Detecting Transliterated Orthographic Variants via Two Similarity Metrics

Kiyonori Ohtake[†], Youichi Sekiguchi[‡] and Kazuhide Yamamoto[‡]

† ATR Spoken Language Translation Research Laboratories, Japan

‡ Nagaoka University of Technology, Japan

August 24, 2004

Background

- Huge corpora are available
- Corpus-based Natural Language Processing

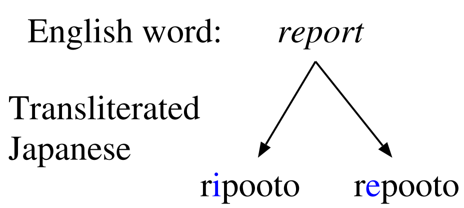
Orthographic variants cause problems:

- mismatch at looking up a dictionary
- raise perplexity
- etc.

Objectives

- To detect transliterated orthographic variants in a corpus.
- To detect mis-typed words in a corpus.

Transliterated variants



- Not only Japanese; e.g., in English, Chinese proper noun: Shanhaiguan, Shanhaikwan, or Shanhaikuan

Approaches

- Rule-based
  high accuracy, weak at irregular variants
- Back-transliteration
  very difficult
- Approximate string matching
  robust, but not accurate

Approaches

- Rule-based
  high accuracy, weak at irregular variants
- Back-transliteration
  very difficult
- Approximate string matching[Take!]

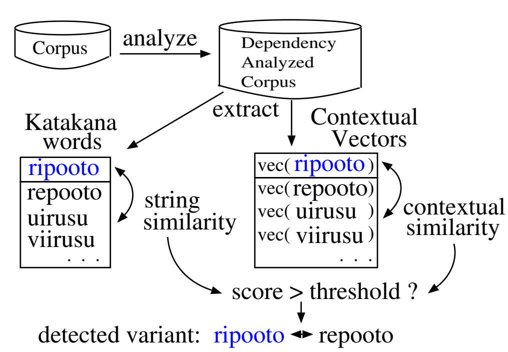
  Enhancements for high accuracy:
  - extend edit distance
  - use contextual information

Katakana transliteration

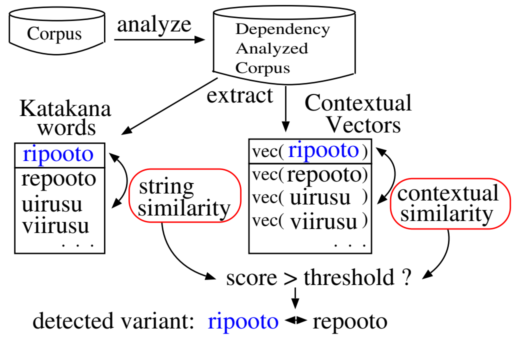
- approximate pronunciation
  - there is no specific guideline
  - several source languages
    e.g., English word *virus*
    uirusu (Latin) ↔ viirusu (German)
- sometimes transliterated from its spelling

  There is considerable variation.

Overview of detecting method



Overview of detecting method



String similarity

Input: katakana words w_1 and w_2

$S_1[1..m]$, $S_2[1..n]$: romanized strings for w_1 and w_2

$$Sim_s(w_1, w_2) = 1 - \frac{ED_k(S_1, S_2)}{m+n}$$

$$ED_k(S_1, S_2) = D(m, n)$$

Edit distance for katakana

$$D(i,j) = \min \begin{bmatrix} D(i-1,j) + id(i,j), \\ D(i-1,j-1) + 2t(i,j), \\ D(i,j-1) + id(i,j) \end{bmatrix},$$

where $t(i,j) = 0$ if $S_1(i) = S_2(j)$; otherwise the value follows $t(i,j)$-table, and $id(i,j)$ follows $id(i,j)$-table that assigns an insertion-deletion distance.

$t(i,j)$-table

rules for katakana matching (20 rules)

$i-3$ / $j-3$	$i-2$ / $j-2$	$i-1$ / $j-1$	i / j	$i+1$ / $j+1$	$i+2$ / $j+2$	$i+3$ / $j+3$	$t(i,j)$
*	*	t	[ou]	u	*	*	0.4
*	*	t	[ou]	u	*	*	
*	*	*	[dz]	i	*	*	0.25
*	*	*	[dz]	i	*	*	

'*' means any character.
'[ ]' means character class in a regular expression.

$id(i,j)$-table

similar to $t(i,j)$-table (13 rules)

Some characters easily inserted and deleted in particular context → relax

For example:
English word *decanter*
dekyantaa ↔ dekantaa

Consonant insertion-deletion → penalize

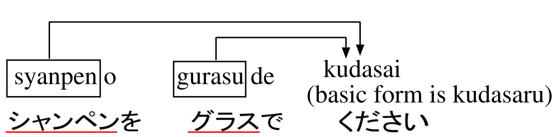
Contextual similarity

given by inner product of two vectors:

$$sim_c(kw_i, kw_j) = \cos(vec(kw_i), vec(kw_j)),$$

where $vec(kw_i)$ is the contextual vector that corresponds to the katakana word kw_i.

Contextual vector



(A glass of champagne, please.)

vec(syanpen) =[N;gurasu:1, P;kudasaru:1, PP;o-kudasaru:1]
vec(gurasu) =[N;syanpen:1, P;kudasaru:1, PP;de-kudasaru:1]

Weighting element of vector

frequently appear ≠ important

There are words

- co-occur with many other words,
- co-occur with a specific word.

Load tf-idf-like weight onto each element of the contextual vector.

