

Week 7

Selecting Good Dictionary Examples with (Local) MapReduce

Using sentence length, readability, and collocations

GDEX: ForBetterEnglish.com

- Visit GDEX at <http://forbetterenglish.com>

difficulty

Search

difficulty (n)

pp_of	puzzle :	The level of difficulty of the puzzles can be selected to suit the audience .
object_of	learn :	The target group is young children with learning difficulties under the age of eight years .
	overcome :	Major efforts have been made to overcome difficulties with the numerous issues of the JTMP software .
	encounter :	Now all I have to do is to convert all my webpages over to the new format - I`ve encountered two difficulties .
	face :	Anyone who is prosecuted will also face difficulty in getting a licence in the future .
	experience :	There is nothing to suggest that a keen 12 year old would experience more difficulties with the theory than a keen 14 year old .
	pose :	Some Masters applicants would not know their results until September so an early start date might pose difficulties for some postgraduate degrees .
a_modifier	behavioural :	For 10 years prior to joining CWFS , he was Chairman of Governors for a residential school for children with emotional and behavioural difficulties .
	financial :	His last years were also clouded by financial difficulties , to which the last building phase at Florence Court must have materially contributed .
	mental :	Supported housing schemes for teenage parents , people with acquired brain injury and people with mental health difficulties are currently under development .
	severe :	This is mainly due to the severe difficulties in obtaining quality crystals of sufficient size to perform the experiments .
	technical :	Once a few technical difficulties were overcome , Chris Goddard 's dazzling fiddle playing had the audience enthralled .
	considerable :	Some patients might be expected to benefit from this ; others have reported considerable difficulties trying to come off Prozac itself .
modifies	breathing :	Miss Y had difficulty breathing and staff from the home took her to see a GP (who was employed by the Trust) .
	walking :	The large ground floor bedroom is easily accessible to guests in wheelchairs or those that have some difficulty walking .

GDEX Examples for DIFFICULTY

1. **difficulty of STH**

- The level of *difficulty of the* **puzzles** can be selected to suit the audience.

2. **v. ~ difficulty**

- Major efforts have been made to **overcome** *difficulties* with the numerous issues of the JTMP software.

- Now all I have to do is to convert all my webpages over to the new format - I've **encountered** *two difficulties*.
- Anyone who is prosecuted will also **face** *difficulty* in getting a licence in the future.

3. **adj. difficulty**

- or 10 years prior to joining CWFS, he was Chairman of Governors for a residential school for children with emotional and **behavioural** *difficulties*.

- His last years were also clouded by **financial** *difficulties*, to which the last building phase at Florence Court must have materially contributed.

4. **difficulty DOING**

- Miss Y had *difficulty* **breathing** and staff from the home took her to see a GP (who was employed by the Trust).
- The large ground floor bedroom is easily accessible to guests in wheelchairs or those that have some *difficulty* **walking**.

Citeseer Examples for DIFFICULTY

1. difficulty of STH

- A system is built to assess the *difficulty of the* **problem**.
- The *difficulty of the* **task** is exceeded only by its importance.

2. STH of difficulty

- We included a lot of relevant exercises of varying **degree of difficulty**.

- Security can have varying **levels** *of difficulty* for implementation.

3. v. ~ difficulty

- To **overcome** *this difficulty*, we consider an AD-movement.
- Why do physicians **have** *difficulty* in making a diagnosis?
- Robust estimation techniques are used to **address** *this difficulty*.

4. adj. difficulty

- However, this algorithm is of enormous **computational difficulty**.
- A **key difficulty** facing networks science is data acquisition.
- **Major difficulty** is found in analyzing the weighted function.

5. difficulty in DOING

- The *difficulty in applying* these procedures occurs at two levels.

- Moreover, these methods have *difficulty in* **detecting** new types of attack.

6. difficulty of **DOING**

- One reason is the *difficulty of* **designing** successful experimental evaluations.
- However, he did not make any claims about the *difficulty of* **determining** aliases.

7. difficulty **DOING**

- A key *difficulty* **facing** networks science is data acquisition.
- Have you ever had *difficulty* **finding** the purpose of a reading?

What makes a sentence a good dictionary example?

- Sentence length: with 10 to 25 words long was preferred
- Word frequencies: with words in the commonest 17,000 words.
- self-contained: not containing pronouns (e.g., this, that, it, one) *this, that*等等的代名詞，很難知道這些代名詞是在指上文的哪一個字
- Target words: with main clause containing target words
- Whole sentences: beginning with a capital letter and ending with. or ! or?

