Information Retrieval

ค้นคืน คือ ค้นอะไร คืนอะไร

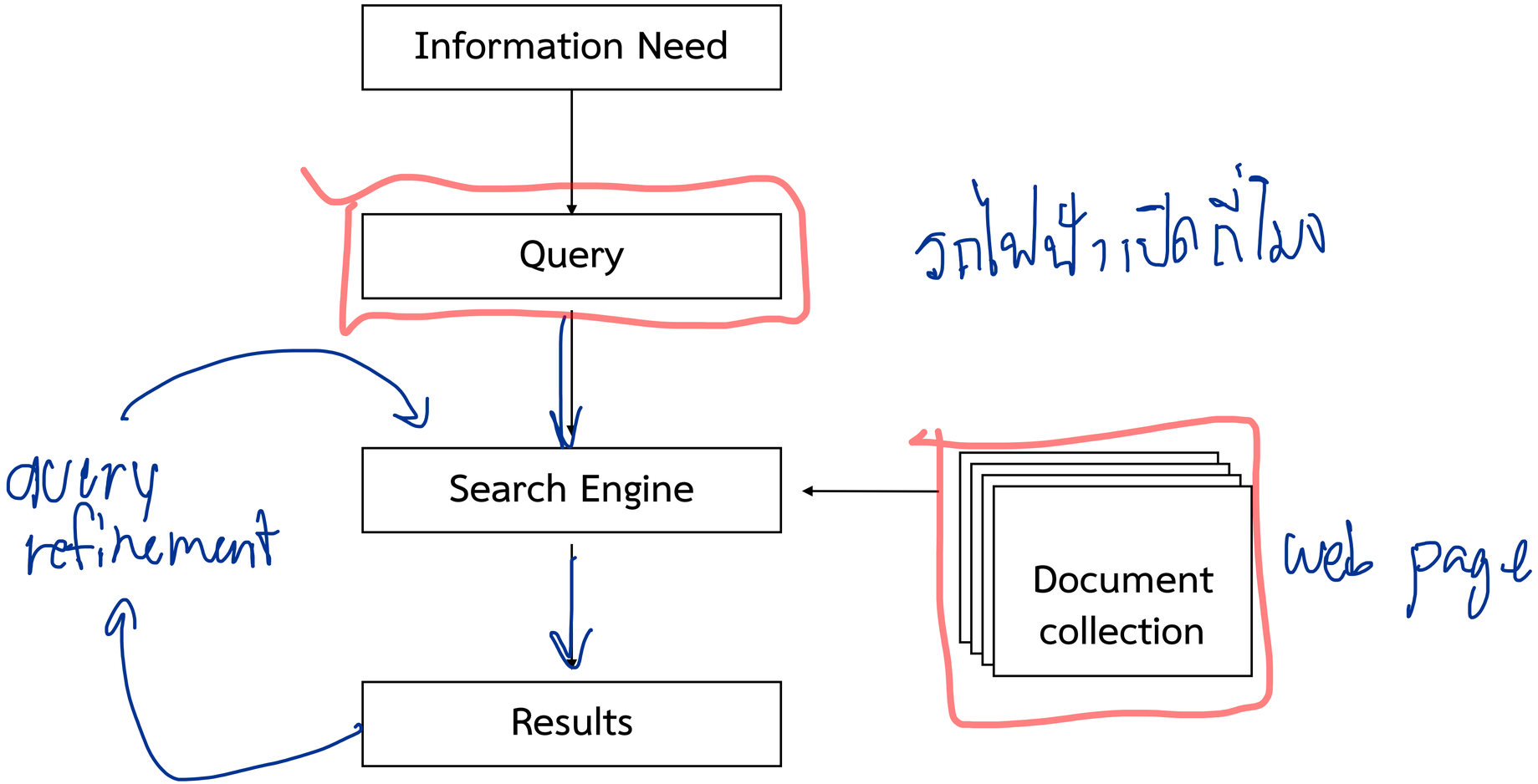
- วิธีการใช้ Dictionary บน Python
- หนังสือเรื่อง อยู่กับก๋ง
- อีเมล์เชิญไปสัมภาษณ์งาน
- รูปภาพของเฌอปราง
- รถไฟฟ้าเปิดกี่โมง

Information Retrieval การค้นคืนข้อมูล

material collection information need

- การค้นหา**เนื้อหา**จาก**กองข้อมูล**ที่ตอบโจทย์**ความต้องการทางข้อมูล** (Information need)