How to decide a variant

follows a decision list considering the following points:

- length
- frequency in the corpus
- string similarity with ordinal edit distance
- string similarity with edit distance for katakana words
- contextual similarity
- dictionary (almost 8,000 entries)

Experiment

Corpus: ATR Basic Travel Expression Corpus [200k sentences]

160k: used for parameter estimation, and verification of rules

40k: used as a test set

Conditions:

Dictionary (8k entries): use / not use

Contextual similarity: use / not use

Experimental result

Dic.	Context	Recall	Precision	F
yes	yes	0.827 (62/75)	0.886 (62/70)	0.855
yes	no	0.907 (68/75)	0.872 (68/78)	**0.889**
no	yes	0.800 (60/75)	0.822 (60/73)	0.811
no	no	0.880 (66/75)	0.725 (66/91)	0.795

Dictionary: recall ↑ precision ↑

Contextual similarity: recall ↓ precision ↑

Discussion

- dictionary helped detection for short words and proper nouns
- some mis-types were detected; e.g., buraun' ↔ buran' (*brown*)
- contextual similarity caused side effect
- data sparseness → statistical approach may be unfit for variants detection

Future works

- To automate estimation of parameters
- To use large dictionary (e.g., more than 100k entries)
- To detect other types of variant e.g., cross-script orthographic variants (kanji vs. hiragana vs. katakana)

Conclusions

- modified edit distance for katakana words
- contextual similarity didn't work with ATR corpus
- dictionary worked very well
- performed almost 90% in F-measure

Thank you very much.

Errata

Formulas (1) and (2) (pages 711 & 712, Sections 3 and 3.1)

$$2ED \rightarrow ED$$

Formula (5) (p. 713, Section 3)

$$W(kw_i, e_i) = f(kw_i, e_i)\log\left(\frac{N}{sf(kw_i)}\right)$$

Parameter (p. 713, Section 4, 2nd paragraph)

$TH_{st1} = 9.4 \rightarrow TH_{st1} = 0.94$

Appendix - weighting element of vector

Very simple tf-idf like weighting

$$W(kw_i, e_i) = f(kw_i, e_i) \log \left(\frac{N}{sf(kw_i)} \right)$$

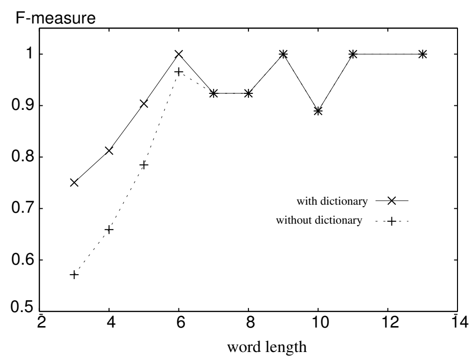
kw_i: katakana word
e_i: element of a vector
$f(kw_i, e_i)$: frequency of e_i which is a element of kw_i-vector
N: # of katakana words in the corpus
$sf(kw_i)$: sentence frequency which includes kw_i

Appendix - Decision list

length	frequency	sim_{ed}	sim_s	sim_c	decision
$> TH_{len}$	$*$	$> TH_{ed1}$	$> TH_{st1}$	$*$	variant
$<= TH_{len}$	$> TH_{freq}$	$*$	$*$	$< TH_{cos1}$	not variant
$< TH_{len}$	$*$	$*$	$*$	$> TH_{cos2}$	variant
Both words have entries in pre-defined dictionary					not variant
$*$	$*$	$> TH_{ed2}$	$> TH_{st2}$	$*$	variant
$*$	$*$	$*$	$*$	$*$	not variant

'$*$' means any conditions.

Appendix - dictionary impact



Appendix - closed test

Dic.	Con.	Recall	Precision	F
yes	yes	0.820 (296/361)	0.931 (296/318)	0.872
yes	no	0.850 (307/361)	0.930 (307/330)	0.889
no	yes	0.823 (297/361)	0.903 (297/329)	0.861
no	no	0.850 (307/361)	0.862 (307/356)	0.856

Appendix - detected examples

- successfully detected
  aisyadoo - aisyadou (*eye shadow*)
  pikurusu - pikkuruzu (*pickles*)
- mis-detected
  mari (*Mari*) - marii (*Mary*)
  maaton (*Murton*) - maton (*mutton*)

Appendix - Dictionary sample

@aisyeedo@
@aisyadoo@@aisyadou@
...
@uirusu@@biirusu@@viirusu@...
...
@syunookeru@@sunookeru@
...

Appendix - trick at $t(i, j)$

English word: *simulate*

s	i	m	y	u	r	e	e	t	o	M=0
M	S	S	S	S	M	M	M	M	M	S=2
s	y	u	m	i	r	e	e	t	o	ED=8

s	i	m	y	u	r	e	e	t	o	R=1
M	R	R	R	R	M	M	M	M	M	ED_k=4
s	y	u	m	i	r	e	e	t	o	

Appendix - trick at $t(i, j)$

English word: *simulate*

s	i	m	y	u	r	e	e	t	o	M=0
M	S	R	S	S	M	M	M	M	M	S=2 R=-2
s	y	u	m	i	r	e	e	t	o	ED_k=4

Transliteration for foreign words

3 types of Japanese characters:
hiragana and katakana → syllabary
and kanji (Chinese character)

Katakana: used to transliterate foreign words

Appendix - Romanization

Katakana corresponds to one or two phonemes

We use romanization to capture pronunciation and to make matching rules simple.

Overview of contextual vector

Context: a sentence

- What nouns co-occur
- How to depend a verb, and what verbs are depended

Construct contextual vector by using a dependency analyzer

Using dictionary

We have already known

- the words that are not similar, but they are variants, and
- the words that are very similar, but they are not variants.

↓

A dictionary will help detecting variants.