Vol.2012-NL-208 No.4
2012/9/2

潜在トピックを考慮したBayes n-gram言語モデル

能地 宏[1,a)]　持橋 大地[2,b)]　石塚 満[1,c)]

概要：文書の潜在トピックを捉え，トピックに応じた適切な n グラムを用いて予測を行う Bayes 的な n グラム言語モデルを提案する．文章には，単語の出現が文書のトピックに依存して決まる内容語と，文法的な関係のみで決まる機能語が存在する．我々はこれらの単語の出現が，文脈によっておおまかに決まることに着目し，適切な箇所でのみトピックを考慮した予測を行うモデルとして，2 種類のモデルを提案し，比較を行う．トピック別の n グラムモデルを，通常の Gibbs サンプリングで学習したのではすぐに局所解に陥ってしまうことを実験的に示し，それを回避するための新しい Blocked Gibbs サンプリングを提案する．提案法は，パープレキシティの比較において，Unigram Rescaling と同等以上の性能を示しながら，予測時間の大幅な改善を行うことを確認した．

キーワード：Bayes 言語モデル，トピックモデル，Pitman-Yor 過程，Gibbs サンプリング

Latent Topic Aware Bayesian n-gram Language Model

Noji Hiroshi[1,a)]　Mochihashi Daichi[2,b)]　Ishizuka Mitsuru[1,c)]

Abstract: We study n-gram language models that can leverage latent topics of a given document, and predict words using n-gram models of these topics. Words in a sentence will appear because of short-range syntactic constraints or semantic dependencies of the document. We observe that whether next word given some context is function word or content word can roughly be predictable and propose two novel n-gram language models that predict words using topic-specific n-grams only when context has strong *topicality*. To avoid getting stucked in a local minima, novel Blocked Gibbs sampling based on a table in the internal Chinese Restaurant is proposed. We show that our models give superior or comparative performance than the traditional method, Unigram Rescaling in terms of test set perplexity, while drastically reduce the time for prediction.

Keywords: Bayesian Language Model, Topic Model, Pitman-Yor Process, Gibbs Sampling

1. はじめに

文章の生成確率を，単語毎に直前の $n-1$ 語のみでモデル化する n グラム言語モデルは，そのシンプルさと性能の良さから，機械翻訳や音声認識などで広く使われている．しかしながら，n グラムモデルが用いる情報は単語の周辺の局所的な情報に限られるため，文書や会話の内容など，大局的な情報に基づいた予測を行うことができない．一方，文書の大局的な内容を捉える言語モデルとして，Latent Dirichlet Allocation[2] を始めとしたトピックモデルが存在する．Gildea らは，文書のトピックを捉えることにより n グラム言語モデルの性能させる目的で，n グラムモデルとトピックモデルとの積モデルである Unigram Rescaling[4] を提案しており，音声認識への応用も報告されている [7]．しかしながら，Unigram Rescaling ではトピックをユニグラム分布でしか捉えることができず，トピックと n グラムとの完全な融合にはなっていない．また，予測の際に全単語に渡って正規化を行う必要があるため，非常に計算コス

[1] 東京大学大学院 情報理工学系研究科
Graduate School of Information Science and Technology, University of Tokyo
[2] 統計数理研究所
The Institute of Statistical Mathematics
[a)] noji@mi.ci.i.u-tokyo.ac.jp
[b)] daichi@ism.ac.jp
[c)] ishizuka@i.u-tokyo.ac.jp

トが大きいという問題が存在する．

これに対し，本研究ではトピック別に異なる n グラムを学習する Bayes 的な n グラム言語モデルについて取り扱う．n グラムのカウントは非常にスパースであるため，これらを単純にトピック毎に分割してしまうと精度が悪化してしまうことが指摘されている [10]．また，文章中の単語は大きく機能語と内容語に分けられる．このうち内容語は文書のトピックに大きく依存するのに対して，機能語はトピックに依存しないため，これらを分けてモデル化することが重要である [5]．ここで次の単語が機能語であるか，内容語であるかは文脈に応じておおまかに予測を行うことができる．例えば *in order* という文脈では次に出現する単語は *to* や *that* でほぼ決定するが，*in the* などの文脈では様々な単語が出現する可能性がある．

そこで我々は，上記 2 つの洞察に基づいたモデルを 2 種類提案し，比較を行う．どちらのモデルも，上記のトピックに依存しない大域的な言語モデルと，各トピックに固有の言語モデルとを，文脈に応じて切り替えるものである．1 つは大域的な言語モデルを，全てのトピック別言語モデルが共有する隠れ言語モデルと見なす．Graphical Pitman-Yor Process(GPYP)[19] を用いることで，トピック別 n グラム分布を，トピック別 $(n-1)$ グラム分布と大域的な n グラム分布とを用いて補間することができる．さらにその補間係数を文脈に応じて変化させることで，大域的な言語モデルとトピック別言語モデルの重みを，文脈に応じて調整する．これを Doubly Hierarchical Pitman-Yor Topic Model(DHPYTM) とよぶ．別のモデルとして，大域的な言語モデルに基づく予測と，トピック別言語モデルに基づく予測とを，文脈に応じて直接切り替えるモデルを提案する．DHPYTM とは異なり，大域的な言語モデルは，各トピック別言語モデルとは独立に生成され，どちらを使うかが，文脈に応じて決定される．これを composite Hierarchical Pitman-Yor Topic Model(cHPYTM) とよぶ．提案法は階層的なトピック分布 [17] を考えることで，トピック数の選択の問題を回避でき，Infinite Markov Model(IMM)[10] を組み合わせることで，最適な n グラムオーダーを文脈に応じて選択することができる．

推定の際，通常の token 毎に隠れ変数の推定を行う Gibbs サンプリングでは，n グラムの階層の深さ故に簡単に局所解に陥ってしまい学習がうまくいかない．そこで我々は，モデル内部の Chinese Restaurant の中で，1 つのテーブルを基にした交換可能な変数のブロックを構成し，それらをまとめてサンプルする新しい Table-based Sampling を提案する．

BNC コーパスや Brown コーパスなどの均衡コーパス，及び NIPS コーパスを用いて性能評価を行い，どちらのモデルも既存の混合モデルに基づく n グラムトピックモデルから性能を大きく改善させることを確認した．また

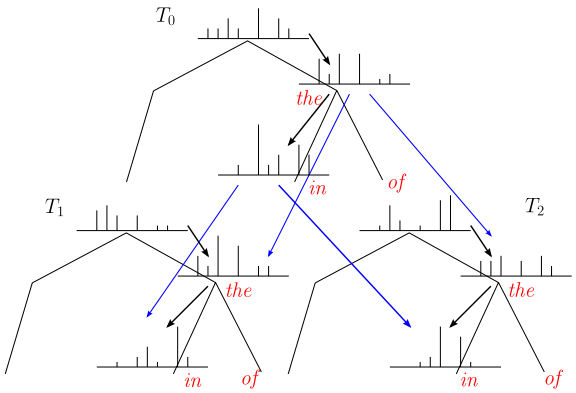

図 1　Suffix Tree 上の DHPYLM の生成過程の一部．T_i がドメイン i の言語モデルを表す．

GHPYTM と cHPYTM との比較では，cHPYTM のほうが安定して良い性能を示し，Unigram Rescaling との比較を行ったところ，予測時間を大幅に削減したうえで，予測性能で同等もしくは若干の向上が見られた．

以下では，まず 2 章で GPYP 及び，DHPYLM について説明する．3 章では我々の 2 つの提案モデル，及びその可変長モデルへの拡張について述べる．4 章では推論についてまとめる．通常の Token-based Sampling を説明し，その問題点，及び Table-based Sampling の構成法について述べる．5 章で関連研究についてまとめる．6 章で実験結果を示し，7 章で考察と今後の展望についてまとめる．

