Natural Language Processing CSCI 4152/6509 — Lecture 8 Similarity Based Classification

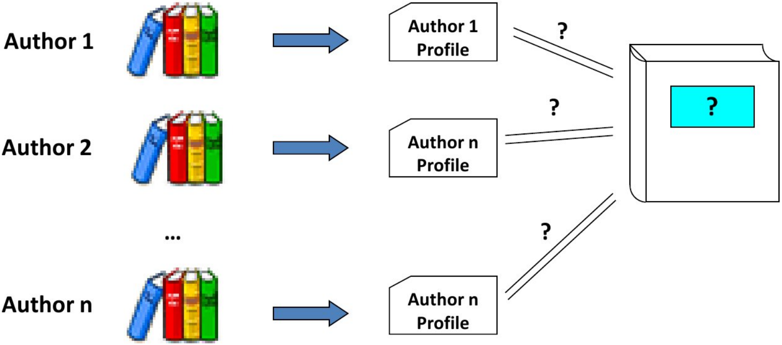
Instructors: Vlado Keselj
Time and date: 16:05 – 17:25, 1-Oct-2024
Location: Carleton Tupper Building Theatre C

Previous Lecture

- IR evaluation measures review
- Recall-precision curve review
- Text classification review
- Evaluation measures for Text Classification review
- Discussion about evaluation methods for classifiers
- Similarity-based Text Classification

CNG Method for Text Classification

- A simple method, initially used for authorship attribution
- Authorship attribution problem:



CNG Method Overview

- Method based on character n-grams
- Language independent
- Based on creating n-gram based author profiles
- Similarity based (a type of kNN method—k Nearest Neighbours)
- Similarity measure:

$$\sum_{g \in D_1 \cup D_2} \left(\frac{f_1(g) - f_2(g)}{\frac{f_1(g)+f_2(g)}{2}} \right)^2 = \sum_{g \in D_1 \cup D_2} \left(\frac{2 \cdot (f_1(g) - f_2(g))}{f_1(g) + f_2(g)} \right)^2 \quad (1)$$

where $f_i(g) = 0$ if $g \notin D_i$.

Example of Creating an Author Profile

Preparing character n−gram profile (n=3, L=5)

```
M a r l e y  w a s  d e a d :  to begin with.
There is no doubt whatever about that...
```

(from Christmas Carol by Charles Dickens)

Example of Creating an Author Profile

Preparing character n–gram profile (n=3, L=5)

$\boxed{\textbf{Mar}}$ley was dead: to begin with.
There is no doubt whatever about that...

(from Christmas Carol by Charles Dickens)

n=3
Mar

Example of Creating an Author Profile

Preparing character n–gram profile (n=3, L=5)

```
Marley was dead: to begin with.
There is no doubt whatever about that...
```

(from Christmas Carol by Charles Dickens)

n=3
Mar
arl

Example of Creating an Author Profile

Preparing character n−gram profile (n=3, L=5)

```
Ma[rle]y was dead: to begin with.
There is no doubt whatever about that...
```

(from Christmas Carol by Charles Dickens)

n=3
Mar
arl
rle

Example of Creating an Author Profile

Preparing character n−gram profile (n=3, L=5)

```
Mar**ley** was dead: to begin with.
There is no doubt whatever about that...
```

(from Christmas Carol by Charles Dickens)

n=3
Mar
arl
rle
ley

Example of Creating an Author Profile

Preparing character n−gram profile (n=3, L=5)

```
Marley was dead: to begin with.
There is no doubt whatever about that...
```

(from Christmas Carol by Charles Dickens)

n=3
Mar
arl
rle
ley
ey_

Example of Creating an Author Profile

Preparing character n−gram profile (n=3, L=5)

```
Marley was dead: to begin with.
There is no doubt whatever about that...
```

(from Christmas Carol by Charles Dickens)

n=3

```
Mar
arl
rle
ley
ey_
y_w
```

Example of Creating an Author Profile

Preparing character n–gram profile (n=3, L=5)

```
Marley was dead: to begin with.
There is no doubt whatever about that...
```

(from Christmas Carol by Charles Dickens)

n=3

```
Mar
arl
rle
ley
ey_
y_w
_wa
was
...
```

Example of Creating an Author Profile

Preparing character n−gram profile (n=3, L=5)

```
Marley was dead: to begin with.
There is no doubt whatever about that...
```

(from Christmas Carol by Charles Dickens)

n=3
Mar
arl
rle
ley
ey_
y_w
_wa
was
...

sort by frequency

_th	0.015
___	0.013
the	0.013
he_	0.011
and	0.007
_an	0.007
nd_	0.007
ed_	0.006

Example of Creating an Author Profile

Preparing character n–gram profile (n=3, L=5)

```
Marley  was dead: to begin with.
There is no doubt whatever about that...
```

(from Christmas Carol by Charles Dickens)

n=3

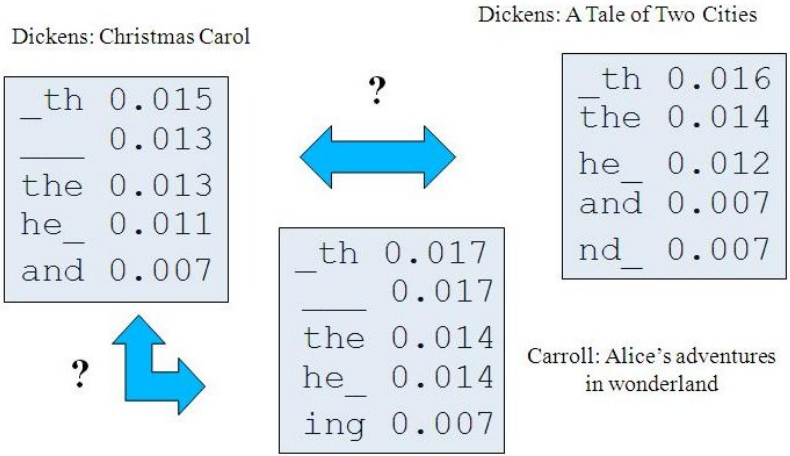
```
Mar
arl
rle
ley
ey_
y_w
_wa
was
...
```

sort by frequency

`_th`	0.015
`___`	0.013
`the`	0.013
`he_`	0.011
`and`	0.007
`_an`	0.007
`nd_`	0.007
`ed_`	0.006

L=5

How to measure profile similarity?



