皮肉文の検出

speaker: Hirakata Kennai

2014/10/20

Introduction

こんな論文を読んだ

ICWSM - A Great Catchy Name: Semi-Supervised Recognition of Sarcastic Sentences in Online Product Reviews

Semi-Supervised Recognition of Sarcastic Sentences in Online Product Reviews

Goal: レビューのようなテキストが「皮肉文」かどうか推定する。

- 皮肉の例文
  - “Great for insomniacs” (本についてのレビュー)
  - “Trees died for this book?” (本)

上のレビューは本を絶賛してるわけではない

- 意見抽出する際の誤りを減らせる
- 要約への利用

コーパス

皮肉である度合い (level) を 5 段階で評価することを考える

- 皮肉である (level $=$ 3,4,5)
- 皮肉でない (level $=$ 1,2)

この度合いがアノテートされたコーパスを用い、
教師つきの学習を行う

n-gram feature (今回は使わない)

- テキスト分類の素性には通常 n-gram が使われる (n は高々3)
- tri(3)-gram なら、連続した 3 words を用いる

例えば

- “I feel like I put the money through the . . . .”

put the money に注目して、これが文中に出現するかどうかを素性とする

Phrase Pattern

Pattern

n-gram の一般拡張と考えられる

語を変数で置き換える

- put the money → put X money
- put the money → put X X

コーパスやそのドメインに特化しすぎない素性が期待できる

Pattern matching (1/2)

パターンのトークンが連続して出現するときマッチ

- パターン put X money
  - match: “I feel like I put the money through the . . . .”
  - match: “I feel like I put my money through the . . . .”
  - no match: “I feel like I threw my money through the . . . .”

パターン中の変数は任意の語にマッチする

- note: 変数が無い場合、n-gram と同じ

素性には次のように使える

```
f_p document
  | match = 1
  | otherwise = 0
```

Pattern matching (2/2)

さらに細かく実数値を振り分ける

```
f_p document
  | exact_match       = 1
  | sparse_match      = alpha -- 0 < alpha < 1
  | incomplete_match = n / N
  | otherwise         = 0
```

- exact_match: さっきの (完全な) マッチ
- sparse_match: パターンに語を挿入したらマッチ
- incomplete_match: パターンの変数を削除したらマッチ

n-gram のような 0/1 よりも細かく設定できる

Data

Data

Amazon のレビューを用いる

訓練データ

- 505 文に手でアノテート: seed training set
- 上を元に Yahoo!で拡張する: enriched training set

拡張

level がアノテートされた文 A と、文頭が一致するような文を見つけて同じ level を設定する

- 合わせて positive 471 文 + negative 5020 文

テストデータ

66,000 reviews

統計

Product	reviews	price	sarcastic	sarcastic ratio (%)
Shure E2c	782	99$	51	6.5
da Vinci Code	3481	9.99$	79	2.3
SONY MDR-NC6	576	69.99$	34	6.0
The God Delusions	1022	27$	19	1.9
Kindle eReader	2900	489$	19	0.7

Method

Outline

- Preprocessing
- Pattern selection
- Pattern features
- Clustering

Preprocessing

特定の企業名、商品名等をメタなトークンに置き換える

- “looking for a SONY camera”
- “looking for a [company] camera”

SONY ドメインに引っ張られないようにするため

Pattern selection [Davidov and Rappoport, 2006]

- 語を変数 X に置き換えるかを選択するので単純にやると指数
- 頻度の低い語を変数 X に置き換えてマッチ頻度がある程度あるパターンを集める
  - 最大で 6 words + 6 patterns からなるパターンを見つける
- 例
  - “X does not X much”
  - “about X X or X X”

