A log-linear model for unsupervised text normalization

Yi Yang and Jacob Eisenstein

Georgia Institute of Technology

Lexical variation

- Social media language is different from standard English.
  - **Tweet**: ya ur website suxx brah
  - **Standard**: yeah, your website sucks bro

Lexical variation

- Social media language is different from standard English.
  - **Tweet**: ya ur website suxx brah
  - **Standard**: yeah, your website sucks bro
- **Word variants**: phonetic substitutions, abbreviations, unconventional spellings, etc.

Lexical variation

- Social media language is different from standard English.
  - **Tweet**: ya ur website suxx brah
  - **Standard**: yeah, your website sucks bro
- **Word variants**: phonetic substitutions, abbreviations, unconventional spellings, etc.
- **Normalization**: map from orthographic variants to standard spellings.

Lexical variation

- Social media language is different from standard English.
    - **Tweet**: ya ur website suxx brah
    - **Standard**: yeah, your website sucks bro
- **Word variants**: phonetic substitutions, abbreviations, unconventional spellings, etc.
- **Normalization**: map from orthographic variants to standard spellings.
    - Large label space.
    - No labeled data.

Why normalization?

- Improve downstream NLP applications.
    - Part-of-speech tagging [6]
    - Dependency parsing [11]
    - Machine translation [7, 8]

Why normalization?

- Improve downstream NLP applications.
  - Part-of-speech tagging [6]
  - Dependency parsing [11]
  - Machine translation [7, 8]
- Is there more systematic variation in social media?
  - Mine patterns in how words are spelled.
  - Discover coherent orthographic styles.

Related work

Method	Surface	Context	Final System
Han & Baldwin 2011	edit distance LCS	language model syntax	linear combination
Liu et al. 2012	character-based translation	distributional similarity	decoding with language model
Hassan et al. 2013	edit distance LCS	random walk	decoding with language model
Our approach	edit distance LCS, word pairs	language model	**unified** model with **features**

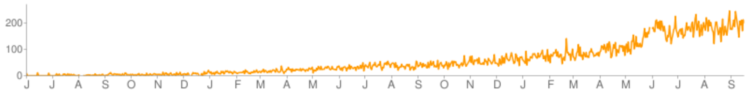
Our approach

- **Unsupervised**
- **Featurized**
- **Context-driven**
- **Joint**

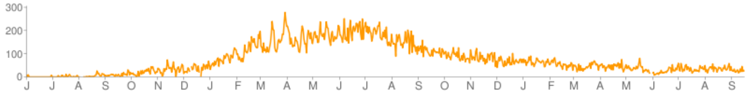
Characteristics

Unsupervised

- Labeled data for Twitter normalization is limited.
- Language changes, annotations may become stale and ill-suited to new spellings and words.



Term frequency of “af” over months



Term frequency of “lml” over months

Characteristics

Featurized: permitting overlapping features

- String similarity features.
- Lexical features that memorize conventionalized word pairs, e.g. you/u, to/2.

Characteristics

Context-driven

Give me <u>suttin</u> to believe in

- Unsupervised normalization needs strong cue of local context.
- String similarity needs to be overcome by contextual preference.
  - Edit-Dist(button/suiting, suttin) $= 2$
  - Edit-Dist(something, suttin) $= 5$

Characteristics

Joint

Gimme suttin 2 beleive innnn

- No words are in the standard vocabulary.
- Joint inference is needed to take advantage of context information.

Normalization in a probabilistic model

Maximize likelihood of observed (Twitter) text, marginalizing over normalizations.

$P(\mathbf{s})$ $P($ya ur website suxx brah$)$

Normalization in a probabilistic model

Maximize likelihood of observed (Twitter) text, marginalizing over normalizations.

$$P(\boldsymbol{s}) \qquad P(\text{ya ur website suxx brah})$$

$$= \sum_{\boldsymbol{t}} P(\boldsymbol{s}, \boldsymbol{t}) \qquad P(\text{ya ur website} \ldots, \text{yam urn website} \ldots)$$

$$+ P(\text{ya ur website} \ldots, \text{yak your website} \ldots)$$

$$+ P(\text{ya ur website} \ldots, \text{yea cure website} \ldots)$$

$$+ \ldots$$

A noisy channel model

$$P(\boldsymbol{s}, \boldsymbol{t}) = P_t(\boldsymbol{t}) P_e(\boldsymbol{s} | \boldsymbol{t})$$

$$P_e(\text{ya ur} \ldots | \text{yam urn} \ldots) \times P_t(\text{yam urn website} \ldots)$$

