6. 初歩の言語モデル

内山将夫@NICT
mutiyama@nict.go.jp

言語モデルの紹介

言語モデルというのは，任意の文字列について，それが日本語文等である確率を付与する確率モデルである．

n-gram　言語モデルとは，単語列 $w_1, w_2, \ldots, w_i$ が与えられたときに，その後に単語 x がくる確率 $P(x|w_1, w_2, \ldots, w_i)$ を，すぐ前の $n-1$ 個の単語を条件とした確率として計算する．つまり

$$P(x|w_1, w_2, \ldots, w_i) = P(x|w_{i-n+2}, w_2, \ldots, w_i)$$

以下では，単純な

- 1-gram 言語モデル
- 2-gram 言語モデル

について説明する．

1-gram 言語モデル

- 言語モデルの役割は，与えられたテキストに確率を割当ることである．
- 良くでるテキストには高い確率を割当て，そうでないものには低い確率を割当てたい．

$$
\begin{aligned}
P(\text{テキスト}) &= P(\text{単語}1, \text{単語}2, \ldots, \text{単語}m) && (1)\\
&= P(\text{単語}1)P(\text{単語}2)P(\text{単語}3)\cdots && (2)\\
&= \prod_{i=1}^{m} P(\text{単語}i) && (3)
\end{aligned}
$$

ただし，

- m はテキスト中の単語数
- 単語 i はテキストに i 番目に出現した単語

1-gram 言語モデルは，単語の確率を計算するときに文脈を考慮しない．

1-gram 言語モデルの確率推定

- 「坊っちゃん」(夏目漱石) を例に 1-gram 言語モデルを利用した確率推定の例を示す.
- 原文は ChaSen で単語に分ける.
- 全部で 2711 文あるので，先頭の 2611 文を訓練に利用して，残りの 100 文のうち 50 文をパラメタ調整，残りの 50 文をテストに使用する
- 訓練とは確率を推定することであり，テストとは推定結果を評価することである.

  (原文) 親譲り の 無鉄砲 で 小 供 の 時 から 損 ばかり し て いる 。小学校 に 居る 時分 学校 の 二 階 から 飛び降り て 一 週間 ほど 腰 を 抜かし た 事 が ある 。なぜ そんな 無闇 を し た と 聞く 人 が ある かも 知れ ぬ 。別段 深い 理由 で も ない 。新築 の 二 階 から 首 を 出し て い たら 、同級生 の 一 人 が 冗談 に 、いくら 威張っ ても 、 そこ から 飛び降りる 事 は 出来 まい 。弱虫 や ー い 。と 囃し た から で ある 。小使 に 負ぶさっ て 帰っ て 来 た 時 、 おやじ が 大きな 眼 を し て 二 階 ぐらい か ら 飛び降り て 腰 を 抜かす 奴 が ある か と 云っ た から 、 この 次 は 抜かさ ず に 飛ん で 見せ ます と 答え た 。

頻度上位の単語100語

- 延べ語数 = 55161

、 2742 。 2362 て 2092 の 2074 は 1643 が 1630 た 1599 を 1586 に 1535 と 1503 だ 1035 で 929 ない 770 から 670 し 643 も 631 な 519 おれ 451 へ 439 か 419 う 350 ん 326 」 309 「 309 ある 279 事 277 いる 245 もの 214 云う 210 人 199 する 196 たら 190 君 182 です 175 赤 172 来 171 云っ 168 い 168 よう 166 なら 166 シャツ 164 じゃ 163 そう 158 一 147 山 嵐 142 お 140 思っ 136 何 134 この 130 ば 119 てる 119 それ 119 方 114 なっ 114 いい 114 出 113 だろ 113 時 100 なる 98 まで 97 その 93 これ 92 れ 91 学校 90 ばかり 88 清 85 見 84 なり 84 や 83 聞い 81 野 78 ら 78 生徒 77 ね 77 ます 76 顔 75 さ 75 でも 74 … 74 って 73 所 72 気 70 こんな 70 校長 69 み 68 上 67 出し 67 より 67 ―― 66 二 65 行っ 65 奴 62 もし 62 うち 62 中 60 今 60 ませ 59 もん 58 なかっ 58 ちゃ 57

