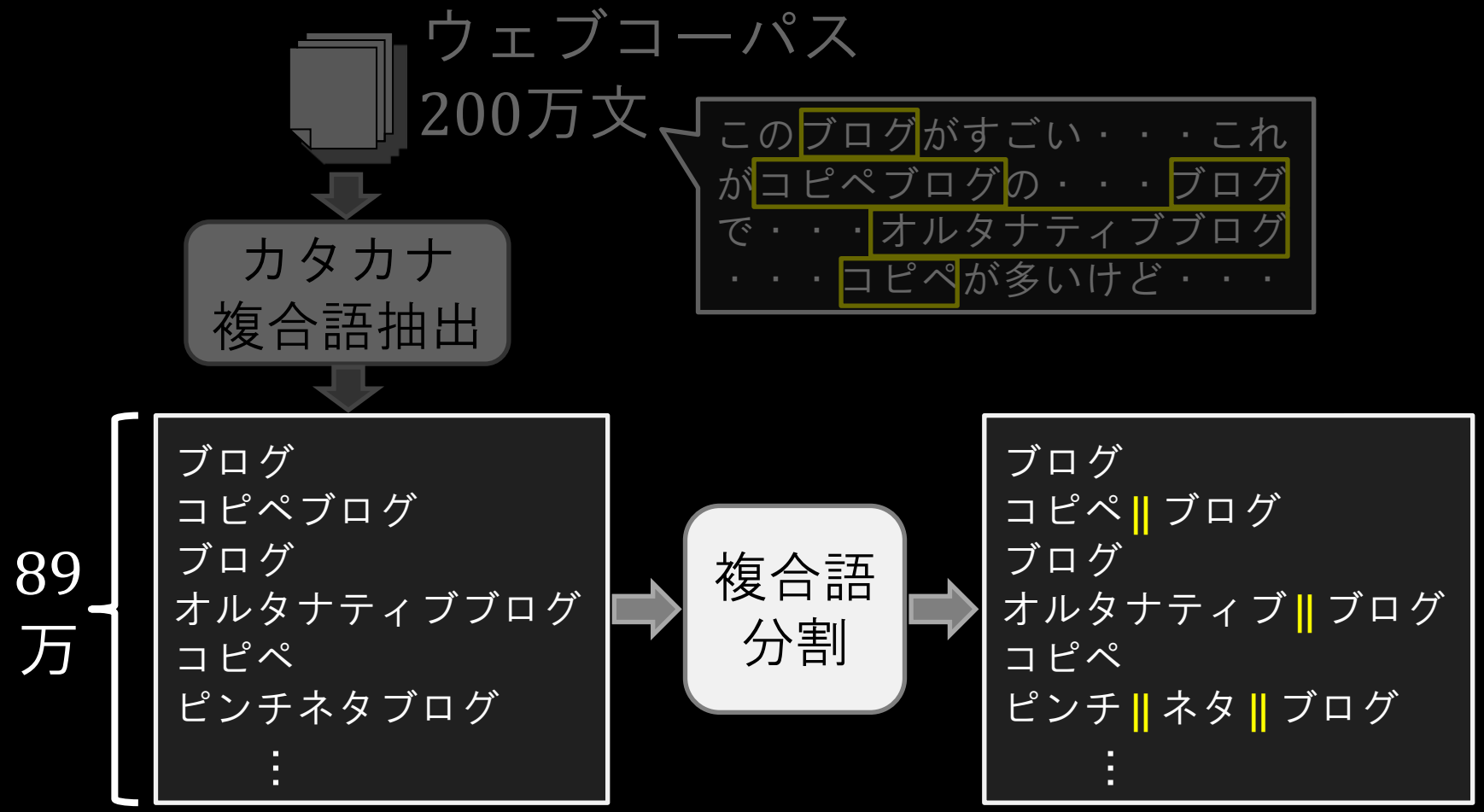
ベイズ学習による
カタカナ複合語の分割

2012年3月14日

京都大学
村脇 有吾　岸本 侑也　黒橋 禎夫

言語処理学会第18回年次大会@広島

カタカナ複合語分割


ウェブコーパス
200万文
このブログがすごい・・・これがコピペブログの・・・ブログで・・・オルタナティブブログ・・・コピペが多いけど・・・
カタカナ
複合語抽出
89
万
ブログ
コピペブログ
ブログ
オルタナティブブログ
コピペ
ピンチネタブログ
⋮
複合語
分割
ブログ
コピペ || ブログ
ブログ
オルタナティブ || ブログ
コピペ
ピンチ || ネタ || ブログ
⋮