A noisy channel model

$$P(\boldsymbol{s}, \boldsymbol{t}) = P_t(\boldsymbol{t}) P_e(\boldsymbol{s}|\boldsymbol{t})$$

$$P_e(\text{ya ur} \ldots |\text{yam urn} \ldots) \\ \times P_t(\text{yam urn website} \ldots)$$

- The language model can be estimated offline.
- The emission model is locally normalized and log-linear.

$$P_e(\boldsymbol{s}|\boldsymbol{t}) = \prod_n P_e(s_n|t_n)$$

$$P_e(\text{ya}|\text{yam}) P_e(\text{ur}|\text{urn}) \ldots$$

$$P_e(s_n|t_n) = \frac{\exp\left(\boldsymbol{\theta}' \boldsymbol{f}(s_n, t_n)\right)}{Z(t_n)}$$

$$\frac{\exp\left(\boldsymbol{\theta}' \boldsymbol{f}(\text{ur}, \text{your})\right)}{Z(\text{your})}$$

Features

String similarity

- Combine edit distance and longest-common subsequence (LCS) [3]:
- Bin to create binary features, e.g.,

$$top5(s,t) = \begin{cases} 1, & t \in Top5(s) \\ 0, & \text{otherwise} \end{cases}$$

Features

String similarity

- Combine edit distance and longest-common subsequence (LCS) [3]:
- Bin to create binary features, e.g.,

$$top5(s, t) = \begin{cases} 1, & t \in Top5(s) \\ 0, & \text{otherwise} \end{cases}$$

Word pairs

- Assign weights to commonly occurring word pairs, e.g. you/**u**, sucks/**suxx**.
- This allows the model to "memorize" frequent substitutions.

Learning

Our goal is to learn to weight these features.
We compute the gradient of the likelihood...

Learning

Our goal is to learn to weight these features.
We compute the gradient of the likelihood...

CRF

$$\ell(\boldsymbol{\theta}) = \log P(\boldsymbol{y}|\boldsymbol{x})$$

$$\frac{\partial \ell}{\partial \boldsymbol{\theta}} = f(\boldsymbol{x}, \boldsymbol{y}) - E_{\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta}}[f(\boldsymbol{x}, \boldsymbol{y})]$$

Learning

Our goal is to learn to weight these features.
We compute the gradient of the likelihood...

CRF

$$\ell(\boldsymbol{\theta}) = \log P(\boldsymbol{y}|\boldsymbol{x})$$

$$\frac{\partial \ell}{\partial \boldsymbol{\theta}} = f(\boldsymbol{x}, \boldsymbol{y}) - E_{\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta}}[f(\boldsymbol{x}, \boldsymbol{y})]$$

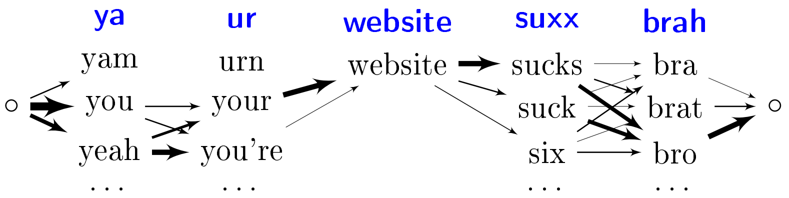
Our model (MRF)

$$\ell(\boldsymbol{\theta}) = \log P(\boldsymbol{s})$$

$$\frac{\partial \ell}{\partial \boldsymbol{\theta}} = E_{\boldsymbol{t}|\boldsymbol{s}}[f(\boldsymbol{s}, \boldsymbol{t}) - E_{\boldsymbol{s}'|\boldsymbol{t}}[f(\boldsymbol{s}, \boldsymbol{t})]]$$

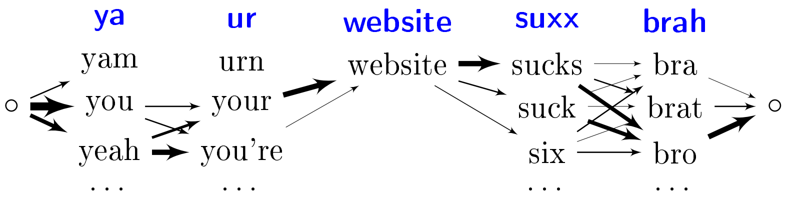
Dynamic programming

In a locally-normalized model, these expectations can be computed from the marginals $P(t_n|\boldsymbol{s}_{1:N})$ [1].