- Collocations: containing collocation of the target words
- Context: introducing a context, and then present the target words

Source: GDEX: Automatically finding good dictionary examples
in a corpus

Kilgarriff, et al. (2009).

<http://kilgarriff.co.uk/Publications/2008-KilgEtAl-euralex-gdex.doc>

Preparation

- Use MapReduce framework to do the following:
 - word count
 - ngram count
 - collocations (with distance between a head-collocate pair).
- **MapReduce** (<https://en.wikipedia.org/wiki/MapReduce>) is a programming model for processing **big data** with a **parallel, distributed** algorithm on a **cluster** of commodity personal computers.
- A MapReduce program is composed of
 - mapper: performs filtering data and generate (key, value) pair (e.g., key = word, value = count = 1)

- reducer: performs the summary operation (e.g., counting instances of a word)
- The MapReduce framework (implementation) manages the processing by running mapper and reducer tasks in parallel, controlling all communications and data transfers, and providing for redundancy and fault tolerance.
- For simplicity and convenience, we show how to do MapReduce locally and make use of the multiple CPU cores found in today's personal computer—Local MapReduce

Local MapReduce and Examples

- See <https://github.com/d2207197/local-mapreduce>
- Usage

```
./lmr <chunk size> <#reducer> <mapper> <reducer> <directory>
```

- <chunk size>: Split data into chunks with <chunk size>
- <#reducer>: Each output line from mappers would then be hashed into <num of reducer> different reducer
- <mapper>, <reducer>: Shell command/Python program
- <directory>: The output directory

Local MapReduce and a Word Count Example

- Mapper and Reducer

把a-zA-z以外的都換成\n

```
tr -sc "a-zA-Z" "\n"
```

```
uniq -c
```

如果有連續的”非a-zA-Z”，則視為一個，不然有可能會連續換行，不好

(s = **Squeeze**; c = **Complement**)

以外的

(c = add Count)

- Testing mapper

```
$ echo 'Colorless green ideas \n sleep furiously' | tr -sc "a-zA-Z" "\n"
```

```
Colorless
```

```
green
```

```
ideas
```

```
sleep
```

```
furiously
```

Mapper: 只是把句子切成一個個單字

- Testing reducer

```
$ echo $'Colorless green ideas \n sleep furiously' | tr -sc "a-zA-Z" "\n"  
| sort | uniq -c  
  1 Colorless  
  1 furiously  
  1 green  
  1 ideas
```

Reducer: 把Mapper輸出的結果排序好

Ngram Count

- Mapper

```
import re, sys

def tokens(str1): return re.findall('[a-z]+', str1.lower())
def ngrams(sent, n):
    return [ ' '.join(x) for x in zip(*[sent[i:] for i in range(n)
        if i <= len(sent) ] ) ]

for line in sys.stdin:
    sent = tokens(line)
    for n in range(2, 6):
        for ngram in ngrams(sent, n):
            print ('%s\t%s' % (ngram, 1))
```

會根據句子的長度做2-gram ~ 6-gram
如果句子太短，就會提前結束

- Testing mapper

```
echo $'Colorless green ideas \n sleep furiously' | python nc-mapper.py
```

```
colorless green 1
```

```
green ideas 1
```

```
colorless green ideas 1
```

```
sleep furiously 1
```

- Reducer

```
import sys
from collections import Counter, defaultdict

ngm_count = defaultdict(Counter)
for line in sys.stdin:
    ngm, count = line.split('\t'); n = ngm.count(' ')+1
    ngm_count[n][ngm] += int(count)

for n in range(2, 6):
    for ngm in ngm_count[n]:
        if ngm_count[n][ngm] >= 3:
            print( '%s\t%s' % (ngm, ngm_count[n][ngm]) )
```

- Testing reducer

```
echo $'Colorless green ideas \n sleep furiously' | python nc-mapper.py  
| sort | python nc-reducer.py
```

```
colorless green 1  
green ideas 1  
sleep furiously 1  
colorless green ideas 1
```

- Running local MapReduce

```
echo '$Colorless green ideas \n sleep furiously'  
| ./lmr 5m 16 'python nc-mapper.py' 'python nc-reducer.py' out
```

```
hashing script hashing.py.BWar
```

```
>>> Temporary output directory for mapper created: mapper_tmp.YZ4i
```

```
>>> Mappers running...
```

```
>>> Reducer running. Temporary input directory: mapper_tmp.YZ4i
```

```
>>> Cleaning...
```

```
>>> Temporary directory deleted: mapper_tmp.YZ4i
```

```
* Output directory: out
```

```
* Elapsed time: 0:00:02
```

```
$ cat out/*
```

```
sleep furiously 1
```

```
colorless green ideas 1
```

```
colorless green 1
```

```
green ideas 1
```