Negative resultsです

モデル		F値
分割なし		.296
JUMAN 7.0 (辞書: 基本)		.532
JUMAN 7.0 (辞書: 基本 + ウェブ + Wikipedia)		**.815**
MeCab (辞書: ipadic)		.559
KyTea 4.0 (モデル: デフォルト)		.745
提案手法	Unigram モデル	.717
	前向きBigramモデル	.626
	後ろ向きBigramモデル	.673

※予稿集と異なる結果を報告

スコア関数と探索

- モデル = スコア関数

ベイズ・モデル

score(ブログ
コピペ || ブログ
ブログ
オルタナティブ || ブログ
コピペ
ピンチ || ネタ || ブログ
⋮) = -15113729

※大きいほど良い

- 探索
  argmax score(…)

Gibbsサンプリングによる近似的探索

※実際のモデルでは、単語列以外に補助変数の値もスコア関数に与える必要がある

ベイズ学習による複合語分割 1/3

1. Unigramモデル

   P(コピペ || ブログ) = P(コピペ) P(ブログ)

2. 前向きBigramモデル

   P(コピペ || ブログ) = P(コピペ|$) P(ブログ|コピペ) P($|ブログ)

3. 後ろ向きBigramモデル

   P(コピペ || ブログ) = P(ブログ|$) P(コピペ|ブログ) P($|コピペ)

ベイズ学習による複合語分割 2/3

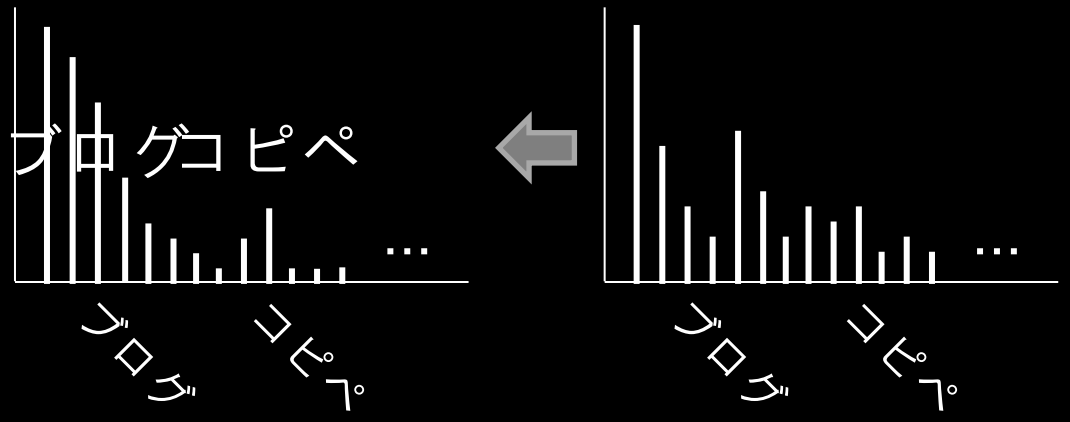
事前分布をおいてベイズ化 (Goldwater+, 2009)(Teh+, 2006)

- 事前分布: 文字N-gramを基底分布とするPitman-Yor過程

Unigramモデルから生成された単語列

G: Unigram分布

P_0: 文字N-gram分布



モデルの性質

- 各単語をデータ中で使いまわすとスコアが上がる
- ただし分割しすぎるとスコアが下がる

ベイズ学習による複合語分割 3/3

- 利点
  - 全体を最適化するモデル
  - 理論的裏付け
  - 拡張性
- 欠点
  - 人間の分割基準になかなか従ってくれない
  - スケールしない

手がかりの追加
- 正解分割複合語
- 言い換え表現
- 共起する用言

型別サンプリング
を用いて中規模
データに適応

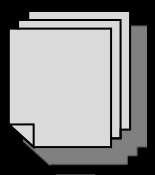
手がかり1:

