Collocation Extraction

Extracting Collocations from ngram

自然語言處理實作
Natural Language Processing Lab

Entire contents © 2011 Jason S. Chang. All rights reserved.

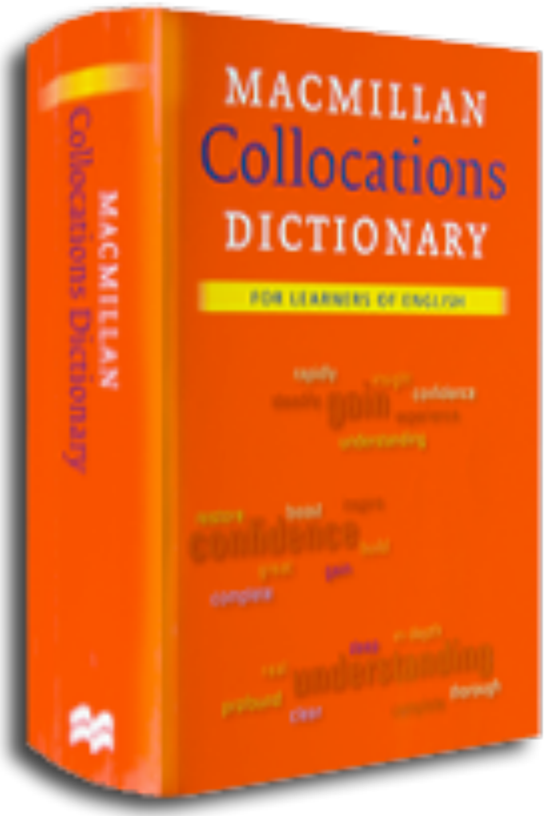
What is Collocation Extraction?

- def = Identifying collocations automatically from a corpus using a computer
- Collocation = a pair / sequence of words co-occurring more often than chance, e.g.,
  - –middle management (*intermediate management)
  - –nuclear family
  - –six sigma (6σ)
  - –plastic surgery
  - –riding boots (*horse boots)
  - –motor cyclist

- Conveying more meaning than the surface words

Entire contents © 2011 Jason S. Chang. All rights reserved.

- Why is collocation so important?
  - These prefabricated chunks are a key to fluency
  - Collocation as a key to meaning
- Methods and resources
  - Collocation tools
    - TANGO, WordSketch, Just-the-Word
  - Collocations Dictionaries
    - Oxford Collocations Dictionary
    - Macmillan Collocations Dictionary



http://www.macmillandictionaries.com/features/how-dictionaries-are-written/macmillan-collocations-dictionary/

Entire contents © 2011 Jason S. Chang. All rights reserved.

傳統紙版詞典與電子詞典

role *noun* ² position and importance

ADJ.
central, crucial, decisive, dominant, essential, fundamental, important, key, leading, major, pivotal, primary, prominent, significant, vital

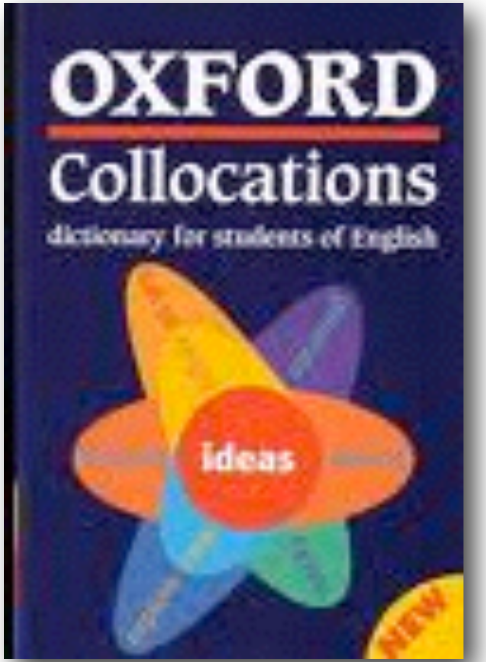
Every member of staff must have a clear role.

VERB + ROLE
occupy, perform, play, serve

Regional managers occupy a crucial role in developing a strategic framework.

