2007－NL－178 (10)
2007／3／29

# Pitman-Yor過程に基づく可変長n-gram言語モデル

持橋 大地　隅田 英一郎
ATR 音声言語コミュニケーション研究所 自然言語処理研究室 /
独立行政法人 情報通信研究機構
daichi.mochihashi@atr.jp
eiichiro.sumita@atr.jp

**概要**

本論文では, n-gram 分布の階層的生成モデルである階層 Pitman-Yor 過程を拡張することで, 各単語の生まれた隠れた文脈を推定することのできるベイズ言語モデルを提案する. 無限の深さをもつ Suffix Tree 上の確率過程を考えることにより, 句を確率的に発見し, 適切な文脈長を学習する. 本手法は言語モデルだけでなく, マルコフモデル一般について, そのオーダーをデータから推定できる可変長生成モデルとなっている. 英語および中国語の標準的なコーパスでの実験により, 提案法の有効性を確認した.

**キーワード:** ノンパラメトリックベイズ, Pitman-Yor 過程, トピック適応, マルコフモデル, 言語モデル

# Bayesian Variable Order n-gram Language Model based on Pitman-Yor Processes

Daichi Mochihashi　Eiichiro Sumita
ATR Spoken Language Communication Research Laboratories /
National Institute of Communications Technology
daichi.mochihashi@atr.jp
eiichiro.sumita@atr.jp

**Abstract**

This paper proposes a variable order n-gram language model by extending a recently proposed model based on hierarchical Pitman-Yor processes. Introducing a stochastic process on an infinite depth suffix tree, we can infer the hidden n-gram context from which each word originated. Experiments on standard large corpora showed validity and efficiency of the proposed model.

**Keywords:** Nonparametric Bayes, Pitman-Yor process, Topic adaptation, Markov models, Language Modeling

## 1　はじめに

単語間のマルコフ過程によって文の確率を計算する $n$ グラムモデルは, Shannon [1] の歴史的な論文で最初に導入されて以来, 自然言語処理の様々な場面に適用され, その有効性が示されてきた, 基礎的で重要な方法である.

$n$グラムモデルは, 直前の $(n-1)$ 個の単語列を状態とした $(n-1)$ 次のマルコフモデルにより, 次の単語の条件つき確率を計算していく. このとき, 状態数は単語の総数を $V$ とすると $V^{n-1}$ のオーダーとなり, $n$ を 1 増やすと総パラメータ数は通常数万倍となり, 指数的に爆発する. このため, 様々なスムージング法を用いた場合でも, 通常 $n=3$ (トライグラム) から最大でも $n=4,5$ 程度が限界であり, それ以上の長い相関は実際上取り扱えないという問題があった.

しかしながら, 現実の言語データには "The United States of America" のように, トライグラムを超える長いフレーズや固有名詞が頻出する. これらをチャンクとして分類し, 一単語とみなす方法もあるが, これには教師データが必要なために主観に依存する上, 慣用句のような系列を全てカバーすることは難しいという問題がある.[1] 特に, 日本語や中国語のように単語境界が曖昧な言語の場合, 品詞体系によっては短い助詞の連続などによる長い$n$グラムが頻出する可能性があり, 「分割の粒度に依存しない言語モデル」が特に重要だと考えられる.

また逆に, "superior to" のように短い$n$グラムで充分な文法的関係も多いことを考えると, $n$ を常に 3 などの固定値とするのではなく, 文脈に応じて必要なだけの長さを用いる「可変長$n$グラム言語モデル」の意義は言語学的にみても高いと考えられる.

しかしながら, これまで提案されてきた "可変長$n$グ

[1] 何が句であるかには 0/1 の正解はなく, 確率的にしかとらえられないと考えている [2]. さらに, この区切りは固定ではなく, 一般に文脈にも依存する.

ラムモデル" はいずれも実際には, 最大オーダーの$n$グラムを最初に作り, それを枝刈りするか [3][4], または頻度閾値でカットオフを行うもの [5] であった. この際の閾値や MDL などの基準はモデルとは別に外部から与えるものであり, さらに, 指数的に大きくなる最大モデルを事前に作る必要がある点で, 可変長モデルの構成意図と矛盾していた. バイオインフォマティクス [6] および統計学 [7] の分野で最近提案された可変長マルコフモデルにおいても, この問題点は依然解決されていない.

これまで, この問題に理論的な解決が存在しなかった理由は, $n$グラム分布を階層的に生成する確率モデルが存在しなかったためだと考えられる. しかし, 最近 [8][9] により, 階層 Pitman-Yor 過程とよばれるノンパラメトリックな確率過程によって, 適切にスムージングされた$n$グラム分布を 0-gram→1-gram→2-gram→$\cdots$ とベイズ統計の枠組から階層的に生成/推定できることが明らかになった.

