資料探勘
DATA MINING

張家瑋 博士
助理教授
國立臺中科技大學資訊工程系

NATURAL LANGUAGE PROCESSING
自然語言處理的原理與應用

自然語言處理的主要範疇

- 機器翻譯 (Machine Translation)
- 自然語言理解/語意分析 (Natural Language Understanding / Semantic Analysis)
  1. 問答系統 (Question Answering)
  2. 萃取式摘要 (Extractive Summarization)
  3. 文件分類 (Text Categorization)
- 自然語言生成 (Natural Language Generation)
  1. 進階問答系統 (Advanced Question Answering)
  2. 抽象式摘要 (Abstractive Summarization)
  3. 聊天機器人 (Chatbot)
- 語法分析 (Syntactic Parsing)
  1. 中文斷詞 (Chinese word segmentation)
  2. 詞性標註 (Part-of-speech Tagging)
  3. 實體辨識 (Named Entity Recognition)
  4. 詞彙依存 (Typed Dependencies)
  5. 文法樹 (Parse Tree)
- 語音辨識 (Speech Recognition)
- 文字轉語音 (Text to Speech)
- 語音轉文字 (Speech to Text)

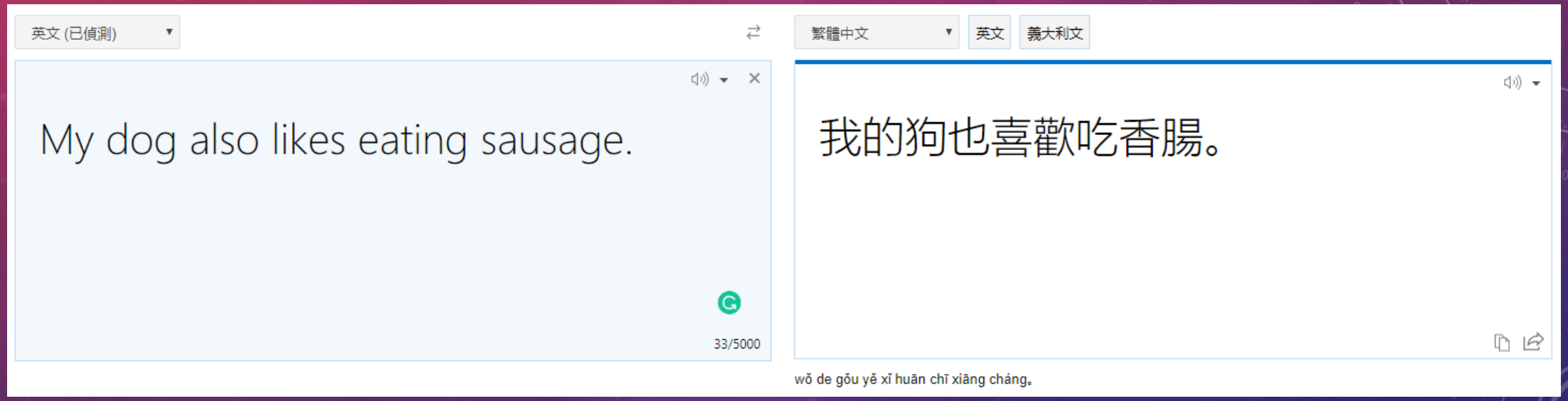
機器翻譯

MACHINE TRANSLATION

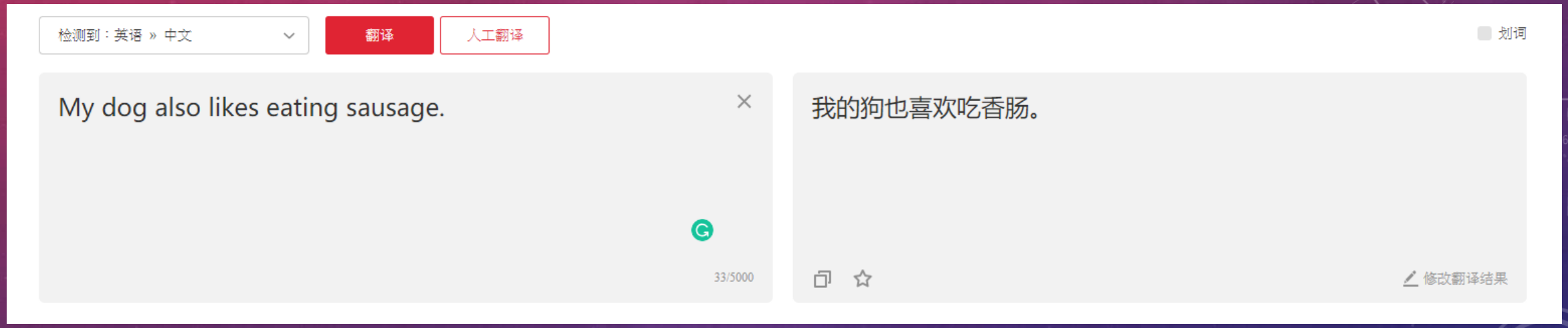
GOOGLE 翻譯


Google
翻譯
關閉即時翻譯
英文
中文
日文
偵測語言
中文(繁體)
英文
中文(簡體)
翻譯
My dog also likes eating sausage.
33/5000
我的狗也喜歡吃香腸。
提出修改建議
Wǒ de gǒu yě xǐhuān chī xiāngcháng.

BING 翻譯

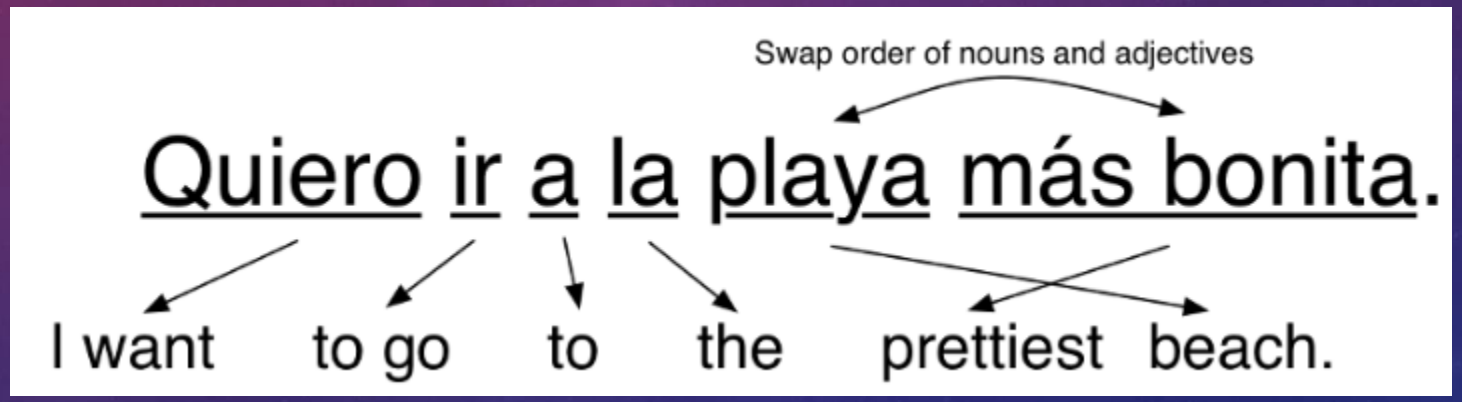

英文 (已偵測)
My dog also likes eating sausage.
33/5000
繁體中文
英文
義大利文
我的狗也喜歡吃香腸。
wǒ de gǒu yě xǐ huān chī xiāng cháng。

有道翻譯


检测到：英语 » 中文
翻译
人工翻译
划词
My dog also likes eating sausage.
33/5000
我的狗也喜欢吃香肠。
修改翻译结果

平行語料

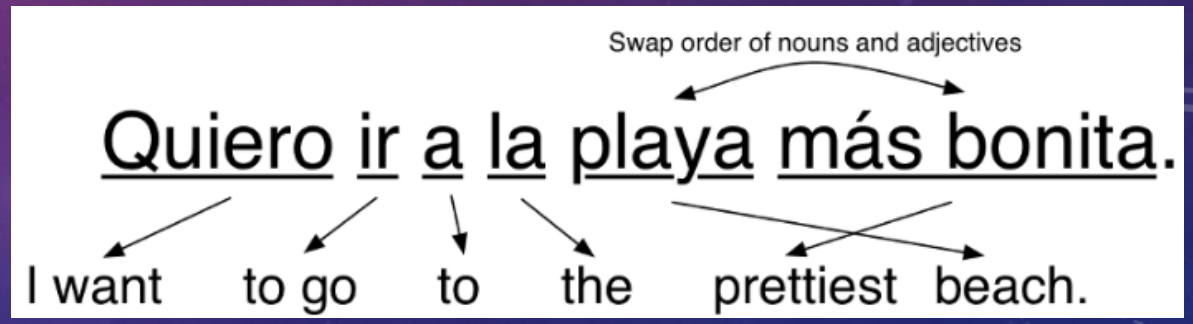

Quiero ir a la playa más bonita.
I want to go to the beach more pretty.
We just replace each Spanish word with the matching English word.