Classic Search Model

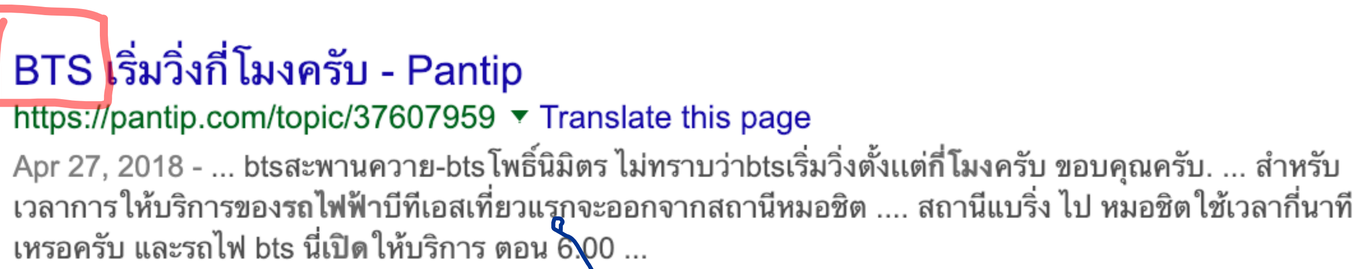


NLP + Information Retrieval

query

document

- "รถไฟฟ้าเปิดกี่โมง" --->



- "รถไฟฟ้าเปิดกี่โมง" ---> รถไฟฟ้า BTS สายสุขุมวิทเปิดเวลา 5:15

Google

who's the prime minister of thailand

All Images News Maps Videos More Settings Tools

About 69,300,000 results (1.13 seconds)

Thailand / Prime minister

Prayut Chan-o-cha

Since 2014

Prayut Chan-o-cha is a Thai politician, retired Royal Thai Army general officer, head of the National Council for Peace and Order, and concurrently serves as the Prime Minister of Thailand. Wikipedia

Document เอกสาร

- วิธีการใช้ Dictionary บน Python
- หนังสือเรื่อง อยู่กับก๋ง
- อีเมล์เชิญไปสัมภาษณ์งาน
- รูปภาพของเฌอปราง
- รถไฟฟ้าเปิดกี่โมง

wongnai

ป้าข้าวเหนียวไก่อักษร

4.0 ★ 26 รีวิว ฿ ปิดอยู่

อาหารจานเดียว, อาหารไทย

เมนูเด็ด: ข้าวเหนียวไก่ทอด, ข้าวเหนียวไก่ทอด

agoda

2017 Baiyoke Sky Hotel

★★★★ | HOTEL | Pratunam, Bangkok - View on map

Excellent location

Breakfast | 2+

Price includes | Free cancellation

Special discount on your first booking

Recommended by 82% of guests

Popular now! Last booked 5 hours a...

Our last room at this price!

Coupon code 24HOURSALE applied - ฿ 132 off!

Shopee

smartmi 智米

pre order เครื่องวัด PM 2.5

1660.-

Challenges

- ทำความเข้าใจ query และ document เพื่อสนองความต้องการทางข้อมูลของผู้ใช้
- ต้องทำให้ได้เร็ว (อย่าเกิน 1-2 วินาที)
- ต้องอย่าเปลืองที่เก็บข้อมูลและเครื่องคอมพ์