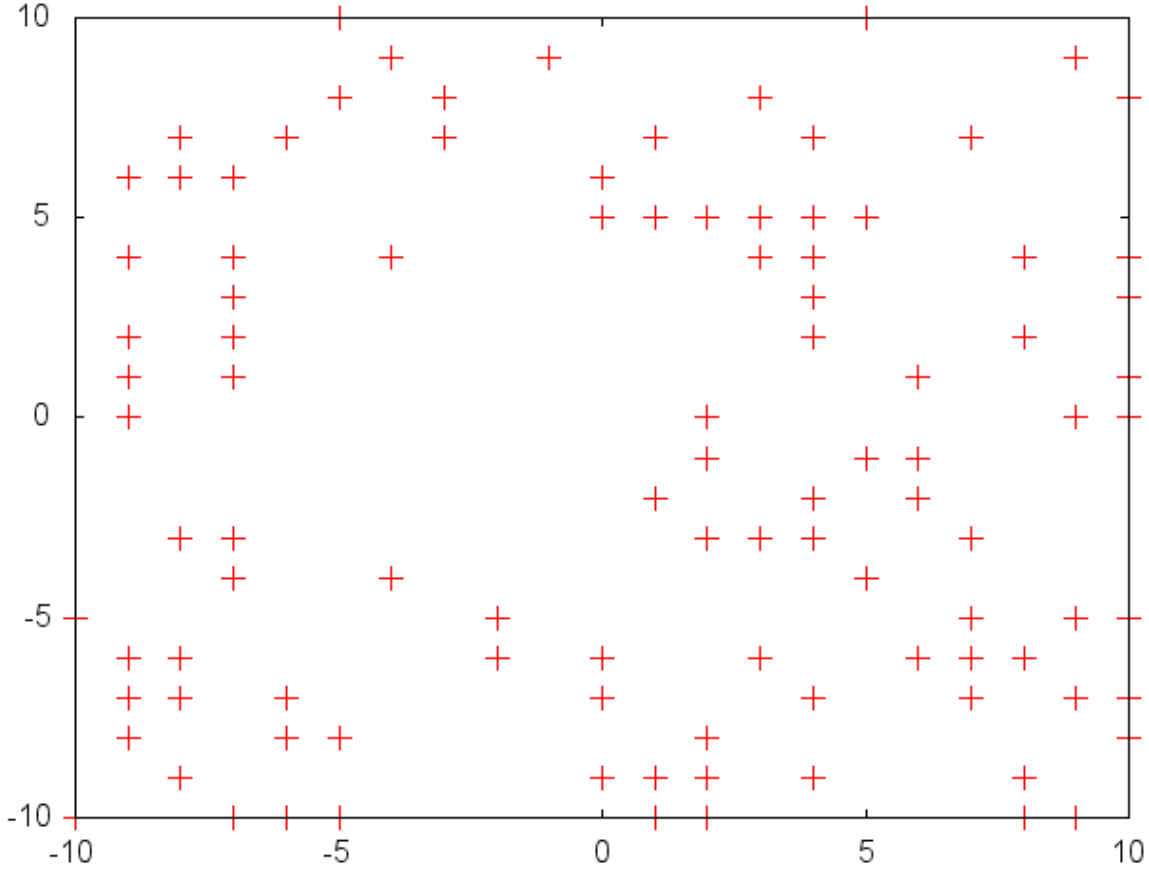
Pattern features

- パターンの列 $[p_1, \ldots, p_n]$
- それぞれのマッチ (0 以上 1 以下の実数) をベクトルで表す

$$[f_1, \ldots, f_n](0 \leq f_i \leq 1)$$

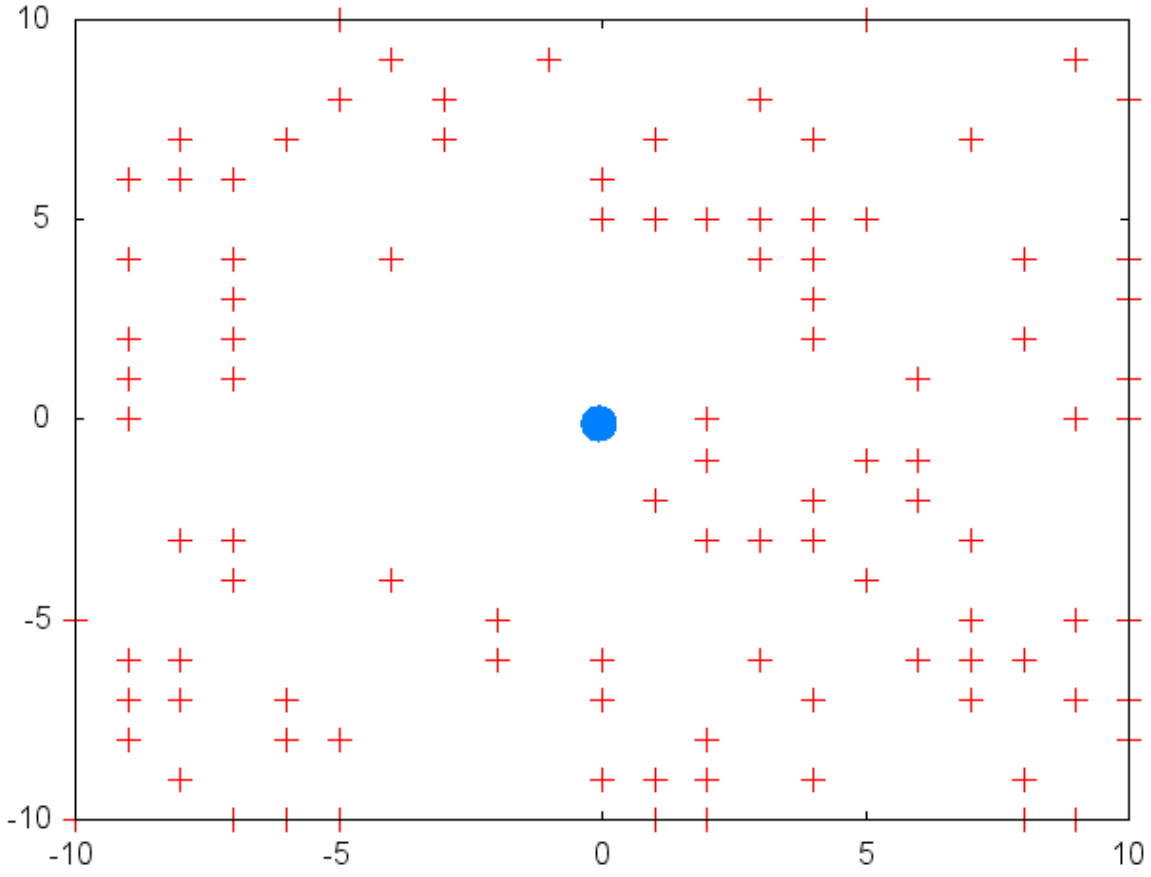
Clustering

ベクトルについてユークリッド距離に基づくクラスタリングを行う



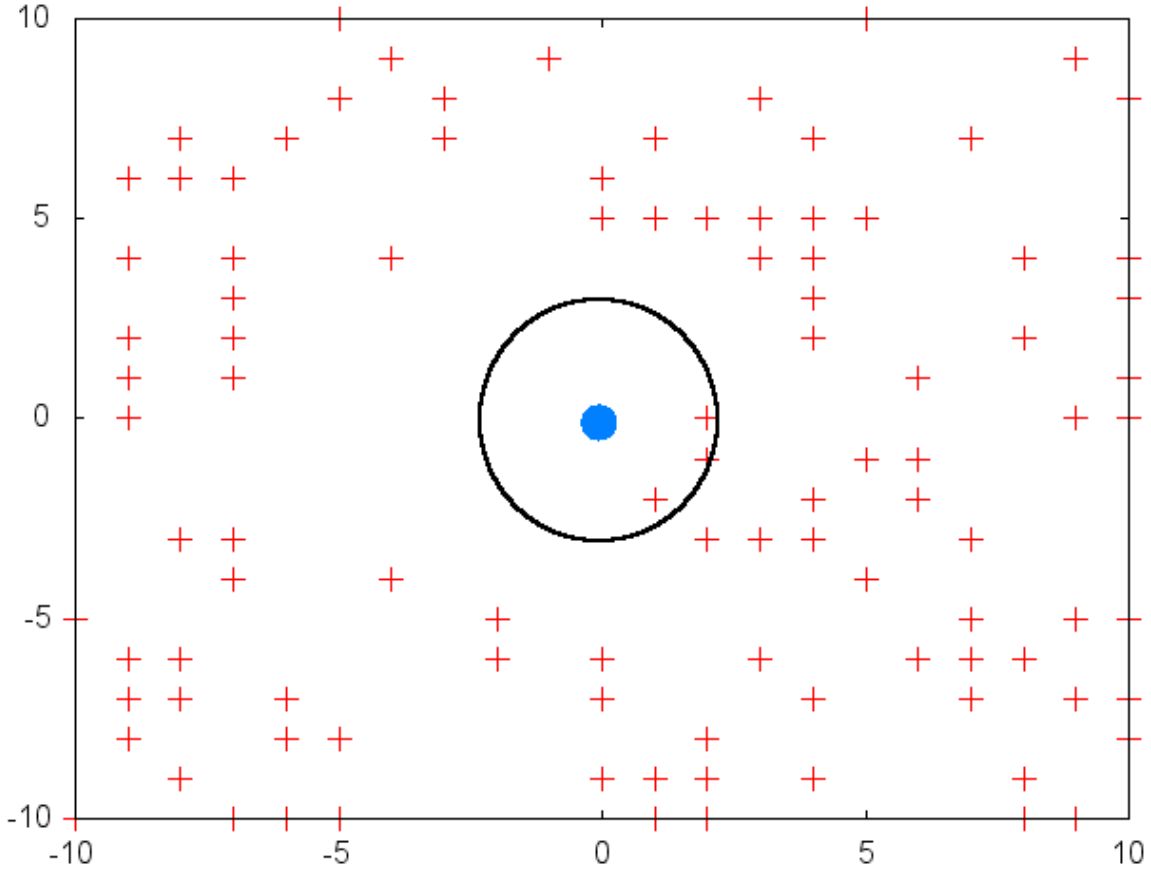
Clustering

- 赤い点がコーパス中の文 (level がついてる)
- 青い点 (レビュー) の level を知りたい



k-NN

- 青い点を中心に k 個、赤い点が入る円を描く
- 多数決



k-NN like clustering

level を連続した実数値と見做したとき、多数決より平均を取る方が良い

$$Level(v) = \left[\frac{1}{k} \sum_{i=0}^{k} weight(t_i) Level(t_i) \right]_{round}$$