Swap order of nouns and adjectives
Quiero ir a la playa más bonita.
I want to go to the prettiest beach.

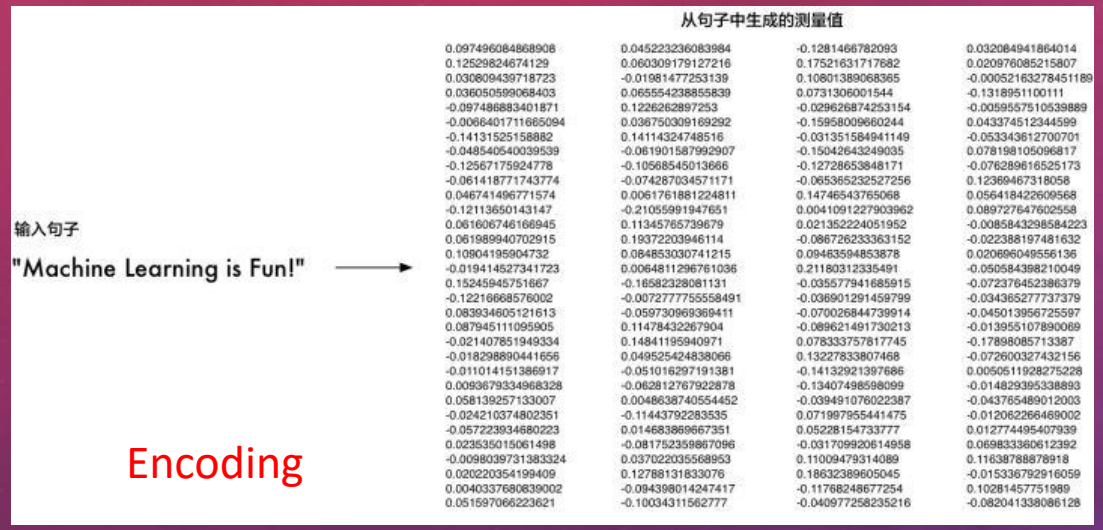
統計式機器翻譯之原理

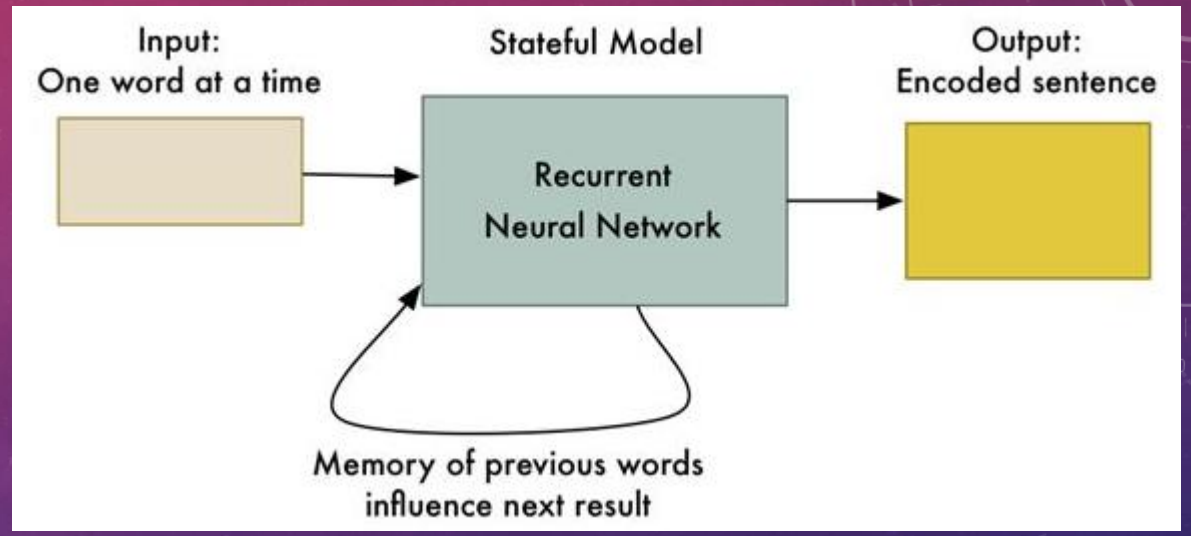

Quiero ir a la playa más bonita.
I want
to go
to
the
beach
more
pretty.
We just replace each Spanish word with the matching English word.

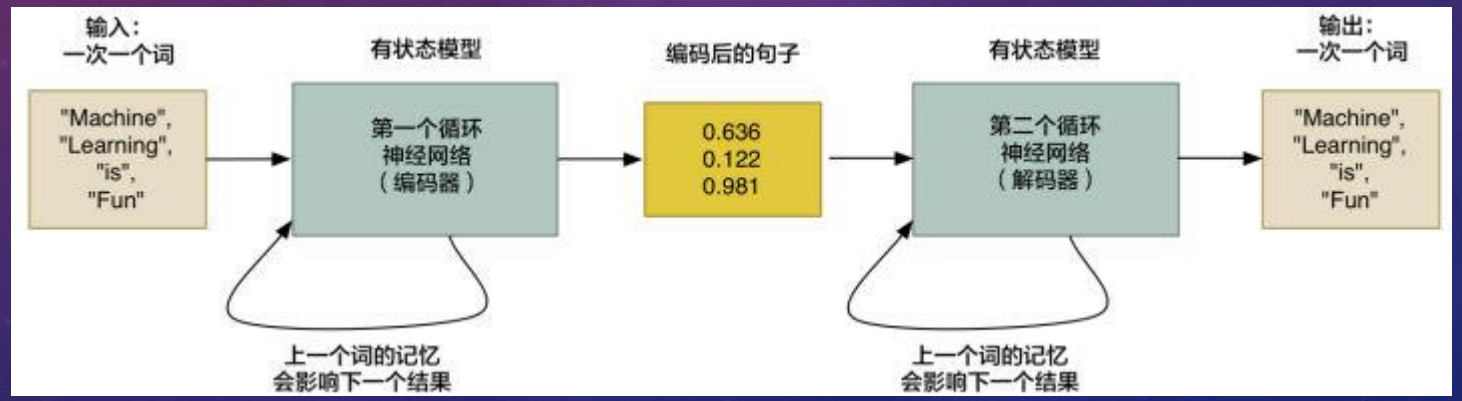

Quiero ir a la playa más bonita.
- I want
- I love
- I like
- I try
- I mean
- to go
- to work
- to run
- to appear
- to be on
- to be
- to leave
- to pass away
- to forget
- to
- at
- per
- the beach
- the seaside
- the open space
- more pretty
- most pretty
- more lovely
- most lovely
- more tidy
- most tidy
Even the most common phrases have lots of possible translations.


Swap order of nouns and adjectives
Quiero ir a la playa más bonita.
I want
to go
to
the
prettiest
beach.