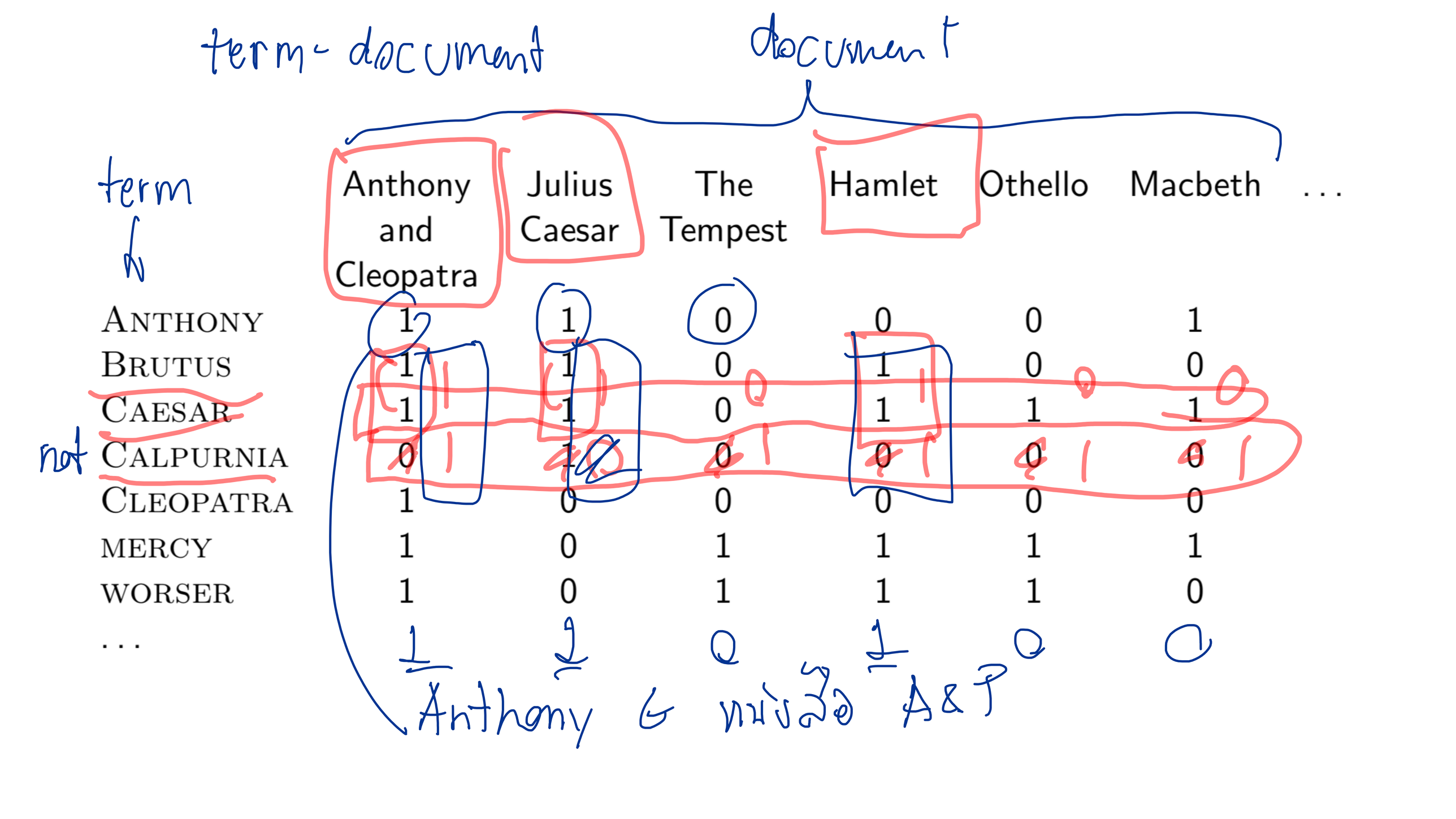
Boolean Retrieval

Search แบบง่ายสุด

- อยากหาไฟล์ที่มีคำว่า Caesar และคำว่า Brutus

  grep 'Caesar' docs/*.txt | grep 'Brutus'

- อยากหาไฟล์ที่มีคำว่า Caesar และ Brutus แต่ไม่มีคำว่า Calpurnia


term-document
document
term
Anthony and Cleopatra
Julius Caesar
The Tempest
Hamlet
Othello
Macbeth
...
ANTHONY 1 1 0 0 0 1
BRUTUS 1 1 0 1 0 0
CAESAR 1 1 0 1 1 1
not CALPURNIA 0 1 0 0 0 0
CLEOPATRA 1 0 0 0 0 0
MERCY 1 0 1 1 1 1
WORSER 1 0 1 1 1 0
... 1 1 0 1 0 0
Anthony & หมายถึง A&P

Term-Doc Matrix ในความเป็นจริง

- N = 1 ล้าน document แต่ละ doc มี 1000 คำ
- ค่าเก็บข้อมูล
  - ตัวอักษรละ 1 byte คำนึงมีประมาณ 6 ตัวอักษรโดยเฉลี่ย
  - ต้องใช้ Hard drive ขนาด 1 byte x 6 x 1000 x 1M = 6000MB = 6GB
- ค่าเก็บข้อมูล term-doc matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	1	1	0	0	0	1	
BRUTUS	1	1	0	1	0	0	
CAESAR	1	1	0	1	1	1	
CALPURNIA	0	1	0	0	0	0	
CLEOPATRA	1	0	0	0	0	0	
MERCY	1	0	1	1	1	1	
WORSER	1	0	1	1	1	0	
...							

