

Perlで入門テキストマイニング

たつを

2007.10.1

Shibuya Perl Mongers

テキストマイニング(1)

- 評判情報。ポジティブ、ネガティブ
- プロフィール。ブロガーの性別、年齢、地域
- そのページに関連した広告とか。
- 関連語。
- もしかして〇〇？
- クラスタリング。グルーピング。

テキストマイニング(2)

- その目的
 - 意外な関係を探し出す！
 - 当たり前前の関係を大量に自動で！
- 後者で

関連語を自動生成

- ヤフー
 - グーグル、楽天、六本木、ライブドア、...
- 文教堂
 - 書店、川崎、有隣堂、株主優待、...
- Perl
 - Ruby, PHP, dankogai, miyagawa, ...

関連語の「関連」って？

- 同じドキュメントにいっしょに現れる単語は関連してそうよね。
 - → 共起 (co-occurrence)
- 例えば、「渋谷」「新宿」

共起のワナ

- 例えば、対象ドキュメント＝Webページ、のとき
- 「トップ」や「ホーム」や「サイト」や「情報」や「メニュー」のようないろんなWebページにまんべんなく現れそうな単語には注意
- 例えば、「渋谷」という単語と一番一緒に現れる単語は「情報」
 - Ref. Web単語共 <http://yapi.ta2o.net/tangokyouki/>
- 「渋谷」と「情報」は関連する語と言えるか？
 - →微妙

単単語の出現頻度を 考慮する必要がある

- 今回「関連度」として使用するのは
 - シンプソン係数

$$\frac{|X \cap Y|}{\min(|X|, |Y|)}$$

- 例:「パンダ」と「ペンギン」(by Yahoo!検索API)
 - パンダの単体出現数 $|X|$ は1130万ドキュメント、
ペンギンの単体出現数 $|Y|$ は860万、
パンダとペンギンの同時出現数 $|X \cap Y|$ は141万、
でしたので、シンプソン係数は約0.16になります。
 - Ref. リアルとWebのネットワーク分析: 先端研ブログ - CNET Japan <http://blog.japan.cnet.com/sentan/archives/002672.html>

やってみよう！

手元にあるドキュメントを対象に
シン普森係数を用いて
関連語DBを作る

手順

1. ドキュメントごとに形態素解析し名詞だけ取り出す
2. 頻度をカウントし、シンプソン係数を計算
3. 仕上げ

1. 形態素解析

- 形態素解析とは？
 - → 文章を単語に分ける
 - → 文章_(名詞)/を_(助詞)/単語_(名詞)/に_(助詞)/分ける_(動詞)
- 何を使う？
 - → Yahoo! の日本語形態素解析API

Yahoo!デベロッパーネットワーク

[トップ](#) > [テキスト解析](#) > [日本語形態素解析](#)

Webサービス
Yahoo!検索
Yahoo!カテゴリ
Yahoo!オークション
Yahoo!ミュージック
Yahoo!地図情報
テキスト解析 ▶ 日本語形態素解析
Yahoo!家電ナビ
Yahoo!ウィジェット
Yahoo!ニュース
RSS配信
検索パラメータ仕様
Yahoo!検索(ウェブ検索)

メニュー
アプリケーションIDの登録
アプリケーションIDの管理
はじめに
よくある質問
ドキュメント
SDKダウンロード
コミュニティー
クレジット表示

日本語形態素解析Webサービス

Version 1

日本語文を形態素に分割し、品詞、読みがなの付与、統計情報を取得できる機能を提供します。

リクエストURL

<http://api.jlp.yahoo.co.jp/MAService/V1/parse>

リクエストパラメータ

「[RESTリクエストの構築](#)」をご参照ください。

パラメータ	値	説明
appid(必須)	string	アプリケーションID。詳細は こちら をご覧ください。
sentence(必須)	string	解析対象のテキストです。
results(必須)	string: <i>ma uniq</i>	解析結果の種類をコンマで区切って指定します。 <ul style="list-style-type: none"> “ma”: 形態素解析の結果を <i>ma_result</i> に返します。 “uniq”: 出現頻度情報を <i>uniq_result</i> に返します。 無指定の場合は “ma” になります。
response	string: <i>surface, reading_pos, baseform, feature</i>	<i>ma_response, uniq_response</i> のデフォルト設定です。 word に返される形態素情報をコンマで区切って指定します。 無指定の場合は “surface,reading,pos” になります。
filter	string	<i>ma_filter, uniq_filter</i> のデフォルト設定です。解析結果として出力する品詞番号を “ ” で区切って指定します。 filterに指定可能な品詞番号: <ul style="list-style-type: none"> 1: 形容詞 2: 形容動詞 3: 感動詞 4: 副詞

