TokyoCL 第2回勉強会

論文紹介:
A Log-Linear Model for Unsupervised Text Normalization

(Yang and Eisenstein, EMNLP 2013)

持橋大地
統計数理研究所
daichi@ism.ac.jp

2015-11-20 (金)

論文の概要

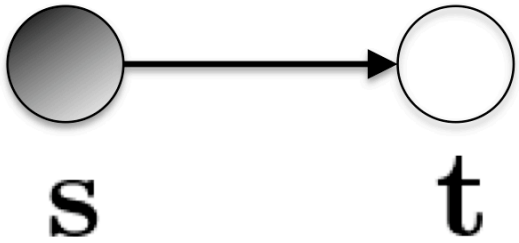
- “gimme suttin 2 beleive inn.” のような文を
  教師なしで
  “give me something to believe in.”
  に変換する
- 隣の単語も間違っている可能性
  →単語別にやってもダメ、系列全体を探索
- 通常の動的計画法では、$O(V^2)$の行列を
  計算する必要があって無理 (V=10000以上)
  →Particle Filterで効率的に微分を計算

教師なし正規化の重要性

- Twitterやチャットのような口語的なメディアは、崩れた表記がごく普通 (「不自然言語処理」)
  - “Finna hit a lick on my gramma for that burple slurrr”
  - “If dis video doing rounds in watsap on bihar electns is true Mayb d election commisn”
- 日本語ではそもそも異表記が普通
  - 「みる」「なく」「ふむ」
  - 「日弁連」「かにしの」「こなみ」・・
- 教師データを全部作るのは不可能 (固定ではない)

問題の定式化

- 以下、Twitterのようなテキストから普通の言葉に変換する問題を考える



Tweet　　English

- 意味的には s, t が逆だが (tが本当は情報源)、ここでは以下、論文の記法に従う

目的関数

Tweet English

- 観測した文の確率を最大化する

$$\text{maximize } \log p(\mathbf{s}) = \log \sum_{\mathbf{t}} p(\mathbf{s}, \mathbf{t})$$

$$= \log \sum_{\mathbf{t}} p(\mathbf{s}|\mathbf{t}) p(\mathbf{t})$$

- ここで p(t) は正しい文の言語モデル、
  p(s|t) は対数線形モデルによる誤りモデル

$$p(\mathbf{s}|\mathbf{t}, \theta) = \frac{\exp(\theta^T f(\mathbf{s}, \mathbf{t}))}{Z}$$

誤りモデル＝翻訳モデル

$$p(\mathbf{s}|\mathbf{t},\theta) = \frac{\exp(\theta^T f(\mathbf{s},\mathbf{t}))}{Z}$$

- Z(θ)は正規化項で、

$$Z(\theta) = \sum_{\mathbf{s}} \exp(\theta^T f(\mathbf{s},\mathbf{t}))$$

- 表記誤りについての素性 f(s,t) はContractor+(2010)のものを用いる
  - “you”-“u” のような直接の単語ペア
  - 単語sが単語tに文字列として何番目に近いか

目的関数の微分

- パラメータはθなので、θに関して目的関数を微分

$$L = \log \sum_{\mathbf{t}} p(\mathbf{s}|\mathbf{t}, \theta) p(\mathbf{t})$$

$$\frac{\partial L}{\partial \theta} = \frac{1}{p(\mathbf{s})} \sum_{\mathbf{t}} p(\mathbf{t}) \frac{\partial}{\partial \theta} p(\mathbf{s}|\mathbf{t}, \theta)$$

- $\frac{\partial}{\partial \theta} p(\mathbf{s}|\mathbf{t}, \theta)$ が問題だが..

目的関数の微分 (2)

$$p(\mathbf{s}|\mathbf{t},\theta) = \frac{\exp(\theta^T f(\mathbf{s},\mathbf{t}))}{Z(\theta)}$$

なので、

$$\begin{aligned}
&\frac{\partial}{\partial\theta} p(\mathbf{s}|\mathbf{t},\theta) \\
&= \frac{(\exp(\theta^T f(\mathbf{s},\mathbf{t})))' Z(\theta) - \exp(\theta^T f(\mathbf{s},\mathbf{t})) Z(\theta)'}{Z(\theta)^2} \\
&= \frac{f(\mathbf{s},\mathbf{t})\exp() \cdot Z(\theta) - \exp() \cdot Z(\theta)'}{Z(\theta)^2}
\end{aligned}$$

目的関数の微分 (3)