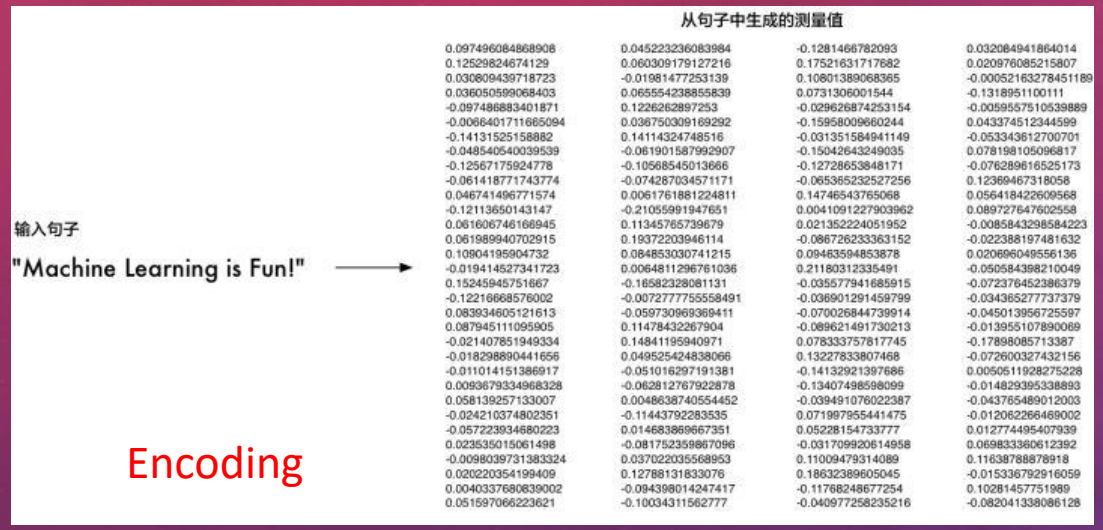
深度學習於機器翻譯之原理

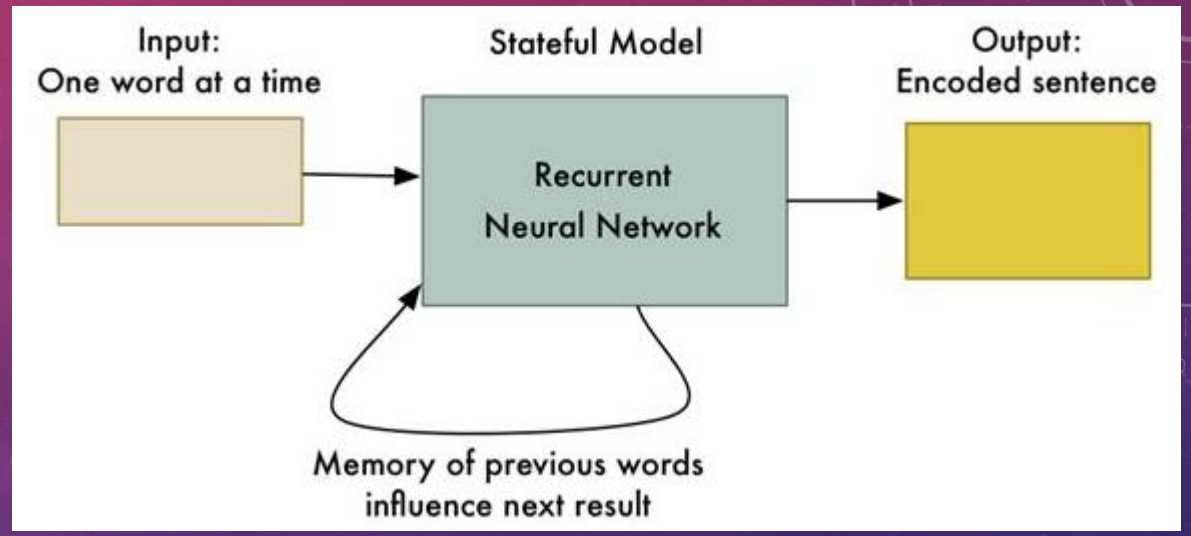


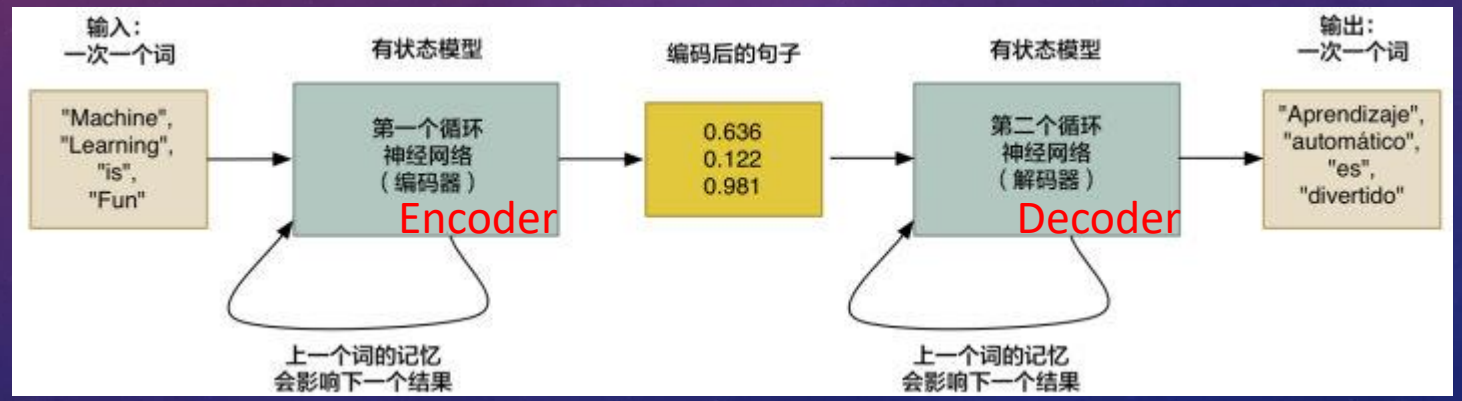




深度學習於機器翻譯之原理







自然語言理解

NATURAL LANGUAGE UNDERSTANDING

WORD-SENSE DISAMBIGUATION

- Ambiguity: a word or phrase with multiple meanings.
  1. “procure” (I will get the drinks)
  2. “become” (she got scared)
  3. “have” (I have got three dollars)
  4. “understand” (I get it)