2. GPYP

まず提案モデルのうち DHPYTM の基礎となる，Graphical Pitman-Yor Process(GPYP) 及び，それを利用した言語モデルである Doubly Hierarchical Pitman-Yor Language Model(DHPYLM) について説明する [19]．これは，Teh による階層 Pitman-Yor 言語モデル (HPYLM)[13] を拡張したものである．

n グラム言語モデルでは，あらゆる長さ $(n-1)$ の文脈 $\mathbf{u}$ に対して，次に出現する単語分布 $G_{\mathbf{u}}$ を考える．ここで，$G_{\mathbf{u}}$ は語彙数 W 次元のベクトルであり，各要素 $G_{\mathbf{u}}(w)$ は文脈 u に続く単語 w の出現確率を表す．従来，この確率モデルとしては様々なスムージング法が提案されてきた．HPYLM では，$G_{\mathbf{u}}$ は，それに対応する $(n-1)$ グラムの確率である $G_{\mathbf{u}'}$ を基底分布とした Pitman-Yor 過程によって生成された，と考える．ここで $\mathbf{u} = w_{i-n+1} \cdots w_{i-1}$ はある長さ $(n-1)$ の文脈を表し，$\mathbf{u}' = w_{i-n+2} \cdots w_{i-1}$ はそこから左端の単語を取り除いた文脈を表す．HPYLM における n グラムの生成過程は，

$$G_{\mathbf{u}} \sim \mathrm{PYP}(a_{|\mathbf{u}|}, b_{|\mathbf{u}|}, G_{\mathbf{u}'}) \qquad (1)$$

と書ける．ここで $\sim$ は，左辺の量が右辺の確率分布に従って生成されることを示す．$a_{|\mathbf{u}|}, b_{|\mathbf{u}|}, G_{\mathbf{u}'}$ は Pitman-Yor 過程のパラメタであり，それぞれディスカウントパラメータ，

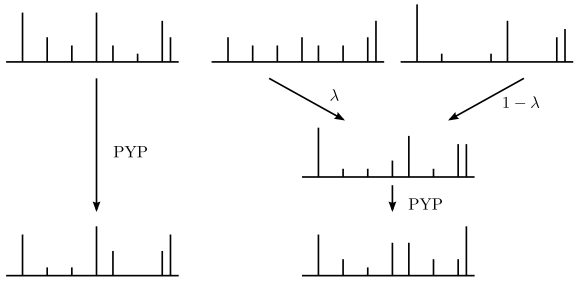

図 2　PYP と GPYP の分布の生成過程の比較．左が PYP，右が基底分布が 2 つの混合分布からなる GPYP を表す．

集中度パラメータ，基底分布と呼ぶ．また，HPYLM は n グラムの文脈を Suffix Tree 上で表現することができる．ここで，1 つ 1 つのノードはある n' グラム $(0 \leq n' \leq n)$ の文脈に対応し，$\mathbf{u}$ に対する $\mathbf{u}'$ は，その親ノードに対応する．3 グラムモデルの場合の，Suffix Tree の一部と，その上の生成過程を図 1 に示す． GPYP は，PYP における基底分布を任意個の分布の混合分布に置き換えるものである．HPYLM では，文脈 $\mathbf{u}$ を表すノードに対する親ノードは一意に定まった．ここではその制限を取り払い，$\mathbf{u}$ に対応する親ノードの集合が存在し，それらに対応する文脈の集合を $\pi(\mathbf{u})$ と表す．ここで，$\mathbf{v} \in \pi(\mathbf{u})$ が，1 つ 1 つの親ノードの文脈である．このとき GPYP では，$\mathbf{u}$ における単語分布を，次のように生成する．

$$G_{\mathbf{u}} \sim \mathrm{PYP}(a_{|\mathbf{u}|}, b_{|\mathbf{u}|}, \sum_{\mathbf{v} \in \pi(\mathbf{u})} \lambda_{\mathbf{v}\to\mathbf{u}} G_{\mathbf{v}}). \qquad (2)$$

ここで $\lambda_{\mathbf{v}\to\mathbf{u}}$ は混合係数であり，$\sum_{\mathbf{v}\in\pi(\mathbf{u})} \lambda_{\mathbf{v}\to\mathbf{u}} = 1$ を満たす．PYP と GPYP との分布の生成過程の違いを，図 2 に示す．

DHPYLM は，GPYP を用いて n グラム言語モデルのドメイン適応を教師ありで行うものである．ここで，適応元コーパスの言語モデルを $G^0_{\mathbf{u}}$，適応先の k 番目の言語モデルを $G^k_{\mathbf{u}}$ と表すこととする．まず，通常のコーパスを用いて，$G^0_{\mathbf{u}}$ の学習を行う．次に適応したいドメイン k のコーパスを用意し，このドメインの言語モデル $G^k_{\mathbf{u}}$ の学習を行うことを考える．その際，$G^k_{\mathbf{u}}$ の基底分布として，同じドメインの $(n-1)$ グラム分布である $G^k_{\mathbf{u}'}$ と，グローバル言語モデルにおける n グラム分布である $G^0_{\mathbf{u}}$ との混合分布をおく．こうすることにより，適応ドメインの学習量の少なさを，$G^0_{\mathbf{u}}$ を利用して補うことができる．すなわち，$G^k_{\mathbf{u}}$ は次のように生成されたと考える．

$$G^k_{\mathbf{u}} \sim \mathrm{PYP}(a_{|\mathbf{u}|}, b_{|\mathbf{u}|}, \lambda^k_{\mathbf{u}} G^k_{\mathbf{u}'} + (1-\lambda^k_{\mathbf{u}}) G^0_{\mathbf{u}}) \qquad (3)$$

ここで基底分布の混合係数である $\lambda^k_{\mathbf{u}}$ は，一般に k と文脈 $\mathbf{u}$ によって決まる変数である．Wood らはこの変数に対し，同じ深さのノードに共通の値を用いている．すなわち $\lambda^k_{\mathbf{u}} = \nu\ (\mathbf{u} \in \{\mathbf{u} : |\mathbf{u}| = l\})$ となる．この様子を，図 1 にまとめた．

3. 提案モデル

3.1 DHPYTM

我々の目的は，文書の潜在的なトピックを推定し，そのトピックに応じた n グラム言語モデルを用いて予測を行うことである．また冒頭で述べた通り，スパースネスの緩和のために，文脈に応じて大域的な言語モデルと，トピック別の言語モデルとを切り替えることが重要である．これは，前節で述べた DHPYLM をトピックモデルの枠組みに組み込むことで可能となる．すなわち DHPYLM において手動で与えていたドメインを，トピックモデルにおけるトピックに置き換え，トピック毎の言語モデルを教師なしで学習する．さらに，文書のトピックに依らない大域的な言語モデルをトピック 0 として捉え，全てのトピックがこのトピック 0 を共有することで，スパースネスの緩和が行える．また未知の文書に対してそのトピック分布を推定することで，効率よく予測を行うことができる．

ここで 0 以外のトピックの n グラム単語分布は式 (3) に従って生成される．[19] ではこの式における基底分布の混合係数 $\lambda^k_{\mathbf{u}}$ に対し，各トピック毎に深さに応じた値を設定していた．しかしながら，DHPYTM ではトピック 0 は文書のトピックに依らない n グラムを捉えるためのものであり，さらに冒頭で述べたように，予測の際にトピック 0 の予測を重視するか，各トピックの予測を重視するかは，文脈に応じてある程度決定される．そのため我々は，$\lambda^k_{\mathbf{u}}$ に対し，各文脈 $\mathbf{u}$ 毎に，全トピックで共通の値を設定する．つまり，$\lambda^k_{\mathbf{u}} = \lambda_{\mathbf{u}}\ (k \in 1, \cdots, K)$ とおく．