Dynamic programming

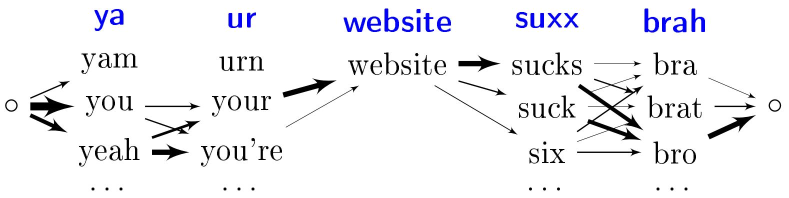
In a locally-normalized model, these expectations can be computed from the marginals $P(t_n|\boldsymbol{s}_{1:N})$ [1].



- **Outer expectation** $E_{\boldsymbol{t}|\boldsymbol{s}}$: forward-backward
- **Inner expectation** $E_{\boldsymbol{s}'|\boldsymbol{t}}$: sum over all possible s_n at each n.

Dynamic programming

In a locally-normalized model, these expectations can be computed from the marginals $P(t_n | \boldsymbol{s}_{1:N})$ [1].



- **Outer expectation** $E_{\boldsymbol{t}|\boldsymbol{s}}$: forward-backward
- **Inner expectation** $E_{\boldsymbol{s}'|\boldsymbol{t}}$: sum over all possible s_n at each n.
- **Time complexity**: $\mathcal{O}(NV_T^2 + NV_T^2 V_S)$ ...
  But $V_T, V_S > 10^4$

Sequential Monte Carlo

A randomized algorithm to approximate $P(\boldsymbol{t}|\boldsymbol{s})$ as a weighted sum,

$$P(\boldsymbol{t}|\boldsymbol{s}) \approx \sum_k \omega^{(k)} \delta_{\boldsymbol{t}^{(k)}}(\boldsymbol{t})$$

$$P(\boldsymbol{t}|\text{ya ur website} \ldots) \approx \begin{cases} 0.2 \times \delta(\text{you your website} \ldots) \\ 0.6 \times \delta(\text{yeah your website} \ldots) \\ \ldots \\ 0.1 \times \delta(\text{you you're website} \ldots) \end{cases}$$

- efficient and simple
- easy to parallelize
- number of samples K provides intuitive tuning between accuracy and speed

Sequential Importance Sampling (SIS)

At each n, for each sample k

- Draw $t_n^{(k)}$ from the **proposal distribution** $Q(t)$.
- Update $\omega_n^{(k)}$ so that

$$\omega_n^{(k)} \propto \frac{P(\boldsymbol{t}_{1:n}^{(k)} | \boldsymbol{s}_{1:n})}{Q(\boldsymbol{t}_{1:n}^{(k)})}$$

$$= \dots$$

$$= \frac{\overbrace{P_e(s_n | t_n^{(k)})}^{\text{Emission model}} \; \overbrace{P_t(t_n^{(k)} | t_{n-1}^{(k)})}^{\text{Language model}}}{\underbrace{Q(t_n^{(k)} | t_{n-1}^{(k)}, s_n)}_{\text{Proposal distribution}}} \omega_{n-1}^{(k)}$$

Computing the gradient from SIS

- **Outer expectation**:

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

Computing the gradient from SIS

- **Outer expectation**:

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

- **Inner expectation**: draw L samples of $s_n'|t_n$:

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

Computing the gradient from SIS

- **Outer expectation**:

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

- **Inner expectation**: draw L samples of $s'_n | t_n$:

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

(In practice, we set $K = 10$ and $L = 1$)

Example

s_n	ya	ur	website	$\dots$
	yam	urn	website	$\dots$
$t_n^{(k)}$	you	your		$\dots$
	yeah	you're		$\dots$
	$\dots$	$\dots$		$\dots$
$\omega_n^{(k)}$				
$s_n^{(\ell)}$				

Example

s_n	ya	ur	website	$\cdots$
	yam	urn	website	$\cdots$
$t_n^{(k)}$	you	your		$\cdots$
$\searrow$	yeah	you're		$\cdots$
	$\cdots$	$\cdots$		$\cdots$
$\omega_n^{(k)}$				
$s_n^{(\ell)}$				

Sample $t_n^{(k)}$ according to $Q(\boldsymbol{t} \mid \circ, ya)$.