- 4 bytes x 1M document x 100,000 (vocab size) = 400GB

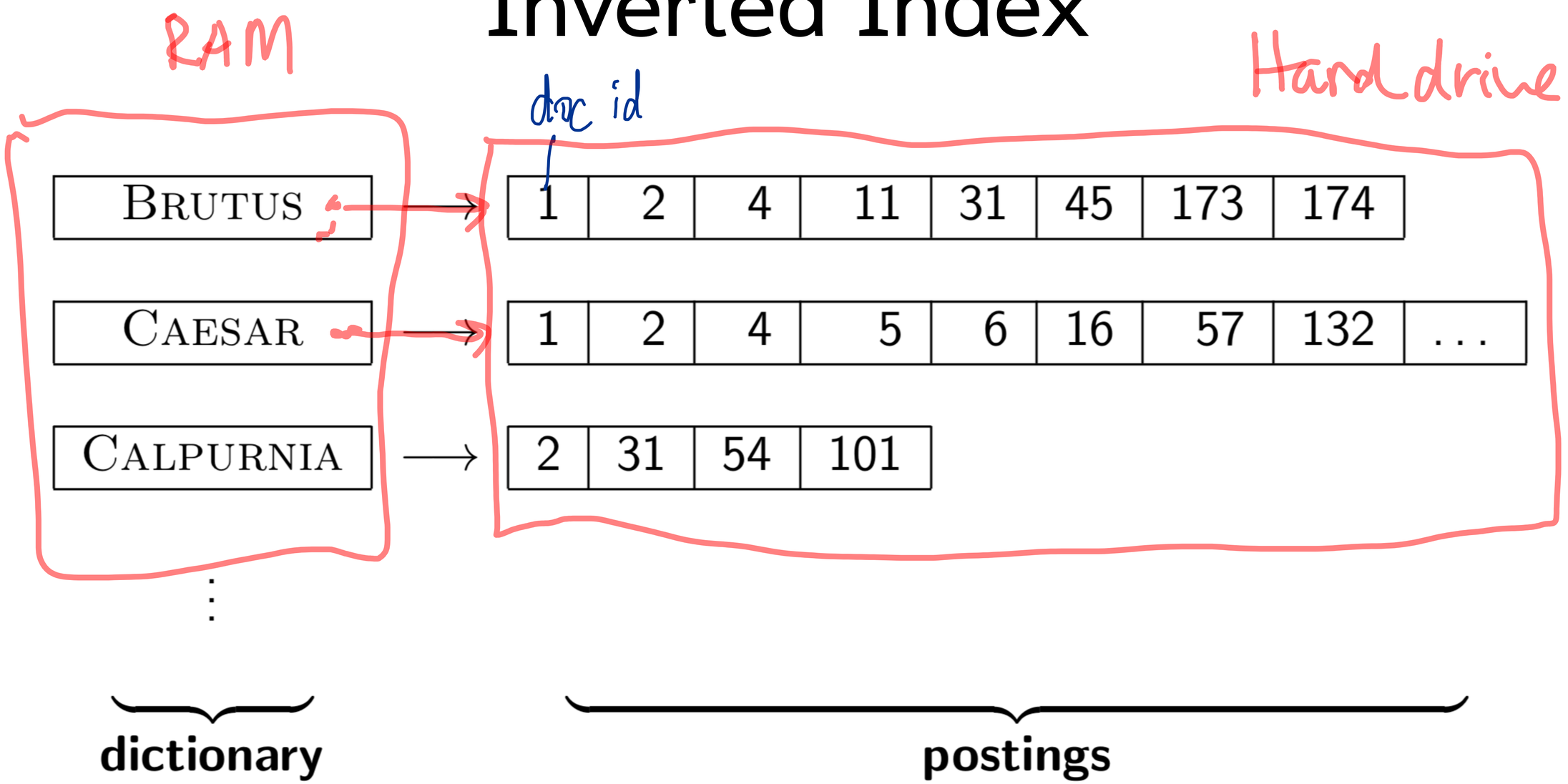
  99% ของ 400GB เก็บแต่ 0 เอาไว้

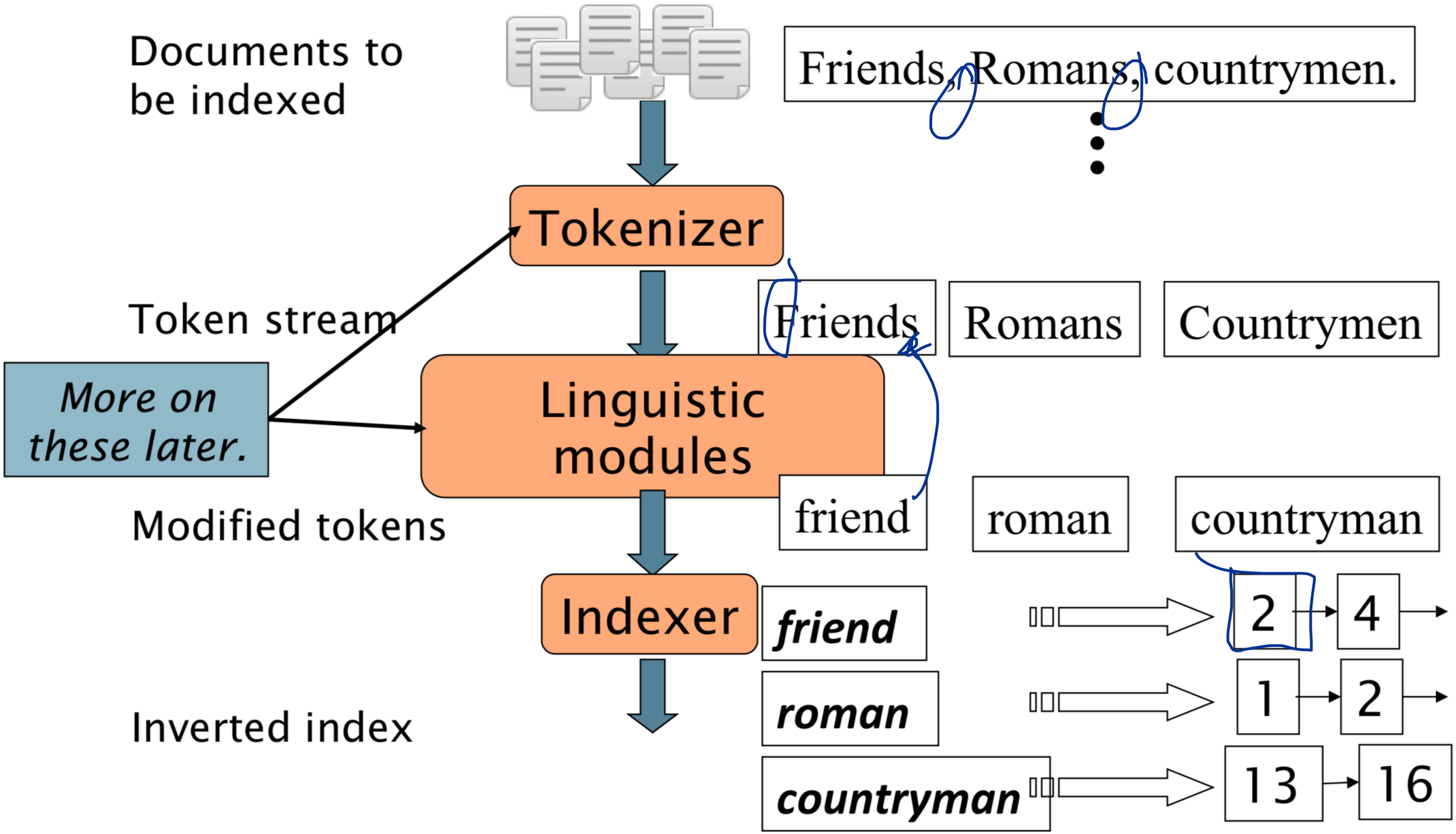
Inverted Index

Inverted Index

- datat structure ที่เป็น sparse matrix แบบหนึ่ง search engine ทุกประเภทบนโลกนี้ยังใช้กันอยู่
- จุดมุ่งหมาย คือ ประหยัดที่และประหยัดเวลา