しかしながら，全てのノードに独立の $\lambda_{\mathbf{u}}$ をおいてしまうと，スパースネスの問題により学習がうまくいかない．そのため我々は $\lambda_{\mathbf{u}}$ に対し，Suffix Tree 上で階層的な Beta 事前分布を設定した．これにより，あるノードにおける $\lambda_{\mathbf{u}}$ の値が，その子ノードに受け継がれることになる．これは HPYLM におけるある $(n-1)$ グラム分布と n グラム分布との関係と実質的に同等である．階層的な Beta 分布は，2 値の階層 Dirichlet Process と見なすことができ，次式で与えられる．

$$\lambda_{\mathbf{u}} \sim \mathrm{Beta}(\gamma_{|\mathbf{u}|}\lambda_{\mathbf{u}'}, \gamma_{|\mathbf{u}|}(1-\lambda_{\mathbf{u}'})) \qquad (4)$$

$$= \mathrm{DP}(\gamma_{|\mathbf{u}|}, \lambda_{\mathbf{u}'}). \qquad (5)$$

ここで $\gamma_{|\mathbf{u}|}$ は DP の集中度パラメータであり，我々はノードの深さ毎に値を設定した．我々はルートノードの λ_ϕ に対し，$\lambda_\phi \sim \mathrm{Beta}(a^\lambda, b^\lambda)$ とおき，それ以外のノードの $\lambda_{\mathbf{u}}$ は，式 (5) により生成されるとした．

トピック別の n グラム言語モデルが生成されたうえで，実際に観測される文書は，次のように生成されたと考える．まず，各文書 j のトピック分布 $\boldsymbol{\theta}_j$ が，Dirichlet 分布から生成される．LDA[2] では，トピックに応じたユニグラム分布

から単語が生成されるが，DHPYTM ではトピックに応じた n グラム分布から単語が生成される．つまり，文書 j の i 番目の単語 w_{ji} は，そのトピック z_{ji} が生成されたうえで，トピック z_{ji} の，直前の $(n-1)$ 語 $\mathbf{u}_{ji} = w_{j,i-n+1} \cdots w_{j,i-1}$ で決まる単語分布 $G^{z_{ji}}_{\mathbf{u}_{ji}}$ から生成される．以上をまとめると，DHPYTM の生成過程は以下のようになる．

DHPYTM

$\lambda_\phi \sim \mathrm{Beta}(a^\lambda, b^\lambda)$ [draw root λ]
For each $\mathbf{u} \in \Sigma^*$:
  $G^0_{\mathbf{u}} \sim \mathrm{PYP}(a_{|\mathbf{u}|}, b_{|\mathbf{u}|}, G^0_{\mathbf{u}'})$ [draw word distribution of topic 0]
  $\lambda_{\mathbf{u}} \sim \mathrm{DP}(\gamma_{|\mathbf{u}|}, \lambda_{\mathbf{u}'})$ [draw weights of local topics]
  For each topic $k \in \{1, \cdots, K\}$:
    $G^k_{\mathbf{u}} \sim \mathrm{PYP}(a_{|\mathbf{u}|}, b_{|\mathbf{u}|}, \lambda_{\mathbf{u}} G^k_{\mathbf{u}'} + (1-\lambda_{\mathbf{u}}) G^0_{\mathbf{u}})$ [draw word distribution of topic k]
For each document $j \in \{1, \cdots, D\}$:
  $\boldsymbol{\theta}_j \sim \mathrm{Dir}(\alpha)$ [draw topic distribution]
  For each word $i \in \{1, \cdots, N_j\}$:
    $z_{ji} \sim \boldsymbol{\theta}_j$ [draw topic]
    $w_{ji} \sim G^{z_{ji}}_{\mathbf{u}_{ji}}$ [draw word]

ここで，DHPYTM と他のモデルの関連についてまとめておく．$a^\lambda = 0$ とおいた場合，全ての文脈において $\lambda_{\mathbf{u}} = 1$ となる．この場合，このモデルは Mochihashi らの HPYLM と LDA との組み合わせである HPY-LDA[10] に一致する．また HPY-LDA は Wallach らの Bigram Topic Model(BTM)[15] の一般化となっており，本モデルはそれら全ての一般化となっている．

3.2 cHPYTM

DHPYTM を用いることにより，文脈に応じたトピック性の強さを $\lambda_{\mathbf{u}}$ を通して学習し，必要な箇所でのみ文書のトピックに応じた n グラムモデルを用いた予測を行うことが可能となる．しかし DHPYTM の 1 つの問題点として，$\lambda_{\mathbf{u}}$ の小さい文脈から生じた単語に対しても，$\{1, \cdots, K\}$ のうちいずれかのトピックを割り当ててしまうという点が挙げられる．$\lambda_{\mathbf{u}}$ の小さい文脈から出現する単語は多くが機能語であり，これらの単語に対してトピックを割り当ててしまうと，文書のトピック分布が真の分布とずれたものとして学習されてしまう可能性がある．そのため我々は，文脈に応じたトピック性の強さをモデル化しつつ，トピック 0 に関して DHPYTM とは異なる混合方法を与えるモデルを考え，比較を行うことにした．

cHPYTM の生成過程は，以下のようになる．まず，DHPYTM と同じように特別なグローバルトピックであるトピック 0 の言語モデルが生成される．また，同じように文脈によって階層的に決まる変数 $\lambda_{\mathbf{u}}$ をおく．cHPYTM では，この $\lambda_{\mathbf{u}}$ の使われ方が異なる．DHPYTM では，$\lambda_{\mathbf{u}}$ は，ある文脈における各トピックの予測において，その基底分布におけるトピック 0 の予測の混合比を決めるものである．一方 cHPYTM では，ある文脈において，次に出現する単語がトピック 0 に割り当てられることに対する事前確率として定義する．こうすると，DHPYTM とは異なり，単語によっては直接トピック 0 に割り当てられる．そして，トピック 0 に割り当てられなかった語が，文書のトピック分布に従って，トピック $k \in \{1, \cdots, K\}$ に割り当てられる．cHPYTM の生成過程をまとめると，以下のようになる．

cHPYTM

$\lambda_\phi \sim \mathrm{Beta}(a^\lambda, b^\lambda)$ [draw root λ]
For each $\mathbf{u} \in \Sigma^*$:
  $\lambda_{\mathbf{u}} \sim \mathrm{DP}(\gamma_{|\mathbf{u}|}, \lambda_{\mathbf{u}'})$ [draw probability of topic $k \neq 0$]
  For each topic $k \in \{0, \cdots, K\}$:
    $G^k_{\mathbf{u}} \sim \mathrm{PYP}(a_{|\mathbf{u}|}, b_{|\mathbf{u}|}, G^k_{\mathbf{u}'})$ [draw word distribution of topic k]
For each document $j \in \{1, \cdots, D\}$:
  $\boldsymbol{\theta}_j \sim \mathrm{Dir}(\alpha)$ [draw topic distribution]
  For each word $i \in \{1, \cdots, N_j\}$:
    $z_{ji} \sim (1-\lambda_{\mathbf{u}_{ji}})\delta(0) + \lambda_{\mathbf{u}_{ji}} \boldsymbol{\theta}_j$ [draw topic]
    $w_{ji} \sim G^{z_{ji}}_{\mathbf{u}_{ji}}$ [draw word]