PREP.
~ in

Pressure groups played a major role in bringing about the reforms.



- 適合翻譯、寫作
- 搭配數量受限
- 例句數量受限

自動產生搭配

- 可產生更多的搭配、例句
- 可產生特定的搭配
  - 學術英文
  - 商業英文

Academic Collocation List - Pearson

	Pre-collocate	AW		AW	Post-collocate
adj	considerable	research	n	research	efforts
adj	initial		n		effort
adj	earlier		n		purposes
adj	past		n		methodology
adj	original		n		evidence
adj	primary		vpp/adj		published
adj	extensive		vpp/adj		undertaken
adj	little		vpp/adj		conducted
adj	major				
adj	basic				
adj	current				
adj	empirical				
adj	previous				
adj	future				
adj	scientific				
adj	further				
adj	recent				
p/adj	existing				
p/adj	published				
n	field				
v	undertake				
v	conducting				

Source: http://baleap.qmlanguagecentre.on-rev.com/pdf/Ackermann_slides.pdf

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學 National Tsing Hua University

Properties of Collocations

1. Syntactic (government) relations
   - Lexical collocations
     - N-Adj
     - SV, VO
     - V-Adv
     - V+V
   - Grammatical collocations
     - V-Part
     - N-Prep
     - Adj+to+DO
     - Adj-Prep-n

type	example
N-Adj	**"heavy/light [] trading/smoker/traffic"**
N-Adj	**"high/low [] fertility/pressure/bounce"**
N-Adj	**"large/small [] crowd/retailer/client"**
SV	**"index [] rose**
SV	**"stock [] [rose, fell, jumped, continued, declined, crashed, ...]"**
SV	**"advancers [] [outnumbered, outpaced, overwhelmed, outstripped]"**
V-Adv	**"trade ⇔ actively," "mix ⇔ narrowly,"**
V-Adv	**"use ⇔ widely," "watch ⇔ closely"**
VO	**"posted [] gain**
VO	**"momentum [] [pick** *up*, **build, carry** *over*, **gather, loose, gain]"**
V-Part	**"take [] from," "raise [] by," "mix [] with"**
VV	**"offer** *to* **[acquire, buy"]**
VV	**"agree** *to* **[acquire, buy"]**