Inverted Index




Documents to be indexed
Friends, Romans, countrymen.
Tokenizer
Token stream
Friends
Romans
Countrymen
More on these later.
Linguistic modules
friend
roman
countryman
Modified tokens
Indexer
friend
2
4
roman
1
2
countryman
13
16
Inverted index

case folding

Doc 1. I did enact Julius Caesar: I was killed i' the Capitol; Brutus killed me.

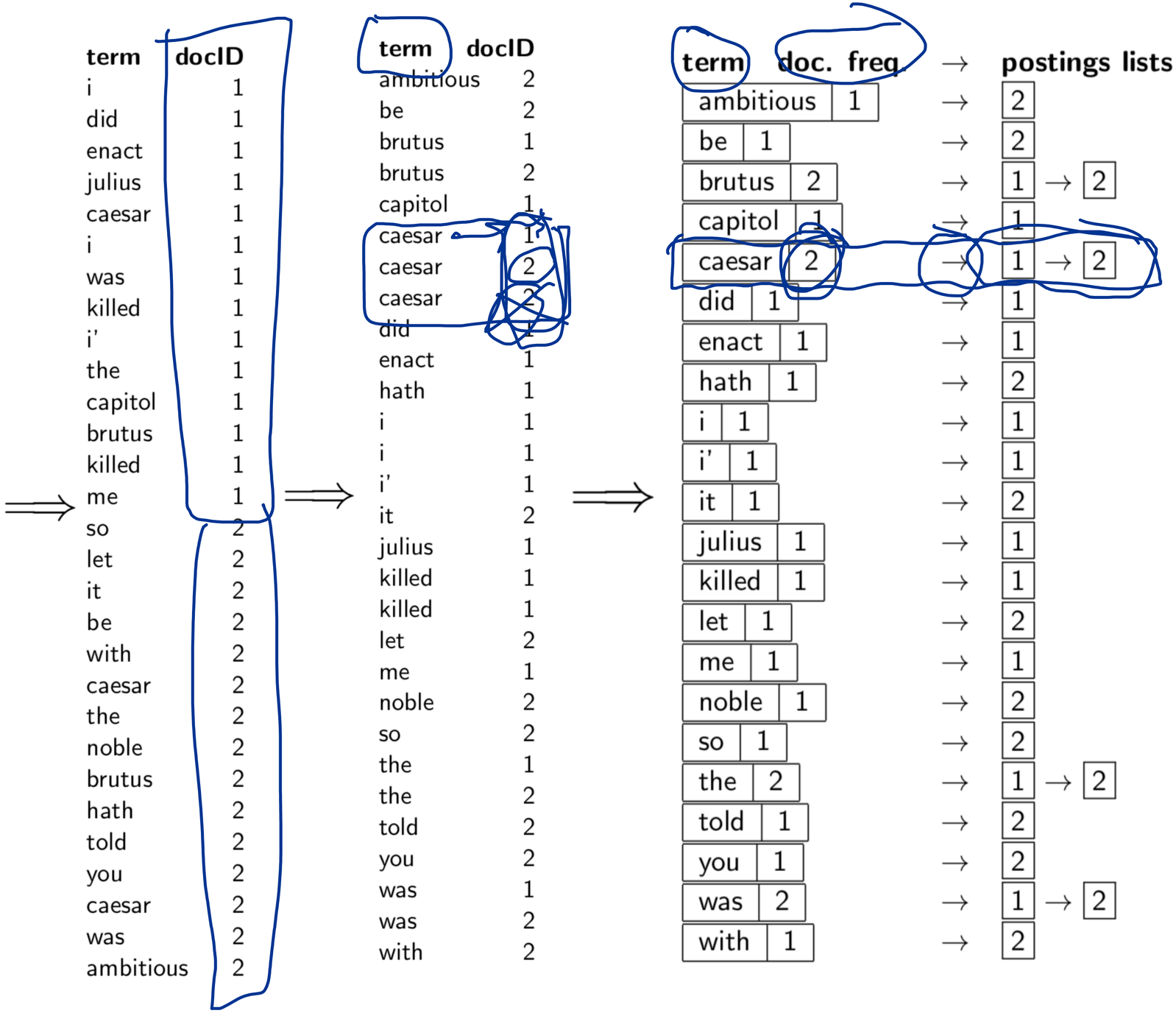
Doc 2. So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:

⟹

Doc 1. i did enact julius caesar i was killed i' the capitol brutus killed me

Doc 2. so let it be with caesar the noble brutus hath told you caesar was ambitious

Doc 1. i did enact julius caesar i was killed i' the capitol brutus killed me
Doc 2. so let it be with caesar the noble brutus hath told you caesar was ambitious