Unigramモデル	-0.007
前向きBigramモデル	+0.024
後ろ向きBigramモデル	+0.025

- 正解基準に従って人
  - 京大コーパス (毎日新聞1995年) から抽出
- 正解分割の出現頻度をモデルに与えておく
  - 正解分割に現れたunigram, bigramをウェブテキストでも再現する方向にバイアスがかかる
- あまり効果がない
  - 複合語の数はウェブテキストが京大コーパスの64倍
  - 京大コーパスによる評価用正解データの単語被覆率はわずか31.2%

手がかり表現2: 言い換え表現

(Kaji+, 2011)



ウェブコーパス
200万文

前向きBigramモデル +0.021

後ろ向きBigramモデル +0.042

カタカナ
複合語抽出

・・・コピペが多いけど・・・
オルタナティブ・ブログで

89万

ブログ
コピペブログ
ブログ
オルタナティブブログ
コピペ
ピンチネタブログ
オルタナティブ|ブログ

この点で必ず区切るという制約

複合語分割

ブログ
コピペ||ブログ
ブログ
オルタナティブ||ブログ
コピペ
ピンチ||ネタ||ブログ
オルタナティブ|ブログ

※入力データが異なるので比較は厳密でない

手がかり表現3: 共起する用言

ライス
カレー || ライス
プライス
ロー ? プ ? ライス
ロープ
ロー || キック
⋮

- 前向きBigramモデル **-0.022**

P(ロー|$) P(プライス|ロー) P(～を提示|プライス)

vs.

P(ロープ|$) P(ライス|ロープ) P(～を提示|ライス)

- 後ろ向きBigramモデル **+0.030**

P(プライス|～を提示) P(ロー|プライス) P($|ロー)

vs.

P(ライス|～を提示) P(ロープ|ライス) P($|ロープ)

探索ベースライン: 複合語単位のブロック・サンプリング

1. 適当に分割を初期化
2. 局所的に分割を変更

以下の候補から確率的に局所分割を選択 (C: 定数)

1. score(“コピペ” + 残り) =log(0.889)+C
2. score(“コ” + “ピペ” + 残り) =log(0.060)+C
3. score(“コピ” + “ペ” + 残り) =log(0.050)+C
4. score(“コ” + “ピ” + “ペ” + 残り) =log(0.001)+C

ブ || ログ
コピペ || ブログ
ブロ || グ
オル || タナ || ティブブログ
コピペ
ピン || チネ || タブ || ログ
⋮

3. 2を繰り返す

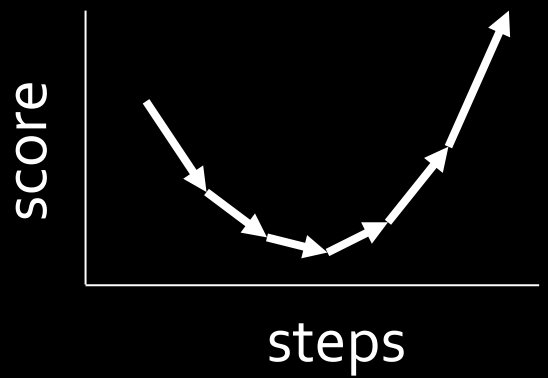
※実際には動的計画法で近似値を求める

(Mochihashi+, 2009)

探索: スケールしない

Step 6

コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
⋮



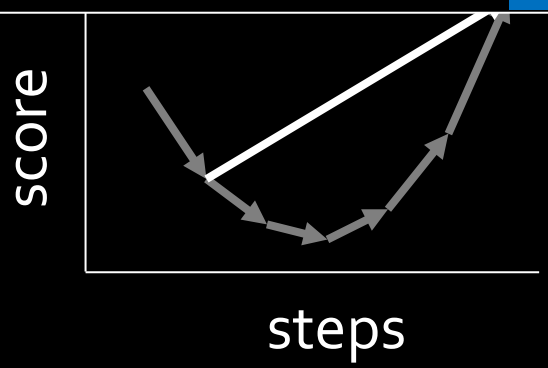
- 最適解にたどりつくためにしばしば谷を下る必要がある
- データサイズが増えると谷が深くなり、現実的な時間内で移れなくなる