WORDNET

WordNet Search - 3.1
- WordNet home page - Glossary - Help

Word to search for: apple Search WordNet

Display Options: (Select option to change) Change

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

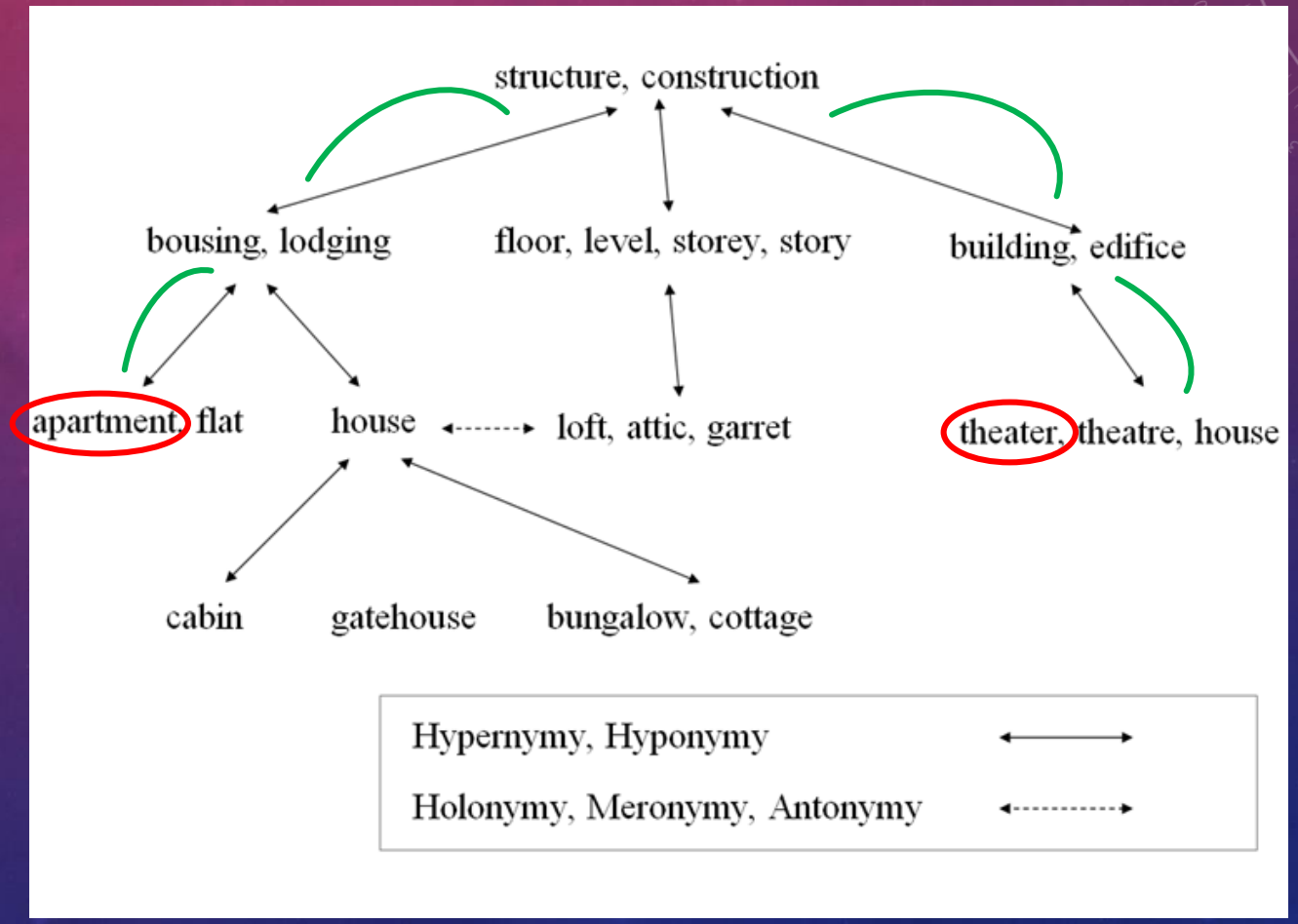
Noun

- S: (n) **apple** (fruit with red or yellow or green skin and sweet to tart crisp whitish flesh)
- S: (n) **apple**, orchard apple tree, Malus pumila (native Eurasian tree widely cultivated in many varieties for its firm rounded edible fruits)

http://wordnetweb.princeton.edu/perl/webwn

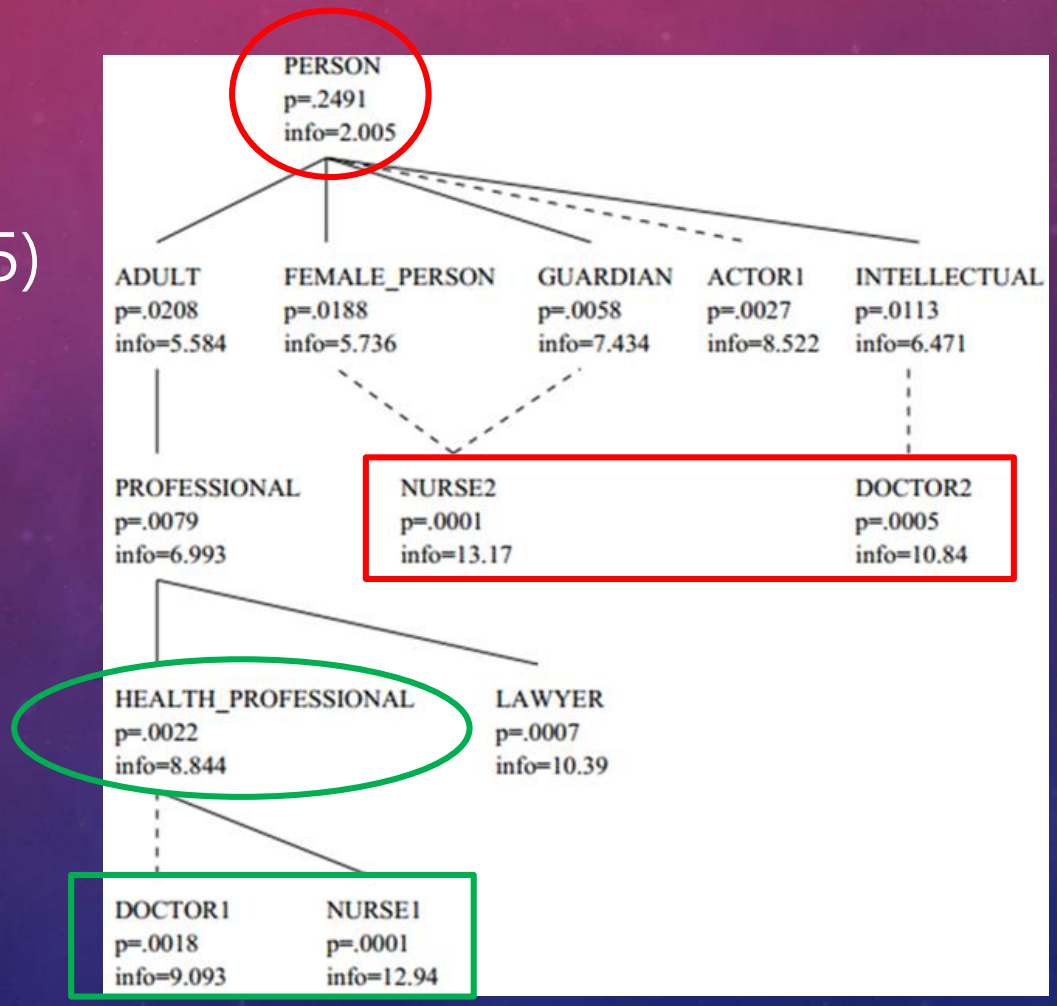
WORD-SENSE DISAMBIGUATION

- Distance-based: PATH (Rada, Mili, Bicknell, & Blettner, 1989)

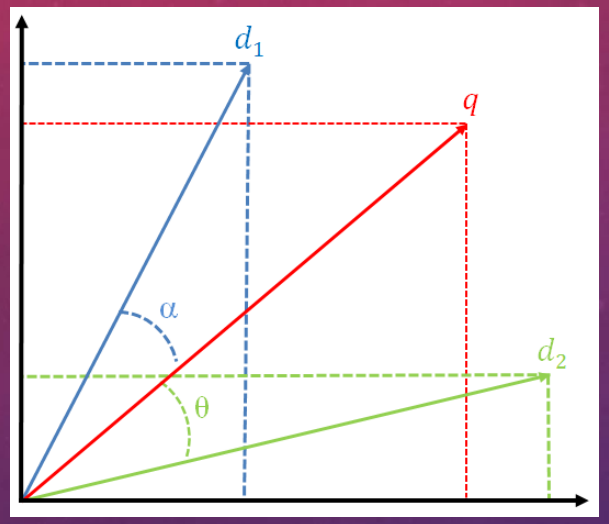


WORD-SENSE DISAMBIGUATION

- Information Content-based: RES (Resnik, 1995)