where $weight(t) = \frac{Count(Level(t))}{\sum_{j=0}^{k} Count(Level(t_j))}$

- `Count` は、その level の文の数をコーパス全体から数える
- 重み $0 < weight \leq 1/k$

Result

Evaluation

人手で附けた positive / negative との比較で評価する

- `Prescition` $= \frac{tp}{tp+fp}$
- `Recall` $= \frac{tp}{tp+tn}$
- `Accuracy` $= \frac{tp+tn}{all}$
- F score is the harmonic mean of `Prescition` and `Recall`

コーパス拡張の比較

	Prec.	Rec.	Acc.	F score
seed	86.8	76.3	94.5	81.2
enriched	91.2	75.6	94.7	82.7

- `seed` は人手コーパス
- `enriched` はそれに拡張を加えたもの

(乱暴な) 拡張でも向上が見られる

Rating star

- レビューには 5 段階評価がついている
- 平均 4.12 (コーパス内で)

27 人中、26人の方が、「このレビューが参考になった」と投票しています。
★★★★★ **まさかこのアニメにハマるとは正直予想できなかった...**
投稿者 な 投稿日 2013/10/1

形式: Blu-ray | Amazonで購入
第1話を視聴したかぎりでは掴みもあんまり、キャストさんも木戸さん以外は皆新人で演技もお世辞には上手いと言えない感じで
正直「微妙」の一言に尽きる。そんな評価でした。
それでも一応観続けることにした結果...

Baseline - Star-sentiment

- 皮肉とは
  - 書いてあることと言いたいことが違うもの
- 次のようなものを皮肉文と推定する
  - rating が低く (3 以下)
  - positive な語を使ったレビュー (‘great’, ‘excellent’, . . .)

Baseline との比較結果

	Prec.	Rec.	False Pos	False Neg	F score
Baseline	50.0	16.0	5.0	44.0	24.2
提案手法	76.6	81.3	11.0	12.0	78.8

- Baseline は取りこぼしが多い

成功例 (1/2)

“Silly me, the Kindle and the Sony eBook can’t read these proceted formats. Great!”

これに対してマッチしたパターンは以下

- me, the X and [product] can’t
- [product] can’t X these X X. great !
- cant X these X X
- these X X. great !

“great!” のような皮肉に特に使われる語が学習できている

成功例 (2/2)

- “If you are under the age of 13 or have nostalgia for the days when a good mystery required minimal brain effort then this Code’s for you”
- “I feel like I put the money through the paper shredder I shelled out for these.”

このような複雑な文章も正しく推定できた

失敗例

次の二文の区別がつかなかった

- “This book was really good until page 2!”
- “This book was really good until page 430!”

一つ目は明らかに皮肉だが、 二つ目はきっと違う

- “This book was really good until page 2! what an achievement!”

これくらい文が続いてれば正しく推定できた

まとめ

まとめ

- n-gram の拡張であるパターンを用いたテキスト分類の一例を示した
- 分類のタスクに特化しない汎用な学習が可能

感想

- クラスタリングが単純すぎる (SVM とか)

補足 (2014/10/27)

index

- 皮肉文の定義
- 結果の考察

皮肉文の定義

意味することと書いてあることとが異なり、怒りや呆れを示すような文言や発言 (sarcastic と irony を区別してない?)

1. “[I] Love The Cover” (book)
   - 本の内容を褒めてない
2. “Great for insomniacs” (book)
   - insominiacs → sleep
3. “Where am I?” (GPS device)
   - GPS の文脈においては皮肉
4. “Are these iPods designed to die after two years?” (music player)
   - naive-like question

考察

- seed: 160 reviews (505 sentences)
- enriched: 66,000 reviews

	Prec.	Rec.	Acc.	F score
seed	86.8	76.3	94.5	81.2
enriched	91.2	75.6	94.7	82.7

- true negative 増
- false positve 減

	true positive	false positve	false negative	true negative
seed	11.9	1.8	3.7	82.6
enriched	12.6	1.2	4.1	82.0

- 皮肉文の検出件数自体は 11.9% → 12.6% と増

その他

二値分類ではなく level にした理由

- 強さ (つまり重み) を持たせるため、だと思います。
- 皮肉の level が強いものとパターンが似てたらそちらに寄せたい

関連論文

- SVM, to determine newswire articles are true of satirical (Burfoot and Baldwin, 2009)
  - bi-gram, uni-gram (, lex, validity)
  - P, R, F = 0.958, 0.680, 0.795
  - ベースラインは all-positive