Stopword Graphs and Authorship Attribution in Text Corpora

R. Arun, V. Suresh, C. E. Veni Madhavan (2009)

Idea

- Identify interactions of stopwords (noisewords) in text corpora
- View interactions as graphs where stopwords are nodes and interactions weights of edges between stopwords
- Interactions defined as distance between pairs of words

Idea

- Given: List of possible authors, graphs for each autor are computed
- i.e. closed case authorship attribution
- Authorship of unknown text attributed due to closeness of the graphs
- Use Kullback-Leibler-Divergence to compute closeness

Stop Words

Table I
SOME FUNCTION WORDS AND THEIR GRAMMATICAL CATEGORIES

Function Words	Examples
Prepositions	of, at, in, without, between
Pronouns	he, they, anybody, it, one
Determiners	the, a, that, my, more, much, either, neither
Conjunctions	and, that, when, while, although, or
Modal verbs	can, must, will, should, ought, need, used
Auxilliary verbs	be (is, am, are), have, got, do

- „Words that convey very little semantic meaning, but help to add detail“
- Stop words similar to function words, but may lists include more words
- „Words that convey very little semantic meaning, but help to add detail“
- Defined based on prevalence in text (occupy ~ 50 % of text)
- Lists used: 571 stopwords (~480 in my approach)

The kids are playing in the garden.

Stop Words

Table I
SOME FUNCTION WORDS AND THEIR GRAMMATICAL CATEGORIES

Function Words	Examples
Prepositions	of, at, in, without, between
Pronouns	he, they, anybody, it, one
Determiners	the, a, that, my, more, much, either, neither
Conjunctions	and, that, when, while, although, or
Modal verbs	can, must, will, should, ought, need, used
Auxilliary verbs	be (is, am, are), have, got, do

- „Words that convey very little semantic meaning, but help to add detail“
- Stop words similar to function words, but may lists include more words
- „Words that convey very little semantic meaning, but help to add detail“
- Defined based on prevalence in text (occupy ~ 50 % of text)
- Lists used: 571 stopwords (~480 in my approach)

The kids are playing in the garden.

Construction of the Graphs

- Stopwords considered as nodes of graphs
- Distance captured by edge weights
- More weight for stopwords with smaller distances
- Distance: Number of words between them

Example: The kids are playing in the garden.

d(The, the) > d(The, in) > d(the, are) = d(are, in) > d(in, the)

w(The, the) < w(The, in) < w(the, are) = w(are, in) < w(in, the)

(d: distance function, w: weight function)

Construction of the Graphs

for every occurrence of w_s, at position p_s
(note: p_i=0 until w_i appears in the corpus)
$\forall i = 1 \ldots n$
update weight of edges $(w_i, w_s), (w_s, w_i)$
if$(p_i \neq 0)$ $W_{i,s} = W_{s,i} \leftarrow W_{s,i} + e^{-|p_i - p_s|}$
(p_i : most recent occurrence of w_i)

Example: The kids are playing in the garden.

	are	in	the
are	0	e^{-2}	$e^{-2} + e^{-3}$
in	e^{-2}	0	$e^{-4} + e^{-1}$
the	$e^{-2} + e^{-3}$	$e^{-4} + e^{-1}$	e^{-5}

Kullback-Leibler Divergence

P, Q discrete probability distributions:

$$D(P\|Q) = KL(P,Q) = \sum_{x \in X} P(x) \cdot \log \frac{P(x)}{Q(x)}$$

Properties: (i) KL(P,Q) is non-negative
(ii) KL(P,Q) = 0 iff P = Q a.s.

(Proof: Follows directly from Gibb‘s inequality.)

Kullback-Leibler Divergence

Since KL Divergence is not symmetric, we use:

$$D_{\mathrm{KL}}(P\|Q) + D_{\mathrm{KL}}(Q\|P)$$

- The more similar P and Q, the smaller KL(P,Q)

Calculation of KL Divergence

Input: $G_{trn_1}, G_{trn_2}, G_{tst}$
Output: 1, if $G_{tst} \simeq G_{trn_1}$; -1 if $G_{tst} \simeq G_{trn_2}$ note: if $w_s \in V_{trn_1}, V_{trn_2}$ and $\notin V_{tst}$:
set $(w_i, w_s) = (w_s, w_i) = 0; \forall i = 1 \ldots n$

Normalize Edge Weights of $G_{trn_1}, G_{trn_2}, G_{tst}$
replace all $(w_i, w_j) = 0$ with $(w_i, w_j) = \epsilon$

$KL_1 = 0, KL_2 = 0$
for each stop_word w_i
$P_1 = \{W_{i,s}\} : W_{i,s} \in E_{trn_1}, \forall s = 1 \ldots n$
$P_2 = \{W_{i,s}\} : W_{i,s} \in E_{trn_2}, \forall s = 1 \ldots n$
$Q = \{W_{i,s}\} : W_{i,s} \in E_{tst}, \forall s = 1 \ldots n$
$kl_1 = (\mathcal{KL}(P_1 \| Q) + \mathcal{KL}(Q \| P_1))/2$
$kl_2 = (\mathcal{KL}(P_2 \| Q) + \mathcal{KL}(Q \| P_2))/2$
$\mathcal{KL}_1 \leftarrow \mathcal{KL}_1 + kl_1$
$\mathcal{KL}_2 \leftarrow \mathcal{KL}_2 + kl_2$

Experiments

- 571 stopwords
- 10 well-known English authors
- Books taken from Project Gutenberg
- Training corpus: 50.000 words
- Test corpus: 10.000 words
- Unclear what texts were used for what purpose…

Results

author	binary accuracy(%)	multi-class accuracy(%)	binary correct/total	multi-class correct/total	classes considered
Hardy	96.67	90	87/90	9/10	10
Haggard	98.89	90	89/90	9/10	10
Trollope	100	100	90/90	10/10	10
Twain	83.3	30	75/90	3/10	10
Wodehouse	97.22	88.9	128/144	32/36	5
Doyle	90.3	80.9	118/126	34/42	4
Maugham	88.89	67	16/18	4/6	4
Christie	100	100	3/3	1/1	4
Dickens	97.22	91.7	188/192	44/48	4
average accuracy	binary 94.72%	multi-class 82.05%			

Observations/Thoughts

- Quality of results influenced largely by training graph
- Which training graph should be used (e.g. Twain)?
- Change of training graph according to time?
- Does it work for other languages?
- How well does it work for shorter texts?

Own implementation

- Python 3.4.
- is running (runtime to be improved!)
- (or was running before I tried to speed it up…)
- Small changes needed
- Waiting for more books to be downloaded so I can get more results

And finally…

- Algorithm fairly easy to reproduce
- (even though I had enough issues…)
- Blanks could be filled in with some common sense
- Clear what to do even though sometimes I would have loved some explanations why…