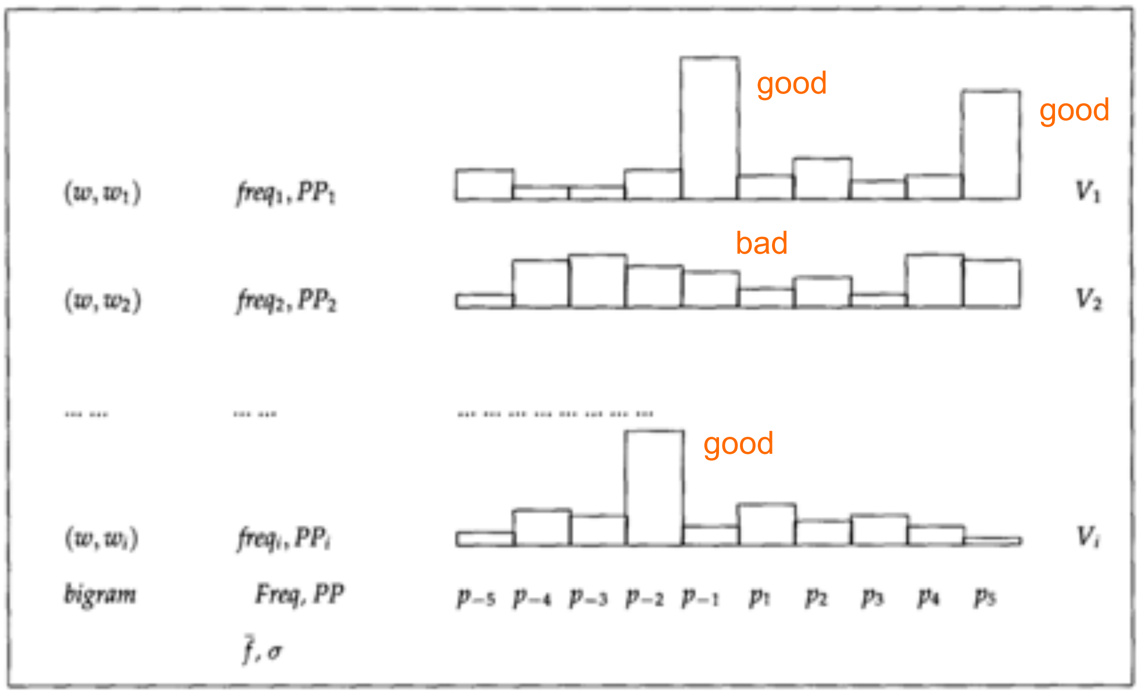
Properties of Collocations

2. Statistical associativity
   - Mutual information, log likelihood ratio (LLR)
   - t-test, chi-square test
   - Reference range (see en.wikipedia.org/wiki/Reference_range or en.wikipedia.org/wiki/Six_Sigma)
3. Syntactic relation by distanced ngram analysis (Smadja 1993)
   - Calculate count for two words (e.g., play and role) at distance *d*
   - play_role (in Google Web 1T 5gram)
     - -4(81230) -3(161358) -2(920270) -1(255149)
     - 4(325548) 3(3452577) 2(1428845) 1(27584)
   - Counts peak at distance of 3, indicating V. + Det. + Adj. + N. relation

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學 National Tsing Hua University

Word pairs, bigram freq, avg f, deviation

- Collocations = word + collocate
- Collocations are relatively high frequency
- Collocations has skew distance distribution
- Peaks at some distance imply grammatical construction: AN, VN, VN (passive)



Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學 National Tsing Hua University

Bigram examples (relatively high-freq)

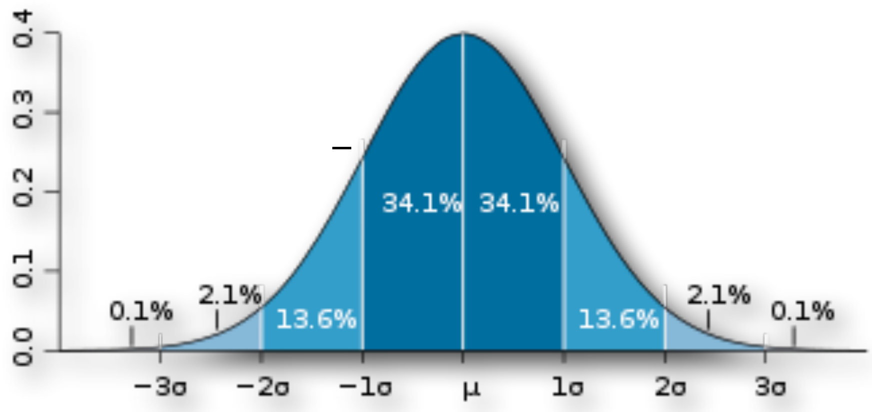
w	w_i	Freq	p_{-5}	p_{-4}	p_{-3}	p_{-2}	p_{-1}	p_1	p_2	p_3	p_4	p_5	
takeover	possible	178	0	13	4	23	138	0	0	0	0	0	Good AN collocation
takeover	corporate	93	2	2	2	1	63	3	2	9	4	5	Good AN collocation
takeover	unsolicited	83	5	30	5	0	42	0	0	1	0	0	
takeover	several	81	2	6	6	6	45	0	0	12	0	4	
takeover	recent	76	5	4	6	5	17	0	0	36	2	1	
takeover	new	75	4	3	6	28	27	0	1	4	2	0	
takeover	unwanted	53	5	0	0	2	46	0	0	0	0	0	
takeover	expensive	52	1	0	0	0	2	0	23	23	3	0	
takeover	potential	50	1	0	1	3	42	0	0	0	2	1	
takeover	big	47	0	0	0	4	15	0	0	5	21	2	
takeover	friendly	41	0	3	3	1	25	0	0	2	3	4	Good AN collocation
takeover	unsuccessful	40	0	1	5	6	27	0	0	0	0	1	
takeover	biggest	35	1	2	1	4	20	0	0	0	5	2	
takeover	largest	32	0	1	3	20	3	0	0	0	0	5	
takeover	old	28	0	8	6	0	14	0	0	0	0	0	
takeover	unfriendly	26	0	0	0	0	18	0	0	0	0	8	
takeover	rival	26	0	1	3	0	3	0	8	5	5	1	Not good
takeover	inadequate	26	5	10	2	0	0	0	0	9	0	0	Not good
takeover	initial	25	0	6	0	0	13	0	0	4	0	2	
takeover	unwelcome	24	4	0	0	0	20	0	0	0	0	0	
takeover	previous	24	0	2	0	4	18	0	0	0	0	0	
takeover	federal	22	4	2	2	0	0	0	2	2	8	2	Not good
takeover	bitter	22	0	0	0	7	14	0	0	0	1	0	
takeover	strong	19	0	4	3	5	4	0	0	1	0	2	
takeover	hostile	16	0	6	0	0	10	0	0	0	0	0	Good AN collocation
takeover	attractive	16	1	0	5	3	7	0	0	0	0	0	
takeover	unfair	13	0	0	0	0	13	0	0	0	0	0	

