架空世界創作のための言語モデル

～統計と計算を言語に組み込む文明的創作のあり方を模索する～

@nymwa

2020/**/**

目次

1 はじめに 2

2 言語モデルと確率 3
2.1 ちょっと確率統計の復習 4
2.1.1 確率・結果・事象 4
2.1.2 確率変数・確率分布・同時確率 5
2.1.3 条件付き確率と周辺確率 5
2.1.4 独立 5
2.1.5 期待値 5
2.2 1-gram 言語モデル 6
2.3 言語モデルの評価 6
2.4 n-gram 言語モデル 6
2.5 Kneser-Ney スムージング 6

3 トキポナ言語モデルを作る 7
3.1 トキポナとは 7
3.2 さまざまな言語モデルで学習させる 7
3.3 ドメイン適応 7
3.4 長さ正規化 7

4 造語支援システムを作る 8
4.1 言語モデルによる生成 8
4.2 ビーム探索 8

5 綴り誤り訂正システムを作る 9
5.1 雑音チャネルモデル 9

6 おわりに 10

A python 3.8 のインストール 11

B python 仮想環境のインストール 12

C 本書サンプルコードのダウンロードと実行 13

第 1 章　はじめに

私は架空世界における言語創作，一般に人工言語と呼ばれているものと，計算機で言語を扱う学問，理学的には計算言語学，工学的には自然言語処理と呼ばれているものが好きです．そのような立場として，架空世界における言語創作に計算言語学的な知見が応用できる可能性があるのかと考えることがあります．

当然指輪物語が書かれた時代に計算機はなかったですし，計算機は言語創作においては必ず必要なものではないかもしれません．しかし，計算言語学の知見はかな漢字変換や綴り誤り訂正，機械翻訳や対話システムなど身の回りにある多くのツールに活かされており，その成果を人工言語へ適用すれば，語彙や例文も微々たる架空言語に対してそのような便利なツールが製作できるでしょう．言語創作に計算機を簡単に応用できるようなツールが多くの人によって作られれば，創作者だけでなく，学習者にとっても利点があるものと信じています．

一方で，計算言語学や自然言語処理の発展も，そのほとんどが高々ここ半世紀のうちに成し遂げられたものに過ぎず，昨今の機械翻訳や対話システムが人間から見て不自然な挙動をする事例からも明らかなように，現時点では計算機で言語を扱うことは非常に難しく，また，高い精度で行おうとすれば大規模な計算機資源が必要になってしまうこともあります．

そのため，今回は CPU が 1 つあればできるような軽量な計算で実現できる古典的な手法のみを用いて計算と統計によって言語創作に有益なツールが作れるのかどうかを模索することに焦点を置いています．特に，簡単に実装・実用が可能な言語モデルと呼ばれるものを用いた応用について検討していきます．

この文書によって人工言語界隈に前よりちょっとだけ計算言語学が普及していい感じなツールが生まれてくれば嬉しいです．

本書で使用するプログラムはすべて python 3.8 で書かれます．実行環境は標準的な UNIX/linux 環境を想定しています．なんか動かなかったりよくわからない場合は twitter:@nymwa に文句を言ってください．頑張って答えます．

第 2 章　言語モデルと確率

以下の 2 文を見比べてみてください.

・色を着けてニスを塗った.

・色を着けテニスを塗った.

最初の文が自然な文であるのに対し，2 番目の文は意味をなさない不自然な文となっています．この 2 文は日本語の話者であればどちらが自然か，不自然かは容易に見分けられます.「テニス」と「塗る」はふつう共起しないことからも，後者の文が不自然なことが説明できます.

しかし，かな漢字変換システムでは後者が最初に候補として示されることもあるかもしれません．これは計算機にとっては人間にとっての文の自然さ・不自然さを理解することが容易ではないためです．計算機と人間の構造が違っているのだからこれはしょうがないことではあるのですが，裏を返せば，計算機に文の自然さを判定させられるようになれば，かな漢字変換や綴り誤り訂正システムなどの便利なソフトウェアが作れるということでもあります.

ここで，理想的に世の中のすべての文に対して，その文が出現する確率を考えることにします．例えば，「色を着けてニスを塗った.」の確率は

$$P(\text{色を着けてニスを塗った．})$$

と書けます．自然な文や流暢な文は，不自然な文やぎこちない文よりも世の中のすべての文全体での出現頻度が高そうです．すると，「色を着けてニスを塗った.」は「色を着けテニスを塗った.」よりも自然な文なので，

$$P(\text{色を着けてニスを塗った．}) > P(\text{色を着けテニスを塗った．})$$

となるはずです．このようにすれば，確率の計算によって自然な文と不自然な文を識別できそうです.

