上海外国语大学
SHANGHAI INTERNATIONAL STUDIES UNIVERSITY

Computational Journalism

Lecture 7: Big Data Analysis and Visualization

Ting Wang

Outlines

- Word Cloud using Python
- Chinese Word Segmentation
- Chinese Word Cloud

上海外国语大学
SHANGHAI INTERNATIONAL STUDIES UNIVERSITY

data visualization with word cloud

Word Cloud using Python

Word Cloud using Python

Case Description

Motivations:

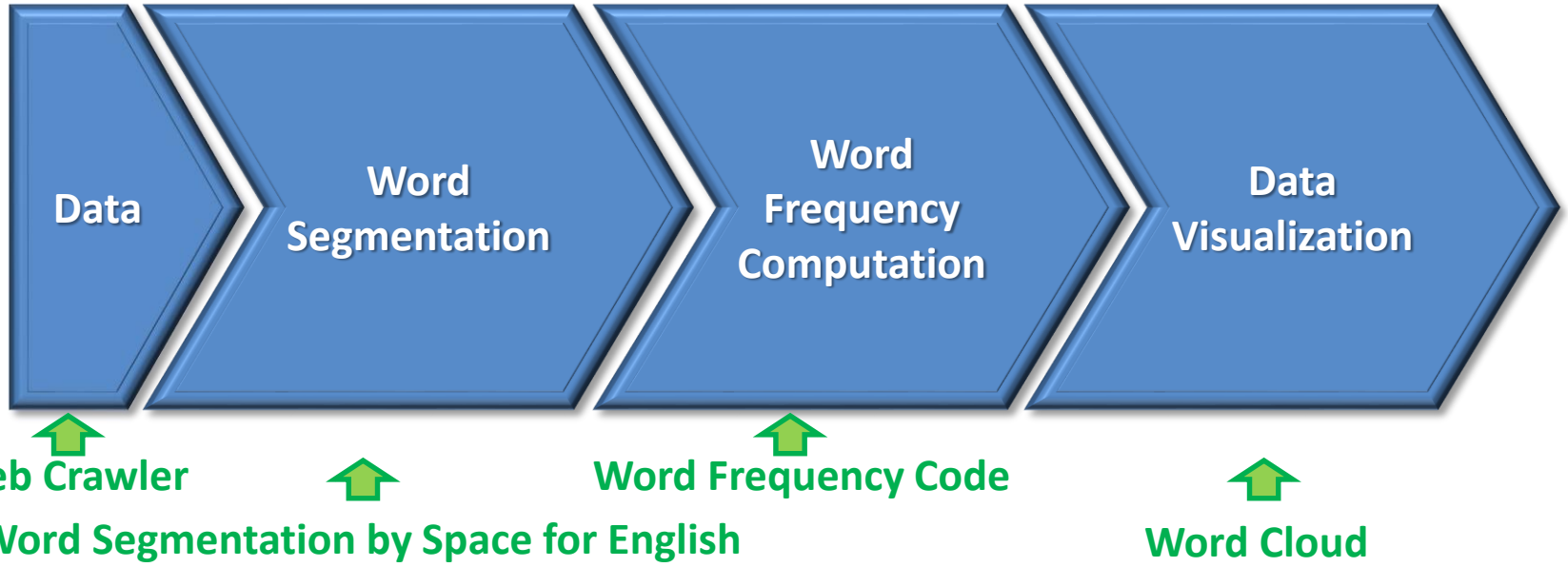
- To measure a news objectively
- To obtain new information efficiently

Methodologies:

- Describe a news report by quantitative method
- Technical integration by computer science, statistics and journalism

Word Cloud using Python

Now, we have data, how to mining it?