庭には二羽ニワトリがいる。

http://api.jlp.yahoo.co.jp/MAService/V1/parse?
appid=YahooDemo&
results=uniq&
uniq_filter=9&
sentence=%E5%BA%AD%E3%81%AB
%E3%81%AF%E4%BA%8C%E7%BE%BD
%E3%83%8B%E3%83%AF%E3%83%88
%E3%83%AA%E3%81%8C%E3%81%84
%E3%82%8B%E3%80%82
(URL)



```
<?xml version="1.0" encoding="UTF-8" ?>
- <ResultSet xmlns:xsi="http://www.w3.org
  xsi:schemaLocation="urn:yahoo:jp:jlp ht
- <uniq_result>
  <total_count>9</total_count>
  <filtered_count>3</filtered_count>
  - <word_list>
    - <word>
      <count>1</count>
      <surface>ニワトリ</surface>
      <reading />
      <pos>名詞</pos>
    </word>
  - <word>
      <count>1</count>
      <surface>二</surface>
      <reading />
      <pos>名詞</pos>
    </word>
  - <word>
      <count>1</count>
      <surface>庭</surface>
      <reading />
      <pos>名詞</pos>
    </word>
  </word_list>
</uniq_result>
</ResultSet>
```

入力

- 1行1エントリの utf-8 テキストファイル

text2mor.pl

```
#!/usr/bin/perl
use strict;
use warnings;
use URI::Escape;
use LWP::Simple;
use XML::Simple;
use utf8;
binmode STDOUT, ":utf8";

while (<>) {
    chomp;
    next if /^¥s*$/;
    my $r_ref = webma({str => $_});
    print join(",",
                grep !/[a-z0-9,]/i,
                map {$_->{surface}}
                @{$r_ref->{uniq_result}->{word_list}->{word}}
            ), "¥n";
}

sub webma {
    my ($args_ref) = @_;
    my $str = $args_ref->{str};
    my $str_ec = URI::Escape::uri_escape($str);
    my $url= "http://api.jlp.yahoo.co.jp/MAService/V1/parse"
        . "?appid=YahooDemo"
        . "&results=uniq"
        . "&uniq_filter=9"
        . "&sentence=$str_ec";
    my $page = get($url);
    return {} unless $page;
    my $xmlsimple = XML::Simple->new(ForceArray => [ 'word' ]);
    my $xml = $xmlsimple->XMLin($page);
    return $xml || {};
}
```

```
% cat a.txt
庭には二羽ニワトリがいる。
今日は庭で休暇ですよ。
今日はニワトリ休暇ですよ。
ニワトリは庭で休暇です。
今日は焼き鳥だ。
% ./text2mor.pl a.txt > a.csv
% cat a.csv
ニワトリ,二,庭
今日,休暇,庭
ニワトリ,今日,休暇
ニワトリ,休暇,庭
今日,焼き鳥
```

計算

- 入力: 1行1エントリの単語CSVファイル
- シンプソン係数を計算
- 各単語ペアのシンプソン係数を出力

mor2pair.pl

```
#!/usr/bin/perl
use strict;
use warnings;

my %freq_pair;
my %freq_word;

while (<>) {
    chomp;
    next if /^%s*$/;
    my @ws = sort split(",", $_);
    for (my $i = 0; $i < @ws; $i++) {
        $freq_word{$ws[$i]}++;
        for (my $j = $i + 1; $j < @ws; $j++) {
            $freq_pair{$ws[$i]."%t".$ws[$j]}++;
        }
    }
}

my %simp;
foreach my $p (keys %freq_pair) {
    next if $freq_pair{$p} == 1;
    next unless $p =~ /^(^[%t]+)%t(^[%t]+)$/;
    my $min = (sort($freq_word{$1}, $freq_word{$2}))[0];
    next if $simpson < 0.1;
    $simp{$p} = $simpson;
}

foreach my $p (sort {$simp{$b} <=> $simp{$a}} keys %simp) {
    print "$p%t$simp{$p}%n";
}
```

```
% cat a.txt
庭には二羽ニワトリがいる。
今日は庭で休暇ですよ。
今日はニワトリ休暇ですよ。
ニワトリは庭で休暇です。
今日は焼き鳥だ。
% ./text2mor.pl a.txt > a.csv
% cat a.csv
ニワトリ,ニ,庭
今日,休暇,庭
ニワトリ,今日,休暇
ニワトリ,休暇,庭
今日,焼き鳥
% ./mor2pair.pl a.csv > a.pair
% cat a.pair
ニワトリ      休暇      0.6666666666666667
ニワトリ      庭        0.6666666666666667
今日      休暇      0.6666666666666667
休暇      庭        0.6666666666666667
```