$\bar{f}, \sigma$

Entire contents © 2011 Jason S. Chang ... reserved.
國立清華大學 National Tsing Hua University

Smadja's Algorithm

- Compute *w*, *c*, *d* (word, collocate, distance)



- Three conditions
  - (C1) *Count(w, c)* > *f* (average) + 1*σ (standard deviation)
    - *w* = "takeover,"
    - C1 selects 167 *collocates* out of 3385
    - 95% rejected (84.2% if normal distribution).
  - (C2) *Count(w, c, d)* spread out non-uniformly (has 1-2 peaks)
  - (C3) Some distances where *Count(word, collocate, d)* peaks

Entire contents © 2011 Jason S. Chang. All rights reserved.

Three Conditions

$$
(C) \quad \left\{ \begin{array}{ll} strength = \frac{freq - \bar{f}}{\sigma} \geq k_0 & (C_1) \\ spread \geq U_0 & (C_2) \\ p_j^i \geq \bar{p}_i + (k_1 \times \sqrt{U_i}) & (C_3) \end{array} \right\}
$$

$$
U_i = \frac{\sum_{j=1}^{10} (p_i^j - \bar{p}_i)^2}{10} \qquad (k_0, k_1, U_0) = (1, 1, 10)
$$

where

strength = normalized frequency

spread(u_i) of *wi* = mean squared difference of freq from avg for *p*

peak = more frequent than avg by k_1 x standard deviation

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學
National Tsing Hua University

Examples
word, collocate, d, strength, spread

w_i	w_j	distance	strength	spread
hostile	takeovers	1	13	97
hostile	takeover	1	13	90
corporate	takeovers	1	8	90
possible	takeover	1	6	73
hostile	takeovers	2	2	70
corporate	takeover	1	3	63
. . .			*Strength > 1: Good*	Spread > 10: Good
takeover	big	4	1	47
takeovers	other	2	1	43
big	takeover	1	1	46
takeovers	major	4	1	46
biggest	takeover	1	.93	53
largest	takeover	2	.82	60

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學 National Tsing Hua University

Datasets

1. *Citeseer* ^{x}
    - lab3.iteseerx100000.tag.txt
    - 100,000 sentences, 2,270,631 words
    - As/IN such/JJ the/DT paper/NN aim/VBZ to/TO establish/VB a/DT steppingstone/NN from/IN which/WDT to/TO launch/VB actual/JJ digital/JJ design/NNS ./.
2. Words in Academic Collocation List
    - lab3.acl.words.txt
    - 1307 words
    - ability abstract ... year younger

Entire contents © 2011 Jason S. Chang. All rights reserved.

Steps

1. Generate ngrams for a given corpus (n = 2, 6)

    e.g. play a role 122
    play an important role 173 ...