- Life-size Test on British National Corpus

```
$ time cat bnc.sent.txt | python nc-mapper.py | sort | python nc-reducer.py 3
```

```
$ grep '^ability ' bnc.ngm.3.plus.txt | sort -k2nr -t $'\t'
```

```
ability to pay 108
```

```
ability to make 97
```

```
ability to cope 64
```

```
...
```

```
ability range 17
```

```
...
```

```
ability and willingness 9
```

```
...
```

```
ability and enthusiasm 6
```

```
ability and motivation 6
```

```
ability could 6
```

```
ability of local 6
```

```
ability of the system 6
```

```
ability tests 6
```


...

ability to conceive and develop 3

ability to conduct 3

ability to construct and convey 3

...

ability to make sense 3

ability to meet the challenges 3

ability to recognise words 3

...

ability to solve problems 3

ability to summon 3

ability to talk and write 3

ability to think logically 3

...

\$

Extracting Collocations with Local MapReduce

- Mapper

```
from collections import defaultdict, Counter
import sys
from nltk.corpus import stopwords

eng_stopwords = set(stopwords.words('english'))
max_distance = 5
skipbigram = defaultdict(Counter)

for line in sys.stdin:
    ngm, count = line.strip().split('\t')
    ngm = ngm.split(); distance = len(ngm)-1
    skipbigram[ngm[0]+' '+ngm[-1]][distance] += int(count)
    skipbigram[ngm[-1]+' '+ngm[0]][-distance] += int(count)

for bigram in sorted(skipbigram.keys()):
    print('%s\t%s\t%s'%(bigram, sum(skipbigram[bigram].values()),
        sorted(list(skipbigram[bigram].items()), key=lambda x: x[1] )) )
```

- Reducer

```
from math import sqrt
from itertools import groupby
```

```

import sys
k0, U0, k1 = 1, 10, 5
def getHighCounts(list1, COUNT, k):
    if not list1:
        return []
    size = len(list1)
    totals = [ COUNT(x) for x in list1 ]
    grandtotal = sum(totals)
    avg = (0.0+grandtotal)/size
    sdv = sqrt( sum( (x-avg)**2 for x in totals )/size )
    return [ x for x in list1 if COUNT(x) >= avg+k*sdv ]

lines = [ line.strip().split('\t') for line in sys.stdin ]
lines = [ x[0].split()+x[1:] for x in lines]
for head, headgroup in groupby(lines, key=lambda x: x[0]):
    cand = [ (x[0], x[1], int(x[2]), eval(x[3])) for x in headgroup]
    cand.sort(key= lambda x: x[2] )
    goodColls = getHighCounts(cand, lambda x: x[2], k0)
    goodColls = [ (head, coll, total,
                    getHighCounts(dCounts, lambda x: x[1], k1) )
                  for head, coll, total, dCounts in goodColls ]
    for head, coll, total, dCounts in goodColls:
        if dCounts: print('%s\t%s\t%s\t%s' % (head, coll, total, dCounts))

```

Lab Work

- Purpose: Selecting good examples for collocations
- Input:
 - SENTS: a set of sentences
 - COLLS: a set of collocation with distance (e.g., ['difficulty', 'task', 3])
 - PRONS: a list of pronouns, 'i, you, your, yours, he, she, they, him, her, them, his, their, it'
- Output:
 - EXAMPLES: A set of word, col, sentence

- Mapper

Read a sentence S in SENTS

For each distance bigram, $S[i]$, $S[i+d]$, where d in $[-5,5]$

If $isCollocation(S[i], S[i+d], d)$ and $10 \leq |S| \leq 25$,

Output ~~$S[i]$ $S[i+d]$~~ <tab> S

$S[i]$ $S[i+d]$ $dist$

- Reducer

For all S in each key group of $(Word, Col, Dist)$

Compute $Score(S)$

$$= \text{location of } Word - \#(\text{words} \in S \ \& \ \notin \text{HiFreWords}) \\ - \#(\text{words} \in S \ \& \ \text{words} \in \text{PRONS})$$

Find S^* with the maximum value of $Score$

Output ~~$Word_Col <tab> S^*$~~

$Word_Col_Dist <tab> S^*$