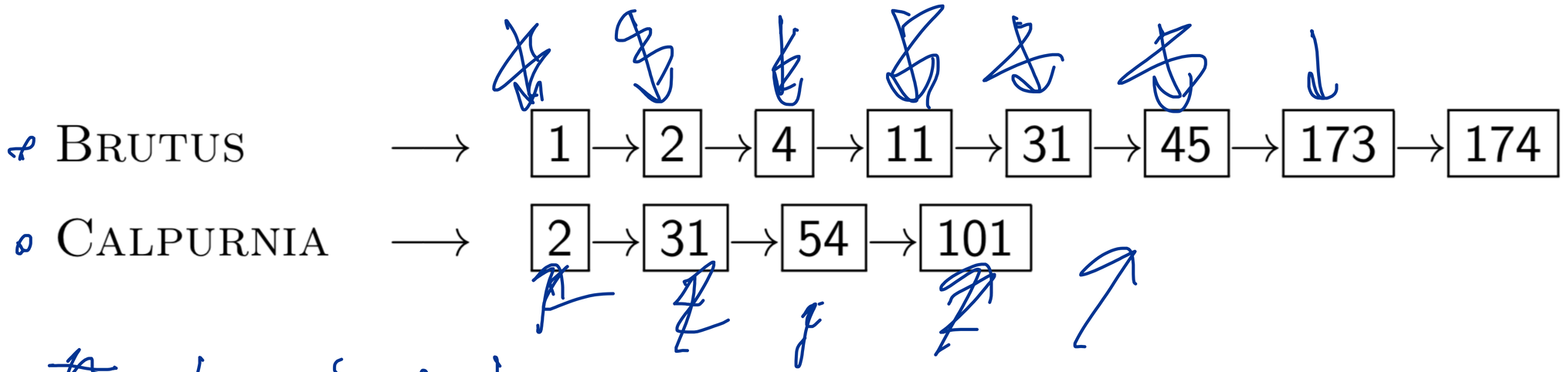


term	docID
i	1
did	1
enact	1
julius	1
caesar	1
i	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

term	docID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
i	1
i	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

term	doc. freq.	→	postings lists
ambitious	1	→	2
be	1	→	2
brutus	2	→	1 → 2
capitol	1	→	1
caesar	2	→	1 → 2
did	1	→	1
enact	1	→	1
hath	1	→	2
i	1	→	1
i'	1	→	1
it	1	→	2
julius	1	→	1
killed	1	→	1
let	1	→	2
me	1	→	1
noble	1	→	2
so	1	→	2
the	2	→	1 → 2
told	1	→	2
you	1	→	2
was	2	→	1 → 2
with	1	→	2

Brutus and Calpurnia



≠ ขยับตัวน้อย

= add to result {2, 31}

```
INTERSECT(p1, p2)
 1  answer ← ⟨ ⟩
 2  while p1 ≠ NIL and p2 ≠ NIL
 3  do if docID(p1) = docID(p2)
 4        then ADD(answer, docID(p1))
 5             p1 ← next(p1)
 6             p2 ← next(p2)
 7        else if docID(p1) < docID(p2)
 8                then p1 ← next(p1)
 9                else p2 ← next(p2)
10  return answer
```

Linguistic Processing of Documents

From words to terms

- แปลงเอกสาร (e.g. email, PDF, word doc) ให้เป็น text file ธรรมดา
- Tokenize: *honey-roasted pork*
- Normalization: *U.S.A, USA —> usa; naïve, naive —> naive*
- Stemming: *authorization, authorize, authorized —> authoriz*
- Stopwords: *the, a, to, of, in —> X*

colour, color -> color

Tokenize ภาษาไทย: *ผลิตรายการ*

- Word segmentation with Machine Learning : ผลิต, รายการ
- Dictionary : ผลิต, ราย, การ, ลิตร
- Character cluster: ผลิต, ราย, การ
- Character ngrams: ผล ลิต ตร รา ยก กา ผลิ ลิตร ตรา ราย ยาก การ

Thai Character Cluster

<TCC> → ‘ก็’ | ‘อึ’ | ‘หึ’

| <Cons> ‘รร’ ,<Cons> ‘◌์’

| <Cons> <BCons> <Cons> ‘◌์’

| <Cons> <TCC1> <Karan>

| <FSara><Cons> <TCC2> <Karan>

<TCC2> → <Cons>‘ าะ’

| ‘เ◌็’<BCons>

| <USara>{<Tone>}<BCons> [‘า’ | ‘ะ’]

| {<Tone>}[‘า’ | ‘าะ’| ‘ะ’]

<TCC1> → <DSara> {<Tone>}

| {<Tone>} ‘◌ำ’

| [‘เ◌็’ | ‘เ◌ี’] {<Tone>}<BCons>

| ‘แ◌’{<Tone>[‘◌ุ’ | ‘◌ิ’]}

| ‘เ◌็’ <BCons>

|<Tone>[<TSara>|<DSara>]{‘ว’}<BCons>

| ‘◌ิ’{<Tone>{<BCons>}}

| ‘เ◌ี’<Tone>

| {<Tone>} <Bsara>

| NULL

<Karan> → <Cons>{<Cons>}{[<DSara>| ‘◌ิ’]}‘◌์’

| NULL

From words to terms

- แปลงเอกสาร (e.g. email, PDF, word doc) ให้เป็น text file ธรรมดา
- Tokenize
- Normalization
- Stemming
- Stopwords