確率推定：最尤推定法

「、」や「。」の確率$P(\text{、})$や$P(\text{。})$を知りたい．そうするとき，最尤推定法では，

$$P(\text{テキスト}) = \prod_{i=1}^{m} P(\text{単語}i) \tag{4}$$

$$= \prod_{\text{単語}\in\text{語彙}} P(\text{単語})^{n(\text{単語})} \tag{5}$$

が最大となるように$P(\text{単語})$を定める

- 4式では，iは1からm (テキスト長) まで動くので，単語iは，i番目の単語という意味である．
- 5式では，テキスト中の同一単語の数を$n(\text{単語})$と数えて，同じ単語はひとまとめにしたものである．

最尤推定つづき

- 語彙におけるi番目の単語の確率をp_i，頻度をn_iとし
- 語彙の大きさをlとする．

$\max \Pi_{i=1}^{l} p_i^{n_i}$なる$p_i$は，テキストの確率を最尤にする．これは，$L = \Sigma_{i=1}^{l} n_i \log(p_i)$としたときに，$\max L$とすれば良い．

課題 (20分)

$\max L$ となるときにおける p_i の値を n_i を利用して表現すること．

ラグランジェの未定乗数法を使わない回答例

まず，$\Sigma_{i=1}^{l} p_i = 1$ より $p_l = 1 - \Sigma_{i=1}^{l-1} p_i$．よって

$$L = \sum_{i=1}^{l-1} n_i \log p_i + n_l \log(1 - \sum_{i=1}^{l-1} p_i) \tag{6}$$

よって，$i = 1, 2, ..., l-1$ について

$$\frac{\partial L}{\partial p_i} = \frac{n_i}{p_i} - \frac{n_l}{1 - \Sigma_{i=1}^{l-1} p_i} = 0 \tag{7}$$

つまり

$$\frac{n_i}{p_i} = \frac{n_l}{1 - \Sigma_{i=1}^{l-1} p_i} = \frac{n_l}{p_l} \tag{8}$$

よって，$K = \frac{n_l}{p_l}$ (定数) とすると，$i = 1, ..., l$ について，

$$\frac{n_i}{p_i} = K \tag{9}$$

次に，$p_i = \frac{n_i}{K}$ だから，

$$\sum_{i=1}^{l} p_i = \sum_{i=1}^{l} \frac{n_i}{K} = 1 \tag{10}$$

よって，$K = \Sigma_{i=1}^{l} n_i$．したがって，

$$p_i = \frac{n_i}{K} = \frac{n_i}{\Sigma_{i=1}^{l} n_i} \tag{11}$$

要するに，普通に出現頻度の割合を求めると，それが最尤推定確率となるということである．

ラグランジェの未定乗数法を使う回答例

$$M = \sum_{i=1}^{l} n_i \log p_i - \lambda(\sum_{i=1}^{l} p_i - 1) \tag{12}$$

とする．$i = 1, 2, ..., l$について，

$$\frac{\partial M}{\partial p_i} = \frac{n_i}{p_i} - \lambda = 0 \tag{13}$$

より，$p_i = \frac{n_i}{\lambda}$．また，制約として，

$$\frac{\partial M}{\partial \lambda} = -(\sum_{i=1}^{l} p_i - 1) = 0 \tag{14}$$

より，

$$\sum_{i=1}^{l} p_i = \sum_{i=1}^{l} \frac{n_i}{\lambda} = 1 \tag{15}$$

よって，

$$\lambda = \sum_{i=1}^{l} n_i \tag{16}$$

よって，

$$p_i = \frac{n_i}{\lambda} = \frac{n_i}{\Sigma_{i=1}^{l} n_i} \tag{17}$$

この場合には，二つの回答例であまり複雑さはかわらないが，一般には，ラグランジェの未定乗数法を利用した方が簡単である．

問題 (10分)

テキスト中の全単語の出現頻度の和 $(\Sigma_i^l n_i)$ が 55161 で,

単語	頻度
おれ	451
は	1643
蕎麦	15
が	1630
大好き	2
で	929
ある	279

という出現頻度のときに

$$P(\text{おれ，は，蕎麦，が，大好き，で，ある}) \quad (18)$$

の確率を 1-gram 言語モデルにより計算すること.