仕上げ

- word → word, word, word, ...
- というフォーマットにする。

```
#!/usr/bin/perl
use strict;
use warnings;
```

pair2reldb.pl


```
my %rel_word;
```

```
while (<>) {
    chomp;
    my @cols = split("¥t", $_);
    next unless @cols == 3;
    $rel_word{$cols[0]}{$cols[1]} = $cols[2];
    $rel_word{$cols[1]}{$cols[0]} = $cols[2];
}
```

```
foreach my $w (keys %rel_word) {
    my $wr = $rel_word{$w};
    print "$w¥t"
        .join(",",
            map {"$_:$wr->{$_}"}
            sort {$wr->{$b} <=> $wr->{$a}} keys %$wr
        ), "¥n";
}
```

```
% cat a.txt
庭には二羽ニワトリがいる。
今日は庭で休暇ですよ。
今日はニワトリ休暇ですよ。
ニワトリは庭で休暇です。
今日は焼き鳥だ。
% ./text2mor.pl a.txt > a.csv
% cat a.csv
ニワトリ,ニ,庭
今日,休暇,庭
ニワトリ,今日,休暇
ニワトリ,休暇,庭
今日,焼き鳥
% ./mor2pair.pl a.csv > a.pair
% cat a.pair
ニワトリ      休暇      0.6666666666666667
ニワトリ      庭        0.6666666666666667
今日      休暇      0.6666666666666667
休暇      庭        0.6666666666666667
% ./pair2reldb.pl a.pair
庭      ニワトリ:0.6666666666666667,休暇:0.6666666666666667
今日      休暇:0.6666666666666667
ニワトリ      庭:0.6666666666666667,休暇:0.6666666666666667
休暇      庭:0.6666666666666667,今日:0.6666666666666667,ニワトリ:0.6666
```

実例！

- 対象ドキュメント
 - 「たつをの ChangeLog」7年分の記事(9000件)
たつをの 
ChangeLog
 - このブログの記述をベースとしているので、かたよりがある。
 - 関連語というよりも、たつを的関連語
- テキストデータ
 - chalow → cl.itemlist を nkf -w
 - 他のブログツール → ...

- スパム 送受信:0.5,送信者:0.5,悪:0.4150943,フレッシュネス:0.4,
- タマちゃん アザラシ:0.6666666,多摩川:0.1176470
- スタウト ピルスナー:1,ビール:1,ギネス:0.75,スーパー:0.5,ドラフト:0.5,瓶:0.15384
- 直也 伊藤:1,達彦:0.7,宮川:0.7,氏:0.4,本:0.4,はてな:0.4,会場:0.4,
- 新宿 都庁:1,滝沢:1,西新宿:1,中村屋:1,ハイチ:1,タイムズ:0.5,
- 銀座 シャンネル:1,伊東:0.545454,アップル:0.5,写真展:0.230769,
- ポン酢 夕食:0.75,野菜:0.75,シシトウ:0.6666667,ピーマン:0.5,
- ラーメン屋 新福:1,菜館:1,滞在中:1,夫婦:0.5,梶ヶ谷:0.5,家系:0.5,醤油:0.27906,
- サックス フルート:1,人:0.25,ベース:0.25,トリオ:0.1666666,ギター:0.1333333,
- にんにく 田子:1,半熟:0.666666,スタミナ:0.666666,ひき肉:0.4,唐辛子:0.33333
- 引越し くろねこ:1,物品:1,旧居:1,箱詰め:0.8181818,一足先:0.666666
- 表現 多様性:1,独学:1,正規:0.892857142857143,詳説:0.75,聞き手:0.666,
- 原宿 カメラ:0.285,ムービー:0.285,山手線:0.125,車窓:0.125,都内:0.105
- ボーダフォン 買収:0.28571,番号:0.285714,ソフトバンク:0.1176470
- プロジェクト 見返り:1,熊:1,ピーターズ:0.75,ウエア:0.75,インタラクシオン
- 花火 大会:0.6875,動画:0.3125,画像:0.25,横浜:0.25,場所:0.25,多摩川:0.25,
- 東急東横線 相鉄線:0.5,横浜:0.22222,目黒:0.142857
- テレビ東京 ワールドビジネスサテライト:1,おもしろ:0.6,たま:0.6,トレ:0.6
- じゃがいも レンジ:0.5,食事:0.375,ごはん:0.375,野菜:0.3125,バナナ:0.25
- 体育館 エアロビ:0.6470588,会社:0.588237,健康:0.4705882,サウナ:0.25,