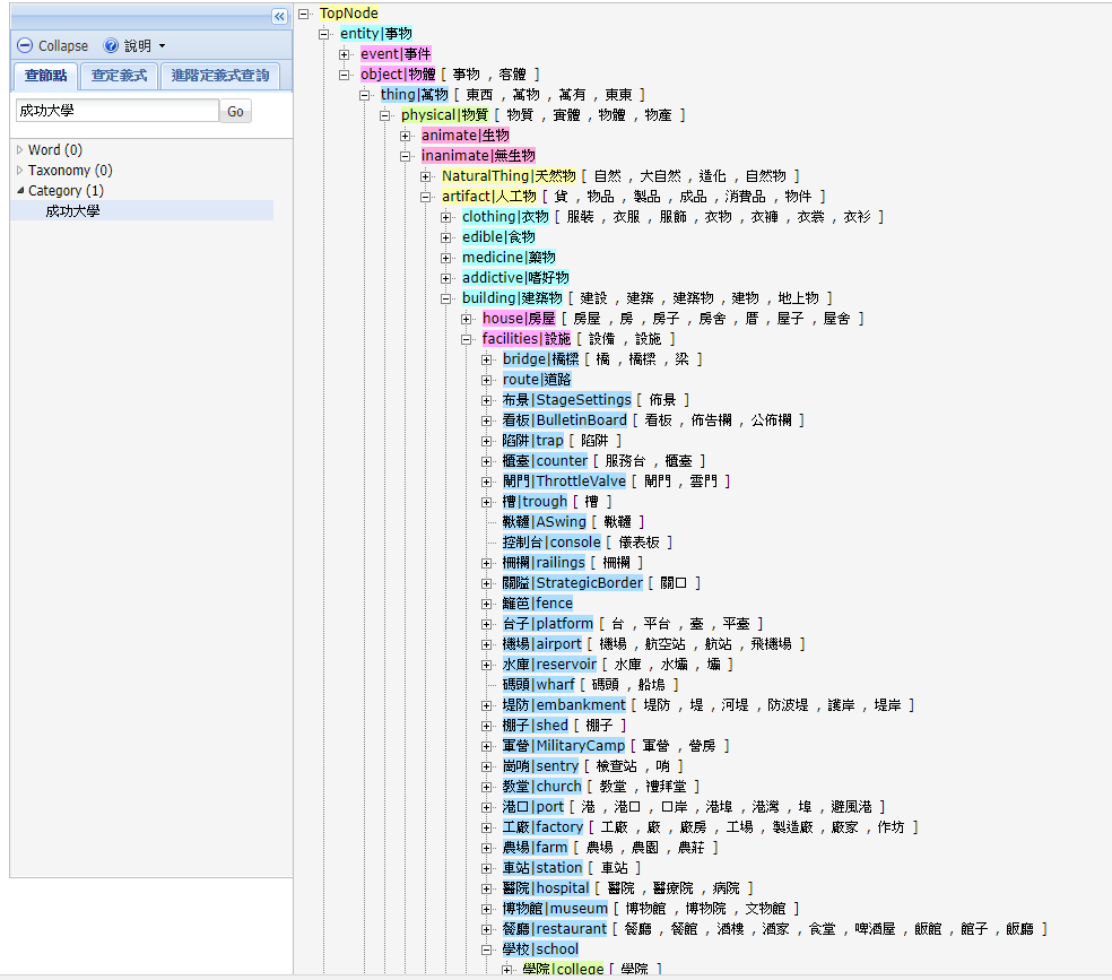
WORD-SENSE DISAMBIGUATION



- Gloss-based: VECTOR (Patwardhan, 2003)

Cute	Cunning
1. attractive especially by means of smallness or prettiness or quaintness	1. attractive especially by means of smallness or prettiness or quaintness
2. obviously contrived to charm	2. marked by skill in deception
	3. showing inventiveness and skill

廣義知網知識本體



http://ehownet.iis.sinica.edu.tw/ehownet.php

VECTOR REPRESENTATION

	w_1	w_2	w_3	..	..	..	w_{n-1}	w_n	label
D_1	0.11	0.23	0	..	..	..	0.57	0	0
D_2	0	0	0	..	..	..	0.29	0.7	1
D_3	0	0.81	0.44	..	..	..	0	0	0
D_4	0	0.37	0	..	..	..	0	0.16	1
..	..	..	..	..	..	..	..	..	..
D_k	..	..	..	..	..	..	..	..	1

TF-IDF

- TF: term frequency: $\mathrm{tf}_{\mathrm{i,j}} = \frac{n_{i,j}}{\sum_k n_{k,j}}$
- IDF: inverse document frequency: $\mathrm{idf}_{\mathrm{i}} = \log \frac{|D|}{|\{j : t_i \in d_j\}|}$

where:

- $|D|$: total number of documents in the corpus
- $|\{j : t_i \in d_j\}|$: number of documents where term t_i appears

Then:

- $\mathrm{tfidf}_{\mathrm{i,j}} = \mathrm{tf}_{\mathrm{i,j}} \times \mathrm{idf}_{\mathrm{i}}$

Document 1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

Document 2	
Term	**Term Count**
this	1
is	1
another	2
example	3

- The calculation of tf–idf for the term "this" is performed as follows:
  - $$\mathrm{tf}(''\text{this}'', d_1) = \frac{1}{5} = 0.2$$
    $$\mathrm{tf}(''\text{this}'', d_2) = \frac{1}{7} \approx 0.14$$
    $$\mathrm{idf}(''\text{this}'', D) = \log\left(\frac{2}{2}\right) = 0$$
  - So tf–idf is zero for the word "this", which implies that the word is not very informative as it appears in all documents.

$$\mathrm{tfidf}(''\text{this}'', d_1) = 0.2 \times 0 = 0$$
$$\mathrm{tfidf}(''\text{this}'', d_2) = 0.14 \times 0 = 0$$

Document 1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

Document 2	
Term	**Term Count**
this	1
is	1
another	2
example	3

- A slightly more interesting example arises from the word "example", which occurs three times only in the second document:
  - $$\mathrm{tf}(''\text{example}'', d_1) = \frac{0}{5} = 0$$
  - $$\mathrm{tf}(''\text{example}'', d_2) = \frac{3}{7} \approx 0.429$$

$$\mathrm{idf}(''\text{example}'', D) = \log\left(\frac{2}{1}\right) = 0.301$$

$$\mathrm{tfidf}(''\text{example}'', d_1) = \mathrm{tf}(''\text{example}'', d_1) \times \mathrm{idf}(''\text{example}'', D) = 0 \times 0.301 = 0$$
$$\mathrm{tfidf}(''\text{example}'', d_2) = \mathrm{tf}(''\text{example}'', d_2) \times \mathrm{idf}(''\text{example}'', D) = 0.429 \times 0.301 \approx 0.13$$

潛藏語意分析(LSA)

- 奇異值分解
  - Singular Value Decomposition (SVD)

Index Words	Titles								
	T1	T2	T3	T4	T5	T6	T7	T8	T9
book			1	1					
dads						1			1
dummies		1						1	
estate							1		1
guide	1					1			
investing	1	1	1	1	1	1	1	1	1
market	1		1						
real							1		1
rich						2			1
stock	1		1					1	
value				1	1				

=

book	0.15	-0.27	0.04
dads	0.24	0.38	-0.09
dummies	0.13	-0.17	0.07
estate	0.18	0.19	0.45
guide	0.22	0.09	-0.46
investing	0.74	-0.21	0.21
market	0.18	-0.30	-0.28
real	0.18	0.19	0.45
rich	0.36	0.59	-0.34
stock	0.25	-0.42	-0.28
value	0.12	-0.14	0.23

*

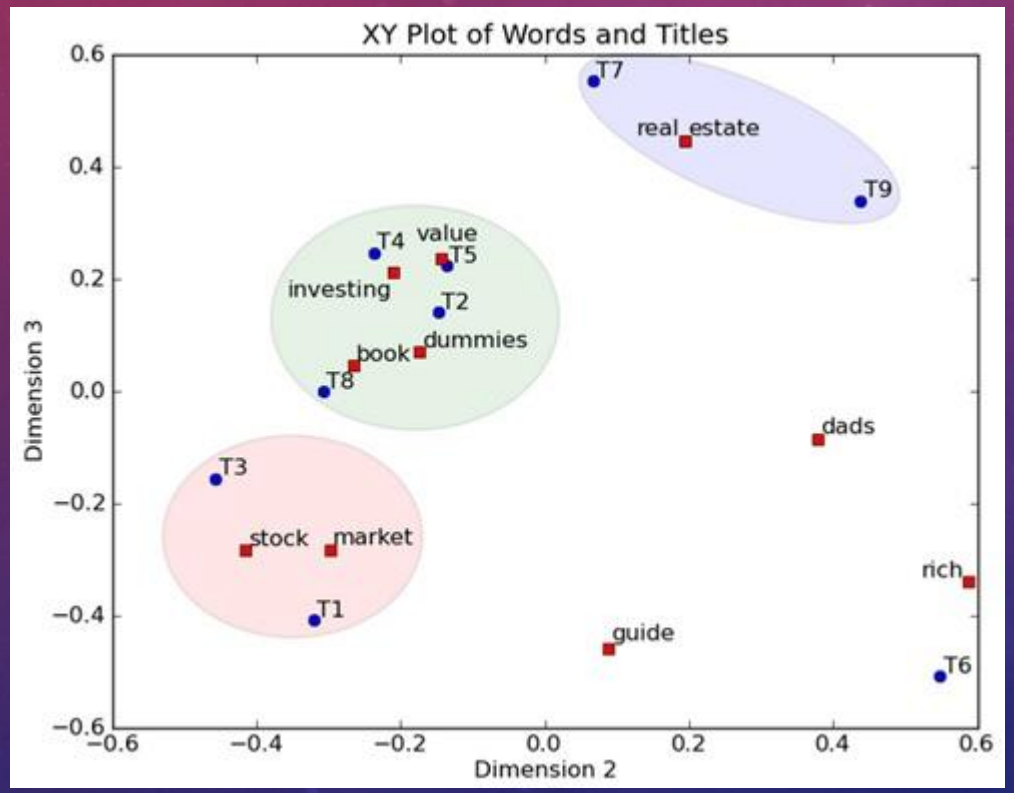
3.91	0	0
0	2.61	0
0	0	2.00

*

T1	T2	T3	T4	T5	T6	T7	T8	T9
0.35	0.22	0.34	0.26	0.22	0.49	0.28	0.29	0.44
-0.32	-0.15	-0.46	-0.24	-0.14	0.55	0.07	-0.31	0.44
-0.41	0.14	-0.16	0.25	0.22	-0.51	0.55	0.00	0.34