2. Generate skip bigrams from ngrams (-5 <= d <= 5) per 100 m. words

    e.g. play_role 669 67 102.4 -5(4) -4(8) -3(16) -2(92) -1(25)
    1(2) 2(142) 3(345) 4(32) 5(3)

    <bgram> <f> <avg over d> <root mean square> <d(cnt)>*10

3. Generate average frequency and standard deviation of a word (e.g., play) with all its collocate (e.g., role, game)

    E.g. play 169 200

4. Discard weak collocates

    E.g., for play_role: strength = (669-169)/200 = 2.5 > 1 (C1)

5. Generate collocations (spread not evenly + peak at distance d)

    keep play_role (good pair) because 102.4 > 10 (C2)
    keep play_role 3 (peak d) because 345 > 67+1x$(102.4)^{0.5}$ (C3)
    keep play_role 2 (second peak) because 142 > 67+1x$(102.4)^{0.5}$

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學
National Tsing Hua University

1. Generate skip bigrams

- Use dictionary to count ngram and skip bigram with d
- CASE 1 from sentences
  - (w1, w_n count)
    - input = w1, w2, …, wn,
    - input = <w1 w2, 1>; <w1 w3, 2>, …, <w1 wk+1, k> (k = 5)
      <w2 w3, 1>; …, <w2 wk+2, k>
      …
      <wk+1 w1, -k>; <wk w1, -k+1>, …, <w2 w1, -1>
      <wk+2 w2, -k>; <wk+1 w2, -k+1>, …, <w3 w2, -1>
      …
- CASE2 from gram
  - input = (w1, ,,, ,wn count) (for n = 2, 5)
  - output = <w1 wn, n-1, count> and <wn, w1, -n+1, count>
- See Hint #2 on page 23

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學 National Tsing Hua University

1. Generate distance counts

- Generate a dictionary that store skip bigram and <stance, count> pairs
  - E.g. play_role -5(4) -4(8) -3(16) -2(92) -1(25) 1(2) 2(142) 3(345) 4(32) 5(3)
- See Hint #3 on page 24
  - Use defaultdict to store word, collocate, distance, count
  - from collections import defaultdict;
  - dic = defaultdict(lambda: defaultdict(lambda: defaultdict(lambda: 0)))
- dic1 = defaultdict(0) #a dictionary mapping distance to count
  - store — dic1[3] += 1
  - (e.g., dic1 = {-4:11, -3:23, -2:23, -1:38, 1:35, 2:125, 3:524, 4:101}
- dic2 = defaultdict(defaultdict(0)) # dictionary mapping word to <distance, count>
  - store — dic2[‘role’][3] += 1 #(or count)
- dic = defaultdict(lambda: defaultdict(lambda: defaultdict(lambda: 0))) # a dictionary mapping word to a dictionary of word and <distance, count>
  - store — dic[‘play’][‘role’][3] += 1 #(count)
    (e.g., dic = {‘play’: {’role’: {-4:11, -3:23, -2:23, -1:38, 1:35, 2:125, 3:524, 4:101}}}

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學 National Tsing Hua University

Step 3 Compute statistics: strength, std

- Input = bigram file: w_wi <d, count>, … <d, count>
- Compute
    - Total, avg, mean-sq-offset
    - Strength
- Example: play_role 669 67 102.4

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學
National Tsing Hua University

Step 4 Check C1

- For each key group of *w*
  - For each bigram
    - Calculate strength and discard weak bigram (C1)
      E.g., play_role 669 67 102.4 -5(4) -4(8) ...
      strength = (669-169)/200 = 2.5 > 1
- Generate bigrams <*w*, *c*> for good *c* candidates (e.g., <play, role>)

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學
National Tsing Hua University

Step 5 Check C2, C3

- input = good bigrams
- For each bigram
  - Check spread and peak conditions

    E.g., play_role is ok because spread 102.4 > 10 (C2)

    play_role 3 is ok because 345 > $67+1\text{x}(102.4)^{0.5}$ (C3)
- Generate collocation <play, role, 3>

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學
National Tsing Hua University

Lab Work

- Corpus: Citeseer X
- Step 1 has been done
- Start with Citeseer ngrams
- Generate collocation for the word ‘difficulty’

BONUS

- Write up an algorithm for generate collocations for 930 words in Academic Keyword List (AKL)
- AKL is available at www.uclouvain.be/en-372126.html

Entire contents © 2011 Jason S. Chang. All rights reserved.