DHPYTM と cHPYTM とは，一見すると非常によく似ている．両者の違いは，予測の際のトピック 0 の扱われ方であるとも考えられる．DHPYTM では，単語 w のトピックは $\{1, \cdots, K\}$ のいずれかが割り当てられるので，文書 d，文脈 $\mathbf{u}$ に続く単語 w の出現確率は，$p(w|\mathbf{u}, d) = \sum_{k=1}^K \theta_{dk} G^k_{\mathbf{u}}(w)$ で与えられる．トピック 0 の予測 $G^0_{\mathbf{u}}(w)$ はこの中には陽に出現せず，各トピックの予測確率 $G^k_{\mathbf{u}}(w)$ $(k \neq 0)$ の中で，バックオフ確率として用いられる．一方 cHPYTM では，同様の予測確率は $p(w|\mathbf{u}, d) = (1-\lambda_{\mathbf{u}}) G^0_{\mathbf{u}}(w) + \lambda_{\mathbf{u}} \sum_{k=1}^K \theta_{dk} G^k_{\mathbf{u}}(w)$ で与えられる．ここで明らかなように，$G^0_{\mathbf{u}}(w)$ は，文脈毎の混合係数 $\lambda_{\mathbf{u}}$ によって直接他のトピックの予測と混合が行われる．すなわち両者のモデルの違いは，予測の際のトピック 0 の混合を，バックオフ時に行うか，単語の直接の予測時に行うか，という点であると理解することができる．

3.3 可変長モデル

提案法は，いずれも Mochihashi らの Infinite Markov Model(IMM)[10] を組み込むことで，メモリ使用量を抑えたまま ∞ グラムモデルへの拡張が可能となる．モデルの詳細については省略するが，これは各ノード上に，そのノードを通過するか，停止するかを表す変数 $\eta_{\mathbf{u}}$ を設定することで可能となる．簡単のため，ここでは全てのトピックにおいて $\eta_{\mathbf{u}}$ は共通の値を用いた．IMM との組み合わせがうまく働けば，不必要な n グラムを保持せずに，高次 n グラムモデルと同等の性能を示すことが期待される．また，トピック別 n グラムのスパースネスを考えると，必要な箇所で n グラムのオーダーを下げる IMM は，我々のモデルと相性が良いと考えられる．これらは 6 章で検討を行う．また可変長モデルの副産物として，各トピック毎に固有の可変長 n グラムのフレーズを，教師なしで獲得することが可能となる．

4. 推論

これまで定義したモデルはいずれも n グラムの複雑な混合モデルとなっており，高度な Bayes 推定を必要とする．事後分布の推定のため，我々は Bayes n グラム言語モデルで標準的に用いられる，Gibbs サンプリングに基づく手法を開発した．これは，各ノード上に存在する単語分布 $G^k_{\mathbf{u}}$ を推定する代わりに，それを積分消去することで得られる Chinese Restaurant の内部状態を推定するものである．

以下では，まず HPYLM と Chinese Restaurant Process(CRP) の関係について述べた後，文書中の各単語について，そのトピック割り当てと CRP 上での割り当てを推定する Token-based Sampling について説明する．その後，推論を効率化するための Table-based Sampling について述べ，その効果の検証を行う．最後に，ハイパーパラメータの推定法についてまとめる．

4.1 言語モデルの CRP 表現

HPYLM と CRP の関係について簡単に述べ，我々が推定すべき変数についてまとめる．詳細は [12][13][10] などを参照されたい．

これまで我々は Suffix Tree 上の各ノードにおける単語分布 $G^k_{\mathbf{u}}$ を推定すべき変数として，説明を行ってきた．しかしながら単語分布は数万次元以上と非常に高次元であり，このような高次元分布の推定を行うことは，推定精度及びメモリ使用量の観点からみて，現実的でない．実際は，$G^k_{\mathbf{u}}$ を積分消去することで得られる，n グラムの単語分布 $p(w|k,\mathbf{u})$ を，訓練文書中に出現する単語のカウントを用いることで表現できる．このとき，$G^k_{\mathbf{u}}$ を積分消去することで得られるモデルの内部状態は，CRP の Seating Arrangement[12] と呼ばれる．具体的には，$G^k_{\mathbf{u}} \sim \mathrm{PYP}(a_{|\mathbf{u}|}, b_{|\mathbf{u}|}, G^k_{\mathbf{u}'})$ の $G^k_{\mathbf{u}}$ を積分消去することで得られる単語分布は，次のようになる．

$$p(w|k,\mathbf{u}) = \frac{c^k_{\mathbf{u}w\cdot} - a_{|\mathbf{u}|}t^k_{\mathbf{u}w}}{c^k_{\mathbf{u}\cdot\cdot} + b_{|\mathbf{u}|}} + \frac{at^k_{\mathbf{u}\cdot} + b_{|\mathbf{u}|}}{c^k_{\mathbf{u}\cdot\cdot} + b_{|\mathbf{u}|}} p(w|k,\mathbf{u}') \quad (6)$$

ここで $c^k_{\mathbf{u}wi}$ で，トピック k に対応する Suffix Tree T_k の文脈 $\mathbf{u}$ に対応するノードに存在するレストランの中で，単語 w に対応するテーブルのうち，i 番目のテーブルの周りに座っている客の総数を表す．$p(w|k,\mathbf{u}')$ は基底分布での w の出現確率であり，これも同じように再帰的に CRP の確率として計算される．ただし式 (6) は GPYP を用いない場合，すなわち cHPYTM での各トピックでの予測確率である．DHPYTM では，各トピックの基底分布は混合分布となるため，$p(w|k,\mathbf{u}')$ の代わりに，混合分布 $\lambda_{\mathbf{u}}p(w|k,\mathbf{u}') + (1-\lambda_{\mathbf{u}})p(w|0,\mathbf{u}')$ が使われることに注意する．また $\cdot$ は対応する変数の周辺化を表す．例えば $c^k_{\mathbf{u}w\cdot} = \sum_i c^k_{\mathbf{u}wi}$ は，$k,\mathbf{u}$ で定まるレストランにおける単語 w のテーブルの周りの客の総数である．

このように CRP を用いることにより，我々は $G^k_{\mathbf{u}}$ の推定を，$k,\mathbf{u}$ 毎に定まるレストラン内部の，テーブルと客の配置の推定に置き換えることができる．

4.2 λ に対する CRP 表現

我々は，各ノードの λ に対して階層的な Beta 分布を配置したが，この Beta 分布も積分消去することで，CRP の枠組みで推定を行うことができる．具体的には，あるノードにおける，λ に対するレストランに存在する客とテーブルの数を $c^\lambda_{\mathbf{u}l}, t^\lambda_{\mathbf{u}l}$ $(l \in \{0,1\})$ と表すと，$\lambda_{\mathbf{u}}$ の事後分布は次の式で与えられる．

$$p(l|\mathbf{u}) = \frac{c^\lambda_{\mathbf{u}l}}{c^\lambda_{\mathbf{u}\cdot} + \gamma_{|\mathbf{u}|}} + \frac{\gamma_{|\mathbf{u}|}}{c^\lambda_{\mathbf{u}\cdot} + \gamma_{|\mathbf{u}|}} p(l|\mathbf{u}') \quad (7)$$

4.3 Token-based Sampling

Token-based Sampling は，LDA などのモデルにおける Collapsed Gibbs Sampling と同じ手法と見なせる．以下ではまず DHPYTM の場合を説明し，続いて cHPYTM の場合について述べる．

4.3.1 DHPYTM の場合

DHPYTM の場合，具体的なサンプリングの流れは以下のようになる．まず，訓練データ中の全単語のトピック割り当てを，ランダムに初期化する．その後，割り当てられたトピックに応じて，各単語を言語モデル T_k に追加する．この際，通常の HPYLM と同じように T_k の文脈 $\mathbf{u}_{ji}$ に対応するノードに，単語 w_{ji} に対応する客を追加する．ここで，追加された客は式 (6) 右辺の第二項から発生した，とされた場合，親レストランにも再帰的に客を追加する．通常の HPYLM では，親レストランは一意に決まるので，そこに客を追加すれば良い．しかし，DHPYTM では基底分布が混合分布であるので，対応する親レストランはどちらか，まで含めてサンプリングを行う必要がある．これは追加されたテーブルに $\{0,1\}$ のラベルを割り振ることで行われる．ここでラベル 0 は，親レストランがトピック 0 であると判断された場合，ラベル 1 は親レストランが現在のトピック k の親ノードのレストランであると判断された場合を表す．テーブルが追加されたら，対応する親レストランに客を追加するだけでなく，そのノードの λ に対するレストランにも，ラベルに応じた $\{0,1\}$ の客を追加する．

