法の一つ (ex. Sadakane 2000)
- 以下，基本的なアイデアを紹介

Ron, Singer, Tishby (1994)

- Prediction Suffix Tree (PST)



↑マッチ

入力列 h = 00100101110 $\rightarrow p(0|h) = 0.6,\ p(1|h) = 0.4$

Ron, Singer, Tishby (1994)(2)

- PST は確率オートマトンと等価
- ノード s (例: 010) について，子ノード σs $(\sigma \in \{0,1\})$ の追加 (例: 010 $\rightarrow$ 0010) は, 追加した時のエントロピー差

$$p(\sigma s) \cdot D(p(\cdot|\sigma s)||p(\cdot|s)) \tag{19}$$

  が大きいときのみ (= Yodo (1998))

- 聖書のテキスト
  - エントロピー差の閾値 $\epsilon = 0.001$
  - $N = 30$ gram (文字ベース)
    $\Downarrow$
  - 状態数 ≤ 3000, 状態の例: ‘shall be’, ‘there was’

Pereira et al. (1995)

- PST の Mixture (CTW (Willems et al. (1995)) をベース)
  - オンラインベイズ推定になっている
- 訓練単語列を $\mathbf{w} = w_1 w_2 w_3 \cdots w_N$ (N very large) とすると，PST T について

$$p(w_1 \cdots w_N | T) = \prod_{i=1}^{N} \gamma_{C_T(w_1 \cdots w_{i-1})}(w_i) \tag{20}$$

このとき,

$$p(w_{N+1} | w_1 \cdots w_N) = \frac{p(w_1 \cdots w_{N+1})}{p(w_1 \cdots w_N)} \tag{21}$$

$$= \frac{\sum_{T \in \mathcal{T}} p(T) p(w_1 \cdots w_{N+1} | T)}{\sum_{T \in \mathcal{T}} p(T) p(w_1 \cdots w_N | T)} \tag{22}$$

$p(T)$: PST T の prior.

Pereira et al. (1995) (2)

- Tree Mixture:

$$\sum_{T \in \mathcal{T}} p(T) p(w_1 \cdots w_N | T) = L_{\mathrm{mix}}(\epsilon) \tag{23}$$

は再帰的に計算できる.

$$L_{\mathrm{mix}}(s) = \alpha \underbrace{L_n(s)}_{\text{自ノードの emission 確率}} + (1-\alpha) \underbrace{\prod_{\sigma \in W} L_{\mathrm{mix}}(\sigma s)}_{\text{子ノードの emission 確率}} \tag{24}$$

α : Tree prior. $(0 \leq \alpha \leq 1)$

Pereira et al. (1995) (3)

- 実際には,

$$p(w_n|w_1 \cdots w_{n-1}) = \tilde{\gamma}_\epsilon(w_n) \tag{25}$$

ここで，PST のノード s での単語 w_n の出力確率 $\tilde{\gamma}_s(w_n)$ は,

$$\tilde{\gamma}_s(w_n) = \begin{cases} \gamma_s(w_n) & : s \text{ が葉} \\ q_n \gamma_s(w_n) + (1 - q_n)\tilde{\gamma}_{w_{n-|s|}s}(w_n) & : s \text{ が枝} \end{cases} \tag{26}$$

- 補間比 q_n をオンラインでアップデートできることと, $\tilde{\gamma}(w)$ の推定が再帰的になっているところがミソ.

Language Model evaluation

言語モデルをどのように評価するか?

- パープレキシティ (Perplexity)
- クロスエントロピー (Cross Entropy)

その前に..