回答

単語	頻度	確率
おれ	451	$451/55161 = 0.0081760664237414 1$
は	1643	$1643/55161 = 0.029785536882942 7$
蕎麦	15	$15/55161 = 0.000271931255778539$
が	1630	$1630/55161 = 0.029549863127934 6$
大好き	2	$2/55161 = 3.625750077047 19\text{e-}05$
で	929	$929/55161 = 0.016841609107884 2$
ある	279	$279/55161 = 0.005057921357480 83$

より，確率を全て掛けて，

$$6.043909687399 61e - 18$$

最尤推定法の問題点

- 訓練テキスト中に出現しなかった単語の確率が0となる

たとえば,「坊っちゃん」で確率推定したときには$P(\text{三四郎}) = 0$となってしまう．そのため,

$$P(\text{三四郎，は，蕎麦，が，大好き，で，ある}) = 0 \quad (19)$$

となる．これは困る.
なぜ困るかというと，言語モデルの役割は,

- ある文AとBについて，AとBのうちで，よく出現しそうな文に高い確率をつけること，つまり
- $P(A) > P(B)$か$P(A) < P(B)$か$P(A) = P(B)$かを推定することだが
- 最尤推定だと，AとBに未知語(訓練データにない語)があると,
- $P(A) = P(B) = 0$となり,
- 未知語以外がどんなに違っていても同じ確率となってしまう.

未知語への対処法

- 訓練データ中にでない単語は UNK という1つの仮想的な単語と考え
- $P(\mathrm{UNK})$を推定する.

つまり,

$$P(\text{三四郎}) = P(\text{明暗}) = P(\mathrm{UNK}) \tag{20}$$

のように，でてこない単語は，1つのクラスUNKでまとめる

- この$P(UNK)$をどう推定するか

が問題である.
最尤推定だと$P(\mathrm{UNK}) = 0$となってしまうので，別の方法を使う必要がある.

未知語を含むテキストにも確率 > 0 を割当てる方法

これにはとてもたくさんの方法がある．これらの方法は，一般に，

- スムージング

と呼ばれている．
ここでは，スムージング法の中でも比較的簡単な

- 補間法

について述べる．
ngram 言語モデルを実際に利用するには，

- SRILM

等のツールキットを使うと良い．

補間法

- 複数の確率分布の重み付き平均を確率とする方法

この場合には，最尤推定による単語wの確率を$P_{ML}(w)$とすると，これにP_{ML}よりも滑らかな確率を組合せる．そのような分布として,「坊っちゃん」に出現した5507の異なり単語にUNK を加えた5508単語(K=5508とする)が一様に出現する一様分布を考えると

$$P_U = \frac{1}{K} = \frac{1}{5508} = 0.000182 \tag{21}$$

これらを足して

$$P(w) = \lambda P_{ML}(w) + (1 - \lambda)P_U \tag{22}$$

とする．ただし，$\lambda(0 \leq \lambda \leq 1)$は，重み調整用のパラメタである．

$P(\mathrm{UNK})$がどうなるか

$$
\begin{aligned}
P(\mathrm{UNK}) &= \lambda P_{ML}(\mathrm{UNK}) + (1-\lambda)P_U && (23)\\
&= \lambda \times 0 + (1-\lambda)P_U && (24)\\
&= (1-\lambda)P_U && (25)\\
&= \frac{1-\lambda}{K} && (26)
\end{aligned}
$$

これよりテキストに出現しない単語(UNK)についても確率>0が割当てられるようになった．なお，$\lambda = 1$のときには，$P(\mathrm{UNK}) = 0$である．

問題(5分)

このように定義した確率が，確率の定義を満すことを確かめること

回答例

確率の定義は, V が UNK を含む語彙として

$$\sum_{w \in V} P(w) = 1 \tag{27}$$

が成立して，かつ，全ての単語 w について

$$0 \leq P(w) \leq 1 \tag{28}$$

が成立することである.

$$\begin{aligned}
\sum_{w \in V} P(w) &= \sum_{w \in V} \{\lambda P_{ML}(w) + (1-\lambda) P_U\} && (29)\\
&= \lambda \sum_{w \in V} P_{ML}(w) + (1-\lambda) P_U \sum_{w \in V} 1 && (30)\\
&= \lambda \cdot 1 + (1-\lambda) \frac{1}{K} K && (31)\\
&= 1 && (32)
\end{aligned}$$