Word Cloud using Python

EXAMPLE 1:
Word Cloud for English News

Word Cloud using Python

Preparation: Take a news for an example.

https://edition.cnn.com/2018/12/22/politics/shutdown-mattis-whitaker-trump/index.html

Word Cloud using Python

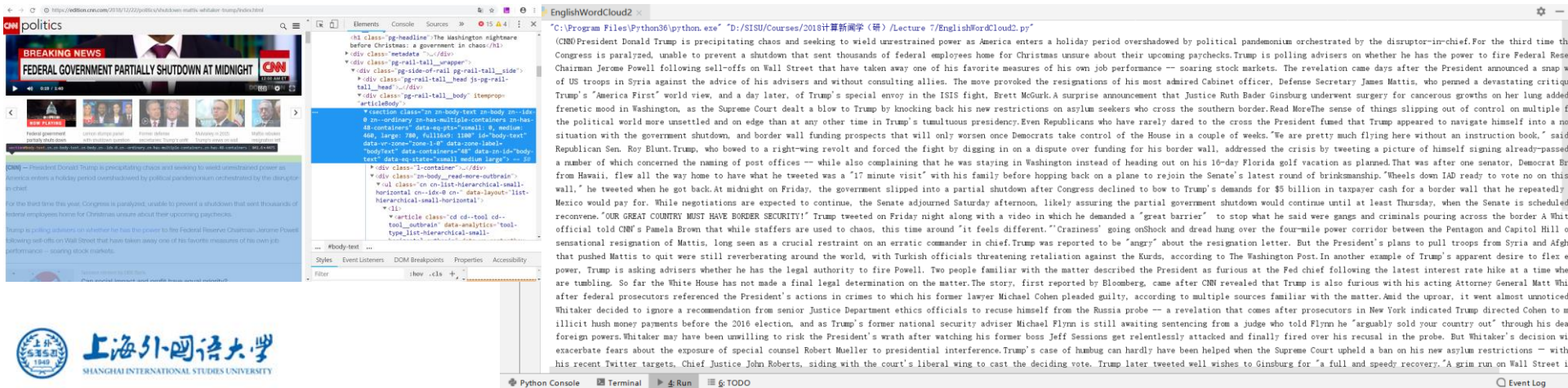
- Step 1-1: **Build a Web Crawler**, and extract the text from this page.

```
import urllib.request
response = urllib.request.urlopen('https://edition.cnn.com/2018/12/22/politics/shutdown-mattis-whitaker-trump/index.html')
HTMLText = response.read()

print(HTMLText)
```

Word Cloud using Python

- Step 1-2: Build a Web Crawler, and **extract the text** from this page.

```
import urllib.request
from bs4 import BeautifulSoup

response = urllib.request.urlopen('https://edition.cnn.com/2018/12/22/politics/shutdown-mattis-whitaker-trump/index.html')
HTMLText = response.read()
BSobj = BeautifulSoup(HTMLText, "html.parser")
Content = BSobj.find("section", {"class": "zn zn-body-text zn-body zn--idx-0 zn--ordinary zn-has-multiple-containers zn-has-48-containers"})

print(Content.get_text())
```



Word Cloud using Python

Step2: Do you remember the word frequency code in Lecture 3? Revise it to a function.

INPUT

```
with open('text_flu.txt', 'rU') as f:
    s_array = str(f.read())
s_array = s_array.replace(',',' ')
s_array = s_array.replace('.',' ')
s_array = s_array.replace('\'',' ')
s_array = s_array.replace('\"',' ')
s_array = s_array.replace(':',' ')
s_array = s_array.replace(';',' ')
s_array = s_array.replace('?',' ')
s_array = s_array.replace('`',' ')
s_array = s_array.replace('~',' ')
s_array = s_array.replace('!',' ')
s_array = s_array.replace('@',' ')
s_array = s_array.replace('#',' ')
s_array = s_array.replace('%',' ')
s_array = s_array.replace('$',' ')
s_array = s_array.replace('^',' ')
s_array = s_array.replace('&',' ')
s_array = s_array.replace('*',' ')
s_array = s_array.replace('(',' ')
s_array = s_array.replace(')',' ')
s_array = s_array.replace('-',' ')
s_array = s_array.replace('+',' ')
s_array = s_array.replace('_',' ')
s_array = s_array.replace('=',' ')
s_array = s_array.replace('{',' ')
s_array = s_array.replace('}',' ')
s_array = s_array.replace('[',' ')
s_array = s_array.replace(']',' ')
s_array = s_array.replace('|',' ')
s_array = s_array.replace('\\',' ')
s_array = s_array.replace('<',' ')
s_array = s_array.replace('>',' ')
s_array = s_array.split()
```