- ここで

$$Z(\theta) = \sum_{\mathbf{s}} \exp(\theta^T f(\mathbf{s}, \mathbf{t}))$$

$$\frac{\partial}{\partial \theta} Z(\theta) = \sum_{\mathbf{s}} f(\mathbf{s}, \mathbf{t}) \exp(\theta^T f(\mathbf{s}, \mathbf{t}))$$

なので、

目的関数の微分 (4)

$$\frac{\partial}{\partial\theta}\, p(\mathbf{s}|\mathbf{t},\theta)$$

$$= \frac{f(\mathbf{s},\mathbf{t})\exp() \cdot Z(\theta) - \exp() \cdot Z(\theta)'}{Z(\theta)^2}$$

$$= \frac{f(\mathbf{s},\mathbf{t})\exp()}{Z(\theta)} - \frac{\exp() \cdot \sum_{\mathbf{s}'} f(\mathbf{s}',\mathbf{t})\exp()}{Z(\theta)^2}$$

$$= p(\mathbf{s}|\mathbf{t},\theta)\left[f(\mathbf{s},\mathbf{t}) - \sum_{\mathbf{s}'} f(\mathbf{s},\mathbf{t})p(\mathbf{s}'|\mathbf{t},\theta)\right]$$

- よって、

目的関数の微分 (5)

$$
\begin{aligned}
\frac{\partial L}{\partial \theta} &= \frac{1}{p(\mathbf{s})} \sum_{\mathbf{t}} p(\mathbf{t}) \frac{\partial}{\partial \theta} p(\mathbf{s}|\mathbf{t}, \theta) \\
&= \frac{1}{p(\mathbf{s})} \sum_{\mathbf{t}} p(\mathbf{t}) p(\mathbf{s}|\mathbf{t}, \theta) \left[f(\mathbf{s}, \mathbf{t}) - \sum_{\mathbf{s}'} f(\mathbf{s}', \mathbf{t}) p(\mathbf{s}'|\mathbf{t}, \theta) \right] \\
&= \sum_{\mathbf{t}} p(\mathbf{t}|\mathbf{s}) \left[f(\mathbf{s}, \mathbf{t}) - \sum_{\mathbf{s}'} p(\mathbf{s}'|\mathbf{t}, \theta) f(\mathbf{s}', \mathbf{t}) \right] \\
&= E_{\mathbf{t}|\mathbf{s}} \left[f(\mathbf{s}, \mathbf{t}) - E_{\mathbf{s}'|\mathbf{t}} [f(\mathbf{s}', \mathbf{t})] \right]
\end{aligned}
$$

何が問題か?

$$\frac{\partial L}{\partial \theta} = E_{\mathbf{t}|\mathbf{s}}\left[f(\mathbf{s},\mathbf{t}) - E_{\mathbf{s}'|\mathbf{t}}[f(\mathbf{s}',\mathbf{t})]\right]$$

1. 単語列 t, s' に関する期待値が二重になっている
2. t やs'だけ取っても、動的計画法 (通常のForward-Backward) は O(V^2) の計算量がかかるため不可能 (V: 語彙数、10000〜100000程度)