Example

s_n	ya	ur	website	$\cdots$
	yam	urn	website	$\cdots$
$t_n^{(k)}$	you	your		$\cdots$
$\searrow$	yeah	you're		$\cdots$
	$\cdots$	$\cdots$		$\cdots$
$\omega_n^{(k)}$	$\frac{P(\text{yeah}, ya \mid \circ)}{Q(\text{yeah} \mid \circ, ya)}$			
$s_n^{(\ell)}$				

Example

s_n	ya	ur	website	$\cdots$
	yam	urn	website	$\cdots$
$t_n^{(k)}$	you	your		$\cdots$
$\searrow$	yeah	you're		$\cdots$
	$\cdots$	$\cdots$		$\cdots$
$\omega_n^{(k)}$	$\frac{P(\text{yeah}, ya \mid \circ)}{Q(\text{yeah} \mid \circ, ya)}$			
$s_n^{(\ell)}$	yea			

Sample $s_n^{(l)}$ according to $P(\mathbf{s} \mid yeah)$.

Example

s_n	ya	ur	website	$\dots$
	yam	urn	website	$\dots$
$t_n^{(k)}$	you	your		$\dots$
	yeah	you're		$\dots$
	$\dots$	$\dots$		$\dots$
$\omega_n^{(k)}$	$\frac{P(\text{yeah}, ya \mid \circ)}{Q(\text{yeah} \mid \circ, ya)}$	$\omega_1^{(k)} \frac{P(\text{your}, ur \mid \text{yeah})}{Q(\text{your} \mid \text{yeah}, ur)}$		
$s_n^{(\ell)}$	yea	youu		

Example

s_n		ya		ur		website	$\cdots$
		yam		urn	$\nearrow$	website	$\cdots$
$t_n^{(k)}$	$\searrow$	you	$\nearrow$	your			$\cdots$
		yeah		you're			$\cdots$
		$\cdots$		$\cdots$			$\cdots$
$\omega_n^{(k)}$		$\frac{P(\text{yeah}, ya \mid \circ)}{Q(\text{yeah} \mid \circ, ya)}$	$\omega_1^{(k)}$	$\frac{P(\text{your}, ur \mid \text{yeah})}{Q(\text{your} \mid \text{yeah}, ur)}$	$\omega_2^{(k)}$	$\frac{P_t(\text{website}, website \mid \text{your})}{Q(\text{website} \mid \text{your}, website)}$	$\cdots$
$s_n^{(\ell)}$		yea		youu		website	$\cdots$

Example

$$\begin{tikzcd}
s_n & \text{ya} & \text{ur} & \text{website} & \cdots \\
 & \text{yam} & \text{urn} & \text{website} & \cdots \\
t_n^{(k)} \arrow[dr] & \text{you} & \text{your} \arrow[ur] & & \cdots \\
 & \text{yeah} \arrow[ur] & \text{you're} & & \cdots \\
 & \cdots & \cdots & & \cdots \\
\omega_n^{(k)} & \frac{P(\text{yeah},\textit{ya}\mid\circ)}{Q(\text{yeah}\mid\circ,\textit{ya})} & \omega_1^{(k)}\frac{P(\text{your},\textit{ur}\mid\text{yeah})}{Q(\text{your}\mid\text{yeah},\textit{ur})} & \omega_2^{(k)}\frac{P_t(\text{website},\textit{website}\mid\text{your})}{Q(\text{website}\mid\text{your},\textit{website})} & \cdots \\
s_n^{(\ell)} & \text{yea} & \text{youu} & \text{website} & \cdots
\end{tikzcd}$$

Update:

$$\omega_N^{(k)}\big(\boldsymbol{f}(\text{yeah},\text{ya}) - \boldsymbol{f}(\text{yeah},\text{yea}) + \boldsymbol{f}(\text{your},\text{ur}) - \boldsymbol{f}(\text{your},\text{youu}) + \ldots\big)$$

Example

s_n	ya	ur	website	$\cdots$
	yam	urn	website	$\cdots$
$t_n^{(k)}$	you	your		$\cdots$
	yeah	you're		$\cdots$
	$\cdots$	$\cdots$		$\cdots$
$\omega_n^{(k)}$	$\frac{P(\text{yeah}, ya \mid \circ)}{Q(\text{yeah} \mid \circ, ya)}$	$\omega_1^{(k)} \frac{P(\text{your}, ur \mid \text{yeah})}{Q(\text{your} \mid \text{yeah}, ur)}$	$\omega_2^{(k)} \frac{P_t(\text{website}, website \mid \text{your})}{Q(\text{website} \mid \text{your}, website)}$	$\cdots$
$s_n^{(\ell)}$	yea	youu	website	$\cdots$

