Vol.2013-SLP-97 No.5
2013/7/26

生成型アプローチによる Latent Words Language Modelの N-gram近似

増村 亮[1,a)]　政瀧 浩和[1,b)]　大庭 隆伸[1,c)]　吉岡 理[1,d)]　高橋 敏[1,e)]

概要：今日の大語彙連続音声認識において，デコーディングとの相性から，N-gram モデルが最も実用的な言語モデルとして利用されている．N-gram モデルは，膨大なモデルパラメータに起因するデータスパースネスの問題を持つことが知られており，この問題を解決するために，スムージングや次元削減に基づく様々なアプローチが検討されてきた．これに対して我々は，学習データ自体を新たに生成し，生成したデータに基づき N-gram モデルを推定する生成型アプローチを検討する．本稿では，高性能な生成モデルである Latent Words Language Model(LWLM) をデータ生成のためのモデルとして利用する．LWLM 自体を音声認識に直接利用することは，計算コストの問題から非現実的ではある．しかし，確率過程に基づきデータ生成を行うことは比較的容易である．ベイズ推定に基づく LWLM は，スムージングと次元削減を同時に考慮した非常に柔軟なモデル構造を持つ．よって，生成されたデータから推定した N-gram モデルは，学習データから直接推定した N-gram モデルとは異なる性質を持ち，様々なタスクで頑健に動作することが期待できる．そこで本稿では，複数タスクで様々な言語モデルとの比較実験を行うことで，LWLM を用いた生成型アプローチの有効性を検証する．

キーワード：言語モデル，N-gram モデル，生成型アプローチ，Latent Words Language Model

1. はじめに

今日の大語彙連続音声認識で最も利用されている言語モデルは，N-gram モデルである．N-gram モデルは，シンプルかつコンパクトな構造を持ち，重みつき有限状態トランスデューサ (WFST) としての表現も容易であり，音声認識のデコーディングとの相性が非常に良い．しかしながら N-gram モデルは，膨大なモデルパラメータを持ち，モデル学習におけるデータスパースネスの問題が避けられない [1].

データスパースネスを回避するための方法として最も基本的なアプローチは，スムージングである [2]．これまで様々なスムージング手法が検討されており，Witten-Bell スムージングや Kneser-Ney スムージングは，代表的なスムージングとして広く利用されている [3]．近年では，スムージングのメカニズムに関するベイズ的な解釈も明らかになり，階層 Pitman-Yor 言語モデルは，従来よりも強力なスムージング付き N-gram モデルとして注目を集めている [4]．実際に，階層 Pitman-Yor 言語モデルは，音声認識の枠組みでも有効性が実証されている [5].

同様に，データスパースネスを避けるために検討されているのが次元削減に基づく方法である．クラス N-gram モデルでは，単語系列をクラス系列として表現し，そのクラス系列から N-gram 言語モデルを構成することでデータスパースネスを回避する [6]．また，決定木言語モデル，Random Forest 言語モデルでは，文脈情報をクラスタリングすることでデータスパースネスを緩和させている [7], [8]. さらに，Neural Network に基づく言語モデルでは，非線形関数を用いて単語空間の次元削減を行うことによりデータスパースネスを緩和させている [9], [10].

これらに対して我々は，学習データ自体を増やすことでデータスパースネスの問題を回避することを試みる．具体的には，限られた学習データを元に，学習データ自体を新たに生成し，生成したデータに基づき N-gram モデルを推定する生成型アプローチを検討する．生成型アプローチの利点としては，音声認識に利用する言語モデルのモデル構造を複雑にすることなく，性能改善が期待できる．また，データ生成には，様々な観点を取り込んだ複雑なモデルを

[1] NTT メディアインテリジェンス研究所
NTT Media Intelligence Laboratories, 1-1 Hikarinooka Yokosuka-shi, Kanagawa 239-0847, Japan
[a)] masumura.ryo@lab.ntt.co.jp
[b)] masataki.hirokazu@lab.ntt.co.jp
[c)] oba.takanobu@lab.ntt.co.jp
[d)] yoshioka.osamu@lab.ntt.co.jp
[e)] takahashi.satoshi@lab.ntt.co.jp

適用することも可能である．

我々は，生成型アプローチにおけるデータ生成を行うためのモデルとして，近年機械学習の分野で提案された高性能な生成モデルである Latent Words Language Model(LWLM) に着目する [11]．LWLM は，潜在語と呼ばれる潜在変数を持つ言語モデルである．潜在語はクラス N-gram モデルにおけるクラスに相当するものと考えることができ，ある文脈において意味や構文的な役割が似た単語をグループ化した場合の代表語を表す．通常のクラス N-gram モデルと異なる点は，ソフトクラスタリングの構造を持つ点である．LWLM は，Bayesian Hidden Markov Model と同様に，全ての単語が全ての潜在語に属する構造を持つ [12], [13]．それに加え，潜在語自体の種類数にも特徴があり，語彙サイズと同数の潜在語を持つ．

LWLM はこのような柔軟な構造により，スムージングや次元削減を効率的に実現することができるが，それと引きかえに，LWLM 自体を音声認識に直接利用することは計算コストの問題から非現実的である．一方，生成モデルという特徴を活かして，確率過程に基づきデータ生成を行うことは容易である．LWLM を利用して生成したデータは，LWLM 自体の柔軟なモデル構造が反映されており，元の学習データには含まれない多様な言語表現を含んでいるのではないかと我々は考えている．つまり，生成データから推定した N-gram モデルは，学習データから直接推定した N-gram モデルとは異なる性質を持ち，また様々なタスクで頑健に動作することが期待できる．