そこで本論文では, この階層 Pitman-Yor 過程による$n$グラムモデルをさらに拡張し, データ中の各単語が生成された$n$グラム長を隠れ変数とみなしてベイズ推定を行うことで, 文脈により様々に長さの異なる可変長$n$グラムの生成モデルを提案する. この方法により, $n$グラムモデルの$n$を指定せず, 原理的には無限長とすることができ, 従来不可能だった高次$n$グラムの推定が可能となる. また, 言語モデルの副産物として, 上で述べた可変長の「句」を教師なし学習の枠組から, 確率的に推定することができる.

まず 2 節で, 階層 Pitman-Yor 過程による$n$グラム言語モデルについて説明し, 3 節で Suffix Tree 上の確率過程を考えることにより, これを原理的に無限長に拡張する. 4 節で LDA を用いたそのトピック適応化について述べ, 5 節で実験結果を示す. 6 節で考察および関連研究について述べ, 7 節で全体のまとめと将来の展望についてふれる.

## 2 階層 Pitman-Yor 過程と n-gram 言語モデル

階層 Pitman-Yor 過程による$n$グラム言語モデル (HPYLM) [8][9] は, 初めて提案された$n$グラム分布の階層的生成モデルであり[2], 後に述べるように, 現在最高性能といわれる言語モデルの Kneser-Ney スムージング [10] は, この確率過程の近似となっている.

階層 Pitman-Yor 過程は, ディリクレ過程の階層化である階層ディリクレ過程 [11] の拡張であり, 確率論の分野では 2-パラメータポアソン=ディリクレ過程 [12] と呼ばれている Pitman-Yor 過程の階層化であるが, ここでは直接測度論に基づく式ではなく, それと等価な

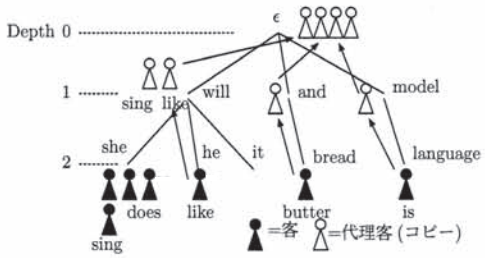


図 1: 階層的 CRP の Suffix Tree による表現. 客一人一人が, モデル上のカウントを表している.

階層的な CRP (Chinese Restaurant Process) [13] を使って, 直感的に説明する.

例として, トライグラムの言語モデルを考えよう. これは図 1 のように, 深さ 2 の Suffix Tree (接尾辞木) によって表すことができる. いま, 文脈 'she will' に続いて 'sing' を予測する場合, この木を根 $\epsilon$ (ユニグラムに対応する) から $\epsilon \to$ will $\to$ she の順に枝をたどり[3], 到達したノードの持つトライグラムのカウント分布を用いて, $p$(sing|she will) を計算する.

さて, HPYLM の学習では, 根だけの初期状態の木から始め, 学習データの 1 つ 1 つのカウントについて, カウント (CRP では, 可算無限個のテーブル = 単語の種類を持つ中国料理店のメタファーから, このカウントを「客」と呼ぶ) が図 1 のように, 深さ 2 の対応するノードに追加されていく.

ここで一般には, どのノードにも全ての種類の単語のカウントがあるわけではない. そこで, 客がノードに追加される時, ある確率で[4]その客のコピー (代理人) が親ノードの客として送られる. たとえば, ノード 'she will' に客 'like' がいなくても, 姉妹ノード 'he will' に 'like' がいれば, そのコピーが共通の親ノード 'will' に送られているため, 確率 $p$(like|she will) は 3-gram 確率 $p$(like|she will)(これは 0) と, 親ノードの持つ 2-gram 確率 $p$(like|will) を適切に補間して計算される. この過程は再帰的に行われ, 親ノードに客 'like' が追加される際にも, またコピーが親ノードに送られ, 2-gram 確率は 1-gram 確率との補間で計算されるため[5], 直接親ノードにない語についても, 必ず確率を求めることができる.

結果として HPYLM においては, 文脈 $h = w_{t-n} \cdots w_{t-1}$ に続く単語 $w$ の確率は,

$$p(w|h) = \frac{c(w|h) - d \cdot t_{hw}}{\theta + c(h)} + \frac{\theta + d \cdot t_{h\cdot}}{\theta + c(h)} p(w|h') \qquad (1)$$

として再帰的に計算される.

ここで $h' = w_{t-n+1} \cdots w_{t-1}$ は最も遠い語を落とした文脈であり, $c(w|h)$ はノード $h$ での $w$ のカウント, $c(h) = \sum_w c(w|h)$ は $h$ での総和である. $t_{hw}$ は $w$ が

[2] これまでの n-gram の確率モデルは各文脈での頻度をディスカウント後, 和を 1 に正規化して確率とするものであり, その頻度の由来を問うものではなかった. また, 各 1-gram, 2-gram,$\cdots$ の確率は, 基本的に独立に計算されていた.

[3] ノードが存在しない場合は, 新しく作成する.

[4] 式 (1) において, 単語が親ノードである第 2 項から生成されたと判断された場合.

[5] 1-gram 確率はさらに, 0-gram 確率 (全ての確率が $1/V$) と補間される.