Update:

$$\omega_N^{(k)}\Big(\boldsymbol{f}(\text{yeah}, \text{ya}) - \boldsymbol{f}(\text{yeah}, \text{yea}) + \boldsymbol{f}(\text{your}, \text{ur}) - \boldsymbol{f}(\text{your}, \text{youu}) + \ldots\Big)$$

Example

$$\begin{tikzcd}
s_n & \text{ya} & \text{ur} & \text{website} & \cdots \\
 & \text{yam} & \text{urn} & \text{website} & \cdots \\
t_n^{(k)} \arrow[dr] & \text{you} & \text{your} \arrow[ur] & & \cdots \\
 & \text{yeah} \arrow[ur] & \text{you're} & & \cdots \\
 & \cdots & \cdots & & \cdots \\
\omega_n^{(k)} & \frac{P(\text{yeah}, ya \mid \circ)}{Q(\text{yeah} \mid \circ, ya)} & \omega_1^{(k)} \frac{P(\text{your}, ur \mid \text{yeah})}{Q(\text{your} \mid \text{yeah}, ur)} & \omega_2^{(k)} \frac{P_t(\text{website}, website \mid \text{your})}{Q(\text{website} \mid \text{your}, website)} & \cdots \\
s_n^{(\ell)} & \text{yea} & \text{youu} & \text{website} & \cdots
\end{tikzcd}$$

Update:

$$\omega_N^{(k)}\big(f(\text{yeah}, \text{ya}) - f(\text{yeah}, \text{yea}) + f(\text{your}, \text{ur}) - f(\text{your}, \text{youu}) + \ldots\big)$$

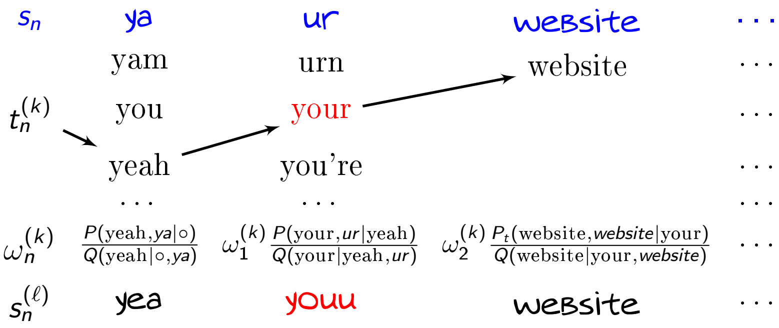
Example

$$\begin{tikzcd}
s_n & \text{ya} & \text{ur} & \text{website} & \cdots \\
 & \text{yam} & \text{urn} & \text{website} & \cdots \\
t_n^{(k)} \arrow[dr] & \text{you} & \text{your} \arrow[ur] & & \cdots \\
 & \text{yeah} \arrow[ur] & \text{you're} & & \cdots \\
 & \cdots & \cdots & & \cdots \\
\omega_n^{(k)} & \frac{P(\text{yeah}, ya \mid \circ)}{Q(\text{yeah} \mid \circ, ya)} & \omega_1^{(k)} \frac{P(\text{your}, ur \mid \text{yeah})}{Q(\text{your} \mid \text{yeah}, ur)} & \omega_2^{(k)} \frac{P_t(\text{website}, website \mid \text{your})}{Q(\text{website} \mid \text{your}, website)} & \cdots \\
s_n^{(\ell)} & \text{yea} & \text{youu} & \text{website} & \cdots
\end{tikzcd}$$

Update:

$$\omega_N^{(k)} \big(\boldsymbol{f}(\text{yeah}, \text{ya}) - \boldsymbol{f}(\text{yeah}, \text{yea}) \\ + \boldsymbol{f}(\text{your}, \text{ur}) - \boldsymbol{f}(\text{your}, \text{youu}) + \ldots \big)$$