このようにしてトピック割り当て及びレストランの状態を初期化したら，コーパス中の全単語をモデルから追加/削除を繰り返すことで，モデルの事後分布を得ることができる．我々は以下の分布に従って，単語 w_{ji} のトピック z_{ji} をサンプリングし，対応する T_k に追加する．

$$p(z_{ji}=k|\mathbf{z}^{-ji},\mathbf{u}_{ji},w_{ji}) \propto p(k|\mathbf{z}^{-ji},j,\mathbf{u}_{ji})p(w_{ji}|k,\mathbf{u}_{ji}) \tag{8}$$

ここで $p(k|\mathbf{z}^{-ji},j,\mathbf{u}_{ji}) = \frac{n_{jk}^{-ji}+\alpha}{N_j-1+K\alpha}$ であり，n_{jk}^{-ji} は，トピック割り当てから z_{ji} を除いた後の，文書 j 中のトピック k の単語の総数である．

4.4 cHPYTM の場合

cHPYTM の場合，基本的な流れは DHPYTM と同じであるが，λ に対するレストランに客の追加/削除を行うタイミングが異なる．cHPYTM では λ は，ある文脈においてトピック 0 へ割り当てられる確率を表すので，ある単語のトピックが 0 とサンプルされた場合，その文脈の λ に対するレストランに，0 に対応する客を追加する．逆に 0 以外のトピックがサンプルされた場合，λ に対するレストランに 1 に対応する客を追加する．また cHPYTM の各言語モデルの内部は GPYP ではないので，テーブルに対してラベルを割り振る必要はなく，テーブルが追加された際には客は一意に対応する親レストランに追加される．また式 (8) における $p(k|\mathbf{z}^{-ji},j,\mathbf{u}_{ji})$ が異なり，以下の式を用いる．

$$p(k|\mathbf{z}^{-ji},j,\mathbf{u}_{ji}) = \begin{cases} 1-\lambda_{\mathbf{u}_{ji}} & (k=0) \\ \lambda_{\mathbf{u}_{ji}}\frac{n_{jk}^{-ji}+\alpha}{N_j-1+K\alpha} & (k\neq 0) \end{cases} \tag{9}$$

4.5 Table-based Sampling

先ほど述べた Token-based Sampling は，通常の LDA の Collapsed Gibbs Sampling と同じように，文書中の 1 つの単語をモデルから削除し，それを追加するプロセスを繰り返すものである．実際，各トピックの言語モデルとしてユニグラムを用い，トピック 0 を考えない場合，これは通常の LDA に一致する．しかしながら，予備実験の結果，トピック内部のモデルが n グラムの場合，この方法では局所解に陥りやすいことが明らかになった．内部のレストランの状態の変化を考えると，あるノードのレストランに客が追加された際，その親ノードの客のカウントが変化するのは，追加された客が新しいテーブルに座ったとされた場合のみである．したがって，例えば 3 グラムモデルであれば，ユニグラム，バイグラムに対応するレストランの客は非常に動きづらく，十分な混合が起きないまま，局所解に陥ってしまう．

これを回避するために，我々は n グラムモデル内部の複数のカウントを同時に動かす，Table-based Sampling を行う．なお，以下ではまず固定長の場合について述べた後，可変長モデルにおいてどう拡張を行うか，について補足する．

まず，1 つの言語モデルの内部に着目する．トピック k の言語モデルの持つ，全ノードの Seating Arrangement をまとめて S^k と表す．このとき，S^k の従う分布は，次のようになる [12]．

$$p(S^k) = \prod_w G_\phi^k(w)^{c^k_{\phi w\cdot}} \prod_{\mathbf{u}} \frac{[b_{|\mathbf{u}|}+a_{|\mathbf{u}|}]^{(t^k_{\mathbf{u}\cdot})}_{a_{|\mathbf{u}|}}}{[b_{|\mathbf{u}|}+1]^{(c^k_{\mathbf{u}\cdot\cdot}-1)}_1} \prod_w \prod_{i=1}^{t^k_{\mathbf{u}\cdot}} [1-a_{|\mathbf{u}|}]^{(c^k_{\mathbf{u}wi}-1)}_1 \tag{10}$$

ここで $[y]_d^{(n)} = \prod_{i=0}^{n-1}(y+id)$ である．この値は，客とテーブルの追加された順番に関して不変，すなわち交換可能であり，これを元に 1 つの客の追加と削除を繰り返すのが Token-based Sampling である．ここで見方を変えて，言語モデル内部の 1 つのテーブルに注目する．HPYLM において，あるノードに存在する 1 つのテーブルは，親ノードの 1 つの客に対応している．逆に言うと，あるノードの 1 つのテーブルの周りに存在する客は全て，その子ノードのあるテーブルと結びついている．ここで，次の手順を再帰的に繰り返すことにより，1 つのテーブルと結びついた，子ノードのテーブル及び客の集合を得ることができる．

(1) トピック k，文脈 $\mathbf{u}$ で定まるレストランの中で，単語 w に対応するテーブルを 1 つ選択する．
(2) 選択したテーブルの周りに存在する客を $c^k_{\mathbf{u}wm}(m \in \{1,\cdots,t^k_{\mathbf{u}w}\})$ とする．
(3) $\mathbf{u}$ の子ノードの集合を $ch(\mathbf{u})$ とする．$ch(\mathbf{u})=\emptyset$，つまり $\mathbf{u}$ が葉ノードの場合，呼び出し元に戻る．$\mathbf{v}\in ch(\mathbf{u})$ の中に存在する，w に対応するテーブルの中から，ランダムに $c^k_{\mathbf{u}wm}$ 個のテーブルを選択する．
(4) 選択したそれぞれのテーブルに対して，再帰的に (2) に戻り，子ノードの客とテーブルの集合を記録する．

以上の手順により得られた客とテーブルの集合を $S^k_{\mathbf{u}wm}$ とおく．式 (10) は，全ての客とテーブルが，追加される順番に不変なことを表しているので，例えば $S^k_{\mathbf{u}wm}$ に対応する客とテーブルの集合（ブロック）が最後に追加された，と考えても問題ないことがわかる．

また，各言語モデルで葉ノードのレストランに存在する客は，それぞれコーパス上の観測と結びついている．ここでコーパス中の全文書のトピック割り当て $\mathbf{z}\in\{1,\cdots,K\}^{\sum_j N_j}$ に対する同時分布を書き下すと，

$$\begin{aligned} p(\mathbf{z}) &= \prod_j p(\mathbf{z}_j) \\ &= \prod_j \frac{\Gamma(\sum_k \alpha_k)}{\Gamma(N_j+\sum_k \alpha_k)} \prod_k \frac{\Gamma(\alpha_k+n_{jk})}{\Gamma(\alpha_k)} \end{aligned} \tag{11}$$

この同時分布も全ての観測 z_{ji} に関して交換可能である．以上の洞察により，以下の Table-based Sampling を構成できる．

まず，あるトピック k の言語モデル中に存在する，1 つのテーブルを抜き出す．そのテーブルは，親ノードの客と結びついているので，親ノードの客を 1 人削除する．そこから上記手順により，選択したテーブルと紐づく全ての客