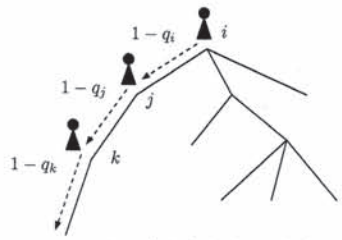


図 2: Suffix Tree 上に客を追加する確率過程. $(1-q_i)$ はノード $i$ の持つ「通過確率」を表す.

$p(w|h)$ からではなく, 親ノード $p(w|h')$ から生成されたと推定された回数であり, $t_{h\cdot} = \sum_w t_{hw}$ と書いた. $d, \theta$ は Pitman-Yor 過程のパラメータであり, Suffix Tree 上のすべての客の分布から, それぞれベータ事後分布, ガンマ事後分布によって推定できる. 詳しい計算については, [8] を参照されたい.

(1) 式は, カウント $c(w|h)$ をディスカウントした分布を 1 つ低いオーダーの分布と補間した確率になっているが, $t_{hw} \equiv 1$ とすると, この式は Kneser-Ney スムージングと一致し, Kneser-Ney スムージングはこの確率過程の近似であることがわかる.

さて, ここでの問題は, 図 1 において客がすべて深さ $(n-1)$ のノードに到着するとしていることである. 実際には, 自然言語の持つ Suffix tree はある所は浅く, ある所は非常に深い, 図 2 のような形をしており, 言語は様々に長さの違う $n$グラム文脈から生成されているはずである. それでは, 自然言語の持つこのような木をどのようにして推定すればよいのだろうか.

## 3 可変長階層 Pitman-Yor 過程

### 3.1 Suffix Tree 上の確率過程

そこで我々は, Suffix Tree の各ノード $i$ が, 木を根からたどる時にそこで止まる確率 $q_i$ (すなわち, $(1-q_i)$ はノード $i$ の「通過確率」を表す) を持っていると考え, 各 $q_i$ は共通のベータ事前分布

$$q_i \sim \mathrm{Be}(\alpha, \beta) \tag{2}$$

から生成されていると仮定する.

ここで, $\mathrm{Be}(\alpha,\beta) = \Gamma(\alpha+\beta)/(\Gamma(\alpha)+\Gamma(\beta)) \times q^{\alpha-1}(1-q)^{\beta-1}$ は二項確率 $q$ の確率分布であるベータ分布の確率密度関数であり, 期待値は $E[q] = \alpha/(\alpha+\beta)$ である. 図 3 に, ハイパーパラメータ $(\alpha,\beta)$ の違いによるベータ分布の例をいくつか示す.

文脈 $h = w_{t-\infty}\cdots w_{t-2}w_{t-1}$ に続いて単語 $w_t$ が観

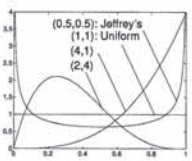


図 3: ベータ分布の例とハイパーパラメータ.

測されたとき, 我々は Suffix Tree を根 $\epsilon$ から始めて, $\epsilon \to w_{t-1} \to w_{t-2} \to \cdots$ の順にノードをたどって降りていくが, この時, 必ず深さ $(n-1)$ で止まるのではなく, このパス上の $q_i$ をそれぞれ $q_0, q_1, q_2, \ldots$ として, 確率

$$p(n = l|h) = q_l \prod_{i=0}^{l-1}(1-q_i) \tag{3}$$

に従って深さ $l$ で停止し, そこに客を追加する (図 2).

この式からわかるように, 非常に深いノードであっても, そのパスに沿った $q_i$ が小さければ (すなわち, 「通過確率」が高ければ) 深い $n$グラムが使われ, 逆に浅いノードでも, $q_i$ が大きければ (「通過確率」が低ければ) そこで止まる確率が大きくなる. 式 (3) より, 深さ $n$ のノードに到達する確率は $n$ に従っておおよそ指数的に減少するが, その度合は木の枝によって異なり, 高頻度の長い系列に対応する深いノードを許すことができる.[6]

### 3.2 可変長ベイズ n-gram 言語モデル

もちろん, われわれは言語の Suffix Tree のノードが持つ真の $q_i$ の値は知らない. それでは, どうやって $q_i$ を推定すればよいだろうか.

ここで, 上の可変長生成モデルでは, データ $\mathbf{w} = w_1 w_2 \cdots w_T$ について, それぞれの単語が生成された隠れた $n$グラム長 $\mathbf{n} = n_1 n_2 \cdots n_T$ が存在していると仮定していることに注意したい. したがって, われわれのモデルでは, データ $\mathbf{w}$ の確率は

$$p(\mathbf{w}) = \sum_{\mathbf{n}} \sum_{\mathbf{s}} p(\mathbf{w}, \mathbf{n}, \mathbf{s}) \tag{4}$$

として表される. ここで, $\mathbf{s}$ は代理客を含む客全体の配置を表す隠れ変数である [8]. この $\mathbf{n}, \mathbf{s}$ は, ギブスサンプリングにより求めることができる.