搜尋 Stackoverflow 來寫簡單的步驟解決

http://stackoverflow.com/questions/16670658/python-variance-of-a-list-of-defined-numbers

```
grades = [(‘john’, 100), 100, 90, 40, 80, 100, 85, 70, 90, 65,
90, 85, 50.5]

def grades_sum(my_list):
    total = 0
    for grade in my_list:
        total += grade
    return total

def grades_average(my_list):
    sum_of_grades = grades_sum(my_list)
    average = sum_of_grades / len(my_list)
    return average

def grades_variance(my_list, average):
    variance = 0
    for i in my_list:
        variance += (average - my_list[i]) ** 2
    return variance / len(my_list)
```


Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學
National Tsing Hua University

Pythonic Way (1) Generate skip bigrams

words = tokens('language plays an important role in learning')

```
>>> [ g for d in range(1, 6) for g in zip(words, words[d:], [d]*len(words)) ]
```

('language', 'plays', 1), ('plays', 'an', 1), ('an', 'important', 1), ('important', 'role', 1), ('role', 'in', 1), ('in', 'learning', 1), ('language', 'an', 2), ('plays', 'important', 2), ('an', 'role', 2), ('important', 'in', 2), ('role', 'learning', 2), ('language', 'important', 3), ('plays', 'role', 3), ('an', 'in', 3), ('important', 'learning', 3), ('language', 'role', 4), ('plays', 'in', 4), ('an', 'learning', 4), ('language', 'in', 5), ('plays', 'learning', 5)]

```
>>> [ g for d in range(1, 6) for g in zip(words[d:], words, [-d]*len(words)) ]
```

[('plays', 'language', -1), ('an', 'plays', -1), ('important', 'an', -1), ('role', 'important', -1), ('in', 'role', -1), ('learning', 'in', -1), ('an', 'language', -2), ('important', 'plays', -2), ('role', 'an', -2), ('in', 'important', -2), ('learning', 'role', -2), ('important', 'language', -3), ('role', 'plays', -3), ('in', 'an', -3), ('learning', 'important', -3), ('role', 'language', -4), ('in', 'plays', -4), ('learning', 'an', -4), ('in', 'language', -5), ('learning', 'plays', -5)]

Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學 National Tsing Hua University

Pythonic Way (2) Sort/group skipgrams

```
from itertools import groupby
def gen_high_counts( counter ):
    ____________________
def sum_skips(skip):
    ____________________
skips.sort(key=lambda x: x)
for head, skips1 in groupby(skips, key=lambda x: x[0]):
    collCounts, collSkips = [], {}
    for collocate, skips2 in groupby(skips1, key=lambda x: x[1]):
        total_counts = ____________________
        collCounts += [  (collocate, total_count) ]
        collSkips[ collocate ] = sum_skips(skip2)

    goodColls = gen_high_counts( dict(collCount) )
    ____________________
    goodSkips = gen_high_counts( ___________ )
```


Entire contents © 2011 Jason S. Chang. All rights reserved.
國立清華大學
National Tsing Hua University

test.defaultdict.py

```
from collections import defaultdict
dic = defaultdict(lambda: defaultdict(lambda: defaultdict(lambda: 0)))
for d in [x-5 for x in range(10) if x != 5]:
    dic['play']['role'][d] += 100+d
print dic['play']['role']print dic[‘play']['role']
```

```
$ python test.defaultdict.py

defaultdict(<function <lambda> at 0x1004cc6e0>,
{1: 101, 2: 102, 3: 103, 4: 104, -2: 98, -5: 95, -4: 96, -3: 97, -1: 99})
```


Entire contents © 2011 Jason S. Chang. All rights reserved.