Example


s_n
ya
ur
website
…
yam
urn
website
…
$t_n^{(k)}$
you
your
…
yeah
you're
…
…
…
…
$\omega_n^{(k)}$
$\frac{P(\text{yeah}, ya \mid \circ)}{Q(\text{yeah} \mid \circ, ya)}$
$\omega_1^{(k)}$
$\frac{P(\text{your}, ur \mid \text{yeah})}{Q(\text{your} \mid \text{yeah}, ur)}$
$\omega_2^{(k)}$
$\frac{P_t(\text{website}, website \mid \text{your})}{Q(\text{website} \mid \text{your}, website)}$
…
$s_n^{(\ell)}$
yea
youu
website
…

Update:

$$\omega_N^{(k)}\big(\boldsymbol{f}(\text{yeah}, \text{ya}) - \boldsymbol{f}(\text{yeah}, \text{yea}) \\ + \boldsymbol{f}(\text{your}, \text{ur}) - \boldsymbol{f}(\text{your}, \text{youu}) + \ldots\big)$$

The proposal distribution

A proposal distribution can guide sampling focuses on high-probability region.

The proposal distribution

A proposal distribution can guide sampling focuses on high-probability region.

1. $Q(t_n|s_n, t_{n-1}) = P(t_n|s_n, t_{n-1}) \propto P(s_n|t_n)P(t_n|t_{n-1})$
   - Accurate (optimal!)
   - Not fast: computing $P(s_n|t_n)$ costs $\mathcal{O}(V_T V_S)$

The proposal distribution

A proposal distribution can guide sampling focuses on high-probability region.

1. $Q(t_n|s_n, t_{n-1}) = P(t_n|s_n, t_{n-1}) \propto P(s_n|t_n)P(t_n|t_{n-1})$
   - Accurate (optimal!)
   - Not fast: computing $P(s_n|t_n)$ costs $\mathcal{O}(V_T V_S)$
2. Our proposal

$$\begin{aligned} Q(t_n|s_n, t_{n-1}) &\propto P(s_n|t_n)Z(t_n)P(t_n|t_{n-1}) \\ &= \exp\left(\boldsymbol{\theta}'\boldsymbol{f}(s_n, t_n)\right) P(t_n|t_{n-1}) \end{aligned}$$

   - Fast: $\mathcal{O}(V_S + V_T)$ to compute $\omega_n^{(k)}$
   - Fairly accurate: biased by a factor of $Z(t_n)$

Implementation details

unLOL: **u**nsupervised **n**ormalization in a **LOg-Linear** model

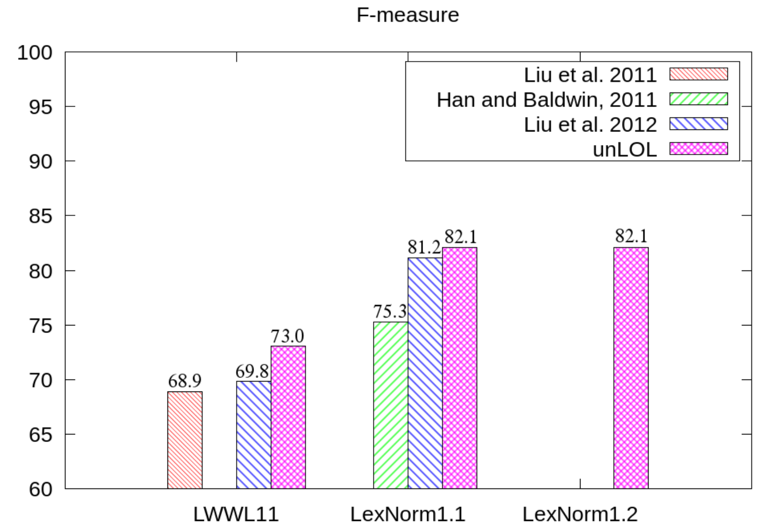
- **Decoding**: propose K sequences $\boldsymbol{t}^{(k)}$, perform Viterbi within this limited set.
- **Normalization targets**:
  - Training: all alphanumeric strings not in aspell
  - Test: normalization targets are given (standard in this task)
- **Language model**: Kneser-Ney smoothed trigram model, from tweets with no OOV words
- **Training data**: For each OOV word in the test set, draw 50 tweets from Edinburgh corpus [10] and Twitter API.