OUTPUT

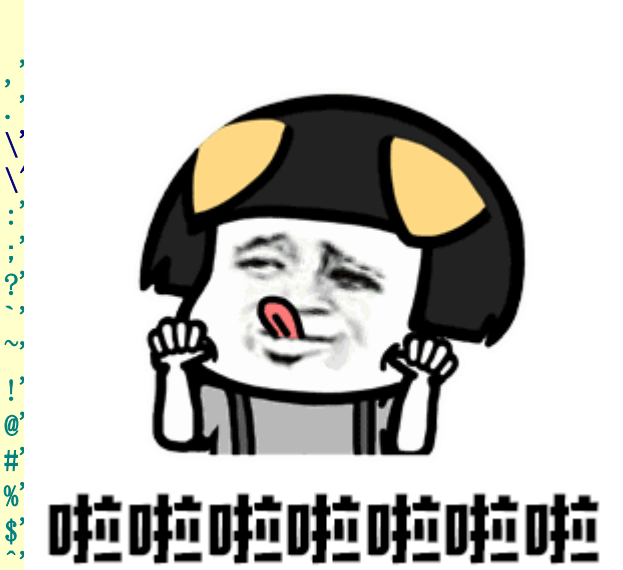
```
word_dict = {}

for i in range(len(s_array)):
    if s_array[i] not in word_dict:
        word_dict[s_array[i]] = 1
    else:
        word_dict[s_array[i]] += 1

with open('result_flu_test.txt', 'w') as f:
    for word, number in word_dict.items():
        result = word + "\t" + str(number) + "\n"
        f.write(result)
```

Make sure that you know what is input, and what is output.

Word Cloud using Python

- Revised function

```
def WordFrequency(InputText):
    s_array = str(InputText)
    s_array = s_array.replace(',',
    s_array = s_array.replace('.',
    s_array = s_array.replace('\'
    s_array = s_array.replace('\
    s_array = s_array.replace(':'
    s_array = s_array.replace(';'
    s_array = s_array.replace('?'
    s_array = s_array.replace('`',
    s_array = s_array.replace('~',
    s_array = s_array.replace('!'
    s_array = s_array.replace('@'
    s_array = s_array.replace('#'
    s_array = s_array.replace('%'
    s_array = s_array.replace('$'
    s_array = s_array.replace('^',
```




```
    s_array = s_array.replace('^', ' ')
    s_array = s_array.replace('&', ' ')
    s_array = s_array.replace('*', ' ')
    s_array = s_array.replace('(', ' ')
    s_array = s_array.replace(')', ' ')
    s_array = s_array.replace('-', ' ')
    s_array = s_array.replace('+', ' ')
    s_array = s_array.replace('_', ' ')
    s_array = s_array.replace('=', ' ')
    s_array = s_array.replace('{', ' ')
    s_array = s_array.replace('}', ' ')
    s_array = s_array.replace('[', ' ')
    s_array = s_array.replace(']', ' ')
    s_array = s_array.replace('|', ' ')
    s_array = s_array.replace('\\', ' ')
    s_array = s_array.replace('<', ' ')
    s_array = s_array.replace('>', ' ')
    s_array = s_array.split()
    dict = {}
    in range(len(s_array)):
        if s_array[i] not in word_dict:
            word_dict[s_array[i]] = 1
        else:
            word_dict[s_array[i]] += 1
    return word_dict
```

You still have to revise one place in your main code,
but I do NOT want to tell you,
could you try it by yourself?