探索: 型別

Unigramモデル	+0.048
前向きBigramモデル	+0.064
後ろ向きBigramモデル	+0.050

Step 1

コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
⋮



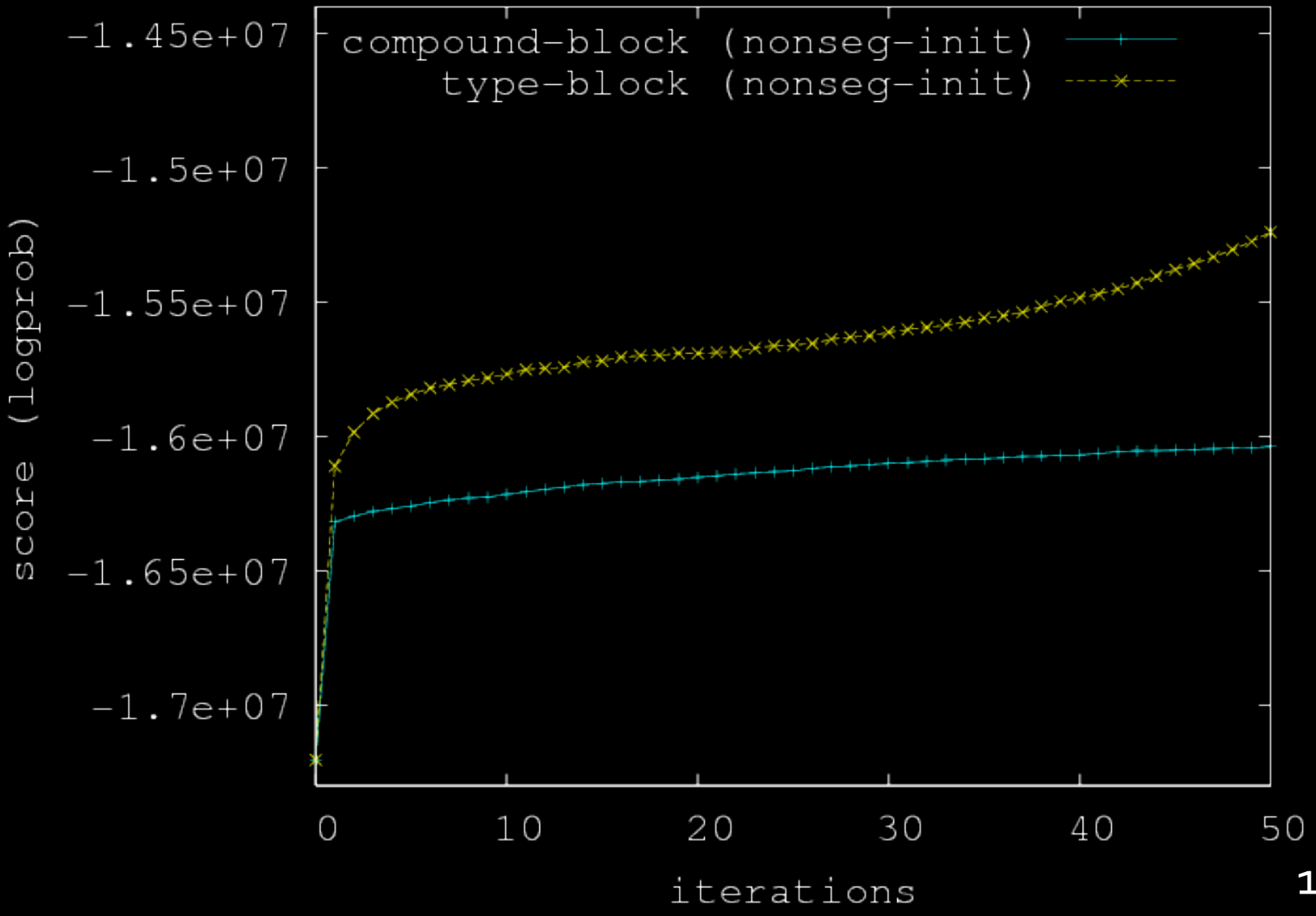
score(コピペブログ
コピペブログ
コピペブログ
コピペブログ
コピペ || ブログ
コピペブログ
⋮)

vs.

score(コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
コピペ || ブログ
⋮)

(Murawaki+, 2011)