Evaluation

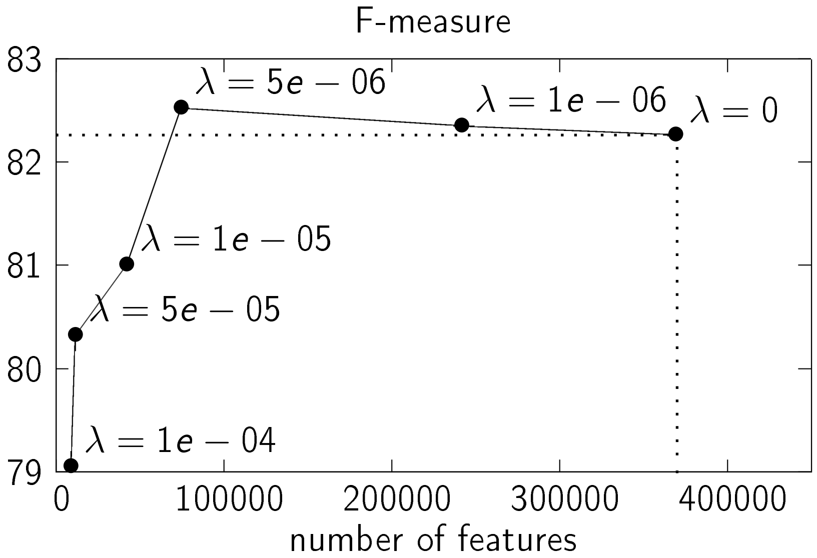
Datasets

- LWWL11 [9]: 3802 isolated words
- LexNorm1.1 [5]: 549 tweets with 1184 nonstandard tokens
- LexNorm1.2: 172 manual corrections to LexNorm1.1 (`www.cc.gatech.edu/~jeisenst/lexnorm.v1.2.tgz`)

Results


F-measure
100
95
90
85
80
75
70
65
60
Liu et al. 2011
Han and Baldwin, 2011
Liu et al. 2012
unLOL
68.9
69.8
73.0
75.3
81.2
82.1
82.1
LWWL11
LexNorm1.1
LexNorm1.2

Adding L1 Regularization

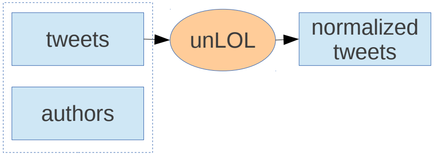


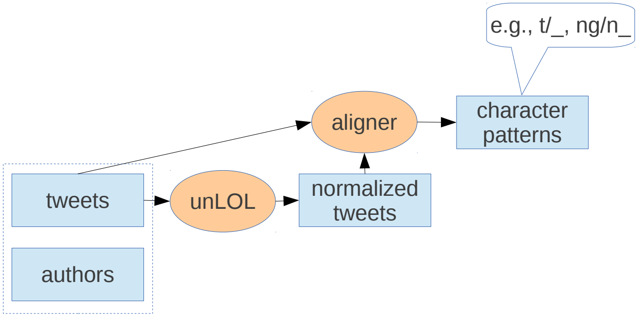
Analysis

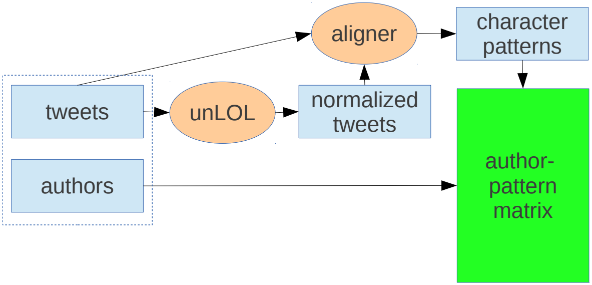
Can normalization help identify orthographic styles?

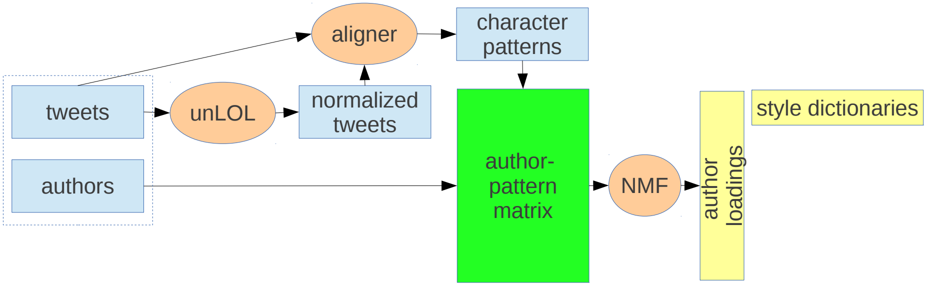
- Use unLOL to automatically label lots of tweets
- Use Levenshtein alignment between original and normalized tweets to find substitution patterns, e.g., ng/n_
- Use matrix factorization to find sets of patterns that are used by the same set of authors.

