データ解析特論 第 9 回
Wikipedia の記事のクラスタリングと評価

201720690 小松 弘人

2018/01/07

1　課題

Wikipedia の記事のクラスタリングと評価を行う。対象となるデータは、以下からダウンロードした 2 つのデータである。

- http://snap.stanford.edu/data/wikispeedia.html
  - wikispeedia_paths-and-graph.tar.gz
  - wikispeedia_articles_plaintext.tar.gz

少なくともネットワークに基づくクラスタリング手法、特徴ベクトルの類似度に基づくクラスタリング手法を 1 つずつ評価すること。また、少なくとも 1 種類の評価指標を用いてクラスタリング結果の良さを定量的に評価すること。

1.1　ネットワークに基づくクラスタリング

links.txt に記事間のリンク関係が記述されている。これを用いて、ネットワークに基づくクラスタリング手法で記事のクラスタリングを行う。

categories.txt には、記事のカテゴリが記述されている。このカテゴリを正解ラベルであるとみなして、何らかの指標でクラスタリングの結果を評価する。カテゴリは、subject.Science.Physics.Space_Astronomy のように階層構造になっている。

1.2　特徴ベクトルの類似度に基づくクラスタリング

plaintext_articles ディレクトリには、各記事の本文がプレーンテキスト形式で保存されている。記事の本文で使用されている単語を基に各記事を特徴ベクトルで表現し、特徴ベクトルの類似度に基づくクラスタリング手法で記事をクラスタリングする。

categories.txt には、記事のカテゴリが記述されている。このカテゴリを正解ラベルであるとみなして、何らかの指標でクラスタリングの結果を評価する。カテゴリは、subject.Science.Physics.Space_Astronomy のように階層構造になっている。

2　データの前処理

データの前処理は、manaba にアップロードされている network_clustering.pl, bag_of_words.pl を用いた。network_clustering.pl は、各記事をノード、リンクをエッジとして表現するグラフのエッジリストおよび各記事のカテゴリの情報を整形するプログラムである。エッジリストは network_hash.txt, カテゴリの情報は category_hash.txt として保存される。bag_of_words.pl は、各記事で用いられる単語の出現頻度のベクトルを出力するプログラムである。確認する単語は、全記事で出現する単語のうち出現頻度の多い 300 単語のみである。各記事のカテゴリと特徴ベクトルは、bow.txt として保存される。ネットワークに基づくクラスタリングには、network_hash.txt, category_hash.txt、特徴ベクトルの類似度に基づくクラスタリングには、bow.txt を用いる。

3　用いたクラスタリング手法

3.1　ネットワークに基づくクラスタリング

Fast Newman 法によってクラスタリングを行った。ネットワークに基づくクラスタリングでは、Modularity と呼ばれるネットワークにおけるクラスタリングの良さを測る指標を最大化することでクラスタリングを行う。Fast Newman 法では、この Modularity の変化 ΔQ に着目し、準最適な最大値を求めることで、高速にクラスタリングを行う。

3.2　特徴ベクトルの類似度に基づくクラスタリング

非階層的な手法である k-means 法によるクラスタリングと、階層的な手法であるデンドログラムを用いたクラスタリングを行う。

k-means 法では、まずクラスタの代表点を k 個決め、各データと各代表点の距離を計算する。次に、距離が最短となる代表点のクラスタをデータの属するクラスタとし、データが属するクラスタが変化したかどうかを調べる。そして、変化していなかった場合はそこで完了とし、変化していた場合は各クラスタに属するデータの中心を新しい代表点として手順を繰り返すことで、データのクラスタリングを行う。

デンドログラムを用いた手法では、まずクラスタの併合関係を表すデンドログラムを計算し、これをカットすることでクラスタリングを行う。

各記事の特徴ベクトルは、出現頻度が多い 300 単語の記事内での出現頻度を表したベクトルを用いる。また、各手法のパラメータは $k = 15$ とする。これは、今回の記事のカテゴリの種類と同じである。

4　用いた評価指標

Purity および F-measure を用いた評価を行った。

4.1　Purity

Purity は、以下の式で表される。ただし、N はデータ数、K はクラスタ数、C_i はクラスタ i に属するデータの集合、A_h は正解ラベルが h であるデータの集合を表す。Purity は、クラスタがどれだけ多数派で占められているかを表す尺度である。

$$\frac{1}{N}\sum_{i=1}^{K}\max_{h}|C_i \cap A_h| \tag{1}$$

4.2　F-measure

F-measure は、以下の式で表される。

$$P_{h,k} = \frac{|A_h \cap C_k|}{|C_k|} \tag{2}$$

$$R_{h,k} = \frac{|A_h \cap C_k|}{|A_h|} \tag{3}$$

$$F_{h,k} = \frac{2R_{h,k}P_{h,k}}{R_{h,k} + P_{h,k}} \tag{4}$$

F-measure は、適合率 (Precision) と再現率 (Recall) の調和平均である。したがって、正確性と網羅性を総合的に表す尺度である。クラスタリングの評価指標として用いる際は、以下の値を用いる。

$$F = \sum_{h=1}^{K} \frac{|A_h|}{N} \max_k F_{h,k} \tag{5}$$

5　評価結果

それぞれの手法に対する評価指標の結果を以下に示す。

表 1　評価結果

手法	Purity	F-measure
Fast Newman 法	0.3364506	0.115294
k-means 法	0.3449326	0.1098105
Dendrogram	0.2346672	0.08339957