そこで本稿では，LWLM を用いた生成型アプローチの有効性について検証実験を行う．具体的には，日本語話し言葉コーパスを用いて様々な言語モデルを構築し，同一タスクのテストデータ，またコンタクトセンタタスク，ボイスメールタスクといった学習データとは異なるテストデータを準備して，性能評価を行う．その際，他のモデルと組み合わせた場合の有効性についても調査する．また，生成型アプローチについて，生成したデータ量と性能の関係や，Entropy Pruning[14] の適用によるモデル縮退時の性能についても調査する．

2. 関連研究

本研究は，生成型アプローチに基づき N-gram モデルを構築するという観点の他に，N-gram モデル以外の言語モデルを N-gram モデルの構造に近似するという観点も持つ．

単純な場合としては，クラス N-gram モデルを単語 N-gram モデルの形で近似することが行われているが [15]，基本的に複雑な構造を持つ言語モデルに対して検討されることが多い．N-gram モデル以外の言語モデルは，計算コストの問題から音声認識の 1 パス目で利用することが難しく，N-best や Confusion Network を利用したリスコアリングにより適用されることが多い [16]．しかしながら，リスコアリングを用いるマルチパスデコーディングは，1 パス目の性能に大きく影響を受けてしまうことが知られている．そこで，複雑な構造を持つ言語モデルを 1 パス目で利用できる形に表現することにより，1 パス目の高精度化を図る研究がなされている．

近年言語モデルとして注目される Neural Network 言語モデルは，基本的にリスコアリングに基づきマルチパスで適用されるモデルであるが，N-gram モデルに近似するアプローチが検討されている [17]．近似しない場合と比較すれば，N-gram モデルに近似することにより性能は劣化するが，1 パス目の性能改善につながり，リスコアリングによるマルチパスデコーディングを行う場合の性能の改善にもつながることが報告されている．同様に，Recurrent Neural Network 言語モデルについても，N-gram モデルに近似するアプローチが検討されている [18], [19]．

また，音声認識のデコーディング時に利用するモデル表現である WFST として近似する方法も提案されており，Recurrent Neural Network 言語モデルを WFST として表現する検討も行われている [20]．

本研究は，LWLM を N-gram モデルの構造に近似するという位置づけであり，本稿では Recurrent Neural Network 言語モデルを N-gram モデルに近似するアプローチについても比較を行う．

3. Latent Words Language Model

3.1 モデル定義

LWLM は機械学習の分野において近年提案されたモデルあり，テキスト中の各単語に対して隠れ変数を持つ生成モデルである [11]．LWLM のモデル構造を図 1 に示す．

LWLM における潜在変数は潜在語と呼ばれる．潜在語 h_k は，潜在語のコンテキスト情報 $\boldsymbol{l}_k = h_{k-N+1}, ..., h_{k-1}$ に基づく遷移確率分布から生成し，実際に観測される単語 w_k は，潜在語 h_k に基づく出力確率分布から生成すると仮定したモデルである．つまり LWLM は，次の (1) 式，(2) 式の生成過程に従う．

$$h_k \sim P(h_k|\boldsymbol{l}_k, \boldsymbol{\theta}_h), \tag{1}$$

$$w_k \sim P(w_k|h_k, \boldsymbol{\theta}_w). \tag{2}$$

$\boldsymbol{\theta}_h$ は，遷移確率分布についてのモデルパラメータであり，$\boldsymbol{\theta}_w$ は，出力確率分布についてのモデルパラメータである．このように，LWLM は一般的なクラス N-gram モデルと非常に類似した構造を持つ [6]．この場合，潜在語はクラスに対応することになる．$P(h_k|\boldsymbol{l}_k, \boldsymbol{\theta}_h)$ は，潜在語に対する N-gram モデルとして表され，$P(w_k|h_k, \boldsymbol{\theta}_w)$ は，観測語と潜在語の関係をモデル化している．もし，観測語 w_k と潜在語 h_k が関連するならば，$P(w_k|h_k, \boldsymbol{\theta}_w)$ は高い確率を持ち，関連しない場合は低い確率を持つ．

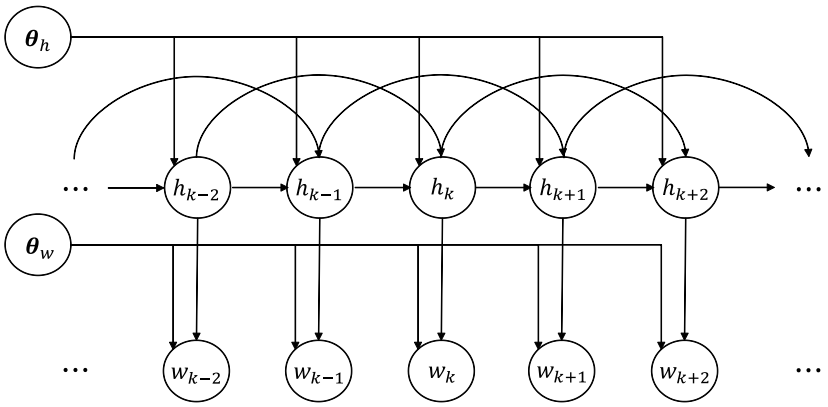

図 1 LWLM の構造

LWLM は，ハードクラスタリング構造を持つクラス N-gram モデルとは異なり，ソフトクラスタリング構造を持つ点が特徴と言える．ハードクラスタリングの場合，ある単語の属するクラスを必ず一意に決定している．しかし，ソフトクラスタリングの場合は，ある語は複数クラスに属し得る．特に LWLM の場合，各単語がすべてのクラスに属す構造を持つ．これに加え，潜在語はモデルの語彙 V に含まれる特定の単語として表現される．つまり，LWLM の潜在語の種類数は，語彙サイズ $|V|$ に一致する．