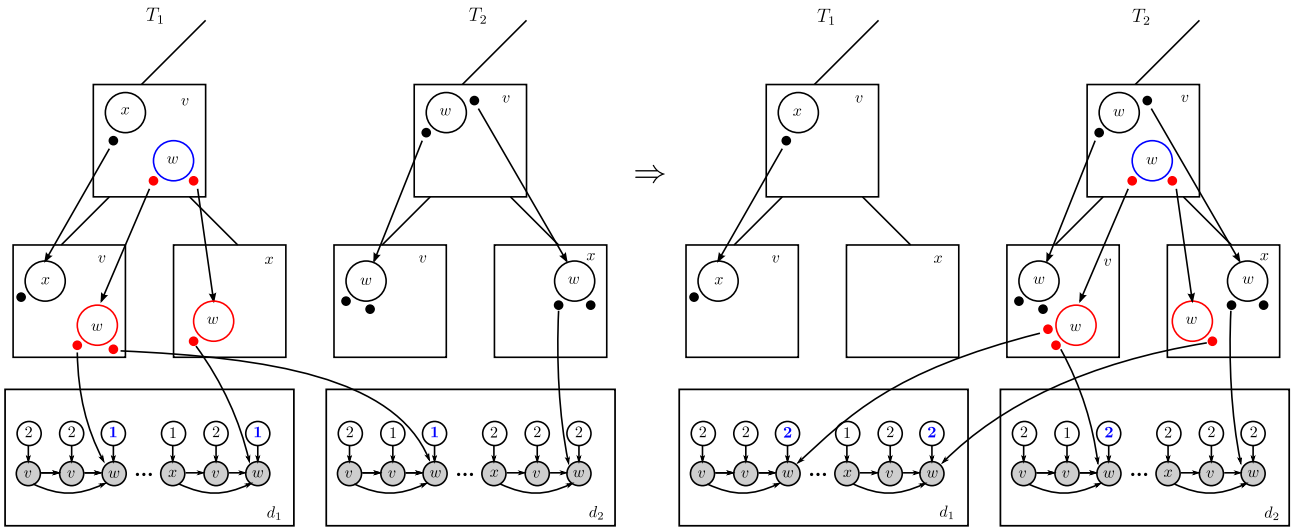

図 3　Table-based Sampling での状態遷移の様子．$\{v, w, x\}$ が単語，$\{1, 2\}$ がトピックの集合である．左がある時点 t，右が時点 $t+1$ での状態を表す．それぞれ上部が Suffix Tree の一部を表し，下部が葉ノードに存在する客とコーパスとの結びつきを表す．各ノードは 1 つのレストランに対応している．青色のテーブルが選択したテーブルであり，その周りの客と対応する子ノードのテーブル及び客を，赤色で示した．

とテーブルの集合 $S^k_{\mathbf{u}wm}$ をランダムに再構成する．$S^k_{\mathbf{u}wm}$ に対応するコーパス中の単語集合に対する，トピック割り当ての集合を $\mathbf{z}^k_{\mathbf{u}wm}$ とおく．我々は以下の分布に従い，$\mathbf{z}^k_{\mathbf{u}wm}$ の値をサンプリングする．

$$p(\mathbf{z}^k_{\mathbf{u}wm} = k' | \mathbf{z}^{-k\mathbf{u}wm}, S^k_{\mathbf{u}wm}, S^{-k\mathbf{u}wm}) \propto p(\mathbf{z}^k_{\mathbf{u}wm} = k' | \mathbf{z}^{-k\mathbf{u}wm}) p(S^k_{\mathbf{u}wm} | k', S^{-k\mathbf{u}wm}) \quad (12)$$

ここで $S^{-k\mathbf{u}wm}$ はモデル全体の Seating Arrangement から $S^k_{\mathbf{u}wm}$ を除いたものを表す．$p(\mathbf{z}^k_{\mathbf{u}wm} = k' | \mathbf{z}^{-k\mathbf{u}wm})$ は，式 (11) から対応する項を取り出したものであり，$p(S^k_{\mathbf{u}wm} | k', S^{-k\mathbf{u}wm})$ も同様に，式 (10) から項を取り出すことで得られる．$S^k_{\mathbf{u}wm}$ に存在するテーブルと客を全てサンプルされたトピックに移動し，親ノードに客を追加する．Table-based Sampling の状態遷移の様子を，図 3 に示す．

4.5.1　cHPYTM の場合

cHPYTM の場合，基本的に DHPYTM と同じだが，若干の注意を要する．DHPYTM では，選択した内部テーブルのラベルは変化しないので，$\lambda_{\mathbf{u}}$ の値はサンプリングの際にキャンセルされ，考える必要がない．一方 cHPYTM の場合，$\lambda_{\mathbf{u}}$ の値はトピック 0 への割り当て確率である．cHPYTM のトピック割り当ての事後分布は式 (9) で与えられるが，ブロックに対するこの式の積を考える場合，$\lambda_{\mathbf{u}}$ の値がそれまでに追加された客の Seating Arrangement に依存して変化してしまうため，$\lambda_{\mathbf{u}}$ を周辺化した確率を閉じた形で求めることができない．これは [3] での type-level sampler と同じ問題である．我々は [3] と同じく，サンプル中に $\lambda_{\mathbf{u}}$ のレストランに，仮想的な客とテーブルを fractional に追加する，という方法で対処した．別の方法として $\lambda_{\mathbf{u}}$ の値を事前に Beta 分布からサンプリングしておく方法が考えられるが，収束が遅くなる可能性があり，我々はこの方法を用いた．

4.5.2　可変長モデル

可変長モデルの場合，テーブルに紐づく Seating Arrangement を抜き出す際，注意を要する．固定長モデルでは，レストラン内の客とコーパス中の単語との結びつきは，葉ノード以外ではありえなかったが，可変長モデルではあらゆるノードのレストランの客とコーパス中の単語が結びつく可能性がある．そのため，内部ノードのテーブルを選択する際，その周りの客がコーパス中から直接追加されたものなのか，子ノードから追加されたものなのかも含めて，選択を行う必要がある．

4.5.3　性能比較

Table-based Sampling の効果を確かめるために，モデル内部の対数尤度を観察し，比較を行う．図 4 に，Brown コーパスを用いた際の DHPYTM，cHPYTM それぞれでの，Token-based のみを行った場合と，Token-based と Table-based を組み合わせた場合のサンプリング回数に対する対数尤度のプロットを示す．どちらのモデルも Table-based を組み合せることで，Token-based のみでは得られない高い尤度が得られていることがわかる．

4.6　ハイパーパラメータの推定

我々は，できる限り予測性能の高い言語モデルを得ることが目標であるので，モデル内部のハイパーパラメータについても，できる限りデータから自動で学習を行う．[12] と同じように，PYP のパラメタ a_i, b_i $(i \in \{0, 1, \cdots\})$ については，それぞれ事前分布 $a_i \sim \mathrm{Beta}(1, 1)$，$b_i \sim \mathrm{Gamma}(1, 1)$

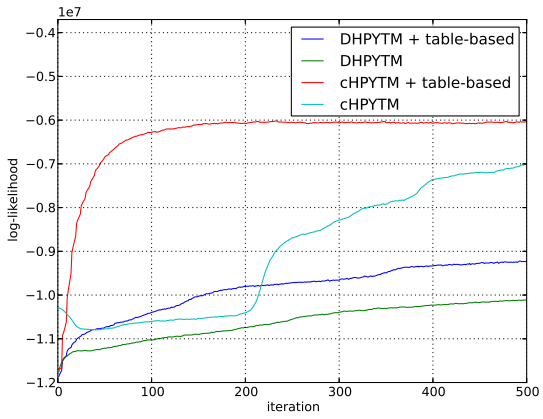

図 4　Brown コーパスでの，サンプル数に対する対数尤度の比較．トピック数 50，$n = 3$ とした．

をおき，事後分布をサンプルした．また γ_i $(i \in \{0, 1, \cdots\})$ 及びトピック分布の α に対しては Gamma(1, 1) の事前分布をおいた．これらは [14] にあるサンプリング法を用いることで，事後分布からのサンプルを得ることができる．我々は 1 回の Token-based Sampling 毎に，これらの変数をサンプリングした．