ノート

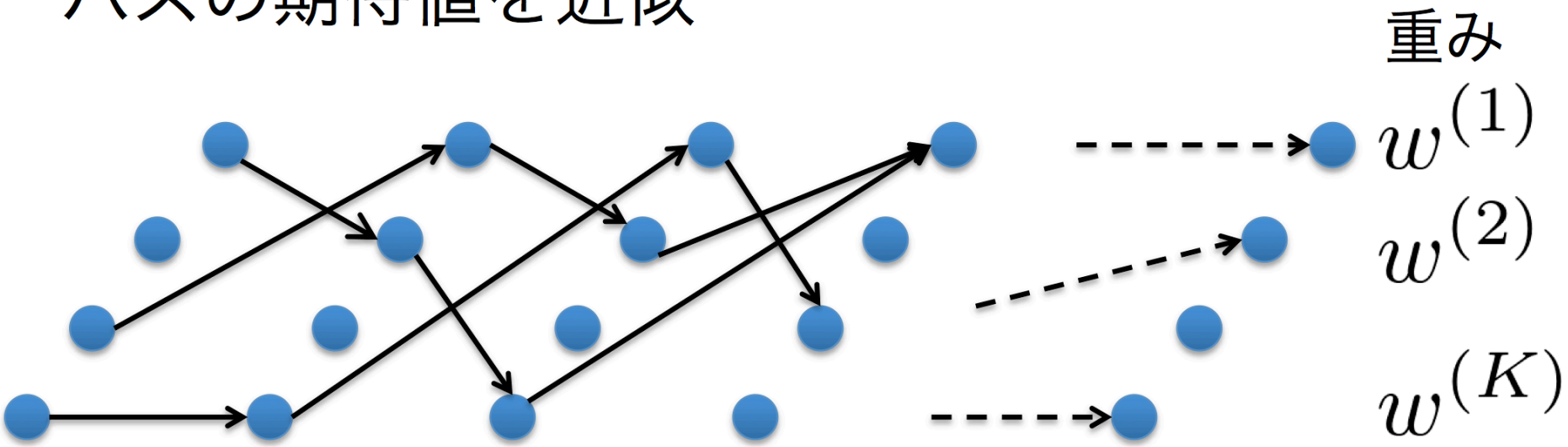
- 実際には、以下誤り確率は単語ごとに分解して考える

$$p(\mathbf{s}|\mathbf{t},\theta) = \prod_n \frac{\exp(\theta^T f(s_n, t_n))}{Z(t_n)}$$

$$Z(t_n) = \sum_s \exp(\theta^T f(s_n, t_n))$$

Sequential Monte Carlo

- 逐次モンテカルロ法 aka. Particle Filter
  - 言語モデルの話に適用したのは、多分私が最初 (Mochihashi and Matsumoto (2005))
  - 実際のモンテカルロサンプルで、Lattice上のパスの期待値を近似



SMCによる期待値

- いまの場合、K個のサンプルを用いて期待値を近似

$$E_{\mathbf{t}|\mathbf{s}}[f(\mathbf{s},\mathbf{t})] = \sum_{k=1}^{K} w_N^{(k)} \sum_{n=1}^{N} f(s_n, t_n^{(k)})$$

  - $\mathbf{s}$ ：観測された単語列 (Tweet)
  - $\mathbf{t}$ ：真の単語列
  - $\mathbf{t}^{(k)}$ ：真の単語列のサンプル
- $w_N^{(k)}$ はサンプル $\mathbf{t}^{(k)}$ の確率に比例

サンプル重みの更新

- $w_n^{(k)}$ ：サンプルkの時刻nでの重み
- 初期値は $w_0^{(k)} = 1$
- SMCの理論から、重みを以下の式で更新

表記誤りモデル

バイグラム言語モデル

$$w_n^{(k)} = w_{n-1}^{(k)} \cdot \frac{p(s_n|t_n^{(k)})p(t_n^{(k)}|t_{n-1}^{(k)})}{q(t_n^{(k)}|t_{n-1}^{(k)}, s_n)}$$

提案分布でのt_nの確率密度

$t_n^{(k)}$: 提案分布 q からの
正しいと思われる単語のサンプル

提案分布

- 正しい分布

$$p(t_n^{(k)}|s_n, t_{n-1}^{(k)}) = \frac{p(s_n|t_n^{(k)})p(t_n^{(k)}|t_{n-1}^{(k)})}{\sum_{t'} p(s_n|t')p(t'|t_{n-1}^{(k)})}$$

$$= \frac{\frac{\exp(\theta^T f(s_n, t_n^{(k)}))}{Z(t_n^{(k)})} p(t_n^{(k)}|t_{n-1}^{(k)})}{Z}$$

はO(V²)の計算量

– 分母と分子でそれぞれ正規化項の計算が必要

提案分布 (2)

- 正規化が一つの確率を提案分布に

$$q(t_n^{(k)}|s_n, t_{n-1}^{(k)}) = \frac{\exp(\theta^T f(s_n, t_n^{(k)}))p(t_n^{(k)}|t_{n-1}^{(k)})}{\sum_{t'} \exp(\theta^T f(s_n, t'))p(t'|t_{n-1}^{(k)})}$$

これによるバイアスは、サンプルの重みで
補正される

内側の期待値

$$\frac{\partial L}{\partial \theta} = E_{\mathbf{t}|\mathbf{s}}\left[f(\mathbf{s},\mathbf{t}) - E_{\mathbf{s}'|\mathbf{t}}[f(\mathbf{s}',\mathbf{t})] \right]$$