Phrase Queries

Phrase Query

- หา document ที่มีคำว่า Stanford University อยู่ติดกัน
  - ไม่เอา doc "I went to university at Stanford"

Doc 1. i did enact julius caesar i was killed i' the capitol brutus killed me
Doc 2. so let it be with caesar the noble brutus hath told you caesar was ambitious

term	docID
i	1
did	1
enact	1
julius	1
caesar	1
i	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

term	docID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
i	1
i	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

term	doc. freq.	→	postings lists
ambitious	1	→	2
be	1	→	2
brutus	2	→	1 → 2
capitol	1	→	1
caesar	2	→	1 → 2
did	1	→	1
enact	1	→	1
hath	1	→	2
i	1	→	1
i'	1	→	1
it	1	→	2
julius	1	→	1
killed	1	→	1
let	1	→	2
me	1	→	1
noble	1	→	2
so	1	→	2
the	2	→	1 → 2
told	1	→	2
you	1	→	2
was	2	→	1 → 2
with	1	→	2

Phrase query ที่ยาวกว่า 2 คำ

- Stanford University Palo Alto —>

  "Stanford University" AND "University Palo" AND "Palo Alto"

Biword Index ไม่ใช่คำตอบ

- Index จะใหญ่เบิ้มมากเสียค่าเก็บ (อย่าลืมว่า RAM แพง)
- ไม่รองรับ phrase ที่ยาวกว่าสองคำ

Positional Index

- (term, freq): [docID, docID, docID, ...]
- (term, freq): [docID: [word position, word position, ...] ,
  docID: [word position, word position, ...] , ...]

Query: “to_1 be_2 or_3 not_4 to_5 be_6”

TO, 993427:

⟨ 1: ⟨7, 18, 33, 72, 86, 231⟩;
2: ⟨1, 17, 74, 222, 255⟩;
4: ⟨8, 16, 190, 429, 433⟩;
5: ⟨363, 367⟩;
7: ⟨13, 23, 191⟩; . . . ⟩

BE, 178239:

⟨ 1: ⟨17, 25⟩;
4: ⟨17, 191, 291, 430, 434⟩;
5: ⟨14, 19, 101⟩; . . . ⟩

Positional Index แพงไปมั้ย

Query: *"to*$_1$ *be*$_2$ *or*$_3$ *not*$_4$ *to*$_5$ *be*$_6$*"*

TO, 993427:

⟨ 1: ⟨7, 18, 33, 72, 86, 231⟩;
2: ⟨1, 17, 74, 222, 255⟩;
4: ⟨8, 16, 190, 429, 433⟩;
5: ⟨363, 367⟩;
7: ⟨13, 23, 191⟩; . . . ⟩

BE, 178239:

⟨ 1: ⟨17, 25⟩;
4: ⟨17, 191, 291, 430, 434⟩;
5: ⟨14, 19, 101⟩; . . . ⟩

สรุป

- ใช้ทั้งสองอย่างไปเลย
  - positional index มันช้า ==> phrase ไหนที่เจอบ่อยๆ ก็เก็บไว้ใน phrase index (caching)

Ranked Retrieval with TF-IDF

Boolean Search

- ไม่เข้า ก็ออกเลย
- ไม่เยอะไป (ขี้เกียจเปิดอ่านหมด) ก็น้อยไป (ไม่มีสิ่งที่อยากได้)

คณะอักษรศาสตร์

All Images Maps Videos News More Settings Tools

About 495,000 results (0.62 seconds)

