Webサービスを実装するとしたら

内山将夫@NICT
mutiyama@nict.go.jp

これまでに紹介したことでWebサービスを実装するとしたらどうするか

- Google suggest
- きざしランキング
- 推薦サービス

念のための注意：ここで実装例として考えるのは，上記のサービスの実際の実装とは，**全く関係なく**，それと似た機能を実現するために必要そうなことを述べるだけである．なお，推薦サービスに関する記述の一部では，以下を参考にしている．Linden, G. Smith, B. York, J. (2003) Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing, Volume: 7, Issue: 1, pp.. 76–80

Google suggest

検索語を入れると，それを拡張するような検索語句のリストを返す．ユーザーは，リストの１つを選択するか，あるいは，何も選択しない．

例

- bass
  - bass guitar
  - bass fishing
- prog
  - programming
  - programming languages
  - progesterone
  - progressive
- duke
  - duke university
  - dukes of hazzard
  - duke nukem
  - duke ellington
  - duke power

パーセプトロンによる定式化

- ユーザー入力を x
- $\mathrm{Gen}(x) = x$ の拡張候補
- 教師信号 $y =$ ユーザーの選択結果
- システムが候補 $\hat{y}$ に与えるスコア $\mathbf{w} \cdot \Phi(x, \hat{y})$
- 一位の候補と y が異なるときに $\mathbf{w}$ を更新する.

残りの問題

- $\Phi($ユーザー入力, 検索候補$)$ $=$ 検索候補のベクトル表現
- $\mathrm{Gen}($ユーザー入力$) =$ 検索候補リスト

をどう設定するか.

$\mathbf{Gen}(x)$の獲得

簡単な方法としては，これまでに入力された検索語をトライ構造に保持しておく，そして，ユーザーが入力した検索語に応じて，トライ構造を下に辿る．辿れるだけ辿ったところの下にある検索語のリストを$\mathrm{Gen}(x)$とする．

$\Phi(x, y)$をどう設定するか

xはユーザー入力で，yは拡張語である．$\Phi(x, y)$は，少なくとも次の2つの要素を持つ．

- xとyの関連の強さ
- yにより検索されるページの人気度

xとyの関連の強さ

簡単な場合として，$x = u, y = uv$で，uとvが単語のときを考える．すると，uとvの共起強度として，検索ログにおける統計量を利用できるだろう．

	v	v 以外	
u	a	b	a+b
u 以外	c	d	c + d
	a+c	b+d	n

a = 「uv」を含む検索語の数
b = uを含む検索語の数 - a
c = vを含む検索語の数 - a
d = 全検索語の数 - a - b - c
n = 全検索語の数

上記の共起テーブルにおける共起強度として，例えば，対数尤度比を利用する．

yにより検索されるページの人気度

- ページランクなどを利用できるだろう

まとめ

以上より，
$\Phi(x, y) =$ [対数尤度比，人気度]
とベクトル表現を定義することが考えられる．
検索語の拡張yのスコアは，

$$w_1\text{対数尤度比} + w_2\text{人気度}$$

であり，w_1とw_2は，パーセプトロンアルゴリズムを適用して，学習できる．

実際に，パーセプトロンのような学習ベースの方法で，上手く検索拡張語を suggest できるかは，不明である．恐らく，これだけだと，検索リストには，似たものだけが集まってしまうことが考えられる．それは，推薦リストの全体としての多様性などを考えていないからである．推薦リストに載せるものとしては，大きめの推薦リストをクラスタリングして，その各クラスタから，１つずつを推薦するということも考えられる．

2008/2/6 の kizasi.jp より

きざしランキングでは，今日の話題の言葉と，その言葉の関連語を提示しているようである．

- E259系
  - 成田エクスプレス
  - 新型車両
  - 253系
  - 秋
  - JR東日本
- スーパーチューズデイ
  - アメリカ
  - 民主党
  - メガチューズデイ
  - 大統領
  - 注目
  - オバマ

今日の話題の言葉をとる

	今日のブログ	今日以外のブログ	
単語w	a	b	a+b
w 以外	c	d	c + d
	a+c	b+d	n

a = wを含む今日のブログのエントリ数
b = wを含む全ブログのエントリ数 - a
c = 今日のブログのエントリ数 - a
d = 全ブログのエントリ数 - a - b - c
n = 全ブログのエントリ数

このテーブルについて，対数尤度比をとると，もし，wが今日のブログに，特によく出現している場合には，その値は高くなる．

話題の言葉uに関連する単語vをとる

	v	v 以外	
u	a	b	a+b
u 以外	c	d	c + d
	a+c	b+d	n

a = uとvを含むブログのエントリ数
b = uを含むブログのエントリ数 - a
c = vを含むブログのエントリ数 - a
d = 全ブログのエントリ数 - a - b - c
n = 全ブログのエントリ数