潛藏語意分析(LSA)

- 文件分類/主題探勘
- 語意分析



Index Words	Titles								
	T1	T2	T3	T4	T5	T6	T7	T8	T9
book			1	1					
dads						1			1
dummies		1						1	
estate							1		1
guide	1					1			
investing	1	1	1	1	1	1	1	1	1
market	1		1						
real							1		1
rich						2			1
stock	1		1					1	
value				1	1				

文字檔案

word2vec

透過向量比對
找出相似的資料

Input:
one document

word
vectors

將被拆解成多個字元

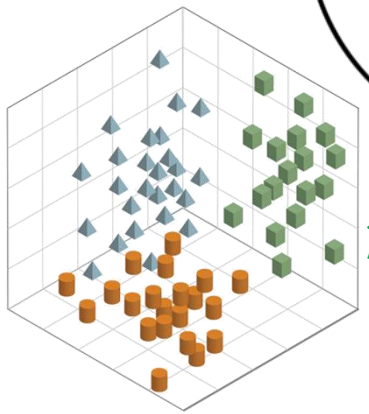
Model:

vector space

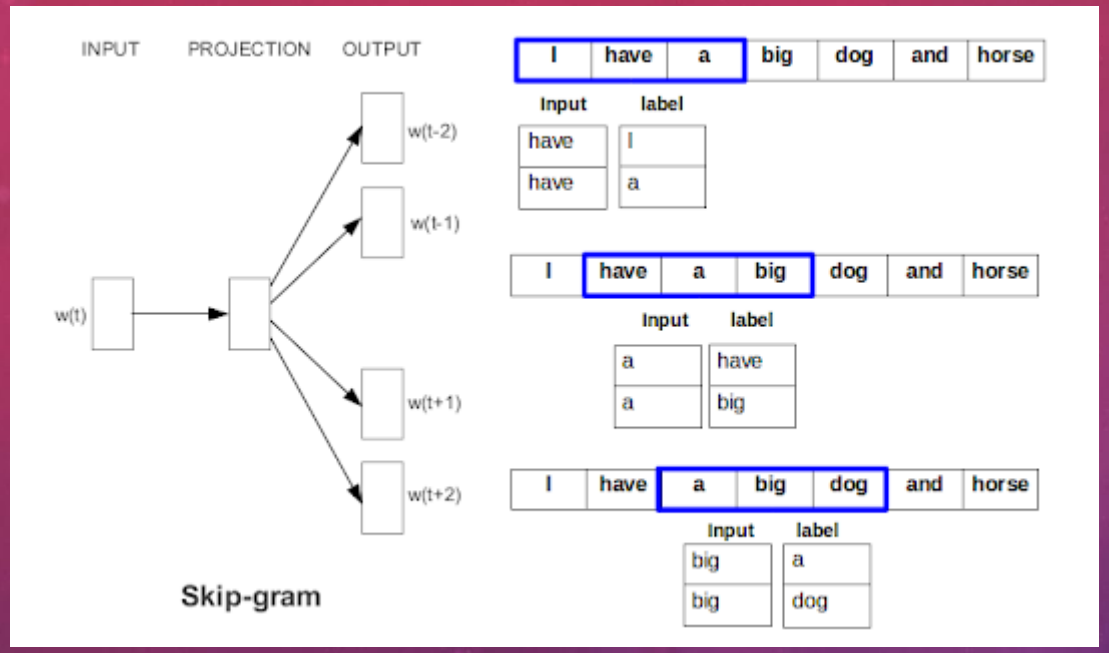
most_similar('france'):

spain	0.678515
belgium	0.665923
netherlands	0.652428
italy	0.633130

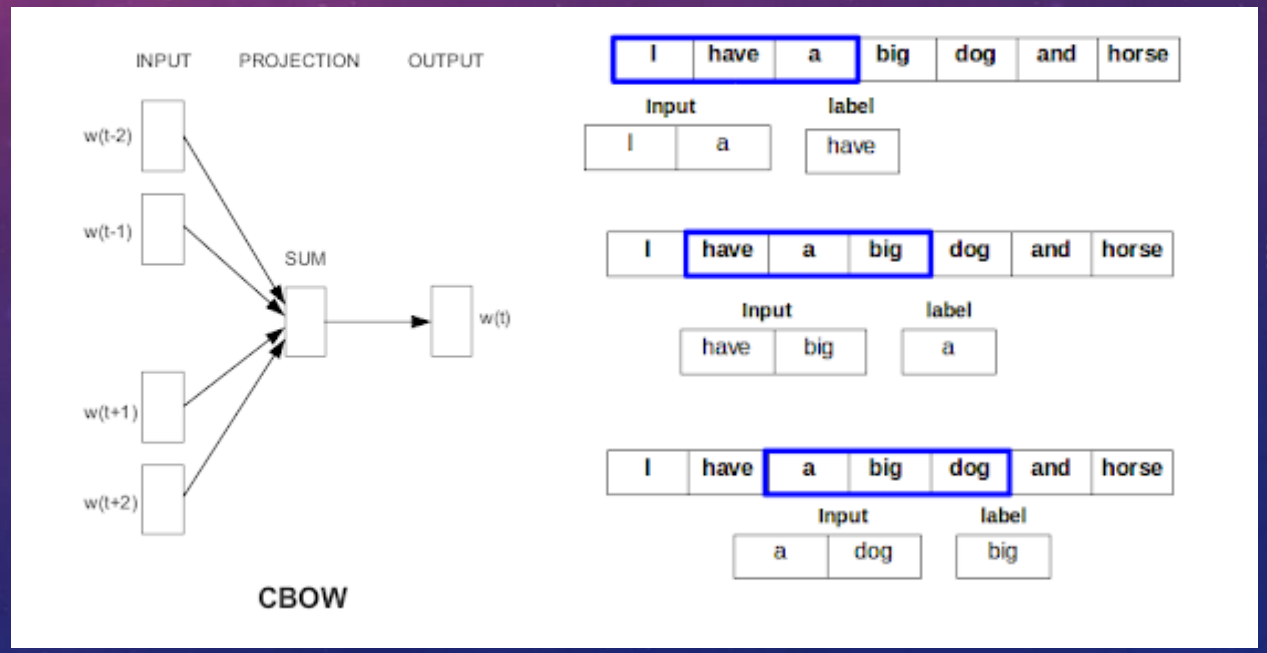
highest cosine
distance values
in vector space
of the nearest
words