文を確率変数とする確率分布のことを言語モデルと言います．言語モデルはテキスト中によく出てくる文に高い確率を割り当て，そうでない文に対しては低い確率を割り当てる必要があります．この章では言語モデルをどのように設計すればいいかについて，基礎的な事項を説明していきます.

2.1 ちょっと確率統計の復習

言語モデルを理解するためには確率と統計の基本的な知識が必要です．用語の使い方を明確にするためにも，最初に確率と統計の基礎的な事項について説明します．

2.1.1 確率・結果・事象

定義 1 (試行と結果). 実験や観測によって偶然に決まる事柄を結果と言い，その結果が偶然に決まった実験や観測のことを試行と言います．

サイコロを振る行為によって出た目はその結果であり，出た目は偶然によって決まるので，サイコロを振る行為は試行と言えます．言語モデルではあるひとつの文が観測されることを試行とみなし，その文を結果とします．

定義 2 (標本点と標本空間). 一回の試行の結果として起こりうるものを標本点と言い，すべての標本点からなる集合を標本空間と言います．標本空間はギリシャ文字 Ω で表します．

例えば，6 面のサイコロを振って出た目を結果とする場合，標本点は $1, 2, 3, 4, 5, 6$ のいずれかで，標本空間 $\Omega = \{1, 2, 3, 4, 5, 6\}$ です．言語モデルの場合は標本点は文なので，標本空間は可能なすべての文です．言語モデルの標本空間は，自然言語では無限集合になるはずです．多くの人工言語でも無限集合になるはずだと思います [1]．

確率は標本空間のそれぞれの標本点に割り当てられる実数値です．この値は標本空間全体での起こりやすさを 1 とした場合のその標本点の起こりやすさを表します．6 面のサイコロを振って 1 が出る確率は，目の出やすさに偏りがない場合 6 回に 1 回ぐらい 1 が出るため，$\frac{1}{6}$ です．

標本点の確率は以下の確率の公理を満たすものとします．

公理 3 (確率の公理). 標本空間 Ω のそれぞれの標本点 x の確率 $P(x)$ は以下の制約を満たします．

1. すべての $x \in \Omega$ に対して，$0 \leq P(x) \leq 1$ です．
2. 標本空間のすべての標本点の確率を足し合わせた和は 1 です．すなわち，$\sum_{x \in \Omega} P(x) = 1$.

[1] 節の再帰ができる言語ではいくらでも長い文が作れ，その標本空間は無限集合になります．詳しくは”Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo.” で検索してください．また，文の標本空間が有限集合となるためには節の再帰ができないことが必要です．

標本点の確率しか求められないと不便なので，標本点の集合の確率も定義します．

定義 4 (事象). 標本空間の部分集合のことを事象といいます [2]．

6 面サイコロを降って 2 以下の目が出る，偶数の目が出るなどが事象の例です．

事象の E 確率は

$$P(E) = \sum_{x \in E} P(x)$$

です． 6 面サイコロを降って 2 以下の目が出る事象の確率は $\frac{1}{3}$ です．

2.1.2 確率変数・確率分布・同時確率

定義 5 (確率変数). hoge

定義 6 (確率分布). hoge

定義 7 (同時確率). hoge

2.1.3 条件付き確率と周辺確率

定義 8 (周辺確率). hoge

定義 9 (条件付き確率). hoge

2.1.4 独立

定義 10 (独立性). hoge

2.1.5 期待値

定義 11 (確率変数の期待値). hoge

[2]「え，文の集合って無限集合だからすべての部分集合をそのまま事象とするのはまずいのでは?」と言われたら多分そうなんだけど，文が標本点になってて文に確率割り振らなきゃいけないし，そんな深いこと考えてたら書き終わらないので許して．雰囲気で確率をやっている，ダメ．

2.2　1-gram 言語モデル

2.3　言語モデルの評価

2.4　n-gram 言語モデル

2.5　Kneser-Ney スムージング

第 3 章　トキポナ言語モデルを作る

3.1　トキポナとは

3.2　さまざまな言語モデルで学習させる

3.3　ドメイン適応

3.4　長さ正規化

第 4 章　造語支援システムを作る

4.1　言語モデルによる生成

4.2　ビーム探索

第 **5** 章　綴り誤り訂正システムを作る

5.1　雑音チャネルモデル

第 6 章　おわりに

第 A 章　python 3.8 のインストール

第 B 章　python 仮想環境のインストール

第 C 章　本書サンプルコードのダウンロードと実行