このテーブルについて，対数尤度比をとると，もし，vがuと共起していることが多ければ，その値は高くなる．

ここまでのまとめ

- 言葉と言葉の共起の強さは，案外と，実用に役立つ
- 簡単な統計量でも，面白い．その理由は，言葉自体に面白い関係が内在するので，それを拾うことができれば，簡単な処理でも，面白いことができるからである．
- サービス = コンテンツ + アルゴリズム

  という考えが成立するとすると，言葉は，コンテンツとして，非常に価値がある場合がある．そのため，アルゴリズムは，複雑なものでなくても，サービスとしては，面白くなる．

  ただし，サービスを実際に展開するには，応答速度など，様々な解決すべき問題があるだろう．

推薦サービス

「ぐりとぐら」を買った人は，以下も買った.

- しろくまちゃんのほっとけーき
- おおきなかぶ
- はらぺこあおむし
- ぐりとぐらのえんそく
- ぐりとぐらのおきゃくさま

「ぐりとぐら」を買った人のリストを (a,b,c) として,「しろくまちゃんのほっとけーき」を買った人のリストも (a,b,c) とする．そうすると,「ぐりとぐら」と「しろくまちゃんのほっとけーき」は，それらを買った人が共通するので,「ぐりとぐら」を買った人には,「しろくまちゃんのほっとけーき」も推薦しよう.

問題

- アイテム (本) をどう表現するか
- アイテムの類似度をどう計算するか

アイテムの表現と類似度

- アイテム $\mathbf{x}_i = [x_{i1}, x_{i2}, ..., x_{in}]$

$$x_{ij} = \begin{cases} 1 \text{ もしアイテム} i \text{を顧客} j \text{が買ったら} \\ 0 \text{ 上記以外} \end{cases}$$

つまり，アイテムを，それを買った人のリストで表現する

- 類似度

$$\cos(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{||\mathbf{x}_i|| ||\mathbf{x}_j||}$$

推薦の方法

アイテム $\mathbf{x}$ を買った人には，DB 中のアイテム $\mathbf{y}$ を $\cos(\mathbf{x}, \mathbf{y})$ により，スコア付けし，その上位を提示する．

アイテム $\mathbf{x}_1, \mathbf{x}_2, ...$ を買った人には，DB 中のアイテム $\mathbf{y}$ を

$$\sum_i \cos(\mathbf{x}_i, \mathbf{y})$$

によりスコア付けし，その上位を提示する．

アイテムの類似度に基づく推薦の利点

オフラインで類似度を計算することができる.

for $\mathbf{x}_i \in$ アイテムのリスト
　for 顧客 $\in \mathbf{x}_i$
　　for $\mathbf{x}_j \in$ 顧客が買ったアイテムリスト
　　内積$(\mathbf{x}_i, \mathbf{x}_j) += 1$

により内積$\mathbf{x}_i \cdot \mathbf{x}_j$を計算でき，これを利用して，類似度$\cos(\mathbf{x}_i, \mathbf{x}_j)$をオフラインで計算できる.
手間の掛る計算をオフラインですることにより，サービスのときには，素早く応答できる.

パーセプトロンによる実装

- 顧客が買ったものを x
- $\mathrm{Gen}(x)$ = 推薦候補リスト
- 教師信号 y = ユーザーが買ったもの
- システムが候補 $\hat{y}$ に与えるスコア $\mathbf{w} \cdot \Phi(x, \hat{y})$
- 一位の候補と y が異なるときに $\mathbf{w}$ を更新する.
- $\Phi(x, \hat{y})$ = たとえば，x と $\hat{y}$ を共に買った人が 1 で，そうでない人が 0 のベクトル.

このような設定の下で，パーセプトロンによる学習をすることもできる．それで上手くいくかはわからない.

この推薦法の問題点など

- 似たものばかりが上位にくることがある．

  そのため，上位に来たものは，既に買っていたりする．その理由は，リストの個々の要素に独立にスコアを与えているから．

もっと，多様な推薦リストが望ましい場合もある．そのためには，個別のアイテムに独立にスコアを与えるのではなく，推薦リストにまとめてスコアを与えて，良い推薦リストを得る必要がある．

- 推薦するものとして，なるべく，もうかるものを推薦するということも考えられる．そうした場合には，買う確率はある程度低くとも，高価なものを推薦するとかも考えられる．

課題

- 適当なWebサービスを自分で実装するとしたら，どうするかを考える
- まだ存在しない，面白いサービスを考えてみる

まとめ

- 生活に密着した中で自然言語処理に関係する技術が使われるようになってきた.
- 自然言語は，コミュニケーションの基本要素の１つであるので，これは，当然であろう.
- 自然言語処理は，今後，ますます，使われるようになると思われる.