3.2 ベイズ推定に基づく LWLM の学習

本稿では，ベイズ推定に基づき LWLM を学習する．学習データ $\boldsymbol{W}$ からベイズ推定により LWLM を学習した場合，観測語列 $\boldsymbol{w}$ についての事後分布 $P(\boldsymbol{w}|\boldsymbol{W})$ は，(3) 式の形で与えられる．

$$\begin{aligned} P(\boldsymbol{w}|\boldsymbol{W}) &= \sum_{\boldsymbol{h}} \int P(\boldsymbol{w}|\boldsymbol{h},\boldsymbol{\Theta}) P(\boldsymbol{h}|\boldsymbol{\Theta}) P(\boldsymbol{\Theta}|\boldsymbol{W}) d\boldsymbol{\Theta}, \\ &= \sum_{\boldsymbol{h}} \int \prod_{k=1}^{L} \left[P(w_k|h_k,\boldsymbol{\Theta}) P(h_k|\boldsymbol{l}_k,\boldsymbol{\Theta}) \right] P(\boldsymbol{\Theta}|\boldsymbol{W}) d\boldsymbol{\Theta}, \end{aligned} \tag{3}$$

ここで，$\boldsymbol{\Theta}$ は，$\boldsymbol{\theta}_h$, と $\boldsymbol{\theta}_w$ を表す．$\boldsymbol{h}$ は，観測語列 $\boldsymbol{w}$ に対する潜在語の割り当て，L は観測語列の長さを表す．このようにベイズ推定では，全てのモデルパラメータの可能性を考慮する．

$\boldsymbol{\Theta}$ についての積分は，解析的に解くことができないため，モンテカルロ積分に基づき (4) 式に近似する．

$$P(\boldsymbol{w}|\boldsymbol{W}) \simeq \frac{1}{T} \sum_{\boldsymbol{h}} \sum_{\tau=1}^{T} P(\boldsymbol{w}|\boldsymbol{h},\boldsymbol{\Theta}_\tau) P(\boldsymbol{h}|\boldsymbol{\Theta}_\tau). \tag{4}$$

このように，事後分布は T サンプルの $\boldsymbol{\Theta}$ によって近似できる．1 つの $\boldsymbol{\Theta}$ を用いる場合は点推定となるが，本稿では複数の $\boldsymbol{\Theta}$ を用いたアンサンブルなモデル化を実施する．実際に，複数モデルパラメータのアンサンブルは，言語モデルにおいて有効であり，Random Clustering 言語モデル [21] や Random Forest 言語モデルにおいて有効性が示されている [8].

$\boldsymbol{\theta}_h$，$\boldsymbol{\theta}_w$ には，任意の事前分布を適用可能である．本稿では，$\boldsymbol{\theta}_h$ には，階層 Pitman-Yor 過程に基づく事前分布を使う．この時，$P(h_k|\boldsymbol{l}_k,\boldsymbol{\theta}_h)$ は (5) 式で与えられる．

$$P(h_k|\boldsymbol{l}_k,\boldsymbol{\theta}_h) = \frac{1}{I} \sum_{i=1}^{I} P_{\mathrm{hpy}}(h_k|\boldsymbol{l}_k,\boldsymbol{S}_i). \tag{5}$$

$\boldsymbol{S}$ は階層 Pitman-Yor 過程の Chinese Restaurant Franchise(中華料理店フランチャイズ, CRF) による表現に基づくものであり，Seating Arrangement と呼ばれる割り当てを表す [22]．$P_{\mathrm{hpy}}(h_k|\boldsymbol{l}_k,\boldsymbol{\theta}_h)$ は，I サンプルの $\boldsymbol{S}$ を得ることで近似的に解を得る．特定の $\boldsymbol{S}$ の際の $P_{\mathrm{hpy}}(h_k|\boldsymbol{l}_k,\boldsymbol{S})$ は (6) 式として計算される．

$$\begin{aligned} P_{\mathrm{hpy}}(h_k|\boldsymbol{l}_k,\boldsymbol{S}) &= \frac{c(h_k,\boldsymbol{l}_k) - d_{|\boldsymbol{u}_k|} t(h_k,\boldsymbol{l}_k)}{\theta_{|\boldsymbol{l}_k|} + c(\boldsymbol{l}_k)} \\ &\quad + \frac{\theta + d_{|\boldsymbol{l}_k|} t(\boldsymbol{l}_k)}{\theta_{|\boldsymbol{l}_k|} + c(\boldsymbol{l}_k)} P_{\mathrm{hpy}}(h_k|\pi(\boldsymbol{l}_k),\boldsymbol{S}), \end{aligned} \tag{6}$$

$\pi(\boldsymbol{l}_k)$ は, $\boldsymbol{l}_k$ の文脈の次数を 1 つ下げたものである．$c(h_k,\boldsymbol{l}_k)$ や $t(h_k,\boldsymbol{l}_k)$ は，CRF に基づくパラメータである．また，$d_{|\boldsymbol{u}_k|}$，$\theta_{|\boldsymbol{u}_k|}$ は，それぞれ Pitman-Yor 過程におけるディスカウントパラメータ，ストレングスパラメータである．

一方，$\boldsymbol{\theta}_w$ の事前分布にはディリクレ分布を用いる．この時，$P(w_k|h_k,\boldsymbol{\theta}_w)$ は (7) 式で与えられる．

$$P(w_k|h_k,\boldsymbol{\theta}_w) = \frac{c_0(w_k,h_k) + \alpha P_0(w_k)}{c_0(h_k) + \alpha}, \tag{7}$$

$P_0(w_k)$ は，学習データ $\boldsymbol{W}$ における unigram 確率の ML 推定値であり，$c_0(w_k,h_k)$，$c_0(h_k)$ は $\boldsymbol{W}$ と $\boldsymbol{H}$ の対応関係から計算可能なカウントである．α はハイパーパラメータである [23].