また，$0 \leq \lambda \leq 1$ より

$$\begin{aligned}
\lambda &\geq 0 && (33)\\
1 - \lambda &\geq 0 && (34)
\end{aligned}$$

かつ

$$\begin{aligned}
P_{ML}(w) &\geq 0 && (35)\\
P_U &> 0 && (36)
\end{aligned}$$

より,

$$P(w) = \lambda P_{ML}(w) + (1-\lambda) P_U \geq 0 \tag{37}$$

λをどう推定するか

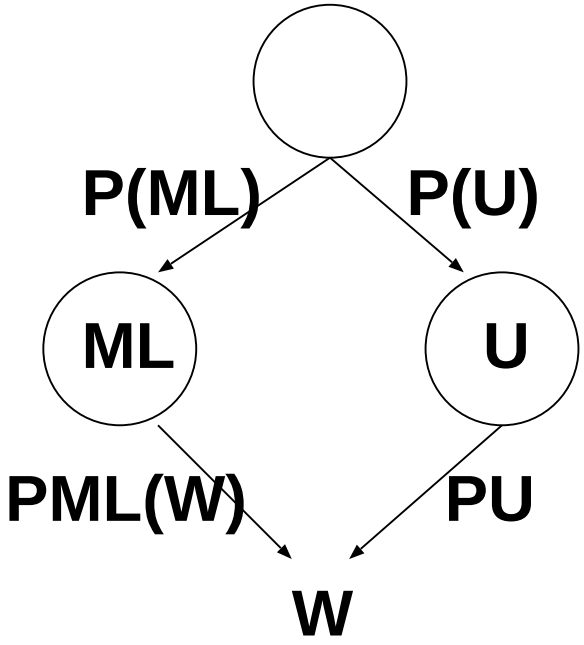
重要な点

λを調整するコーパスは，P_{ML}を推定したコーパスとは別のコーパスとすること．

理由

λの推定は最尤推定によるので，もし，P_{ML}を推定したコーパスを使うと，そこにはUNKがないので，λは 1 となってしまう．

λの推定法 (単語の生成モデル)



2つの状態 ML と U を考え，ある単語wは，MLもしくはUのどちらかからでるとすると，

$$P(w) = P_{ML}(w)P(ML) + P_U P(U) \tag{38}$$

$$P(ML) + P(U) = 1 \tag{39}$$

ところで，テキストにおけるi番目の単語t_iはMLかUのどちらかからでるので，$n(ML)$により，MLからでた単語の総頻度を表し，$n(U)$により，Uから出た単語の総頻度を表すとすると，

$$P(ML) = \frac{n(ML)}{n(ML) + n(U)} \tag{40}$$

$$P(U) = \frac{n(U)}{n(ML) + n(U)} \tag{41}$$

と推定できる．$P(ML)$がλに相当する．

ところが $n(ML)$ や $n(U)$ を直接数えることはできない．なぜなら，単語 w について，それがMLからでたかUからでたかはわからないからである．
そこで $n(ML)$ や $n(U)$ のかわりに，その期待値を使う．$n(ML)$ の期待値は

$$E(ML) = \sum_{w \in V} n(w) P(ML|w) \tag{42}$$

である．ここで，$n(w)$ は単語 w の出現頻度であり，$P(ML|w)$ は，w が与えられたときに，それがMLからでた確率である．上式は，各 w について，それが ML からどれくらいの頻度ででたかを求めて，足している．
同様に

$$E(U) = \sum_{w \in V} n(w) P(U|w) \tag{43}$$

なお，

$$P(ML|w) + P(U|w) = 1 \tag{44}$$

より

$$E(ML) + E(U) = \sum_{w \in V} n(w) = \text{全単語の頻度の和} \tag{45}$$

このとき，

$$P(ML) = \frac{E(ML)}{E(ML) + E(U)} \tag{46}$$

である．なお，$P(U) = 1 - P(ML)$ である．

よって，$P(ML|w)$を求めれば良い．ベイズの定理より

$$P(ML|w) = \frac{P(w|ML)P(ML)}{P(w)} \tag{47}$$

さて，

$P(w|ML) = P_{ML}(w) =$ 既に訓練コーパスより得られた定数 (48)