具体的には, 単語 $w_t$ の持つ隠れた $n$グラムオーダー $n_t$ を,

$$n_t \sim p(n_t|\mathbf{w}, \mathbf{n}_{-t}, \mathbf{s}_{-t}). \tag{5}$$

のようにサンプリングして更新していく. ($s_t$ も, この時自動的にサンプリングされる.) ここで $\mathbf{n}_{-t}, \mathbf{s}_{-t}$ はそれぞれ, $\mathbf{n}, \mathbf{s}$ から $n_t, s_t$ を除いたベクトルである.

上の式で, $n_t$ をサンプリングする際, われわれは他の客 (単語) が Suffix Tree をたどった深さ $\mathbf{n}_{-t}$ を全て知っていることに注意されたい. したがって, 他の客がノード $i$ で止まった回数を $a_i$, 通過した回数を $b_i$ とおくと, $q_i$ の期待値は, ベータ事後分布の期待値として

$$E[q_i] = \frac{a_i+\alpha}{a_i+b_i+\alpha+\beta} \tag{6}$$

と推定することができる. ここで, 式 (5) は

$$p(n_t|\mathbf{w}, \mathbf{n}_{-t}, \mathbf{s}_{-t}) \propto p(w_t|\mathbf{w}_{-t}, \mathbf{n}, \mathbf{s}_{-t})\, p(n_t|\mathbf{w}_{-t}, \mathbf{n}_{-t}, \mathbf{s}_{-t}) \tag{7}$$

[6] 式 (2)(3) の形からわかるように, これは Suffix Tree 上で定義された, Beta two-parameter process の Stick-breaking process [13] である.

For $j = 1 \cdots N$, {
  For $t = \text{randperm}(1 \cdots T)$, {
    if $(j > 1)$ then
      remove_customer (order[$t$], $w_t$, $w_{1:t-1}$),
    order[$t$] = add_customer ($w_t$, $w_{1:t-1}$).
  }
}

図 4: VPYLM のギブスサンプラ.

と展開できるが, この第一項は$n$グラムオーダーが $n_t$ と決まったときの $w_t$ の$n$グラム確率であり, 式 (1) から求められる. 第二項は, この文脈で深さ $n_t$ のノードに到達する事前確率であり, (3) および (6) 式から,

$$p(n_t = l|\mathbf{w}_{-t}, \mathbf{n}_{-t}, \mathbf{s}_{-t}) = \frac{a_l+\alpha}{a_l+b_l+\alpha+\beta} \prod_{i=0}^{l-1} \frac{b_i+\beta}{a_i+b_i+\alpha+\beta} \quad (8)$$

として計算することができる.

これらの確率を用いて, 各単語の持つ隠れた$n$グラムオーダーをサンプリングする, 図 4 のギブスサンプラを構成することができる. このアルゴリズムは HPYLM のギブスサンプリングの拡張であり, 単語 $w_t$ に対応して Suffix Tree の order[$t$] の深さにいる客を一人, ノードをたどって削除し, 新しい$n$グラムオーダーを (7) 式からサンプリングした後, 新しい深さに再配置する. この時, 親ノードの代理客が同時に確率的に削除/追加される.

(7) 式から $n_t$ をサンプリングする際, 実際にはある最大値 $n_{\max}$ を設定し, その中でサンプリングを行うか, または $n_t$ の事前確率 (8) がある小さな値 $\epsilon$ 以下になるまで計算を行う. このとき, $n$グラムモデルにおいて 'n' は必要なくなり, われわれは原理的に, 「$\infty$-gram 言語モデル」を得たことになる.

### 3.3 予測と文脈長確率分布

予測の際にも, われわれは従来のように$n$グラム長を固定していないため, $n$ を隠れ変数とみなして, 文脈 $\mathbf{h}$ に対して

$$p(w|\mathbf{h}) = \textstyle\sum_n p(w, n|\mathbf{h}) \quad (9)$$
$$= \textstyle\sum_n p(w|n, \mathbf{h}) p(n|\mathbf{h}) \quad (10)$$

として予測を行う. ここで第 2 項は文脈 $\mathbf{h}$ の持つ$n$グラム文脈長分布であり, (7) 式で与えられる. 第 1 項はオーダーを $n$ とした HPYLM の予測確率であり, (1) 式で与えられるが, この確率はすでに, Kneser-Ney スムージングと同等に階層的にスムージングされていることに注意されたい. すなわち, VPYLM の予測確率は, HPYLM の$n$グラム確率をさらに $n = 0, 1, \cdots, \infty$ の事後分布で混合したものになっている. 実際には (10) をさらに, 訓練データ $\mathbf{w}$ からの $\mathbf{n}, \mathbf{s}$ の $N$ 個の Gibbs 事後サンプルで平均化して予測を行う. [7]

[7] これは, Gibbs サンプリングが事後分布からの正確なサンプリングであり, 事後分布最大化を行うものではないからである. ディリクレ過程混合モデルに対して最近提案されたモデル探索法 [14] は, この意味で興味深い.

```
struct ngram {          /* n-gram node */
    ngram *parent;
    splay *children;    /* = (ngram **) */
    splay *words;       /* = (restaurant **) */
    int ncounts;        /* c(h) */
    int ntables;        /* t_h. */
    int stop;           /* a_i */
    int through;        /* b_i */
    int id;             /* word id */
};
```

図 5: $n$ グラムノードのデータ構造.