LWLM の学習，つまり (5) 式から (7) 式を計算するには，学習データ $\boldsymbol{W}$ の裏に隠れた潜在語の割り当て $\boldsymbol{H}$ を推定する必要がある．潜在語の割り当ての推定にはギブスサンプリングを利用することができる．ある潜在語 h_k についての条件付き確率は (8) 式にしたがって計算できる．

$$P(h_k|\boldsymbol{W}, \boldsymbol{H}^{-k}, \boldsymbol{\Theta}^{-k}) \propto P(w_k|h_k, \boldsymbol{\Theta}^{-k}) \prod_{j=k}^{k+n-1} P(h_j|\boldsymbol{l}_j, \boldsymbol{\Theta}^{-k}), \quad (8)$$

$\boldsymbol{H}^{-k}$ は，h_k を除く潜在語の割り当てを表す．また，$\boldsymbol{\Theta}^{-k}$ は，$\boldsymbol{\theta}_h^{-k}$ と $\boldsymbol{\theta}_w^{-k}$ を表す．ギブスサンプリングは，(8) 式に示す確率分布に従って新たな潜在語をサンプリングし，k 番目の位置を新たな潜在語に置き換えることを繰り返すことで実現される．この時，潜在語の置き換えを行うごとに，(5) 式から (7) 式の再計算を行う．この一連の流れを収束するまで繰り返すことにより，LWLM を学習できる．

3.3 LWLM に基づく予測分布の計算

LWLM を用いて単語 N-gram モデルと同様の予測分布を得ることは難しい．コンテキスト情報 $\boldsymbol{u}_k$ が与えられた場合の w_k の予測分布は，(9) 式に従う．

$$P_{\mathrm{LW}}(w_k|\boldsymbol{u}_k) \approx \frac{1}{\mathrm{T}} \sum_{\tau=1}^{\mathrm{T}} \sum_{\boldsymbol{l}_k, h_k} P(w_k|h_k, \boldsymbol{\Theta}_\tau) P(h_k|\boldsymbol{l}_k, \boldsymbol{\Theta}_\tau). \quad (9)$$

この式において，$P(w_k|h_k, \boldsymbol{\Theta}_\tau)P(h_k|\boldsymbol{l}_k, \boldsymbol{\Theta}_\tau)$ の計算は，クラス N-gram モデルの計算と同様の計算である．LWLM ではこれに加え，2 種類の総和を考慮する必要がある．

まず，LWLM はソフトクラスタリング構造を持つために，あらゆる潜在語 (クラス) の割り当てについて計算する必要がある．一般的なクラス N-gram モデルのようにハードクラスタリングの構造を持つ場合は，ある単語列を生成するクラスの割り当ては一意に決まっている．しかしながら，ソフトクラスタリングの場合は，全ての $\boldsymbol{l}_k$ と h_k の組を考慮しなければならない．

また，モンテカルロ近似によりベイズ推定を近似しているので，T サンプルのモデルパラメータについてそれぞれ計算を行う必要がある．

つまり，潜在語の種類数が 100,000，サンプルしたモデルパラメータの数が 10 であれば，通常のクラス N-gram モデルの $100,000^3 \times 10$ 倍の計算コストが必要である．したがって，LWLM を単語予測のために音声認識に適用することは非現実的と言え，パープレキシティを測ることすら困難である．

3.4 LWLM を用いたデータ生成

LWLM を用いて直接予測分布を得ることは困難である一方で，生成モデルという特徴を活かし，確率過程に基づきデータ生成を行うことは容易である．モンテカルロ近似を行った LWLM を用いて，確率過程に基づきデータ生成を行う場合は，次の **Algorithm 1** にしたがう．

Algorithm 1 Random sampling on LWLM.

for $\kappa = 1$ to K **do**
  $\boldsymbol{\Theta}_\tau \sim P(\boldsymbol{\Theta}_\tau) = \frac{1}{\mathrm{T}}$
  $h_\kappa \sim P(h_\kappa|\boldsymbol{l}_\kappa, \boldsymbol{\Theta}_\tau)$
  $w_\kappa \sim P(w_\kappa|h_\kappa, \boldsymbol{\Theta}_\tau)$
end for

$\boldsymbol{\Theta}_\tau$ は，サンプルされた τ 個目のモデルパラメータである．このように K 回の繰り返しにより，潜在語系列 $h_1, ..., h_K$，観測語系列 $w_1, ..., w_K$ を生成することが可能である．LWLM から生成したデータは，LWLM 自体の柔軟なモデル構造を反映したデータであり，LWLM 自身の学習データには含まれない多様な言語表現を含むことが期待できる．したがって，生成したデータから N-gram モデルを学習すれば，元の学習データから直接学習した N-gram モデルとは異なる特徴を持ち，また幅広い言語表現に頑健なモデルの構築が期待できる．

表 1 実験に用いる各データの詳細

	ドメイン	総形態素数
学習データ	学会講演 & 模擬講演	7,317,392
開発データ	模擬講演	28,046
テストデータ A	模擬講演	27,907
テストデータ B	コンタクトセンタ	24,665
テストデータ C	ボイスメール	21,044