一方

$$P(ML) = \lambda \tag{49}$$

$$P(w) = \lambda P_{ML}(w) + (1-\lambda)P_U \tag{50}$$

は，今求めたい未知数λを含む．しかし，それにもかかわらず式を書くと

$$P(ML|w) = \frac{\lambda P_{ML}(w)}{\lambda P_{ML}(w) + (1-\lambda)P_U} \tag{51}$$

$$E(ML) = \sum_{w \in V} n(w)P(ML|w) \tag{52}$$

繰り返しによるλの推定

1. $\lambda = 0.5$とする
2. $$P(ML|w) = \frac{\lambda P_{ML}(w)}{\lambda P_{ML}(w) + (1-\lambda)P_U} \tag{53}$$
   $$E(ML) = \sum_{w \in V} n(w) P(ML|w) \tag{54}$$
   を計算する
3. $$\lambda = P(ML) = \frac{E(ML)}{\text{全単語の頻度の和}} \tag{55}$$
4. λが収束したら終了

この方法は，とりあえずλがわかっているものとして$E(ML)$を計算し，その結果を利用して，λを再推定するというように，少しずつλを改善する方法である．これはEM法の簡単な例である．

課題

- EM 法について調べること

EM 法は，確率統計的手法を使うときには，必須の方法ですが，EM 法自体は一般的な方法であり，それを個々の事柄に適用するためには，ここでしたように，個別の事柄にあわせて解法を開発する必要があります.
ここでは，EM 法を利用すると，未知パラメタを推定できるということを覚えておいて下さい.

例：「坊っちゃん」についてλを求める

- 2611文でP_{ML}を推定する
- 50文でλを推定する

繰り返し計算による

$\lambda, E(ML), E(U), L$

の推移をみる.

$$L = \sum_{w \in V} n(w) \log P(w) = \text{対数尤度} \tag{56}$$

$$n(w) = \text{単語} w \text{の開発データでの頻度} \tag{57}$$

$$P(w) = \lambda P_{ML}(w) + (1 - \lambda) P_u \tag{58}$$

```
# lambda=λ
# E(ML)=Pml からでた回数の期待値
# E(UNI) = Puni からでた期待値
# N=開発データの延べ単語数
# LL=対数尤度
#
ruby src/lambda.rb -n 20 botchanj/train.wfreq botchanj/test-develop.wfreq botchanj/pr
延べ単語数 = 55161
異なり単語数 = 5507
Puni = 0.000181554103122731
 1: lambda=0.7703, E(ML)=825.7380, E(UNI)=246.2620, N=1072, LL=-6693.4745
 2: lambda=0.8454, E(ML)=906.2771, E(UNI)=165.7229, N=1072, LL=-6477.1231
 3: lambda=0.8693, E(ML)=931.8799, E(UNI)=140.1201, N=1072, LL=-6451.9807
 4: lambda=0.8777, E(ML)=940.9064, E(UNI)=131.0936, N=1072, LL=-6448.6396
 5: lambda=0.8808, E(ML)=944.2322, E(UNI)=127.7678, N=1072, LL=-6448.1713
 6: lambda=0.8820, E(ML)=945.4790, E(UNI)=126.5210, N=1072, LL=-6448.1047
 7: lambda=0.8824, E(ML)=945.9495, E(UNI)=126.0505, N=1072, LL=-6448.0951
 8: lambda=0.8826, E(ML)=946.1275, E(UNI)=125.8725, N=1072, LL=-6448.0937
 9: lambda=0.8826, E(ML)=946.1949, E(UNI)=125.8051, N=1072, LL=-6448.0936
10: lambda=0.8827, E(ML)=946.2205, E(UNI)=125.7795, N=1072, LL=-6448.0935
11: lambda=0.8827, E(ML)=946.2301, E(UNI)=125.7699, N=1072, LL=-6448.0935
12: lambda=0.8827, E(ML)=946.2338, E(UNI)=125.7662, N=1072, LL=-6448.0935
13: lambda=0.8827, E(ML)=946.2352, E(UNI)=125.7648, N=1072, LL=-6448.0935
14: lambda=0.8827, E(ML)=946.2357, E(UNI)=125.7643, N=1072, LL=-6448.0935
15: lambda=0.8827, E(ML)=946.2359, E(UNI)=125.7641, N=1072, LL=-6448.0935
16: lambda=0.8827, E(ML)=946.2360, E(UNI)=125.7640, N=1072, LL=-6448.0935
17: lambda=0.8827, E(ML)=946.2360, E(UNI)=125.7640, N=1072, LL=-6448.0935
18: lambda=0.8827, E(ML)=946.2360, E(UNI)=125.7640, N=1072, LL=-6448.0935
19: lambda=0.8827, E(ML)=946.2360, E(UNI)=125.7640, N=1072, LL=-6448.0935
20: lambda=0.8827, E(ML)=946.2360, E(UNI)=125.7640, N=1072, LL=-6448.0935
```