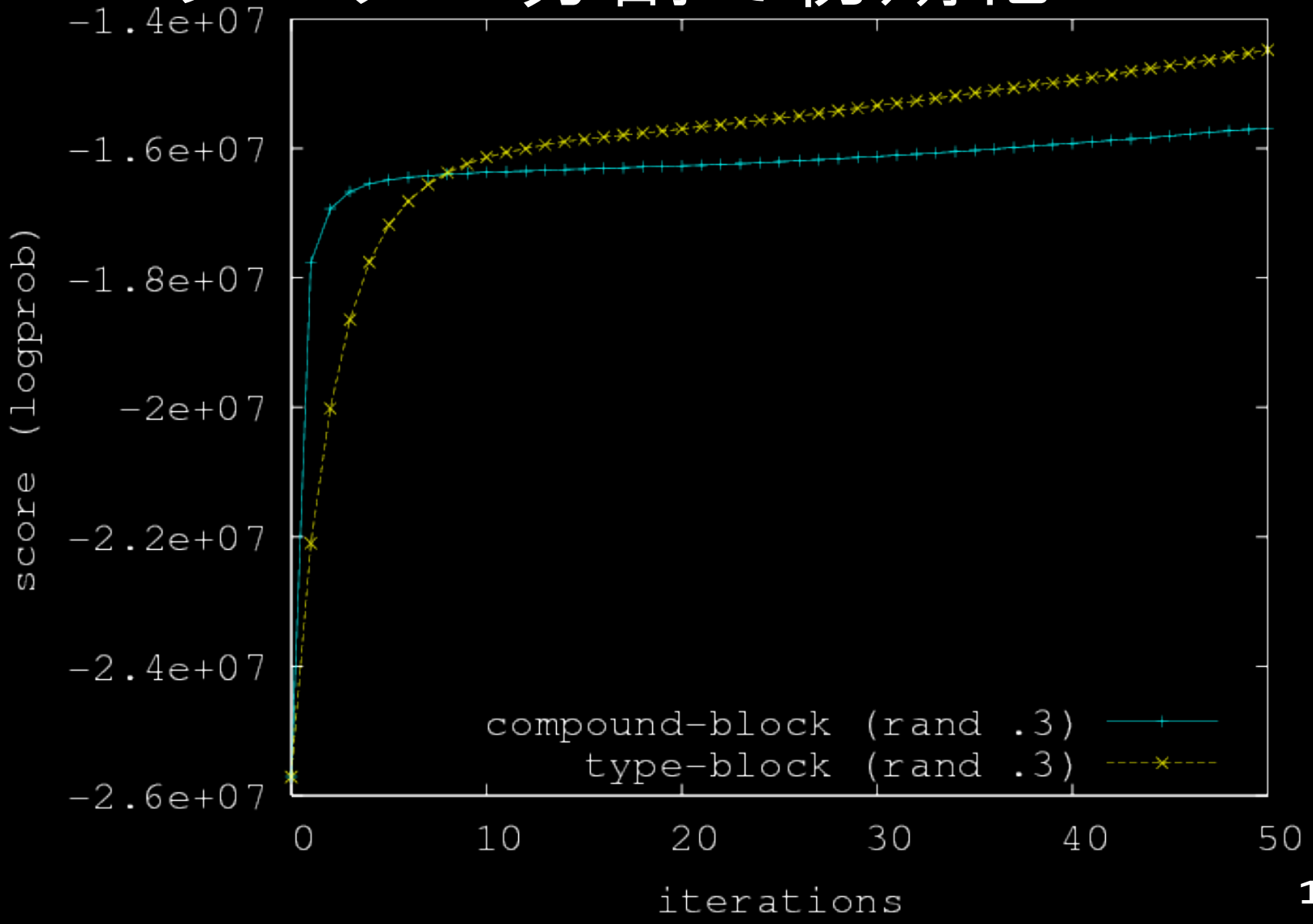
探索: scoreの変化


compound-block (nonseg-init)
type-block (nonseg-init)
-1.45e+07
-1.5e+07
-1.55e+07
-1.6e+07
-1.65e+07
-1.7e+07
score (logprob)
0
10
20
30
40
50
iterations

まとめと今後の課題

- ベイズ学習によるカタカナ複合語分割
  - 手がかり追加による若干の精度向上
  - 型別サンプリングによる中規模データへの適応
- 今後の課題
  - 外来語: 元言語の分かち書き (Kaji+, 2011)
    - トラスティ || ベル ⇔ Trusty ␣ Bell
  - 非外来語: 一般的な表記との対応付け
    - スゴイ || ケイバ ⇔ 凄い/すごい || 競馬

ランダム分割で初期化 1/2



ランダム分割で初期化 2/2