### 3.4 実装

ギブスサンプリングの際, Suffix Tree 上で数千から数万にのぼることがある子ノードをたどる操作を高速化するため, われわれは [5] と同様に, 子ノードの管理にスプレー木 [15] を使用した. スプレー木は $O(\log n)$ で子ノードを探索可能な二分順序木であり, 木の自己組織的な最適化を行うことで, 高頻度なアイテムへの特に高速なアクセスを提供するため, 自然言語のように Power Law の性質を持つ系列での使用に適している. われわれはこのスプレー木を子ノードだけでなく, 各ノードの持つ予測単語[8]の管理にも用いた. 図 5 に, Suffix Tree の持つ各 n-gram ノードのデータ構造を示す.

## 4 可変長ベイズ n-gram 言語モデルのトピック適応

可変長$n$グラムモデルについて, そのベイズ生成モデルが得られたので, 次にそのトピック適応化を自然に考えることができる.

### 4.1 Latent Dirichlet Allocation (LDA)

このためのモデルとして, 我々は LDA[16] を使用した. LDA では, 各文書に, 隠れたトピック混合分布 $\theta_d = (\theta_1, \theta_2, \ldots, \theta_M)$ があり, $\theta$ がディリクレ事前分布

$$\boldsymbol{\theta} \sim \text{Dir}(\boldsymbol{\gamma}) = \frac{\Gamma(M\gamma)}{\Gamma(\gamma)^M} \prod_{t=1}^{M} \theta_t^{\gamma-1} \quad (11)$$

から生成されていると仮定する. 簡単のため, 上ではディリクレ分布のハイパーパラメータは全て同じ値 $\gamma$ を用いた. このとき, 文書 $d = w_1 w_2 \cdots w_N$ の各単語は, 次のようにして生成される.

1. $\boldsymbol{\theta}_d \sim \text{Dir}(\boldsymbol{\gamma})$ をサンプル.
2. For $n = 1 \ldots N$,
   a. $t \sim \text{Mult}(\boldsymbol{\theta}_d)$ をサンプル.
   b. $w_n \sim p(w|t)$ をサンプル.

ここで, トピック言語モデル $p(w|t)$ はオリジナルの LDA ではユニグラム分布が使われているが, LDA は混合モデルであるため, これは任意の分布に置き換えることができ, 例えばトピックごとの VPY 言語モデルとすることができる.

しかしながら予備実験の結果, トピック毎の VPYLM を混合する方法では, トピックを考慮しない時に比べて予測精度が悪化した.

[8] CRP においては, 「レストラン」と呼ばれている.

```
For j = 1 ··· N, {
  For t = randperm(1 ··· T), {
    d = document(w_t).
    k = topic[t].
    if (j>1) then
      remove_customer (table[t]),
    k = draw_topic (w_t,w_{1:t-1},d),
    table[t] = add_customer (w_t,w_{1:t-1},k),
    topic[t] = k.
  }
}
```

図 6: VPYLDA のギブスサンプラ.

この理由は, LDA のギブスサンプリングによりデータを分割し, トピック別の $n$グラム言語モデルを構築すると, $n$グラムのスパース性がより深刻になるためだと考えられる. トピックを動的に推定することによる精度上昇より, 選択されたトピック言語モデルのカバレージ不足による精度悪化の方が重要となるからである.

### 4.2 ギブスサンプリングによるモデル推定

そこでわれわれは, VPY 言語モデルにおいて, データの多いユニグラム分布のみを混合分布 (混合 Pitman-Yor 測度) とすることとした. LDA によって決まるトピック分布 $\boldsymbol{\theta}_d$ に従い, 単語ごとに異なるユニグラム分布を用いた $n$グラム分布から単語が生成される.

このとき, 階層 Pitman-Yor 過程によって生成される $n$グラムカウントは, テーブル[9]ごとに生成されたユニグラム分布が異なるため, ギブスサンプリングの際, カウントがどのテーブルに追加され, その親テーブルは何かを明示的に追跡する必要がある.

この情報を用いて, VPY 言語モデルの LDA 化 (VPYLDA) のギブスサンプリングは, 図 6 のように行うことができる. ここで draw_topic は, 単語 $w_t$ がトピック $k$ に属する事後確率

$$p(k|w_t, w_{1:t-1}) \propto p(w_t|w_{1:t-1}, k)\, p(t|\theta_d) \quad (12)$$

$$\propto p(w_t|w_{1:t-1}, k) \cdot (n^d_{-t,k} + \gamma) \quad (13)$$

から $k$ をサンプルする関数である. $p(w|h,k)$ はユニグラム分布 $k$ を用いた VPY 言語モデルの予測確率 (10), $n^d_{-t,k}$ は文書 $d$ 中でトピック $k$ に割り当てられた単語数 ($w_t$ を除く) である.

## 5 実験

### 5.1 VPYLM on NAB and Sinica

言語モデル研究で使われる英語と中国語の標準的なコーパスを用いて実験を行った.