確率の値

	最尤推定値	一様分布との補間
おれ	0.00817606642374141	0.00723817319206339
は	0.0297855368829427	0.0263124826219221
蕎麦	0.00027193l255778539	0.00026132846771908 5
が	0.029549863l279346	0.026104457435l871
大好き	3.6257500770471 9e-05	5.33032809840486e-05
で	0.0168416091078842	0.0148870992889363
ある	0.0050579213574808 3	0.0044858399521 8444
全積	6.043909687399 61e-18	4.624874917451 99e-18

UNKに確率を与えた分だけ，一様分布との補間の方が，少しずつ確率が少なくなっている．しかし，重みつき平均の結果として,「大好き」については，少し増えている．なお，$P(\mathrm{UNK}) = 2.1299406101735 3e-05$である．

2-gram 言語モデルの導入

1-gram 言語モデルでは，単語の順番と確率が無関係なので，言語のモデル化には不十分である．

P(おれ, は, 蕎麦, が, 大好き, で, ある)

$= P$(おれ)P(は)P(蕎麦)P(が)P(大好き)P(で)P(ある)

$= P$(蕎麦)P(は)P(大好き)P(が)P(で)P(ある)P(おれ)

$= P$(蕎麦, は, 大好き, が, で, ある, おれ)

2-gram 言語モデルでは，一つ前の単語を見ることにより，並び順を少し考慮する．

$$
\begin{aligned}
& P(\text{おれ}, \text{は}, \text{蕎麦}, \text{が}, \text{大好き}, \text{で}, \text{ある}) \\
&= P(\text{おれ})P(\text{は}\,|\,\text{おれ})P(\text{蕎麦}\,|\,\text{は})P(\text{が}\,|\,\text{蕎麦}) \\
&\quad P(\text{大好き}\,|\,\text{が})P(\text{で}\,|\,\text{大好き})P(\text{ある}\,|\,\text{で}) \\
&\neq P(\text{蕎麦}, \text{は}, \text{大好き}, \text{が}, \text{で}, \text{ある}, \text{おれ}) \\
&= P(\text{蕎麦})P(\text{は}\,|\,\text{蕎麦})P(\text{大好き}\,|\,\text{は})P(\text{が}\,|\,\text{大好き}) \\
&\quad P(\text{で}\,|\,\text{が})P(\text{ある}\,|\,\text{で})P(\text{おれ}|\,\text{ある})
\end{aligned}
$$

2-gram 言語モデルの求め方の例

$$P_{ML}(\text{単語}2|\,\text{単語}1) = \frac{\text{単語}1\text{の後に単語}2\text{が続く頻度}}{\text{単語}1\text{の頻度}} \tag{59}$$

例

「蕎麦」の頻度が15のとき

単語1	単語2	頻度	P(単語2|単語1)
蕎麦	屋	5	0.33
	を	4	0.27
	と	2	0.13
	粉	1	0.067
	も	1	0.067
	の	1	0.067
	が	1	0.067