Basic of Basics

- 確率 $p = \dfrac{1}{\text{分岐数}} \quad \therefore \text{分岐数} = \dfrac{1}{p}$
- Shannon 符号長 (自己情報量)

$$\log \frac{1}{p(x)} = -\log(p(x)) \tag{27}$$

- エントロピー = 平均 Shannon 符号長

$$H(p) = -\sum_x p(x) \log p(x) \tag{28}$$

$$= \langle -\log p \rangle_p \tag{29}$$

Perplexity

- パープレキシティ (Perplexity) = 平均分岐数

$$\mathrm{PPL}(w_1^T) = \left(\prod_{t=1}^{T} \frac{1}{p(w_t|w_1^{t-1})} \right)^{1/T} \quad \text{(幾何平均)} \tag{30}$$

$$= \exp\left(\log \left(\prod_{t=1}^{T} \frac{1}{p(w_t|w_1^{t-1})} \right)^{1/T} \right) \tag{31}$$

$$= \exp\left(\frac{1}{T} \sum_{t=1}^{T} -\log p(w_t|w_1^{t-1}) \right). \tag{32}$$

Cross Entropy (1)

- Kullback-Leibler ダイバージェンス

$$D(\mathbf{p}||\mathbf{q}) = \sum_x p(x) \log \frac{p(x)}{q(x)} \tag{33}$$

$$= \langle \log p(x) - \log q(x) \rangle_p \tag{34}$$

$$= \langle (-\log q(x)) - (-\log p(x)) \rangle_p \tag{35}$$

$\therefore$ $p(x)$ のかわりに $q(x)$ で情報源を符号化したときの符号長の差 (の平均).

$\rightarrow$ 自然の持つ確率 $\mathbf{p}$ をモデル $\mathbf{q}$ で近似するとき, $D(\mathbf{p}||\mathbf{q})$ を最小化すればよい.

Cross Entropy (2)

$$D(\mathbf{p}||\mathbf{q}) = \sum p \log \frac{p}{q} \tag{36}$$

$$= \sum p \log p - \sum p \log q \tag{37}$$

$$= -H(p) - \sum p \log q \rightarrow \text{最小化}. \tag{38}$$

- ゆえに, クロスエントロピー $H(\mathbf{p}, \mathbf{q})$ を

$$H(\mathbf{p}, \mathbf{q}) = -\sum_x p(x) \log q(x) \quad (= \langle -\log q(x) \rangle_p) \tag{39}$$

と定義すれば, KL ダイバージェンス最小化は
クロスエントロピー最小化と同値.
(パープレキシティは, 形式的に $p(x)$ が Uniform な場合に相当)

Whole sentence maximum entropy (1)

$$p(\mathbf{s}) \propto p_0(\mathbf{s}) \cdot \exp(\boldsymbol{\Lambda} \cdot \boldsymbol{F}(\mathbf{s})) \quad (40)$$

$p_0(\mathbf{s})$: 文 $\mathbf{s}$ のデフォルト確率

$\boldsymbol{\Lambda} = (\lambda_1\ \lambda_2\ \ldots\ \lambda_n)$: ME パラメータ

$\boldsymbol{F}(\mathbf{s}) = (f_1(\mathbf{s})\ f_2(\mathbf{s})\ \ldots\ f_n(\mathbf{s}))$: 素性ベクトル

注意:

- $p_0(\mathbf{s})$ を n-gram 確率にとれば，Whole sentence ME は常に n-gram に対する改良となる
- Random Field のセルの数 (= 文の長さ) が大きいため，効率的な素性を設定することはやや難しい

Whole sentence maximum entropy (2)

パラメータ推定/制約:

$$\sum_{\mathbf{s}} p(\mathbf{s}) f_i(\mathbf{s}) = \langle f_i \rangle_{\hat{p}} - (\text{正則化項}). \tag{41}$$

- 可能な文全体について和をとるのは不可能
  $\Rightarrow$ 左辺はモンテカルロサンプリングで近似.
- 式 (40) の $p(\mathbf{s})$ に従って「文」を次々と生成する
  - Gibbs sampling (シンプル) (Pietra, Lafferty 1995)
  - Independent Metropolis sampler
  - Importance Sampling from n-gram

Whole sentence maximum entropy (3)

- 生成された文の例 (Rosenfeld et al. 2000)

```
<s> What do you have to live los angeles </s>
<s> A. B. C. N. N. business news tokyo </s>
<s> Be of says I’m not at this it </s>
<s> Bill Dorman been well I think the most </s>
```

- (Pietra, Lafferty 1995) 英単語のスペルのサンプリング

```
was, reaser, in, there, to, will, ,, was, by, homes,
thing, be, reloverated, ther, which, conists, at,
fores, anditing, with, Mr., proveral, the, ***, ...
```