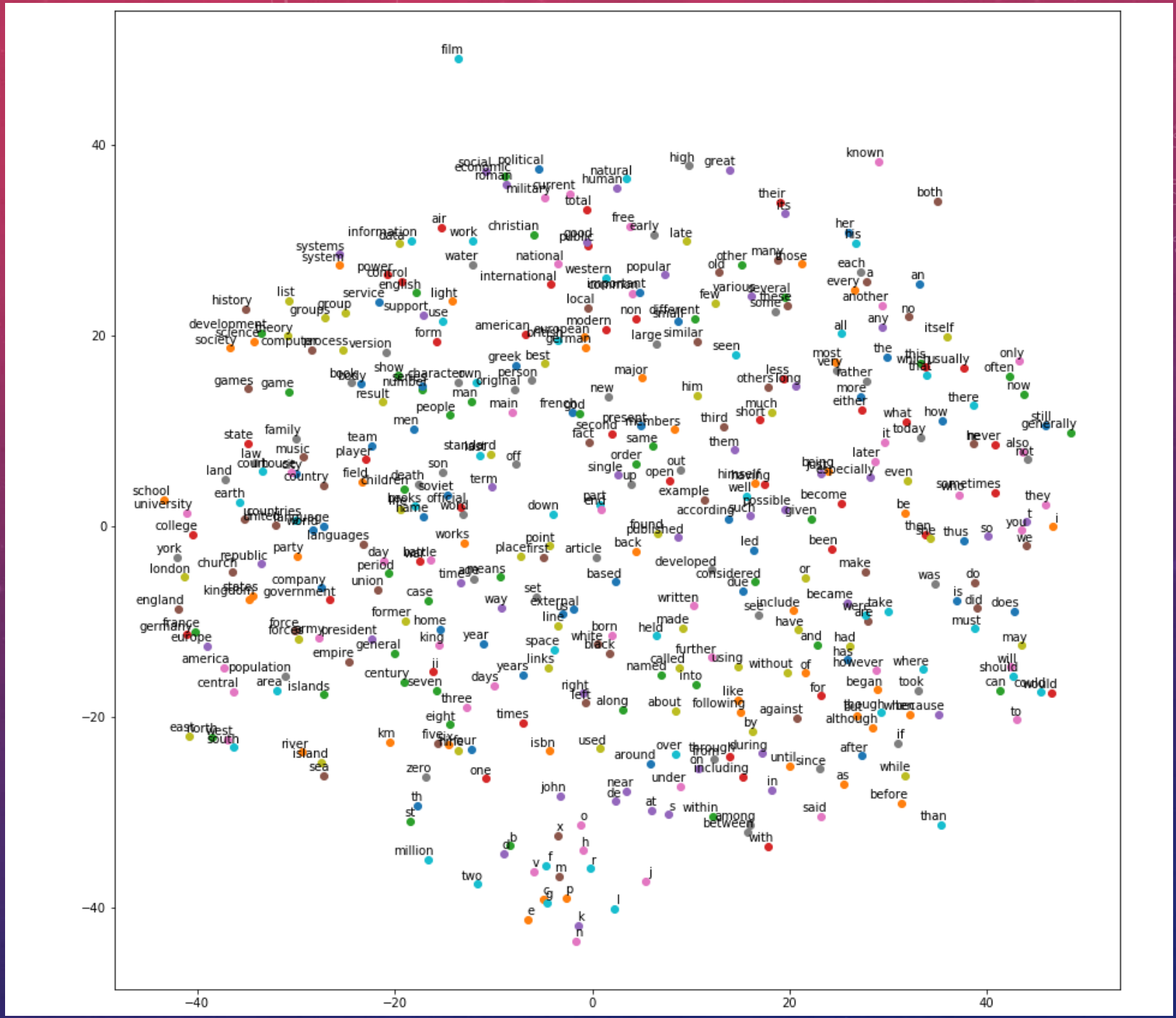


解析成多元維度的向量



http://zongsoftwarenote.blogspot.com/2017/04/word2vec-model-introduction-skip-gram.html




40
20
0
−20
−40
−40
−20
0
20
40
film
social
political
economic
roman
military
current
human
natural
high
great
known
total
their
its
both
air
work
christian
free
early
late
information
data
good
public
her
his
systems
system
water
national
other
many
those
each
every
a
an
power
control
international
western
important
popular
old
various
several
these
some
few
another
no
list
service
light
history
group
groups
support
use
local
non
different
small
any
development
theory
science
form
american
european
modern
all
itself
society
computer
process
version
greek
best
large
similar
seen
most
very
the
this
which
that
usually
only
often
show
character
own
person
major
less
others
rather
more
now
games
game
book
body
number
original
result
man
new
him
either
there
people
main
french
much
short
what
how
still
generally
men
second
present
members
third
today
it
never
also
state
family
team
fact
same
them
later
not
law
music
player
standard
off
order
being
country
field
son
death
single
up
open
out
himself
having
especially
even
land
children
soviet
term
school
earth
books
official
part
example
well
sometimes
university
name
word
end
down
possible
become
they
countries
according
such
given
be
college
language
found
published
then
she
thus
so
you
t
i
we
languages
works
point
back
led
been
york
party
day
battle
place
first
article
developed
make
republic
church
period
time
age
means
based
considered
or
was
do
london
union
due
is
company
states
kingdom
government
case
set
way
external
written
include
became
take
did
does
england
former
line
made
have
were
are
must
france
germany
europe
force
army
president
home
king
year
space
white
black
born
held
and
had
may
general
further
has
america
empire
links
called
using
without
of
however
where
will
should
population
years
named
into
central
area
islands
century
seven
days
right
left
like
began
took
can
could
three
along
about
following
against
for
though
because
although
to
eastorth
west
south
eight
times
by
if
km
five
six
four
isbn
used
river
island
sea
around
over
during
after
zero
one
including
until
since
while
under
as
in
near
john
de
before
at
s
within
said
than
th
st
among
between
with
o
x
b
h
million
v
f
m
r
j
two
g
p
l
e
k
n

語法分析
SYNTACTIC PARSING

STANFORD PARSER

Stanford Parser

Please enter a sentence to be parsed:

My dog also likes eating sausage.

Language: English Sample Sentence Parse

Your query

My dog also likes eating sausage.

Tagging

```
My/PRP$  dog/NN  also/RB  likes/VBZ  eating/VBG  sausage/NN  ./.
```

Parse

```
(ROOT
  (S
    (NP (PRP$ My) (NN dog))
    (ADVP (RB also))
    (VP (VBZ likes)
      (S
        (VP (VBG eating)
          (NP (NN sausage)))))
    (. .)))
```

Universal dependencies

```
nmod:poss(dog-2, My-1)
nsubj(likes-4, dog-2)
advmod(likes-4, also-3)
root(ROOT-0, likes-4)
xcomp(likes-4, eating-5)
dobj(eating-5, sausage-6)
```

The Stanford Natural Language Processing Group people publications research blog software teaching local

Software > Stanford Parser

The Stanford Parser: A statistical parser

About | Citing | Questions | Download | Included Tools | Extensions | Release history | Sample output | Online | FAQ

About

A natural language parser is a program that works out the grammatical **structure of sentences**, for instance, which groups of words go together (as "phrases") and which words are the **subject** or **object** of a verb. Probabilistic parsers use knowledge of language gained from hand-parsed sentences to try to produce the *most likely* analysis of new sentences. These statistical parsers still make some mistakes, but commonly work rather well. Their development was one of the biggest breakthroughs in natural language processing in the 1990s. You can try out our parser online.