CNG Similarity Measure

- Euclidean-style distance with relative differences, rather than absolute
- Example: instead of using $0.88 - 0.80 = 0.10$, we say it is about 10% difference, which is the same for 0.088 and 0.080
- To be symmetric, divide by the arithmetic average:

$$d(f_1, f_2) = \Sigma_{n \in \mathsf{dom}(f_1) \cup \mathsf{dom}(f_2)} \left(\frac{f_1(n) - f_2(n)}{\frac{f_1(n)+f_2(n)}{2}} \right)^2$$

- $dom(f_i)$ is the domain of function f_i, i.e., of the profile i

Classification using CNG

- Create profile for each class using training text
  - done by merging all texts in each class into one long document
  - another option: centroid of profiles of individual documents
- Create profile for the test document
- Assign class to the document according to the closest class profile according to the CNG distance

Edit Distance

Another text similarity measure

Edit Distance: Introduction

- Edit distance is a similarity measure convenient for words and short texts, robust for typos and morphological differences
- Tends to be too expensive for longer texts
- Consider typical errors that cause typos:
  - there → thre (missed a letter)
  - there → theare (inserted an extra letter)
  - there → yhere (mistyped a letter)
- Task: find a word in lexicon most likely to produce incorrect word found in text

Edit Distance: Brute Force Approaches

- one approach: search lexicon and try deleting, inserting, and replacing each of the letters, and compare with mistyped word
- this is already quite expensive, but what with multiple errors?
- Can we find the minimal number of edit operations (deletes, inserts, or substitutions) that would lead from a source string s to the target string t?
- This is *minimal edit distance* — it always exists because we can always delete $|s|$ letters and insert $|t|$ letters, so it is always $\leq |s| + |t|$

Edit Distance: Properties

- Reflexive: $d(s,t) = 0$ if and only if $s = t$
- Symmetric: $d(s,t) = d(t,s)$, because edit operations are reversible
- Transitive: $d(s,t) + d(t,v) \geq d(s,v)$
- Can be parametrized with *cost_d*(c), *cost_i*(c), *cost_s*(c,d) for all characters c and d; positive cost functions with exception *cost_s*$(c,c) = 0$
- If cost is 1 for delete and insert, and 2 for substitute operations, it is also known as the Levenshtein distance [JM] (all cost$= 1$ according to some sources)

Edit Distance: Dynamic Programming Idea

- calculate optimal distance between $s = xe$ and $t = yf$ using optimal distances between xe and y, x and yf, and x and y

Efficient calculation of min. Edit Distance

– a dynamic programming approach

Example:

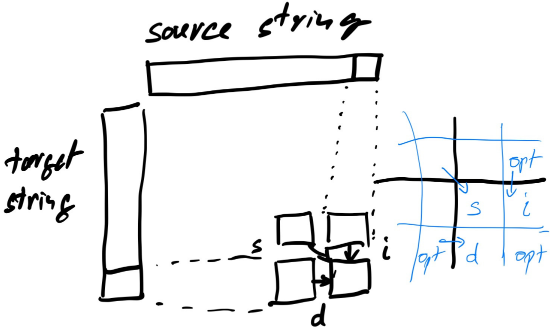
th$\boxed{e}$re

d ↓

thre

↑ i y ↓

$\boxed{y}$thre



	→	t	h	e	r	e
→						
y						
t						
h						
r						
e						

	→	t		h		e		r		e	
→	0	1	1	2	2	3	3	4	4	5	5
y	1	1	2	2	3	3	4	4	5	5	6
	1	2	1	2	2	3	3	4	4	5	5
t	2	1	2	2	3	3	4	4	5	5	6
	2	3	1	2	2	3	3	4	4	5	5
h	3	3	2	1	3	3	4	4	5	5	6
	3	4	2	3	1	2	2	3	3	4	4
r	4	4	3	3	2	2	3	2	4	4	5
	4	5	3	4	2	3	2	3	2	3	3
e	5	5	4	4	3	2	3	3	3	2	4
	5	6	4	5	3	4	2	3	3	4	2

Edit Distance Algorithm

```
Algorithm EditDistance(s,t)
n = len(s); m  = len(t)
d[m+1,n+1] - initialize to 0s
for i=1 to n do d[0,i] = d[0,i-1] + cost_d(s[i-1])
for j=1 to m do d[j,0] = d[j-1,0] + cost_i(t[j-1])
for j=1 to m do
  for i=1 to n do
    d[j,i] = min( d[j-1,i-1] + cost_s(s[i-1],t[j-1]),
                  d[j-1,i]   + cost_i(t[j-1]),
                  d[j,i-1]   + cost_d(s[i-1]) )
return d[m,n]
```

Edit Distance Example (to finish)

- distance between ‘there’ and ‘ythre’