5. 関連研究

本来 Bag-of-Words を仮定したトピックモデルに，n グラムなどの構造を組み合わせる研究は数多く存在する．HMM-LDA[5] は，HMM の状態のうち，1 つの状態が文書のトピックを反映した内用語を出力する，というモデルである．こうすることにより，文章内の機能語，文法構造を HMM で捉えつつ，各文書に固有の内容語のみに，文書のトピックを反映させることができる．HMM-LDA は我々のモデルと非常に関連が深く，特に cHPYTM は，HMM-LDA における隠れ状態を語彙化したものと見なすことができる．これはもともと言語モデルの予測性能を向上させることを目指したモデルではないが，[6] では，音声認識の学習コーパスに HMM-LDA を用いることにより機能語と内容語を分離し，ドメイン適応を行っている．

またトピック別の n グラムを構築するものとして，Bigram Topic Model(BTM)[15] や，トピックに固有のフレーズを獲得するモデル [8][18] が存在する．2 節で述べたように，DHPYTM は BTM の大幅な拡張であり，一般化となっている．[11] では BTM で得られたモデルと，Unigram Rescaling で補間した n グラムモデルとを，更に補間し，言語モデルの性能を向上させている．[20] は文章単位でトピックの混合比を変化させる n グラムモデルであるが，我々のように共通のグローバルトピックを捉えることはしておらず，また 3 グラムで性能の改善がなかったと報告している．

Unigram Rescaling(Rescaling)[4][7] は，n グラムモデルとトピックモデルとの積モデルである．Rescaling では，n グラムモデルとトピックモデルとを別々に学習し，テストデータでは次の式を用いて予測を行う．

$$p(w|\mathbf{u}, d) \propto p(w|\mathbf{u}) \left(\frac{p(w|d)}{p(w)} \right)^{\beta} \qquad (13)$$

ここで $p(w|\mathbf{u})$ は任意の n グラム言語モデルに基づく予測，$p(w|d) = \sum_k p(k|d)p(w|k)$ はトピックモデルに基づく予測，$p(w)$ は訓練データのユニグラムに基づく予測である．このモデルは積モデルであるため，具体的な単語の予測確率を得るためには，式 (13) を全ての語彙について計算し正規化を行う必要があることに注意する．トピック分布 $p(k|d)$ は履歴から動的に推定しなければならないことを考えると，この正規化係数は事前に計算しておくことはできず，そのコストは非常に大きい．

6. 実験

6.1 データ

我々はモデルの性質を調べるためのコーパスとして，Brown 及び BNC，NIPS コーパスを用いた．Brown と BNC は，新聞や小説など，様々なドメインが含まれた均衡コーパスである．我々は様々なドメインに自動適応する言語モデルを設計することが目標なので，新聞記事等より，このようなコーパスでの性能比較が重要である．NIPS コーパスは NIPS の論文を集めたもので，トピックモデルの評価において一般的に用いられる．

Brown コーパスは 15 のカテゴリに分けられた 500 文書からなる．我々はこのうち，各カテゴリから 2 文書ずつ選び，30 文書をテストデータ，残り 470 文書を訓練データとした．訓練データは 11,572,255 語，テストデータは 70,795 語からなる．また頻度 2 以下の単語は特別な未知語と見なし，総語彙数は 19,759 となった．NIPS コーパスは 1,740 文書からなるが，我々はランダムに選んだ 1500 文書を訓練データ，50 文書をテストデータとした．[5] と同様の前処理を行い，訓練データ 5,088,786 語，テストデータ 167,370 語，総語彙数 22,705 を得た．BNC は，書き言葉コーパスからランダムに 400 文書を選択し，さらにそれを 100 文毎に区切り，一文書とした．訓練データとして 6,162 文書，テストデータとして 100 文書をランダムに選択し，頻度 10 以下の単語を特別な未知語とみなした．訓練データは 12,783,130 語，テストデータは 202,994 語からなり，総語彙数は 33,071 となった．

6.2 モデルと予測方法

上記で得た文書集合に対し，様々なモデルを適応し，その性質を調べる．我々は比較のためのモデルとして，トピックを用いない HPYLM[13], VPYLM[10], 及び Rescaling[4] を用いた．HPYLM，VPYLM はそれぞれ 200 回，Rescaling

モデル	K	n	PPL
HPYLM	1	3	242.7
VPYLM	1	∞	243.2
Rescaling	100	3	213.7
HPY-LDA	50	3	223.2
DHPYTM	50	3	214.1
	100	∞	216.2
DHPYTM*	100	∞	214.4
	200	∞	216.9
cHPYTM	50	3	207.3
	100	∞	209.9
cHPYTM*	100	∞	206.3
	200	∞	**205.3**

表 1　Brown コーパスでのパープレキシティの比較．*は階層的なトピック分布をおいたもの．

は 2000 回の Gibbs サンプリングを行い，最後の 1 サンプルを最終的なモデルとした．cHPYTM，VPYTM は合計 500 回の Token-based Sampling を行い，そのうちの 5 回に 1 回，Table-based Sampling を行った．また Table-based は，各トピックのルートノードは除外し，深さ 2 以上のノードに対してのみ実行した．これは，ルートノードのテーブルは非常に大きなブロックを作りやすく，サンプリングに非常に時間がかかるためである．同様に，最後の 1 サンプルを最終的なモデルとした．

テストデータの予測は，文書のトピック分布を動的に推定しながら行う．これには Particle Filter の簡易版である，left-to-right 法 [16] を用いた．また Unigram Rescaling の予測式 (13) における β の値は予備実験の結果 0.7 に設定した．提案法では自ら設定する変数は，トピック数以外に存在しない．可変長モデルでの各ノードの Beta 事前分布は，$\mathrm{Beta}(1,1)$ をおいた．

以下の性能比較において，HPY-LDA はトピック 0 を考えずに，全トピックを独立においたモデル [10] である．また DHPYTM と cHPYTM について，*がついたものは，各文書のトピック事前分布 $\mathrm{Dir}(\alpha)$ に階層的な事前分布を設定したものである．具体的には $\boldsymbol{\theta}_j \sim \mathrm{Dir}(\alpha\boldsymbol{\beta})(j \in \{1,\cdots,D\}), \boldsymbol{\beta} \sim \mathrm{Dir}(\alpha_0)$ とおき，$\alpha \sim \mathrm{Gamma}(a^\alpha, b^\alpha)$ と $\boldsymbol{\beta}$ の事後分布を推定する．これは HDP に基づく無限次元トピックモデル [14] の有限モデル版と見なすことができる．ユニグラム LDA の場合，こうすることで不必要なトピックが縮退し，トピック数を大きくしても性能が低下しないことが示されている [17]．我々は n グラムモデルでもこの性質が成り立つかどうかを確かめる．

6.3 Brown での性能比較

Brown コーパスは非常に少量であるが，予備実験の結果，このコーパスでのモデルの優劣は他のコーパスでもほぼそのまま成り立つことを確認した．表 6.3 に，様々なモデルと設定での，テストセットパープレキシティの結果をまとめる．ここで K はトピック数，n は n グラムのオーダーを表す．表 6.3 を見ると，特に cHPYTM で性能が向上していることがわかる．また，DHPYTM と cHPYTM とも，単純なトピック n グラムモデルである HPY-LDA と比較して性能を大きく向上させている．Rescaling との比較では，DHPYTM はほぼ同等の性能を示し，cHPYTM は大きく上回る性能を示した．トピック数を大きくした場合 ($K = 100$)，いずれのモデルでも，トピック分布を階層化することで，性能を向上させている．これより，上記で述べた不必要なトピックが自動で縮退する効果が得られていることがわかる．$K = 200$ の階層モデルの場合，cHPYTM では更に性能を向上させるが，DHPYTM は性能を低下させている．これは学習が不十分であった可能性がある．DHPYTM の Table-based Sampling では，トピック 0 に存在するテーブルは直接他のトピックに移動させることができないため，トピック数が増えてモデルが複雑になるほど，学習が難しくなるからである．以降の実験では，提案手法については $K = 100$ の階層モデルに絞って，実験結果を示す．