4. 評価実験

4.1 実験条件

本稿では日本語話し言葉コーパス (CSJ) を利用して評価実験を行う [24]．我々は CSJ のデータを，学会講演および模擬講演の 2672 講演を学習データ，学会講演 10 講演を開発データ，学会講演 10 講演をテストデータと分割した．さらに，学習データ・開発データのドメインとは異なるドメインでの性能も調査するため，コンタクトセンタタスク，ボイスメールタスクについてもテストデータを準備した．いずれも，講演音声である CSJ とは大きく異なるタスクと言える．各データの詳細を表 1 に示す．

音響モデルは，テストデータごとに 3000 状態 32 混合 triphone HMM を準備した．音声認識デコーダには，WFST ベースのデコーダである VoiceRex を用いた [25], [26]．また，テキストの形態素解析には JTAG を利用した [27]．

4.2 実験で使用する言語モデル

本稿で学習する N-gram モデルは，一部のものを除き 3-gram モデルとし，その際のカットオフは行わないこととした．なお，学習データに基づく単語辞書の語彙サイズ

表 2 各言語モデルの評価結果

Setup	開発データ (In-Domain)		テストデータ A (In-Domain)		テストデータ B (Out-Of-Dmain)		テストデータ C (Out-Of-Domain)	
	PPL	WER(%)	PPL	WER(%)	PPL	WER(%)	PPL	WER(%)
モデル単体の評価結果								
(1) WBLM	90.61	24.38	76.51	28.40	160.83	49.15	182.01	41.02
(2) MKNLM	86.81	24.26	73.09	28.80	164.07	49.32	189.91	40.78
(3) HPYLM	79.32	23.51	67.50	27.94	158.13	48.72	175.62	40.68
(4) MELM	83.00	23.66	69.80	28.16	150.19	47.38	170.58	39.56
(5) C-HPYLM	82.97	23.91	69.36	28.08	158.92	48.96	180.89	40.96
(6) RNNLM	**70.39**	-	**61.28**	-	145.05	-	158.57	-
(7) HPYLM(g)	82.69	24.16	70.43	28.16	161.56	49.03	177.61	41.07
(8) RNNLM(g)	98.65	24.88	82.23	28.84	153.89	48.31	163.99	39.44
(9) LWLM(g)	79.57	**23.45**	66.93	**27.85**	**141.34**	**46.86**	**147.87**	**38.71**
モデル混合の評価結果								
RNNLM+HPYLM	64.01	-	55.84	-	138.21	-	147.62	-
HPYLM(g)+HPYLM	79.14	23.46	68.04	28.04	158.56	48.88	176.24	40.92
RNNLM(g)+HPYLM	77.96	23.15	66.09	27.36	143.88	46.92	149.81	38.75
LWLM(g)+HPYLM	72.86	**22.47**	62.05	**26.42**	134.65	**46.19**	141.23	**37.92**
LWLM(g)+RNNLM+HPYLM	**61.56**	-	**53.36**	-	**120.21**	-	**133.09**	-

は，83,536 単語である．

まず，基本的な言語モデルとして次の 6 種類を用いた．

(1) **WBLM**: Witten-Bell スムージングを適用した単語 3-gram モデル．

(2) **MKNLM**: Modified Kneser-Ney スムージングを適用した単語 3-gram モデル．

(3) **HPYLM**: 3-gram の階層 Pitman-Yor 言語モデル．Burn-in 期間は 200 回，その後 10 サンプルで近似．

(4) **MELM**: 3-gram までの単語素性を用いた最大エントロピー言語モデル [28]．

(5) **C-HPYLM**: 5000 クラスのクラス 3-gram 階層 Pitman-Yor 言語モデル．クラスの決定は 2-gram の単語素性に基づくクラスタリングを利用 [6]．クラス内確率は最尤推定値．

(6) **RNNLM**: Recurrent Neural Network 言語モデル．中間層の数は 500，また出力層のクラス化は 1000 クラス．開発データに最適になるよう学習 [16]．

次に，本稿で着目する生成型アプローチに基づく言語モデルとして次の 3 種類を用いた．

(7) **HPYLM(g)**: 前述の HPYLM を利用してデータ生成を行い，生成したデータから学習した 3-gram の階層 Pitman-Yor 言語モデル．

(8) **RNNLM(g)**: 前述の RNNLM を利用してデータ生成を行い，生成したデータから学習した 3-gram の階層 Pitman-Yor 言語モデル．

(9) **LWLM(g)**: 提案法．学習データから LWLM を最初に推定．Burn-in 期間として 500 回，その後 10 サンプルで近似．学習した LWLM を利用してデータ生成を行い，生成したデータから学習した 3-gram の階層 Pitman-Yor 言語モデル．

これらのモデルは，RNNLM を除き，言語モデルのデータフォーマットとして標準的な ARPA フォーマットを用いて表現可能である．

実験ではさらに，モデル間を線形補間により混合した場合についても，いくつかのケースについて検討した．なお線形補間による混合を行う場合，モデル間の混合重みは EM アルゴリズムを利用して開発データに対して最適にした．

4.3 各言語モデルの性能評価

我々は各言語モデルを構築し，開発データおよび各テストデータに対して，パープレキシティ (PPL) および単語誤り率 (WER) による評価を行った．生成型アプローチである HPYLM(g)，RNNLM(g)，LWLM(g) については，それぞれ約 10 億単語を生成して，N-gram モデルを学習した．なお，RNNLM は，マルチパスでなければ音声認識に適用できないため，今回はパープレキシティによる評価のみ行い，単語誤り率による評価は行わなかった．この結果を表 2 に示す．