**データ** 英語については, [17] 等でも使われている NAB (North American Business News) コーパスの WSJ セットよりランダムに選択した 409,246 文, 10,007,108 語を訓練データ, さらに 10,000 文を評価データとした. 単語はすべて小文字とし, 頻度 10 以下の単語は同じ特別な語に写像した. 総語彙数は 26,497 語である. 中国語については Sinica バランスドコーパス [18] のうち, ランダムに選んだ 700,000 文を訓練データ, 残り 8,953 文を評価データとして文字単位の言語モデルを構築した. 1 文の平均文字数は 11.8 文字, 漢字の総数は 6,087 個であった.

[9]各 $n$グラムノード $h$ でのカウントは, 単語ごとのテーブルの一つに加算される. このテーブルは, 単語 $w$ ごとに $t_{hw}$ 個存在する.

| $n$ | HPYLM | VPYLM | Nodes(H) | Nodes(V) |
|---|---|---|---|---|
| 3 | 113.60 | 113.74 | 1,417K | **1,344K** |
| 5 | 101.08 | 101.69 | 12,699K | **7,466K** |
| 7 | N/A | **100.68** | N/A | **10,182K** |
| 8 | N/A | **100.58** | N/A | **10,434K** |
| $\infty$ | — | 161.68 | — | **6,837K** |

(a) NAB コーパス (英語)

| $n$ | HPYLM | VPYLM | Nodes(H) | Nodes(V) |
|---|---|---|---|---|
| 3 | 47.72 | 47.76 | 722K | **690K** |
| 5 | 42.74 | 43.39 | 8,705K | **4,599K** |
| $\infty$ | — | 73.16 | — | **3,884K** |

(b) Sinica コーパス (中国語)

表 1: 英語と中国語のコーパスにおける, HPYLM と VPYLM のテストセットパープレキシティ, 及びモデルのノード数. N/A はメモリオーバーフローを表す.

**学習設定** 予備実験の結果と計算量的制約から, HPYLM, VPYLM のそれぞれのモデルについて $N = 200$ 回の Gibbs サンプリングを行い, さらに 50 回の事後サンプルを用いて評価を行った. Suffix Tree の形を決めるベータ事前分布のハイパーパラメータは $(\alpha, \beta) = (4, 1)$ としたが, これはより深い木が得られる $(1, 1)$ の一様分布を用いた場合等とほとんど性能差がなかったためである. $(\alpha, \beta)$ の値による性能差とその最適化については, 6 節を参照されたい. 実験は全て Xeon 3.2GHz, メモリ 4GB の Linux 上で行った.

**結果** 表 1 に HPYLM と VPYLM のパープレキシティを, モデル中の n-gram ノード数とともに示す. モデル次数 $n$ は前者では固定, 後者では最大値である.

この結果から, VPYLM は HPYLM とほぼ同等の性能を40%以上少ないノード数で達成し, HPYLM では推定できない $n = 7, 8$ のような高次 n-gram についても, 必要なもののみを選択的にモデルに加えることで推定が可能であり, より高い性能を持つことがわかる.

また, 同じ (最大) オーダー $n$ の場合でも, VPYLM は HPYLM より学習が高速である. これは, VPYLM において $n$グラムオーダーをサンプリングする計算コストよりも, 不必要に深いノードを追加しないことによる計算量削減が大きいからだと考えられる. 図 7 に, 8-gram VPYLM において推定された $n$グラムオーダーのデータ全体での分布を示す. 文脈長を長くするメリットと, 深いノードに到達するデメリットの間で適切なトレードオフが行われ, $n = 3, 4$ 程度をピークに指数的な減衰が起きていることがわかる.

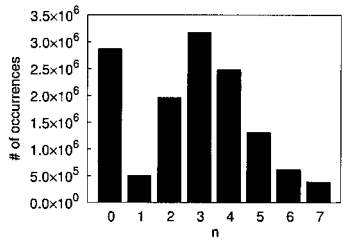


図 7: 8 グラム VPYLM で推定された $n$ グラム文脈長のデータ全体での分布. $n=0$ がユニグラム,$n=1$ がバイグラム,... である.

### 5.2 Suffix Tree と確率的フレーズ

提案法は, Ron ら [19] における Suffix Tree と等価な確率オートマトンに, さらに入力を可変長とする状態遷移確率を加えたものとも考えることができる. すると, 木の根 (初期状態) からたどって, 全ての可変長 $n$ グラム $\mathbf{w} = w_1 w_2 \cdots w_{n+1}$ がモデルから生成される事前確率は,

$$p_0(\mathbf{w}) = \langle q_n \rangle \prod_{i=1}^{n-1} (1 - \langle q_i \rangle) \cdot p(w_{n+1}|w_1 \cdots w_n)$$

として計算することができる. ここで, $\langle q_i \rangle = E[q_i]$ は (6) 式で与えられる, パス $w_n \to w_{n-1} \to \cdots \to w_1$ に沿った確率である. この式は, HPYLM では末端ノード以外の全ての $q_i$ が 0 (必ず深さ $(n-1)$ のノードに到達する) なため無意味なことに注意されたい. 図 8 に, 8-グラム VPYLM から得られた「確率的フレーズ」の例を, 上の確率でソートして示す.