- Gibbs sampling で生成

Whole sentence maximum entropy (4)

- しかし...
  - ベースライン (n-gram) モデル p_0 : PPL = 81.37
  - Whole Sentence ME : PPL = $80.49 \pm .02$
- なぜ??
  - 素性 f の効果: KL ダイバージェンス $D(\hat{p}(f)||p(f))$

$$D(\hat{p}(f)||p(f)) = \hat{p}(f)\log\frac{\hat{p}(f)}{p(f)} + (1-\hat{p}(f))\log\frac{1-\hat{p}(f)}{1-p(f)}$$

$$\simeq \hat{p}(f)\log\frac{\hat{p}(f)}{p(f)}$$

  - $\log(\hat{p}(f))-\log(p(f))$ は大きいが, $\hat{p}(f)$ が非常に小さい
  - ME+隠れ変数モデル? (eg. ME=HMM (Goodman 2002/2004), LME (Wang 2003))

Long Distance Models

- 意味的な “文脈” を考慮するモデル
  - (Old) キャッシュ/トリガモデル
  - 潜在変数モデル

Cache/trigger models

- Cache model
  - 「一度出た単語は再び表れやすい」ため，k 語前の履歴に出てきた単語を “キャッシュ” して, 確率を上げる
  - k はいくつに取ればよい?
  - 同じ単語は直後は繰り返されない (Beeferman 1997a)
- Trigger model
  - ‘hospital’ が出てくると，‘nurse’ や ‘disease’ が出やすくなる
  - 訓練データに存在する有意な単語の組み合わせとその距離効果を ME で学習 (Beeferman 1997b)
  - 可能な組み合わせ $\sim W \times W \simeq 1$ 億 !

Latent Variable Models

- テキストの生成モデルの応用
- テキストや現在の文脈に対して，隠れトピックの Mixture Model を考える
  → 隠れたトピックの混合比がわかれば，次の単語を予測できる!
  - PLSI 言語モデル (Gildea & Hofmann 1998)
  - LDA 言語モデル (三品 2002, 高橋 2003)

PLSI Language model

- PLSI によって, K 個のトピック別ユニグラム $p(w|z_1), p(w|z_2), \ldots, p(w|z_K)$ が EM で計算できる
- このとき，$\mathbf{w} = w_1 w_2 \cdots w_n$ がこれらのユニグラムの混合として

$$p(\mathbf{w}|\boldsymbol{\lambda}) = \prod_j \sum_{i=1}^{K} \lambda_i p(w_j|z_i) \tag{42}$$

$\rightarrow$ 混合比 $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_K)$ がわかれば,

$$p(w|w_1 \cdots w_n) = \sum_{i=1}^{K} \lambda_i p(w|z_i) \tag{43}$$

として計算できる.

LDA Language model

- PLSI LM では, 混合比 $\boldsymbol{\lambda}$ が最尤推定だった
  → オーバーフィットの危険性.
- $\boldsymbol{\lambda}$ 自体に確率分布を与え，積分除去 (ベイズ推定)

$$p(w|\mathbf{w}) = \int_{\triangle} p(w|\boldsymbol{\lambda}) p(\boldsymbol{\lambda}|\mathbf{w}) d\boldsymbol{\lambda} \tag{44}$$

$$= \int_{\triangle} \sum_{i=1}^{K} \lambda_i p(w|z_i) \cdot p(\boldsymbol{\lambda}|\mathbf{w}) d\boldsymbol{\lambda} \tag{45}$$

$$= \sum_{i=1}^{K} \langle \lambda_i | \mathbf{w} \rangle p(w|z_i) \tag{46}$$

Future of Long distance models

- 文脈の時間順序を考慮したモデル. (今の自分の研究)
- Hierarchical Mixture (NIPS 2003) のオンラインアップデート?
- Maxent モデルとの混合
- 文法的要素 (Long distance dependencies).

Conclusion

- Let’s model a language!
- Generative model and Bayesian method make us happy.. :-)
- お疲れさまでした.