表 2 の結果については，学習データと同様のタスクである In-Domain データ，学習データと異なるタスクである Out-Of-Domain データに分けて考察を述べる．

4.3.1 In-Domain データに対する評価結果

まず，In-Domain データである開発データ，およびテストデータ A について結果を述べる．パープレキシティに関しては，単体のモデルとしては RNNLM が最も高い性能を示した．これは，非線形な識別モデルとしてタスクに特化した学習を行っている点や，長距離文脈の考慮に起因す

表 3　生成型アプローチにおける生成データ量と性能の関係

	学習データ量 (生成データ量)	テストデータ A (In-Domain)		テストデータ B (Out-Of-Dmain)		テストデータ C (Out-Of-Domain)	
		PPL	WER(%)	PPL	WER(%)	PPL	WER(%)
HPYLM	7.3 M	67.50	27.94	158.13	48.72	175.62	40.68
HPYLM(g)	10.0 M	72.34	28.44	164.75	49.34	180.54	41.43
	100.0 M	70.88	28.21	162.23	49.23	178.15	41.16
	1000.0 M	70.43	28.16	161.56	49.03	177.61	41.07
RNNLM(g)	10.0 M	98.07	30.19	168.44	49.75	174.58	41.09
	100.0 M	87.07	29.43	158.59	49.11	167.38	40.38
	1000.0 M	82.23	28.84	153.89	48.31	163.99	39.44
LWLM(g)	10.0 M	79.42	29.80	154.41	48.25	164.73	39.85
	100.0 M	70.21	28.36	145.37	47.31	154.41	39.27
	1000.0 M	**66.93**	**27.85**	**141.34**	**46.86**	**147.87**	**38.71**

ると考えられる．しかしながら，RNNLM は 1 パスデコーディングでは直接利用できない．一方，RNNLM を近似したモデルである RNNLM(g) は，In-Domain データに対しては高い性能が得られなかった．

1 パスデコーディングで利用可能なモデルの中では，モデル単体としては提案法である LWLM(g) が最も高い性能を示した．この性能は，HPYLM や MELM の性能と大きく差はないが，確率的に生成したデータに基づいているにも関わらず，In-Domain データに対する言語モデルとして有効であることが分かった．

さらにモデル間を組み合わせることで，1 パスデコーディングで利用可能なモデルの中では，LWLM(g)+HPYLM が最も高い性能を示した．LWLM(g) と HPYLM は，単体では同等の性能であったにも関わらず，両者を組み合わせることで更なる改善を得た．これは，元の学習データと LWLM を利用して生成したデータは，異なる性質を持つことに起因すると考えられる．また，これに RNNLM を加えた LWLM(g)+RNNLM+HPYLM が最も低いパープレキシティを示した．RNNLM もまた，LWLM(g) や HPYLM とは異なる性質を持つモデルであると考えられる．

4.3.2　Out-Of-Domain データに対する評価結果

次に，Out-Of-Domain データであるテストデータ B，テストデータ C について結果を述べる．パープレキシティ，単語誤り率ともに，単体のモデルとして LWLM(g) が他のモデルよりも高い性能を示した．これは，LWLM の柔軟なモデル構造には言語モデルの汎化性能を高める効果があり，LWLM を利用して生成したデータには様々な言語表現が広く含まれていることが起因していると考えられる．それにより，Out-Of-Domain データに対しても有効に動作したと考えられる．従来のハードクラスタリング構造を持つ C-HPYLM は，次元削減を行わない HPYLM とほとんど変わらない性能を示した．つまり，単純なクラス化では汎化性能を高めることはできない．したがって，LWLM が Out-Of-Domain データに対して性能が高い理由は，ソフトクラスタリングの構造や膨大なクラス数が有効に働いていることに起因すると考えられる．

さらに，モデル間を混合することで，1 パスデコーディングで利用可能なモデルの中では LWLM(g)+HPYLM が最も高い性能を示した．そして，In-Domain データに対する評価と同様に，RNNLM を加えた LWLM(g)+RNNLM+HPYLM が最も低いパープレキシティを示した．またこの結果から，モデル間を混合する際のモデル混合比を開発データに対して最適化しているにも関わらず，Out-Of-Domain データに対して有効な混合を行えていることも分かった．

4.4　生成データ量と性能の関係の評価

生成型アプローチは，大量のデータを生成して N-gram モデルの学習を行うため，生成データ量が言語モデルの性能に大きく関わる．そこで我々は，生成型アプローチにおける生成データ量と性能の関係を調査した．生成型アプローチである HPYLM(g)，RNNLM(g)，LWLM(g) のそれぞれについての生成データ量を変化させたときの，各テストデータに対する結果を表 3 に示す．

表 3 の結果から，生成するデータ量が増えるにつれて，各モデルのパープレキシティおよび単語誤り率が改善していることが見て取れる．HPYLM(g) については，生成データ量が少ない場合でもある程度収束していることが分かる．これは HPYLM のモデル構造が単純であるためだと思われる．一方，RNNLM(g) や LWLM(g) は，大量のデータを生成しないと収束しないことが分かる．元々の学習データの総形態素数が約 7 百万形態素であったことと比較すると，LWLM や RNNLM を十分に活かした N-gram モデルを構築するためには，大量のデータ生成が必要と言える．

さらに，生成型アプローチで生成するデータの質を調査するために，生成したデータから学習した N-gram モデルについて，テストデータに対する N-gram ヒットレートを調査した．N のオーダーが高い時の N-gram ヒットレート