tweets
authors


tweets
authors
unLOL
normalized tweets


e.g., t/_, ng/n_
aligner
character patterns
tweets
unLOL
normalized tweets
authors


aligner
character patterns
tweets
unLOL
normalized tweets
authors
author-pattern matrix


aligner
character patterns
tweets
unLOL
normalized tweets
authors
author-pattern matrix
NMF
author loadings
style dictionaries

Observations

Some styles mirror phonological variables:
g-dropping, (TH)-stopping, t-deletion... [4]

style	**rules**	**examples**
g-dropping	g*/_*	goin, talkin, watchin, feelin, makin
	ng/n_	
	g/_	
t-deletion	t*/_*	jus, bc, shh, wha, gota, wea, mus, firts, jes, subsistutes
	st/s_	
	t/_	
th-stopping	h/_	dat, de, skool, fone, dese, dha, shid, dhat, dat's
	*t/*d	
	th/d_	
	t/d	

Observations

Others are known
"netspeak" phenomena, like expressive lengthening [2]

style	rules	examples
(kd)-adding	i_/id	idk, fuckk, okk, backk, workk, badd, andd, goodd, bedd, elidgible, pidgeon
	_/k _/d	
	_*/k*	
o-adding	o_/oo	soo, noo, doo, oohh, loove, thoo, helloo
	_*/o*	
	_/o	
e-adding	_/i	mee, ive, retweet, bestie, lovee, nicee, heey, likee, iphone, homie, ii, damnit
	e_/ee	
	_/e	
	_*/e*	

Observations

We get meaningful styles even from mistaken normalizations! (e.g., i'm/ima, out/outta)

style	rules	examples
a-adding	_/a	ima, outta, needa, shoulda, woulda, mm, comming, tomm, boutt, ppreciate
	_ _/ma	
	_/m	
	_*/a*	

Summary

- unLOL: joint, unsupervised, and log-linear
  - **Features** balance between edit distance and conventionalized word pairs.
  - **Main challenge**: massive label space, in which quadratic algorithms are not practical.
  - **Solution**: approximate gradient with SMC
  - **Proposal distribution** focuses the samples on the high-likelihood part of the search space.
- Normalization enables the study of systematic orthographic variation.

References I

T. Berg-Kirkpatrick, A. Bouchard-Côté, J. DeNero, and D. Klein.
Painless unsupervised learning with features.
In *Proceedings of NAACL*, pages 582–590, 2010.

S. Brody and N. Diakopoulos.
Cooooooooooooooollllllllllllll!!!!!!!!!!!!!!: using word lengthening to detect sentiment in microblogs.
In *Proceedings of EMNLP*, 2011.

D. Contractor, T. A. Faruquie, and L. V. Subramaniam.
Unsupervised cleansing of noisy text.
In *Proceedings of COLING*, pages 189–196, 2010.

J. Eisenstein.
Phonological factors in social media writing.
In *Proceedings of the NAACL Workshop on Language Analysis in Social Media*, 2013.

References II

B. Han and T. Baldwin.
Lexical normalisation of short text messages: makn sens a #twitter.
In *Proceedings of ACL*, pages 368–378, 2011.

B. Han, P. Cook, and T. Baldwin.
Lexical normalization for social media text.
ACM Transactions on Intelligent Systems and Technology, 4(1):5, 2013.

H. Hassan and A. Menezes.
Social text normalization using contextual graph random walks.
In *Proceedings of ACL*, 2013.

References III

W. Ling, C. Dyer, I. Trancoso, and A. Black.
Paraphrasing 4 microblog normalization.
In *Proceedings of the 2013 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Seattle, USA, October 2013. Association for Computational Linguistics.

F. Liu, F. Weng, B. Wang, and Y. Liu.
Insertion, deletion, or substitution?: normalizing text messages without pre-categorization nor supervision.
In *Proceedings of ACL*, pages 71–76, 2011.

S. Petrović, M. Osborne, and V. Lavrenko.
The edinburgh twitter corpus.
In *Proceedings of the NAACL HLT Workshop on Computational Linguistics in a World of Social Media*, pages 25–26, 2010.

References IV

C. Zhang, T. Baldwin, H. Ho, B. Kimelfeld, and Y. Li.
Adaptive parser-centric text normalization.
In *Proceedings of ACL*, pages 1159–1168, 2013.