คณะอักษรศาสตร์ เรียนเกี่ยวกับอะไรหรอคะ? - Pantip

https://pantip.com/topic/36869921 ▾ Translate this page

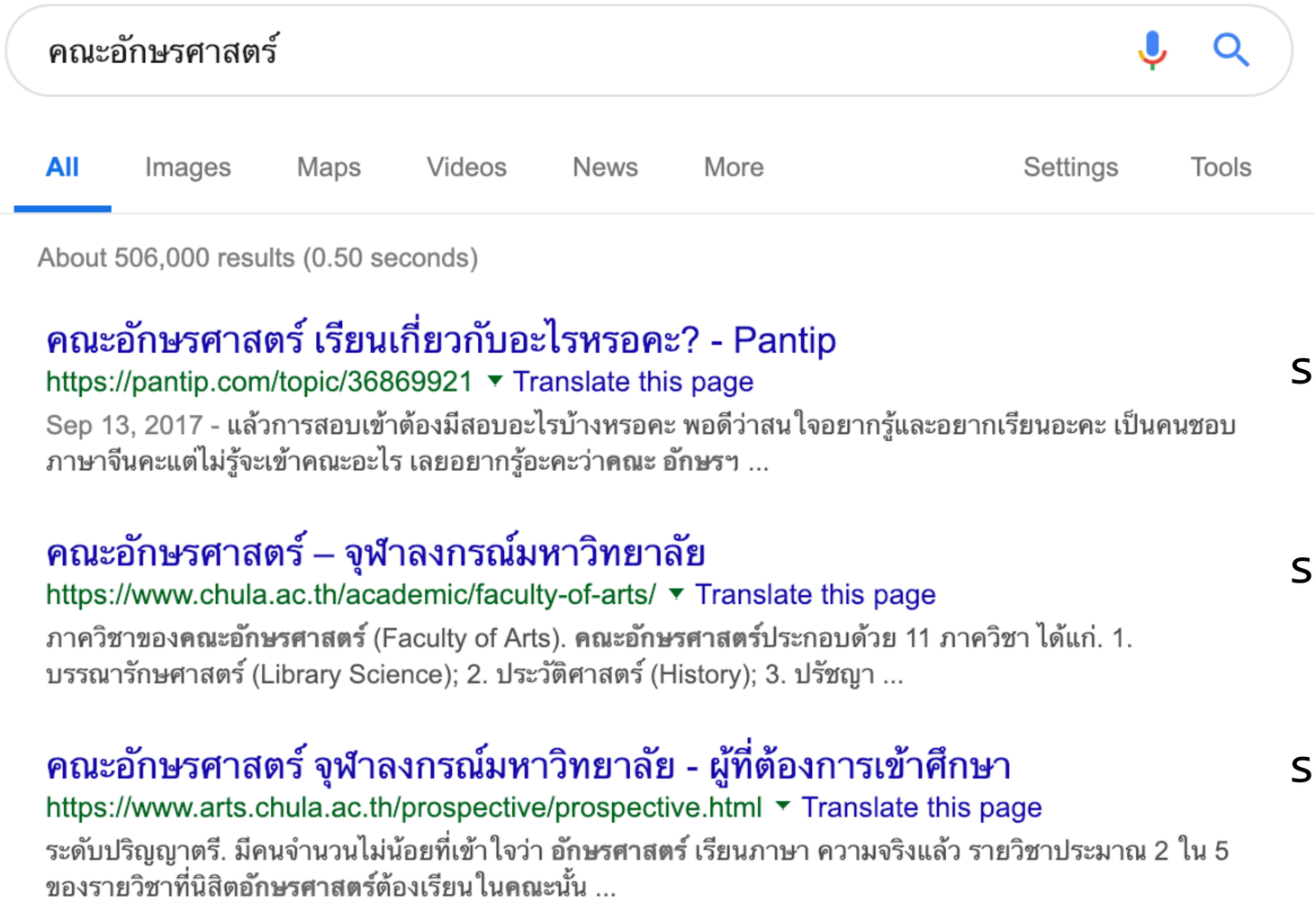
Sep 13, 2017 - แล้วการสอบเข้าต้องมีสอบอะไรบ้างหรอคะ พอดีว่าสนใจอยากรู้และอยากเรียนอะคะ เป็นคนชอบภาษาจีนคะแต่ไม่รู้จะเข้าคณะอะไร เลยอยากรู้อะคะว่า**คณะ อักษรฯ** ...

คณะอักษรศาสตร์ – จุฬาลงกรณ์มหาวิทยาลัย

https://www.chula.ac.th/academic/faculty-of-arts/ ▾ Translate this page

ภาควิชาของ**คณะอักษรศาสตร์** (Faculty of Arts). **คณะอักษรศาสตร์**ประกอบด้วย 11 ภาควิชา ได้แก่. 1. บรรณารักษศาสตร์ (Library Science); 2. ประวัติศาสตร์ (History); 3. ปรัชญา ...

Relevance Score คะแนนความเกี่ยวข้อง



q = คณะอักษรศาสตร์

score(d1, q) = 0.81

score(d2, q) = 0.74

score(d3, q) = 0.68

ถ้าเห็น term นั้นบ่อย doc นั้นยิ่งคะแนนเยอะ

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet
ANTHONY	1	1	0	0
BRUTUS	1	1	0	1
CAESAR	1	1	0	1
CALPURNIA	0	1	0	0
CLEOPATRA	1	0	0	0
MERCY	1	0	1	1
WORSER	1	0	1	1
...				

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet
ANTHONY	157	73	0	0
BRUTUS	4	157	0	2
CAESAR	232	227	0	2
CALPURNIA	0	10	0	0
CLEOPATRA	57	0	0	0
MERCY	2	0	3	8
WORSER	2	0	1	1

```
q = Anthony Brutus
```

Term Frequency

- Frequency = Occurrence = จำนวนครั้งที่เจอ

$$w_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d} & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$$

- $\mathrm{tf}_{t,d}$ = จำนวนครั้งที่เจอ t ใน d

คำทุกคำไม่ได้เท่าเทียมกัน

- term ที่เห็นบ่อยๆ มักจะไม่ค่อย informative (ไม่ได้ช่วยให้หา document ได้แม่นยำขึ้น)
- $tf_{t,d}$ = ความสำคัญของ term นั้นต่อ document นั้น
- ความสำคัญของ term นั้นโดยทั่วไป?

$$\text{idf}_t = \log_{10} \frac{N}{\text{df}_t}$$

term	df_t	idf_t
calpurnia	1	
animal	100	
sunday	1000	
fly	10,000	
under	100,000	
the	1,000,000	

$$\text{idf}_t = \log_{10} \frac{N}{\text{df}_t}$$

TF-IDF weighting

$$w_{t,d} = (1 + \log \mathrm{tf}_{t,d}) \cdot \log \frac{N}{\mathrm{df}_t}$$