6　評価結果に対する考察

Purity の評価指標では k-means 法が最もよく、次いで Fast Newman 法、デンドログラムを用いた方法だったが、F-measure の評価指標では Fast Newman 法が最もよく、次いで k-means 法、デンドログラムを用いた手法となった。このように、評価指標によって評価が異なることが確認できた。

付録 A　評価に用いたプログラム

A.1　評価指標

```
purity = function(ct) {
    sum(apply(ct, 1, max)) / sum(ct)
}

f_measure = function(ct) {
    F_h_k = matrix(0, nrow=nrow(ct), ncol=ncol(ct))
    for(h in 1:ncol(ct)) {
        for(k in 1:nrow(ct)) {
            F_h_k[k, h] = (2 * ct[k, h]) / (sum(ct[k, ]) + sum(ct[, h]))
        }
    }

    F_h = matrix(0, ncol=ncol(ct))
    for(h in 1:ncol(ct)) {
        F_h = sum(ct[, h]) / sum(ct) * max(F_h_k[, h])
    }
    sum(F_h)
}
```

A.2　クラスタリング

A.2.1　Fast Newman 法

```
library(igraph)
source('indicator.R')
g = read.graph("network_hash.txt", directed=F)
g2 = simplify(g)
cluster = fastgreedy.community(g2)
category = read.table("category_hash.txt")
ct = table(cluster$membership,category$V1)
purity(ct)
f_measure(ct)
```

A.2.2　k-means 法

```
source('indicator.R')
bow = read.table("bow.txt", sep=" ")
bow2 = bow
bow2$V1 = NULL
bow2$V2 = NULL
km = kmeans(bow2, 15)
ct = table(km$cluster, bow$V2)
purity(ct)
f_measure(ct)
```

A.2.3　デンドログラムを用いた手法

```
source('indicator.R')
bow = read.table("bow.txt", sep=" ")
bow2 = bow
bow2$V1 = NULL
bow2$V2 = NULL
d = dist(bow2)
hc = hclust(d)
plot(hc)
res = cutree(hc, k=15)
ct = table(res, bow$V2)
purity(ct)
f_measure(ct)
```

付録 B　前処理に用いたスクリプト

B.1　network_clustering.pl

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

open(IN,"wikispeedia_paths-and-graph/categories.tsv") or die;
my %hash;
my $count=0;
open(OUT,">category_hash.txt");
while(my $line=<IN>){
        chomp $line;
        next if($line=~/^#/);
        if($line=~/(.+)\s+(.+)/){
                unless (defined $hash{$1}){
                        $hash{$1}=$count++;
                        my @categories=split(/\./,$2);
                        print OUT $categories[1],"\n";

                }
        }
}
close(OUT);
close(IN);
open(OUT,">network_hash.txt");
open(IN,"wikispeedia_paths-and-graph/links.tsv") or die;
while(my $line=<IN>){
        chomp $line;
        next if($line=~/^#/);
        if($line=~/(.+)\s+(.+)/){
                next unless (defined $hash{$1});
                next unless (defined $hash{$2});
                print OUT $hash{$1}," ",$hash{$2},"\n"
        }
}
close(OUT);
close(IN);
```

B.2 bag_of_words.pl

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

my $dimension=300;

open(IN,"wikispeedia_paths-and-graph/categories.tsv") or die;
my %category;
while(my $line=<IN>){
    chomp $line;
    next if($line=~/^#/);
    if($line=~/(.+)\s+(.+)/){
        unless (defined $category{$1}){
            my @categories=split(/\./,$2);
            $category{$1}=$categories[1];
        }
    }
}
close(IN);

open(IN,"stopwords.txt") or die;
my %stop;
while(my $line=<IN>){
    chomp $line;
    $stop{$line}=1;
}
close(IN);

my %all_article_word_count;
my $dirname="plaintext_articles";
opendir(DIR,$dirname) or die;
my @files=readdir(DIR);
closedir(DIR);

foreach my $file(@files){
    next unless ($file=~/^(.+?)\.txt$/);
    next unless (defined $category{$1});
    open(IN,"$dirname/$file") or die;

    while(my $line=<IN>){
        chomp $line;
        my @array=split(/\s+/,$line);
        foreach (@array){
            next if($_ eq "");
            next if (defined $stop{lc($_)});
            $all_article_word_count{lc($_)}++;
        }
    }
    close(IN);
}


my @word_order=sort {$all_article_word_count{$b}<=>$all_article_word_count{$a}} keys %all_article_word

open(OUT,">bow.txt");
```

```
foreach my $file (@files){
    next unless ($file=~/(.+)?\.txt/);
    next unless (defined $category{$1});
    print OUT $1," ",$category{$1};
    my %hash;
    open(IN,"$dirname/$file") or die;
    while(my $line=<IN>){
        chomp $line;
        my @array=split(/\s+/,$line);
        foreach (@array){
            next if (defined $stop{lc($_)});
            next if($_ eq "");
            $hash{lc($_)}++;
        }
    }
    close(IN);
    my $count=0;
    foreach my $w (@word_order){
        if(defined $hash{$w}){
            print OUT " $hash{$w}";
        }
        else{
            print OUT " 0";
        }
        $count++;
        last if($count>$dimension);
    }
    print OUT "\n";
}

close(OUT);
```