表 4 テストデータに対するに対する N-gram ヒットレート

	学習データ量 (生成データ量)	テストデータ A (In-Domain)			テストデータ B (Out-Of-Dmain)			テストデータ C (Out-Of-Domain)		
		Hit rate (%)			Hit rate (%)			Hit rate (%)		
		1-gram	2-gram	3-gram	1-gram	2-gram	3-gram	1-gram	2-gram	3-gram
HPYLM	7.3 M	8.78	26.75	64.47	15.88	39.36	44.75	17.11	35.13	47.76
HPYLM(g)	10.0 M	10.03	27.84	62.12	17.39	39.00	43.61	17.60	33.66	47.75
	100.0 M	6.06	21.22	73.71	12.55	35.07	52.38	13.27	29.02	57.70
	1000.0 M	3.41	16.45	80.14	8.50	31.76	59.74	8.13	26.73	65.14
RNNLM(g)	10.0 M	10.99	30.10	58.91	16.53	40.42	43.04	16.52	36.18	47.29
	100.0 M	5.95	23.49	70.56	10.76	36.41	52.82	10.90	29.17	59.92
	1000.0 M	3.89	17.13	78.98	7.58	31.56	60.86	7.96	22.59	69.45
LWLM(g)	10.0 M	8.93	28.80	62.27	15.19	39.93	44.88	16.07	34.55	49.39
	100.0 M	4.51	21.33	74.16	10.22	34.20	55.58	10.37	28.49	61.14
	1000.0 M	2.21	14.72	**83.07**	7.37	29.17	**63.46**	7.77	19.40	**72.82**

表 5 モデル縮退時の性能評価

	ファイルサイズ	開発データ (In-Domain)		テストデータ A (In-Domain)		テストデータ B (Out-Of-Dmain)		テストデータ C (Out-Of-Domain)	
		PPL	WER(%)	PPL	WER(%)	PPL	WER(%)	PPL	WER(%)
LWLM(g)+HPYLM	22.0 G	72.86	22.47	62.05	26.42	134.65	46.19	141.23	37.92
	9.8 G	73.39	22.52	62.86	26.48	135.02	46.27	143.05	37.96
	3.9 G	73.74	22.68	63.21	26.60	135.47	46.41	143.91	37.73
	2.2 G	74.09	22.70	63.51	26.64	136.02	46.63	144.44	37.74
	563 M	75.41	22.98	64.69	27.05	137.86	46.75	147.66	38.06
	353 M	76.39	23.18	65.46	27.34	138.95	46.85	149.08	38.36
HPYLM	322 M	79.32	23.51	67.50	27.94	158.13	48.72	175.62	40.68

が高いほど，そのテストデータに対して良いデータができていることになる．各オーダの N-gram ヒットレートの結果を表 4 に示す．

表 4 の結果から，生成するデータ量が増えるにつれて，3-gram ヒットレートが上がることが見てとれる．3-gram ヒットレートが高くなるということは，テストデータに含まれる単語 3 つ組を実際に生成できたことになる．LWLM(g) は，HPYLM(g) や LWLM(g) と比較して，生成データ量が増えるに連れて効率的に 3-gram ヒットレートを上げていることが見てとれ，生成型アプローチのための生成モデルとして有効であることが分かる．

4.5 モデル縮退時の性能評価

生成型アプローチでは，大量のデータを生成してモデルを学習するため，モデルサイズが大きくなってしまうという欠点を持つ．そこで我々は，Entropy Pruning の適用によるモデル縮退時の性能について調査した [14]．ここでは，10 億形態素を生成した場合に構築した LWLM(g) と，HPYLM を線形補間により混合したモデル LWLM(g)+HPYLM に対して，Entropy Pruning を適用し，モデル縮退を試みた．圧縮なし・アスキー形式の ARPA フォーマットとして保存した N-gram モデルのファイルサイズと各テストデータに対する性能を図 5 に示す．

表 5 の結果から，Entropy Pruning を適用することによって，効率的にモデル縮退できていることが分かる．学習データから直接推定した HPYLM と同等のファイルサイズまで縮退した場合，In-Domain データ，Out-Of-Domain データに関わらず，パープレキシティ，単語誤り率の両面で HPYLM と比較して高い性能が得られていることが見て取れる．特に，Out-Of-Domain データに対しては，モデル縮退した場合でも良い性能が得られていることが分かる．したがって，モデルサイズが問題になる場合は，Entropy Pruning によるモデル縮退が有用と言える．

5. まとめ

本稿では，言語モデルのデータスパースネスの問題を解決するために，学習データ自体を新たに生成し，生成したデータに基づき N-gram モデルを学習する生成型アプローチについて検討した．具体的には，柔軟なモデル構造を持つ LWLM に着目し，LWLM の確率過程に従ってデータ生成を行い，N-gram モデルに近似する方法を提案した．

LWLM が持つ柔軟なモデル構造に基づいて生成したデータは様々な言語表現を含み，生成データから学習した N-gram モデルは，様々なタスクで頑健に動作することが分かった．さらに，生成データは元の学習データとは異なる特徴を持ち，元の学習データから推定したモデルと，生

成したデータから推定したモデルを組み合わせることで相乗効果が得られることが分かった．また，生成型アプローチで十分な性能を得るためには，多くのデータを生成する必要があるが，Entropy Pruning に基づきモデル縮退を行うことで，モデルサイズが小さくても性能の高いモデルを構築できることを示した．