### 5.3 VPYLDA on 20news-18828

**データと学習設定** 可変長 $n$ グラム LDA(VPYLDA) の評価データとして, われわれは 20news-18828 データセット [20] を用いた. このうち, 20 個のニュースグループを同等にカバーするように 4,000 文書を訓練データ, 400 文書を評価データとした.

訓練データは NAB コーパスと同様に前処理し, 801,580 語, 語彙数 10,477 語のデータとなった. $N=250$ 回のギブスサンプリングを行った後の単一サンプルを用い, 文脈 $h = w_1 \cdots w_{n-1}$ での単語 $w_n$ の予測確率を, 順に以下のように求めた.

$$p(w_n|h) = \sum_t p(w_n|h,t) \langle p(t|h) \rangle \qquad (14)$$

ここで $p(w_n|h,k)$ はトピック $t$ のユニグラムを用いた可変長 $n$ グラムの予測確率 (10) であり, $\langle p(t|h) \rangle$ は仮想的な文書 $h$ のもつトピック $t$ の事後期待値である. 言語モデルは固定であるから, 高速化のため, この計算には LDA の変分ベイズ EM アルゴリズム [16] を用いた.

**実験結果** 表 2 に, トピック混合数 $M$ を変えたときの VPYLDA のテストセットパープレキシティを示す. $M=1$ のとき, これは VPYLM と等価であり, 最下段に示した.

予測は良くはなっているが, その差はわずかである

| $p$ | Stochastic phrases in the suffix tree |
|---|---|
| 0.9784 | primary new issues |
| 0.9726 | BOS at the same time |
| 0.9556 | american telephone & |
| 0.9512 | is a unit of |
| 0.9394 | to NUM % from NUM % |
| 0.8896 | in a number of |
| 0.8831 | in new york stock exchange composite trading |
| 0.8696 | a merrill lynch & co. |
| 0.7566 | mechanism of the european monetary |
| 0.7134 | increase as a result of |
| 0.6617 | tiffany & co. |
| : | |

図 8: NAB コーパスで学習した 8 グラム VPYLM の Suffix Tree から得られた, 確率的フレーズ.

ことがわかる. 学習されたモデルを観察したところ, この理由は, 高頻度の単語が各トピックユニグラムに均等に分配されないためであることがわかった. 図 1 の階層的 CRP において, トピック $t$ をもつ客のコピーが親ノードに送られるのは, その客が現在のノードではなく, 親ノードから生成されたと確率的に判断された場合であるが, 高頻度の語は多くのノードで既に存在するため, カウントが再利用されて単に 1 増やされ, トピック別ユニグラムまで再帰的に到達しない.

トピックはユニグラムに限られるものではなく, バイグラムやトライグラム以上にも別の混合数で存在する[10]ことを考えると, このような階層的な混合モデルの推定にはまた新しい方法を必要とすると思われる.

## 6 考察および関連研究

本研究は, データ圧縮における Context Tree Weighting 法 [21] を自然言語に拡張した, Pereira らの興味深い研究 [5] をその動機としている. 彼らの手法も様々な深さの木の事後確率を考え, それらを混合するものであるが, $n$グラム分布を階層的に推定するモデルは存在しなかったため, 最終的に各ノードでの Witten-Bell スムージングと頻度によるカットオフに頼っており[11], 性能面でのアドバンテージがないという結果になっていた. これに対し本研究では, [5] で定数だった木の事前確率をベイズ化し, HPY 言語モデルでスムージングされた $n$グラムをさらに混合し, 生成モデルの立場からカウント毎に確率的なプルーニングを行うという点で, 彼

| Model | PPL |
|---|---|
| VPYLDA ($M=5$) | **104.69** |
| ($M=10$) | **103.57** |
| ($M=20$) | **103.28** |
| VPYLM | 105.30 |

表 2: 20news-18828 データセットにおける VPYLDA のテストセットパープレキシティ.

[10] たとえば, 'mixture of Gaussians' と 'mixture of flour' の出現は相補的だと考えられる.

[11] 頻度による一様なカットオフは, [4] により性能がきわめて悪いことが後に示されている.

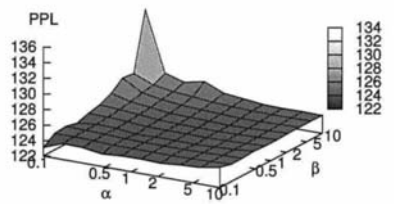


図 9: ハイパーパラメータ $(\alpha, \beta)$ を変えた時の, VPYLM のテストセットパープレキシティ.

らの手法の後継となっていると考えることができる.

5節の実験では, Suffix Tree の形を決める事前分布として $(\alpha, \beta) = (4, 1)$ を用いたが, このパラメータは経験ベイズ法により, (6) 式で各 $q_i$ がもつベータ事後分布から最適化することも可能である. [22] の Newton 法を用いると, 1M 語の NAB コーパスのサブセットの場合, この値は各繰り返しですべて $(0.85, 0.57)$ に収束した.

