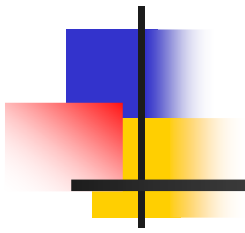
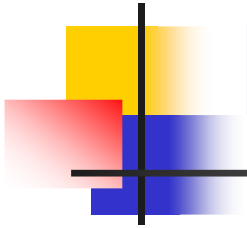


「文章クラスタリングの技法」ゼミ

A. 単一パスアルゴリズム
    1. k-means 法の適用
    2. Willett のアルゴリズム
    3. 平均クラスタリング・アルゴリズム



k-means法の適用(1)

- k-means法とは
  非階層的なクラスタリングにおける
  標準的アルゴリズム

- 文章クラスタリングに応用でき
  例えば、XuとCroftは分散型検索の
  実現を目的としてこれを利用している

k-means法の適用(2)

- k-means法のアルゴリズム(1)
  (1) クラスタの個数をLと決め、
  それぞれの中心をランダム設定
  このクラスタのベクトルを$\mathbf{c}_1,\ldots,\mathbf{c}_L$とする

  (2) N件の分類対象を最も近い
  ベクトル$\mathbf{c}_k$に分類する

  (3) 分類後、各クラスタについて
  それに属する各分類対象とベクトル$\mathbf{c}_k$との
  距離が最も近くなるようベクトル$\mathbf{c}_k$を再設定

k-means法の適用(3)

- k-means法のアルゴリズム(2)
  (4) 再設定したベクトル$\mathbf{c}_k$をもとに分類対象を
  分類し、ベクトル$\mathbf{c}_k$が変化しなくなれば
  処理を終了する
  変化がある限り分類とベクトル$\mathbf{c}_k$の
  再設定を繰り返し行う

k-means法の適用(4)

- 一般的なk-means法ではクラスタベクトル$\mathbf{c}_k$が安定するまでに分類と再設定を反復的に行う

- 反復回数とrとすればその計算量は
  $O(NMLr)$となる

  N:文書数
  M:語数(ベクトルの次元)
  L:クラスタ数

k-means法の適用(4)

- k-means法の利点
  階層的クラスタ分析より計算量が少ない
  (階層的クラスタ分析の計算量はN^2に比例)

- k-means法の問題点
  ・クラスタ個数Lを先験的に与える必要がある
  ・クラスタの中心を初期的に設定する必要がある

  → 教師無しの分類である文書クラスタリングには
  望ましくない条件

Willettのアルゴリズム(1)

- Willettのアルゴリズムの特徴
  ・転置牽引ファイルを使用して
  初期的なクラスタを設定している

  →先験的にクラスタ個数や中心を
  与える必要がない

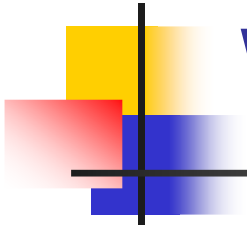
- 転置牽引ファイルとは
  単語と、その単語を含む全ての文書を
  リストとして備えている

Willettのアルゴリズム(2)

- Willettのアルゴリズム

(1) 転置牽引ファイルを使って初期的な
クラスタベクトルをM個生成($\mathbf{c}_1,\ldots,\mathbf{c}_M$)
クラスタ個数LをMに設定する(L←M)

(2)次の(3)、(4)を一定回数反復的に繰り返す
(3) 各文書$d_1,\ldots,d_N$ に対して文書ベクトル$\mathbf{d}_i$と
クラスタベクトル$\mathbf{c}_k$との類似度s ($\mathbf{d}_i$,$\mathbf{c}_k$)を
計算し(k =1,…,L)、その値が最大の
クラスタに文書d_i を割り当てる。
(同点の場合は複数のクラスタに割り当てる。)



Willettのアルゴリズム(3)

(4) 文書が全く割り当てられなかったクラスタを削除し、残ったクラスタの個数をLとする
対応するクラスタベクトル($\mathbf{c}_1,\ldots,\mathbf{c}_M$)とする

- 初期時点では語の数(M個)だけクラスタが存在すると考える
- 手順(3)、(4)を繰り返すことによりクラスタの数が減り、最終的に残ったL個のクラスタが結果として出力される

Willettのアルゴリズム(4)

- クラスタベクトルは一定件数以上に出現する語にのみ重みをつける
  →クラスタC_kに割り当てられた文書の中で語t_jを含んでいる文書数を$\tilde{n}_{j|k}$とし、
  それが閾値τより大きければ重みは1
  τ以下であれば重みは0とする
  →これにより、クラスタのベクトル、文書ベクトル共に2値ベクトルとなる

- Willettでは閾値τを$\tilde{n}_k$/3としている

Willettのアルゴリズム(5)