最尤推定における数値例

コーパス全体での頻度 = 55161

w	v	$n(w)$	$P_{ML}(w)$	$n(v)$	$n(vw)$	$P_{ML}(w\|v)$
おれ	*	451	0.00817607			
は	おれ	1643	0.02978554	451	164	0.36363636
蕎麦	は	15	0.00027193	1643	1	0.00060864
が	蕎麦	1630	0.02954986	15	1	0.06666667
大好き	が	2	0.00003626	1630	1	0.00061350
で	大好き	929	0.01684161	2	1	0.50000000
ある	で	279	0.00505792	929	78	0.08396125

w	v	$n(w)$	$P_{ML}(w)$	$n(v)$	$n(vw)$	$P_{ML}(w\|v)$
蕎麦	*	15	0.00027193			
は	蕎麦	1643	0.02978554	15	0	0.00000000
大好き	は	2	0.00003626	1643	1	0.00060864
が	大好き	1630	0.02954986	2	0	0.00000000
で	が	929	0.01684161	1630	1	0.00061350
ある	で	279	0.00505792	929	78	0.08396125
おれ	ある	451	0.00817607	279	0	0.00000000

- 文の生成確率は，1-gram モデルでは等確率だが，2-gram モデルでは確率が異なる
- 確率 0 となる場合があるのは困る．

2-gram 言語モデルにおける補間

$$P(w|v) = \lambda_2 P_{ML}(w|v) + \lambda_1 P_{ML}(w) + \lambda_0 P_U \tag{60}$$

$$P_{ML}(w|v) = \frac{n(vw)}{n(v)} \tag{61}$$

$$P_{ML}(w) = \frac{n(w)}{\Sigma_v\, n(v)} \tag{62}$$

$$P_U = \frac{1}{|V| + 1} \tag{63}$$

$$\lambda_2, \lambda_1, \lambda_0 \geq 0 \tag{64}$$

$$\lambda_2 + \lambda_1 + \lambda_0 = 1 \tag{65}$$

問題 (15分)

テキスト $w_1, w_2, \ldots, w_N$ が与えられているとする．ただし，w_i はテキストの i 番目の単語とする．このとき，λ_0, λ_1, λ_2 の推定法を示すこと．
1-gram 言語モデルのときと同様に,

1. λ_i の初期値を設定する
2. 各 λ_i に相当する状態における期待頻度 E_i を求める．
3. E_i を利用して，λ_i を求める
4. 収束したら終了

というようにする．

λ_iの計算法

1-gram 言語モデルのときには，語彙の上で $E(ML)$ や $E(U)$ を計算したが，今回は，テキストの上で E_i を計算する．1-gram 言語モデルにおける $n(w)$ は，今回は，テキストの上で単語を動かすので，複数回出現する単語はその回数分だけ重複されてカウントされるので，1-gram 言語モデルで $n(w)$ を掛けるのと同様な効果が得られる．

1. $\lambda_2 = \lambda_1 = \lambda_0 = \frac{1}{3}$ とする
2. $$E_2 = \sum_{i=2}^{N} \frac{\lambda_2 P_{ML}(w_i|w_{i-1})}{\lambda_2 P_{ML}(w_i|w_{i-1}) + \lambda_1 P_{ML}(w_i) + \lambda_0 P_U}$$
$$E_1 = \sum_{i=2}^{N} \frac{\lambda_1 P_{ML}(w_i)}{\lambda_2 P_{ML}(w_i|w_{i-1}) + \lambda_1 P_{ML}(w_i) + \lambda_0 P_U}$$
$$E_0 = \sum_{i=2}^{N} \frac{\lambda_0 P_U}{\lambda_2 P_{ML}(w_i|w_{i-1}) + \lambda_1 P_{ML}(w_i) + \lambda_0 P_U}$$
3. $\lambda_i = \frac{E_i}{E_0+E_1+E_2}$ for $(i = 0, 1, 2)$
4. goto 2 もしくは収束したら終了

- まず初期値を設定する
- 各 w_i について，それぞれが3つの確率分布のどれからでたかの期待度数を求め，その和を各確率分布の期待度数とする．
- 期待度数の比として λ_i を求める
- 収束するまで繰り返す