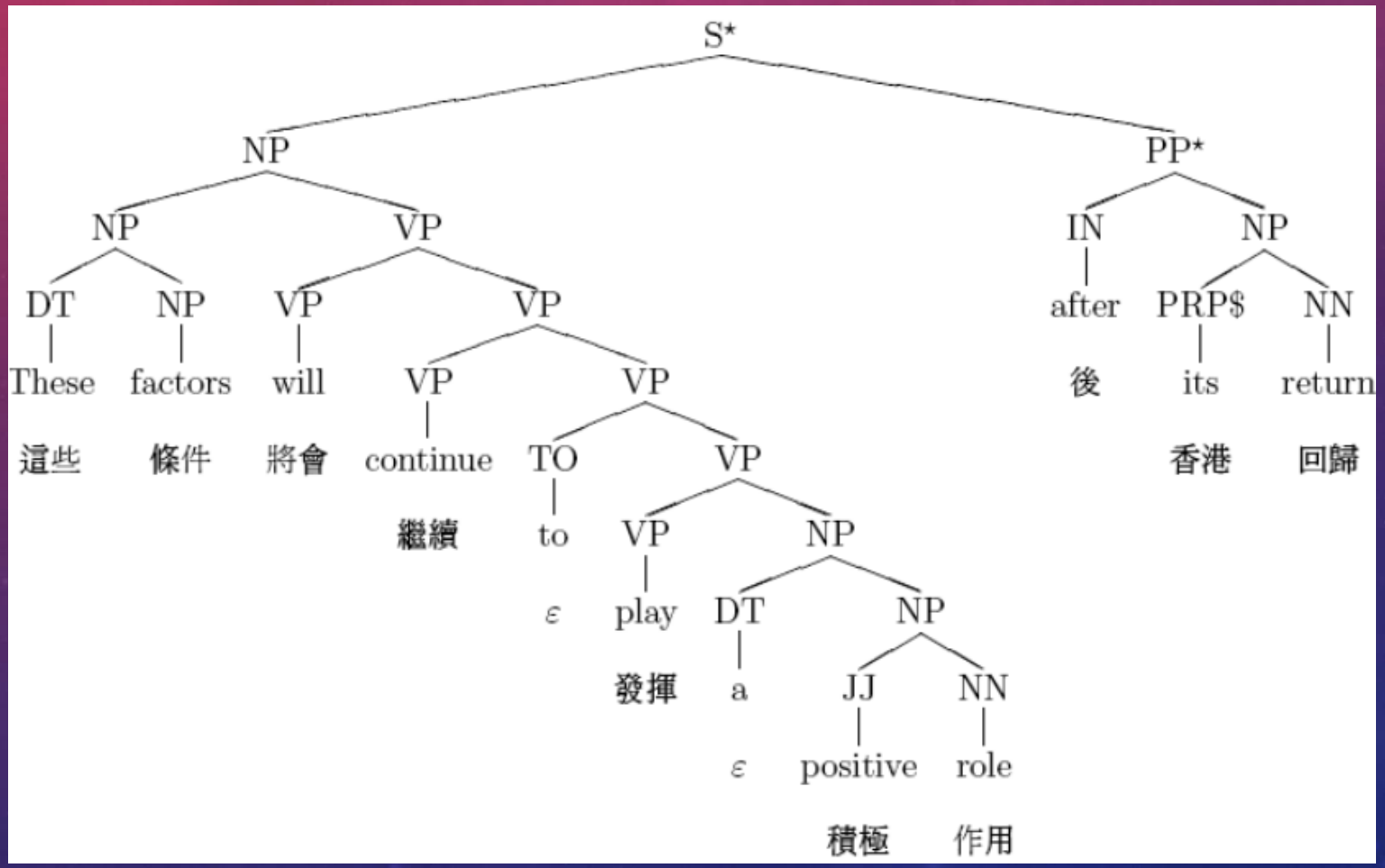
Package contents

This package is a Java implementation of probabilistic natural language parsers, both highly optimized PCFG and lexicalized dependency parsers, and a lexicalized PCFG parser. The original version of this parser was mainly written by Dan Klein, with support code and linguistic grammar development by Christopher Manning. Extensive additional work (internationalization and language-specific modeling, flexible input/output, grammar compaction, lattice parsing, *k*-best parsing, typed dependencies output, user support, etc.) has been done by Roger Levy, Christopher Manning, Teg Grenager, Galen Andrew, Marie-Catherine de Marneffe, Bill MacCartney, Anna Rafferty, Spence Green, Huihsin Tseng, Pi-Chuan Chang, Wolfgang Maier, and Jenny Finkel.

The lexicalized probabilistic parser implements a factored product model, with separate PCFG phrase structure and lexical dependency experts, whose preferences are combined by efficient exact inference, using an A* algorithm. Or the software can be used simply as an accurate unlexicalized stochastic context-free grammar parser. Either of these yields a good performance statistical parsing system. A GUI is provided for viewing the phrase structure tree output of the parser.

- http://nlp.stanford.edu:8080/parser/

PARSE TREE


S*
NP
PP*
NP
VP
IN
NP
DT
NP
VP
VP
after
PRP$
NN
These
factors
will
VP
VP
後
its
return
這些
條件
將會
continue
TO
VP
香港
回歸
繼續
to
VP
NP
ε
play
DT
NP
發揮
a
JJ
NN
ε
positive
role
積極
作用

中研院

傅達仁(Nb) 今(Nd) 將(D) 執行(VC) 安樂死(Na) ，(COMMACATEGORY)

卻(D) 突然(D) 爆出(VJ) 自己(Nh) ２０(Neu) 年前(Nd) 遭(P) 緯來(Nb) 體育台(Nc) 封殺(VC) ，(COMMACATEGORY)

他(Nh) 不(D) 懂(VK) 自己(Nh) 哪裡(D) 得罪(VC) 到(P) 電視台(Nc) 。(PERIODCATEGORY)

實體辨識

傅達仁PERSON今將執行安樂死，卻突然爆出自己20年前遭緯來體育台封殺，他不懂自己哪裡得罪到電視台。

指代消解

傅達仁今將執行安樂死，NULL傅達仁卻突然爆出自己20年前遭緯來體育台封殺，他傅達仁不懂自己哪裡得罪到電視台。

show all　show phrase head　show word head

agent(執行_VC2)=傅達仁_NP
time(執行_VC2)=今_Ndabd
time(執行_VC2)=將_Dd
goal(執行_VC2)=安樂死_NP
theme(爆出_VJ3)=傅達仁_Nba
evaluation(爆出_VJ3)=卻_Dbb
time(爆出_VJ3)=突然_Dd
range(爆出_VJ3)=自己20年前遭緯_NP
complement(爆出_VJ3)=來體育台封殺_VP
experiencer(懂_VK1)=傅達仁_NP
negation(懂_VK1)=不_Dc
goal(懂_VK1)=自己哪裡得罪到電視台_S

agent(執行_VC2)=theme(爆出_VJ3), 1
agent(執行_VC2)=experiencer(懂_VK1), 1
theme(爆出_VJ3)=experiencer(懂_VK1), 1

中研院CKIP Lab – 繁體中文剖析工具
http://ckip.iis.sinica.edu.tw:8080/demo/

FUDANNLP

```
D:\fnlp>java -Xmx1024m -Dfile.encoding=UTF-8 -classpath "fnlp-core/target/fnlp-c
ore-2.1-SNAPSHOT.jar;libs/trove-3.1a1.jar;libs/commons-cli-1.4.jar" org.fnlp.nlp
.cn.tag.POSTagger -s models/seg.m models/pos.m "周杰伦出生于台湾，生日为79年1月1
8日，他曾经的绯闻女友是蔡依林。"
周杰伦/人名 出生于/动词 台湾/地名 ，/名词 生日/名词 为/介词 79年/时间短语 1月/时
间短语 18日/时间短语 ，/动词 他/人称代词 曾经/形容词 的/结构助词 绯闻/名词 女友/
名词 是/动词 蔡依林/人名 。/标点
```



```
D:\fnlp>java -Xmx1024m -Dfile.encoding=UTF-8 -classpath "fnlp-core/target/fnlp-c
ore-2.1-SNAPSHOT.jar;libs/trove-3.1a1.jar;libs/commons-cli-1.4.jar" org.fnlp.nlp
.cn.tag.NERTagger -s models/seg.m models/pos.m "詹姆斯·默多克和丽贝卡·布鲁克斯
 鲁珀特·默多克旗下的美国小报《纽约邮报》的职员被公司律师告知，保存任何也许与电
话窃听及贿赂有关的文件。"
<美国=地名，纽约=地名，詹姆斯·默多克=人名，鲁珀特·默多克=人名，丽贝卡·布鲁克
斯=人名>
```


復旦NLP - 簡體中文剖析工具
https://blog.csdn.net/hhu_lyc/article/details/79179619

以語言學習輔助工具為例

Collocation online suggestion v1.0

英語搭配詞線上檢索系統

介紹　常用搭配詞查詢　整句搭配詞查詢與推薦

整句搭配詞查詢與推薦

輸入句子： We commonly use a small cell for medical research.　清除

送出

輸入的句子為

We commonly use a small cell for medical research .

副詞修飾(V/Adv/Adj組合)

commonly + V/Adv/Adj

#	collocation	freq(%)
1	commonly use	46.5
2	commonly used	4.7
3	commonly find	4.4
4	commonly know	3.3
5	commonly employ	2.4
6	commonly refer	2.2
7	commonly observe	1.9
8	commonly report	1.9
9	commonly encounter	1.4
10	commonly available	1.3

commonly與use的搭配字同義組合

commonly + 搭配同義字

#	collocation	freq(%)
1	commonly use	46.5
2	commonly employ	2.4
3	commonly apply	0.5

同義詞搭配詞組搜尋結果

commonly的同義字 + use 的同義字

#	collocation	count
1	commonly use	296
2	often use	140
3	frequently use	68
4	commonly employ	15
5	frequently employ	9
6	often employ	6
7	frequently apply	5
8	repeatedly use	5
9	routinely use	5
10	frequently utilize	4
11	routinely employ	3
12	commonly apply	3

查詢總時間:0.52sec

以語言學習輔助工具為例

	computer	data	pinch	result	sugar
aprocot	0	0	1	0	1
pineapple	0	0	1	0	1
digital	2	1	0	1	0
information	1	6	0	4	0

$$P(x = information, y = data) = \frac{6}{19} = 0.32$$

$$P(x = information) = \frac{6+4+1}{19} = \frac{11}{19} = 0.58$$

$$P(y = data) = \frac{6+1}{19} = \frac{7}{19} = 0.37$$

$$\begin{aligned} & pmi(x = information, y = data) \\ &= log \frac{P(x = information, y = data)}{P(x = information) \times P(y = data)} \\ &= log 1.49 \\ &= 0.57 \end{aligned}$$

THANK YOU