- クラスタのベクトル、文書ベクトルの
  類似度の計算は以下のDice係数が用いられる

$$s(d_i, c_k) = \frac{2\sum_{j=1}^{M} w_{ij}\tilde{w}_{kj}}{\sum_{j=1}^{M} w_{ij} + \sum_{j=1}^{M} \tilde{w}_{kj}}$$

記号の意味

M：初期的なクラスタベクトルの数

w_{ij}：文書ベクトル$\mathbf{d}_i$における語t_jの重み

$\tilde{w}_{jk}$：クラスタC_kにおける語t_jの重み

Willettのアルゴリズム(6)

- Willettのアルゴリズムの利点
  ・先験的にクラスタ個数Lを与える必要がない

- Willettのアルゴリズムの問題点
  反復計算により、最終的に単一のクラスタにまとまる可能性がある
  →反復回数、Lやクラスタベクトルの収束条件を前もって設定する必要がある

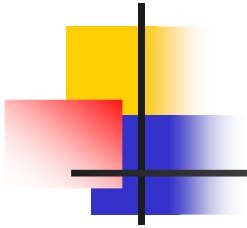
平均クラスタリング・アルゴリズム(1)

- クラスタリングの結果に対する「良し悪し」を評価
  →基準関数を設定する
  例：平方誤差の総和

$$J_e\left(\{C_k\}_{k=1}^{L}\right)=\sum_{k=1}^{L}\sum_{i:d_i\in C_k}\|d_i-m_k\|^2$$

- 記号の意味
  $\mathbf{d}_i$:文書ベクトル
  $\{C_k\}_{k=1}^{L}$:基準に照らして求めた最適な分割
  $\mathbf{m}_k$:重心ベクトル($m_k=\frac{1}{\tilde{n}_k}\sum_{i:d_i\in C_k} d_i$)
  $\tilde{n}_k$:C_kに属する文書数

平均クラスタリング・アルゴリズム(2)



- 平方誤差の総和は各クラスタがどれだけ密集しているかを表す
  → ベクトル間の距離が小さいほど
  平方誤差の総和の値も小さくなり、
  各クラスタがまとまっている事を示す

- 平方誤差の総和のような基準では厳密なクラスタを決定するのは難しい
  →反復的な計算で近似的な最適解を求める

平均クラスタリング・アルゴリズム(2)

- 平方誤差の総和は各文書を、その文書ベクトルと最も重心が近いクラスタに再割り当てすれば小さくなることが期待できる

- 再割り当てによって求められたクラスタを $\tilde{C}_k$ と表記すると以下のようになる

$$\tilde{C}_k = \left\{ d_i \mid k = \arg_{k'} \min \left\| d_i - m_{k'} \right\| \right\} \qquad \text{式（1）}$$

平均クラスタリング・アルゴリズム(3)

- 問題点
  各文書の再割り当てを反復的に繰り返すと
  局所的な最適解になる可能性がある

- 改善方法
  ある1件の文書d_iを別のクラスタに
  割り当て直したとき、その平方誤差の総和J_eが
  最も減少するようなクラスタの分割を求める
  この分割を$\{\hat{C}_k\}_{k=1}^{L}$と表記する

平均クラスタリング・アルゴリズム(4)

- 平均クラスタリング・アルゴリズムは、2つの分割 $\{\tilde{C}_k\}_{k=1}^{L}$ と $\{\hat{C}_k\}_{k=1}^{L}$ とを交互に反復的に求めることにより、よりよいクラスタを得ようとする方法。

平均クラスタリング・アルゴリズム(5)

- 平均クラスタリング・アルゴリズムの手順

(1) 初期的なクラスタの分割 $\{C_k\}_{k=1}^{L}$ の生成と
2つの閾値を θ_1, θ_2 設定する

(2) 文書 $d_1,\ldots,d_N$ に対して式(1)で定義される
分割 $\{\tilde{C}_k\}_{k=1}^{L}$ を求める
もし

$$J_e\left(\{\tilde{C}_k\}_{k=1}^{L}\right) - J_e\left(\{C_k\}_{k=1}^{L}\right) < \theta_1$$

が成り立つとき分割を更新する $\{C_k\}_{k=1}^{L} \leftarrow \{\tilde{C}_k\}_{k=1}^{L}$

平均クラスタリング・アルゴリズム(6)

(3) 文書$d_1,\ldots,d_N$に対して分割$\{\hat{C}_k\}_{k=1}^{L}$を求める
もし

$$J_e\left(\{\hat{C}_k\}_{k=1}^{L}\right)-J_e\left(\{C_k\}_{k=1}^{L}\right)<\theta_1$$

が成り立つなら分割を更新$\{C_k\}_{k=1}^{L}\leftarrow\{\hat{C}_k\}_{k=1}^{L}$

(4) 終了条件が満たされれば処理を終了する
満たされていない場合は(2)に戻る