数値例1

$\lambda_0 = \lambda_1 = \lambda_2 = \frac{1}{3}$のときの各単語の確率と期待度数

おれ, は, 蕎麦, が, 大好き, で, ある

	pml(w|v)	pml(w)	pu	e(w|v)	e(w)	e0
は	0.363636	0.029786	0.000182	0.923865	0.075674	0.000461
蕎麦	0.000609	0.000272	0.000182	0.573041	0.256025	0.170934
が	0.066667	0.029550	0.000182	0.691577	0.306540	0.001883
大好き	0.000613	0.000036	0.000182	0.737989	0.043615	0.218396
で	0.500000	0.016842	0.000182	0.967075	0.032574	0.000351
ある	0.083961	0.005058	0.000182	0.941262	0.056703	0.002035
sum				4.834808	0.771131	0.394061
lambda				0.805801	0.128522	0.065677

2-gramまで考慮した状態$e(w|v)$における期待頻度が他よりも大きいことがわかる.

数値例2

「坊っちゃん」のパラメタ調整用データ50文を利用してλ_iを求める.

- 2611文は，$P_U, P_{ML}(w), P_{ML}(w|v)$の推定に用いる
- 50文は，λ_iの推定に用いる

	λ2	λ1	λ0	E2	E1	E0	尤度
1:	0.5350	0.2985	0.1665	546.77	305.05	170.18	-5217.63371050
2:	0.5749	0.2842	0.1409	587.57	290.48	143.95	-5090.05722072
3:	0.5813	0.2823	0.1364	594.12	288.50	139.38	-5085.40951867
4:	0.5823	0.2823	0.1355	595.06	288.48	138.46	-5085.27277246
5:	0.5824	0.2824	0.1353	595.16	288.61	138.23	-5085.26807026
6:	0.5823	0.2825	0.1352	595.16	288.69	138.15	-5085.26776245
7:	0.5823	0.2825	0.1352	595.15	288.73	138.13	-5085.26772140
8:	0.5823	0.2825	0.1351	595.14	288.74	138.12	-5085.26771436
9:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771310
10:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771287
11:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771283
12:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771282
13:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771282
14:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771282
15:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771282
16:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771282
17:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771282
18:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771282
19:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771282
20:	0.5823	0.2825	0.1351	595.14	288.75	138.11	-5085.26771282

尤度 $=$ $\Sigma_i \log p(w_i|w_{i-1})$が単調に増加していることがわかる.

数値例3

個別の文について各単語の確率を計算し，それを掛けて文の確率を求める．

$P(\text{おれは蕎麦が大好きである}) = 6.456 \times 10^{-14}$

w	v	$P_{ML}(w\|v)$	$P_{ML}(w)$	P_U	P
おれ	*	0.000000	0.008176	0.000182	0.002334
は	おれ	0.363636	0.029786	0.000182	0.220184
蕎麦	は	0.000609	0.000272	0.000182	0.000456
が	蕎麦	0.066667	0.029550	0.000182	0.047192
大好き	が	0.000613	0.000036	0.000182	0.000392
で	大好き	0.500000	0.016842	0.000182	0.295932
ある	で	0.083961	0.005058	0.000182	0.050344

「おれ」「は」や「大好き」「で」は，高い確率で出現する．

$P(\text{蕎麦は大好きがであるおれ}) = 1.683 \times 10^{-18}$

w	v	$P_{ML}(w\|v)$	$P_{ML}(w)$	P_U	P
蕎麦	*	0.000000	0.000272	0.000182	0.000101
は	蕎麦	0.000000	0.029786	0.000182	0.008439
大好き	は	0.000609	0.000036	0.000182	0.000389
が	大好き	0.000000	0.029550	0.000182	0.008372
で	が	0.000613	0.016842	0.000182	0.005140
ある	で	0.083961	0.005058	0.000182	0.050344
おれ	ある	0.000000	0.008176	0.000182	0.002334

- どの単語の確率も低い
- $P_{ML}(w|v) = 0$でも，$P > 0$となっている．

まとめ

- n-gram 言語モデルは，任意の文字列についての日本語らしさ (あるいは他の言語のそれらしさ) の確率を求める
- 確率の推定には，最尤推定ではなくスムージングが必要である

もっと複雑な言語モデル

- 3-gram, 4-gram, 5-gram, ... 言語モデル
- トピックを考慮した言語モデル
- 構文を利用した言語モデル

言語モデルの利用例

- 機械翻訳
- 音声認識
- 仮名漢字変換