- 内側の期待値 $E_{\mathbf{s}'|\mathbf{t}}[]$ には、誤りモデル p(s|t,θ) をそのまま使ってモンテカルロ近似

$$E_{\mathbf{s}'|\mathbf{t}}[f(\mathbf{s},\mathbf{t}^{(k)}] = \frac{1}{L}\sum_{n=1}^{N}\sum_{\ell=1}^{L} f(s_n^{k,\ell}, t_n^{(k)})$$

$$s_n^{k,\ell} \sim p(s|t_n^{(k)},\theta) \qquad \text{i.i.d.}$$

最終的なgradient

$$E_{\mathbf{t}|\mathbf{s}}\left[f(\mathbf{s},\mathbf{t}) - E_{\mathbf{s}'|\mathbf{t}}[f(\mathbf{s}',\mathbf{t})]\right]$$

$$= \frac{1}{\sum_{k=1}^{K} w_N^{(k)}} \sum_{k=1}^{K} w_N^{(k)} \times$$

$$\sum_{n=1}^{N}\left(f(s_n, t_n^{(k)}) - \frac{1}{L}\sum_{\ell=1}^{L} f(s_n^{k,l}, t_n^{(k)})\right) .$$

- SMC+MCによる期待値で、計算量の爆発するForward-Backward (のネスト)を回避
- 論文では、K=10, L=1で充分だったとのこと(!)

実験

- 評価用データセット
  - LWWL11 (Liu+ 2011) : 単語とその正規形のリスト, 3802個
  - LexNorm1.1 (Han&Baldwin 2011) : Tweetとその正規形、549 tweets, 558 nonstandard word types
  - LexNorm1.2 : 著者らが上のものの誤りを訂正
- 辞書にある語については、p(s_n|t_n)=0 に設定
  →書き換えられることはない
  - illを i'll に書き換えることは現状できない

実験結果

Method	Dataset	Precision	Recall	F-measure
(Liu et al. 2011)	LMML11	68.88	68.88	68.88
(Liu et al. 2012)		69.81	69.81	69.81
UNLOL		**73.04**	**73.04**	**73.04**
(Han and Baldwin, 2011)	LexNorm 1.1	75.30	75.30	75.30
(Liu et al. 2012)		84.13	78.38	81.15
(Hassan et al. 2013)		**85.37**	56.4	69.93
UNLOL		82.09	**82.09**	**82.09**
UNLOL	LexNorm 1.2	82.06	82.06	82.06

- 既存研究の精度を大きく上回る

書き換えタイプの分析

- 英語の400000ツイートをUnLOLで正規化して、傾向を分析
  - 単語の書き換えアライメントが得られる
    →Levenstein距離で使われた規則がわかる
- 誰がどんな規則を使ったかの頻度を行列化
  ↓
  NMFで圧縮して、“orthographic style”を求める
- どんなスタイルがあったのか?

書き換えタイプの分析結果

style	rules	examples
1. you; o-dropping	*y*/_ *ou*/_u **y*/⋆_ *o*/_	u, yu, 2day, knw, gud, yur, wud, yuh, u′ve, toda, everthing, everwhere, ourself
2. e-dropping, u/o	*be*/b_ *e*/_ *o*/u *e**/_⋆	b, r, luv, cum, hav, mayb, bn, remembr, btween, gunna, gud
3. a-dropping	*a*/_ **a*/⋆_ *re*/r_ *ar*/_r	r, tht, wht, yrs, bck, strt, gurantee, elementry, wr, rlly, wher, rdy, preciate, neway
4. g-dropping	*g**/_⋆ *ng*/n_ *g*/_	goin, talkin, watchin, feelin, makin
5. t-dropping	*t**/_⋆ *st*/s_ *t*/_	jus, bc, shh, wha, gota, wea, mus, firts, jes, subsistutes
6. th-stopping	*h*/_ **t*/⋆d *th*/d_ *t*/d	dat, de, skool, fone, dese, dha, shid, dhat, dat′s
7. (kd)-lengthening	*i*_/id _/k _/d _*/k⋆	idk, fuckk, okk, backk, workk, badd, andd, goodd, bedd, elidgible, pidgeon
8. o-lengthening	*o*_/oo _*/o⋆ _/o	soo, noo, doo, oohh, loove, thoo, helloo
9. e-lengthening	_/i *e*_/ee _/e _*/e⋆	mee, ive, retweet, bestie, lovee, nicee, heey, likee, iphone, homie, ii, damnit
10. a-adding	_/a __/ma _/m _*/a⋆	ima, outta, needa, shoulda, woulda, mm, comming, tomm, boutt, ppreciate

Table 3: Orthographic styles induced from automatically normalized Twitter text

まとめ

- 教師なしでも、表記の正規化は可能
  - 面倒な教師データを人手で作る必要はない
- 文→文の可能な書き換えの探索は動的計画法では不可能
  → Sequential Monte Carlo法を用いて、gradientを効率的に計算
  - 他のNLPタスクにも役立つ手法
- 書き換えは、人によってスタイルがある
  - どんなスタイルがあるかの教師なし学習