参考文献

[1] Joahua. T. Goodman, “A bit of progress in language modeling,” *Computer Speech & Language*, vol.15, pp.403-434, 2001.
[2] Stanley F. Chen and Joshua Goodman, “An Empirical Study of Smoothing techniques for language modeling,” *Computer Speech & Language*, vol.13, pp.359-383, 1999.
[3] Reinhard Kneser and Hermann Ney, “Improved backing-off for m-gram language modeling,” *In Proc. ICASSP*, Vol.1, pp.181-184, 1995.
[4] Yee Whye Teh, “A Hierarchical Bayesian Language Model based on Pitman-Yor Processes,” *In Proc. COLING/ACL 2006*, pp.985-992, 2006.
[5] Songfang Huang and Marc Yor, “Hierarchical Pitman-Yor language models for ASR in meetings,” *In Proc. ASRU 2007*, pp.124-129, 2007.
[6] Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, Jenifer C. Lai, “Class-based n-gram models of natural language,” *Computational Linguistics*, vol.18, pp.467-479, 1992.
[7] Gerasimos Potamianos and Frederick Jelinek, “A study of n-gram and decision tree letter language modeling methods,” *Speech Communication*, vol.24, no.3, pp.171-192, 1998.
[8] Peng Xu and Frederick Jelinek, “Random forests in language modeling,” *In Proc. EMNLP 2004*, pp.325-332, 2004.
[9] Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin, “A neural probabilistic language model,” *Journal of Machine Learning Research*, vol.3, pp. 1137-1155, 2003.
[10] Stefan Kombrink, Tomas Mikolov, Martin Karafiat, Lukas Burget, “Recurrent Neural Network based Language Modeling in Meeting Recognition,” *In Proc. Interspeech 2011*, pp.2877-2880, 2011.
[11] Koen Deschacht, Jan De Belder, Marie-Francine Moens, “The latent words language model,” *Computer Speech & Language*, vol.26, pp.384-409, 2012.
[12] Sharon Goldwater and Tom Griffiths, “A fully bayesian approach to unsupervised part-of-speech tagging,” *In Proc. ACL 2007*, pp.744-751, 2007.
[13] Phil Blunsom and Trevor Cohn, “A Hierarchical Pitman-Yor Process HMM for Unsupervised Part of Speech Induction,” *In Proc. ACL 2011*, pp.865-874, 2011.
[14] Andreas Stolcke, “Entropy-based pruning of backoff language models,” *In Proc. DARPA Broadcast News Transcription and Understanding Workshop*, pp.270-274, 1998.
[15] Wen Wang, Andreas Stolcke, and Mary P. Harper, “The use of a linguistically motivated language model in conversational speech recognition,” *In Proc. ICASSP 2004*, pp.261-264, 2004.
[16] Tomas Mikolov, Anoop Deoras, Stefan Kombrink, Lukas Burget, Jan Cernocky, “Empirical Evaluation and Combination of Advanced Language Modeling Techniques,” *In Proc. Interspeech 2011*, pp.605-608, 2011.
[17] Ebru Arisoy, Stanley F. Chen, Bhuvana Ramabhadran, Abhinav Sethy, “Converting Neural Network Language Models into Back-Off Language Models for Efficient Decoding in Automatic Speech Recognition,” *In Proc. ICASSP 2013*, pp.8242-8246, 2013.
[18] Anoop Deoras, Tomas Mikolov, Stefan Kombrink, Martin Karafiat, Sanjeev Khudanpur, “Variational Approximation of Long-Span Language Models in LVCSR,” *In Proc. ICASSP 2011*, pp.5532-5535, 2011.
[19] Anoop Deoras, Tomas Mikolov, Stefan Kombrink, and Kenneth Church, “Approximate inference: A sampling based modeling technique to capture complex dependencies in a language model,” *Speech Communication*, vol.55, no.1, pp.162-177, 2013.
[20] Gwenole Lecorve and Petr Motlicek, “Conversion of recurrent neural network language models to weighted finite state transducers for automatic speech recognition,” *In Proc. Interspeech*, 2012.
[21] Ahmad Emami and Frederick Jelinek, “Random clusterings for language modeling,” *In Proc. ICASSP 2005*, vol.1, pp.581-584, 2005.
[22] Yee Whye Teh, Michael. I. Jordan, M. J. Beal, and David. M. Blei, “Hierarchical dirichlet processes,” *Journal of the American Statistical Association*, vol.101, pp.1566-1581, 2006.
[23] David J. C. MacKay and Linda C. Peto, “A hierarchical Dirichlet language model,” *Natural language engineering*, vol.1, pp.289-308, 1994.
[24] Kikuo Maekawa, Hanae Koiso, Sadaoki Furui and Hitoshi Isahara, “Spontaneous speech corpus of Japanese,” *In proc.LREC*, pp.947-952, 2000.
[25] Takaaki Hori, Chiori Hori, Yasuhiro Minami and Atsushi Nakamura, “Efficient WFST-based one-pass decoding with on-the-fly hypothesis rescoring in extremely large vocabulary continuous speech recognition,” *IEEE transactions on Audio, Speech and Language Processing*, vol.15, no.4, pp.1352-1365, 2007.
[26] Hirokazu Masataki, Daisuke Shibata, Yuichi Nakazawa, Satoshi Kobashikawa, Atsunori Ogawa and Katsutoshi Ohtsuki, “VoiceRex Spontaneous speech recognition technology for contact-center conversations,” *NTT Tech. Rev.*, vol.5, no.1, pp.22-27, 2007.
[27] Takeshi Fuchi and Shinichiro Takagi, “Japanese Morphological Analyzer using Word Co-occurence-JTAG,” *In Proc. COLING-ACL*, pp.409-413, 1998.
[28] Tanel Alumae and Mikko Kurimo, “Efficient Estimation of Maximum Entropy Language Models with N-gram features: an SRILM extension,” *In Proc. Interspeech 2010*, pp.1820-1823, 2010.