しかしながら, VPY 言語モデルの性能はハイパーパラメータにはあまり依存しない. 図 9 に, $(\alpha, \beta) \in (0.1 \sim 10) \times (0.1 \sim 10)$ の範囲で変えたときの, 上のデータでのパープレキシティを示す. $\beta \gg \alpha$ となる場合以外は, 性能はほぼ一定であることがわかる.

## 7 まとめと展望

本論文では, 最近提案された n-gram 分布に対するノンパラメトリックな確率過程である階層 Pitman-Yor 過程をさらに拡張することで, 単語の生まれた n-gram 文脈長を適切に推定する可変長 n-gram 言語モデルを示した. 無限の深さをもつ Suffix Tree 上の確率過程を考えることで, これまで固定だった $n$ を原理的に不要とし, データから推定することができる.

ここで, 本研究の手法は言語モデルに限られることなく, 可変長マルコフモデルを実現する一般的なベイズ的方法であることに注意されたい. プルーニングに基づくこれまでの "可変長モデル" と異なり, 提案手法は可変長系列の完全な生成モデルとなっている.

提案法は最大オーダーの従来法と同等の性能を, より少ない空間的および時間的計算量で達成する. また, 言語モデルの副産物として, 学習された Suffix Tree から特徴的な系列を確率つきで取り出せることを明らかにした.

本研究では Suffix Tree の事前分布に, 柔軟だが単純な共通のベータ分布を用いたが, [23] に見られるような事前分布を考えることで, モデルをより適切にすることができると考えている. また, トピック適応化についても, 階層的な混合モデルのよりよい推定法を考えたい.

## 参考文献


[1] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[2] 工藤拓. 形態素周辺確率を用いた分かち書きの一般化とその応用. 言語処理学会全国大会論文集 NLP-2005, 2005.

[3] Manhung Siu and Mari Ostendorf. Variable n-grams and extensions for conversational speech language modeling. *IEEE Trans. on Speech and Audio Processing*, 8:63–75, 2000.

[4] Andreas Stolcke. Entropy-based Pruning of Backoff Language Models. In *Proc. of DARPA Broadcast News Transcription and Understanding Workshop*, pages 270–274, 1998.

[5] Fernando Pereira, Yoram Singer, and Naftali Tishby. Beyond Word N-grams. In *Proc. of the Third Workshop on Very Large Corpora*, pages 95–106, 1995.

[6] Florencia G. Leonardi. A generalization of the PST algorithm: modeling the sparse nature of protein sequences. *Bioinformatics*, 22(11):1302–1307, 2006.

[7] Peter Buhlmann and Abraham J. Wyner. Variable Length Markov Chains. *The Annals of Statistics*, 27(2):480–513, 1999.

[8] Yee Whye Teh. A Bayesian Interpretation of Interpolated Kneser-Ney. Technical Report TRA2/06, School of Computing, NUS, 2006.

[9] Yee Whye Teh. A Hierarchical Bayesian Language Model based on Pitman-Yor Processes. In *Proc. of COLING/ACL 2006*, pages 985–992, 2006.

[10] Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. In *Proceedings of ICASSP*, volume 1, pages 181–184, 1995.

[11] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet Processes. Technical Report 653, Department of Statistics, University of California at Berkeley, 2004.

[12] Jim Pitman and Mark Yor. The Two-Parameter Poisson-Dirichlet Distribution Derived from a Stable Subordinator. *Annals of Probability*, 25(2):855–900, 1997.

[13] Zoubin Ghahramani. Non-parametric Bayesian Methods. *UAI 2005 Tutorial*, 2005. http://learning.eng.cam.ac.uk/zoubin/talks/uai05tutorial-b.pdf.

[14] Hal Daumé III. Fast search for Dirichlet process mixture models. In *AISTATS 2007*, 2007.

[15] Daniel Sleator and Robert Tarjan. Self-Adjusting Binary Search Trees. *JACM*, 32(3):652–686, 1985.

[16] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[17] Stanley F. Chen and Joshua Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. In *Proc. of ACL 1996*, pages 310–318, 1996.

[18] Chu-Ren Huang and Keh-jiann Chen. A Chinese Corpus for Linguistics Research. In *Proc. of COLING 1992*, pages 1214–1217, 1992.

[19] Dana Ron, Yoram Singer, and Naftali Tishby. The Power of Amnesia. In *Advances in Neural Information Processing Systems*, volume 6, pages 176–183, 1994.

[20] Ken Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339, 1995.

[21] F.M.J. Willems, Y.M. Shtarkov, and T.J. Tjalkens. The Context-Tree Weighting Method: Basic Properties. *IEEE Trans. on Information Theory*, 41:653–664, 1995.

[22] Thomas P. Minka. Estimating a Dirichlet distribution, 2000. http://research.microsoft.com/~minka/papers/dirichlet/.

[23] Jim Pitman. Combinatorial Stochastic Processes. Technical Report 621, Department of Statistics, University of California, Berkeley, 2002.