モデル	K	n	NIPS (time)	BNC (time)
HPYLM	1	3	121.4 (6)	180.2 (10)
VPYLM	1	∞	118.3 (18)	173.8 (20)
Rescaling	100	3	104.9 (15060)	138.6 (28774)
	100	4	101.8 (14219)	130.9 (26492)
DHPYTM*	100	3	108.6 (593)	135.9 (475)
	100	∞	107.3 (1204)	132.1 (902)
cHPYTM*	100	3	103.7 (644)	132.1 (477)
	100	∞	**100.0** (1213)	**127.5** (941)

表 2　NIPS,BNC でのパープレキシティと予測時間の比較．予測時間の単位は秒である．*は階層的なトピック分布をおいたもの．

6.4 NIPS,BNC での性能比較

表 6.4 に NIPS 及び BNC の結果をまとめた．どちらのコーパスでも，各モデルの相対的な性能はほぼ同じであることがわかる．ここではパープレキシティに加え，テストデータの予測に必要な時間も合わせて示す．cHPYTM は Rescaling と比べて良い性能が得られているが，その差は有意であるとはいえない．しかしながら提案法は Rescaling と違い，予測の際に正規化が必要ないため，予測時間の大幅な改善が行えていることがわかる．この差は語彙数が大きいほど顕著であり，BNC では 3 グラムで約 60 倍の高速化を達成している．

また，cHPYTM で学習された，訓練データへのトピック 0 への割り当ての一部を図 5 に示す．おおまかに，機能語と内容語が分離されている様子が見てとれる．最後

0	46	83	89	91
according to	BOS support vectors	the hierarchical mixtures	BOS linear discriminant	BOS adaptive network
BOS (#)	in high dimentional	the rbf units	images per class	to test patterns
BOS section #	as decision function	the gating networks	multi-class classification	BOS neural network
techniques such as	set of # observations	grown hme	BOS decision boundaries	local feature detector
BOS (b)	original data set	the modular architecture	references per class	BOS test patterns

表 3 NIPS から抽出された可変長フレーズの一部．モデルは cHPYTM*の $K = 100$ である．各トピックに対し，$p(w, n|\mathbf{u}, k) = p(n|\mathbf{u})p(w|k, n, \mathbf{u})$ の高い順に並べ，一部を取り出した．$p(n|\mathbf{u})$ は IMM により得られる n グラムオーダーに対する確率である．

mr lawson . who has never found it easy to win the affection of the party conference . now faces the __unk__ political test of his career in attempting to avert __unk__ attacks on his competence from the conservative grassroots . tory mps were last night angry about the rise in interest rates and expressed fears that a further rise might be necessary . but ministerial aides said it was better to get the increase out of the way this week . rather than have it happen during the conference .

図 5 cHPYTM における，BNC コーパスでのトピック割り当ての様子．灰色の語がトピック 0 に割り当てられた単語を表す．文書のトピック分布は，黒色の単語に割り当てられたトピックのみで学習される．

に，NIPS で cHPYTM の可変長モデルによって得られたトピック毎のフレーズを表 3 に示す．

7. まとめ

文書のトピックを考慮する Bayes 的 n グラム言語モデルとして，2 種類のモデルを提案し，比較を行った．提案法はいずれも，n グラムのスパースネスを緩和するために文書のトピック分布に依存しないグローバルなトピックを用意したモデルである．DHPYTM では，このグローバルトピックが各トピック分布のバックオフモデルとして使われるのに対して，cHPYTM では他のトピックと直接の混合を行う．いずれも可変長モデルと階層的なトピック分布を組み合わせることにより，文脈に応じた適切な n グラム長とトピック数を自動で学習可能であることを示した．2 つのモデルを Unigram Rescaling と比較し，予測精度および予測時間の比較において，性能の改善を確認した．

DHPYTM と cHPYTM との比較では，cHPYTM のほうが常に良い性能を示した．cHPYTM では限られた n グラムを全てトピック毎に分割してしまうのに対して，DHPYTM では GPYP によって全トピックでいくつかのカウントを共有することができる．DHPYTM のほうが予測性能が悪化した原因として，文書中の全ての単語にトピックを割り当ててしまうことが考えられる．DHPYTM では，cHPYTM でトピック 0 に割り当てられるような機能語などは，すぐにトピック 0 にバックオフされ，全トピックでカウントが共有される．しかしながら，この際に必ず 0 以外のトピックに割り当てられてしまい，それによって文書のトピック分布が，真の分布とずれて学習されてしまう可能性がある．

GPYP によるトピック間でのカウントの共有は，スパースな n グラムトピックモデルにとって非常に重要である．今回我々は，全てのトピックがただ 1 つのトピックとカウントを共有するモデルを考えたが，トピック自体が階層構造を持っていることを考慮すると，階層的なトピック分布と GPYP との組み合わせを考えることができる．これは Dirichlet Tree 分布 [9] や，その無限モデルである [1] などの階層構造をもつ事前分布を用いることで可能になると考えられ，現在検討を行っている．

参考文献

[1] R.P. Adams, Z. Ghahramani, and M. I. Jordan. Tree-structured stick breaking for hierarchical data. *Advances in Neural Information Processing Systems*, 2010.
[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 2003.
[3] P. Blunsom and T. Cohn. A hierarchical Pitman-Yor process HMM for unsupervised part of speech induction. *Proceedings of the Association for Computational Linguistics*, 2011.
[4] D. Gildea and T. Hofmann. Topic-based language models using EM. *Proceedings of EUROSPEECH*, 1999.
[5] T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. Integrating topics and syntax. *Advances in Neural Information Processing Systems*, 2005.
[6] B. P. Hsu and J. Glass. Style & topic language model adaptation using HMM-LDA. In *Proceedings of the Empirical Methods in Natural Language Processing*, 2006.
[7] S. Huang and S. Renals. Unsupervised language model adaptation based on topic and role information in multiparty meetings. *Proceedings of Interspeech*, 2008.
[8] R. V. Lindsey, W. P. Headden III, and M. J. Stipicevic. A Phrase-Discovering Topic Model Using Hierarchical Pitman-Yor Processes. *Proceedings of the Association for Computational Linguistics*, 2012.
[9] T. Minka. The dirichlet-tree distribution. 1999.
[10] D. Mochihashi and E. Sumita. The infinite Markov model. *Advances in Neural Information Processing Systems*, 2007.
[11] Y.C. Tam and T. Schultz. Correlated bigram LSA for unsupervised language model adaptation. *Advances in Neural Information Processing Systems*, 2008.
[12] Y. W. Teh. *A Bayesian Interpretation of Interpolated Kneser-Ney*. NUS School of Computing Technical Report TRA2/06, 2006.
[13] Y. W. Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. *Proceedings of the Association for Computational Linguistics*, 2006.
[14] Y. W. Teh, M. I. Jordan, M.J. Beal, and D. M. Blei. Hi-

erarchical dirichlet processes. *Journal of the American Statistical Association*, 2006.

[15] H. M. Wallach. Topic modeling: beyond bag-of-words. *Proceedings of the International Conference on Machine learning*, 2006.

[16] H. M. Wallach. *Structured topic models for language*. Ph.D. thesis, University of Cambridge, 2008.

[17] H. M. Wallach, D. Mimno, and A. McCallum. Rethinking LDA: Why priors matter. *Advances in Neural Information Processing Systems*, 2009.

[18] X. Wang, A. McCallum, and X. Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. *Proceedings of the International Conference on Data Mining*, 2007.

[19] F. Wood and Y. W. Teh. A hierarchical nonparametric Bayesian approach to statistical language model domain adaptation. *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2009.

[20] 三品 拓也, 山本 幹雄. 文脈適応による複数 n-gram の動的補間を用いた言語モデル. 情報処理学会研究報告 *2003-NL-155*, 2003.