Word Cloud using Python

Results

Run: EnglishWordCloud3

```
"C:\Program Files\Python36\python.exe" "D:/SISU/Courses/2018计算新闻学（研）/Lecture 7/EnglishWordCloud3.py"
{'CNN': 5, 'President': 12, 'Donald': 2, 'Trump': 25, 'is': 14, 'precipitating': 1, 'chaos': 2, 'and': 25, 'seeking': 1, 'to': 59, 'wield': 1, 'unrestrained': 1, 'power': 4, 'as': 10, 'America': 2, 'enters': 1, 'a': 49, 'holi
'period': 1, 'overshadowed': 2, 'by': 11, 'political': 5, 'pandemonium': 1, 'orchestrated': 1, 'the': 91, 'disruptor': 1, 'in': 27, 'chief': 4, 'For': 1, 'third': 1, 'time': 5, 'this': 8, 'year': 3, 'Congress': 3, 'paralyzed
'unable': 1, 'prevent': 1, 'shutdown': 6, 'that': 24, 'sent': 1, 'thousands': 1, 'of': 39, 'federal': 2, 'employees': 1, 'home': 2, 'for': 15, 'Christmas': 2, 'unsure': 1, 'about': 4, 'their': 1, 'upcoming': 1, 'paychecks':
'polling': 1, 'advisers': 4, 'on': 22, 'whether': 2, 'he': 15, 'has': 6, 'fire': 2, 'Federal': 1, 'Reserve': 1, 'Chairman': 1, 'Jerome': 1, 'Powell': 2, 'following': 3, 'sell': 1, 'offs': 1, 'Wall': 2, 'Street': 2, 'have': 1
'taken': 1, 'away': 1, 'one': 3, 'his': 30, 'favorite': 1, 'measures': 1, 'own': 2, 'job': 1, 'performance': 1, 'soaring': 1, 'stock': 1, 'markets': 2, 'The': 12, 'revelation': 2, 'came': 2, 'days': 1, 'after': 8, 'announced
'snap': 1, 'withdrawal': 1, 'US': 2, 'troops': 2, 'Syria': 2, 'against': 2, 'advice': 1, 'without': 3, 'consulting': 1, 'allies': 1, 'move': 2, 'provoked': 1, 'resignations': 1, 'most': 2, 'admired': 1, 'Cabinet': 1, 'office
'Defense': 1, 'Secretary': 1, 'James': 1, 'Mattis': 5, 'who': 8, 'penned': 1, 'devastating': 1, 'critique': 1, 's': 23, 'First': 1, 'world': 4, 'view': 1, 'day': 3, 'later': 2, 'special': 2, 'envoy': 1, 'ISIS': 1, 'fight': 3
1, 'McGurk': 1, 'A': 4, 'surprise': 1, 'announcement': 1, 'Justice': 3, 'Ruth': 1, 'Bader': 1, 'Ginsburg': 2, 'underwent': 1, 'surgery': 1, 'cancerous': 1, 'growths': 1, 'her': 1, 'lung': 1, 'added': 3, 'frenetic': 1, 'mood'
'Washington': 5, 'Supreme': 2, 'Court': 2, 'dealt': 1, 'blow': 1, 'knocking': 1, 'back': 3, 'new': 2, 'restrictions': 2, 'asylum': 2, 'seekers': 1, 'cross': 2, 'southern': 1, 'border': 8, 'Read': 1, 'MoreThe': 1, 'sense': 3,
1, 'slipping': 1, 'out': 4, 'control': 2, 'multiple': 2, 'fronts': 1, 'left': 1, 'more': 2, 'unsettled': 1, 'edge': 1, 'than': 1, 'at': 8, 'any': 1, 'other': 3, 'tumultuous': 1, 'presidency': 1, 'Even': 2, 'Republicans': 2,
1, 'dared': 1, 'fumed': 1, 'appeared': 2, 'navigate': 1, 'himself': 3, 'into': 5, 'no': 2, 'win': 2, 'situation': 1, 'with': 13, 'government': 6, 'wall': 8, 'funding': 5, 'prospects': 1, 'will': 4, 'only': 3, 'worsen': 1, 'o
'Democrats': 2, 'take': 2, 'House': 13, 'couple': 1, 'weeks': 1, 'We': 1, 'are': 8, 'pretty': 1, 'much': 3, 'flying': 1, 'here': 1, 'an': 5, 'instruction': 1, 'book': 1, 'said': 7, 'Missouri': 1, 'Republican': 3, 'Sen': 2,
'Blunt': 1, 'bowed': 1, 'right': 1, 'wing': 2, 'revolt': 1, 'forced': 1, 'digging': 1, 'dispute': 1, 'over': 5, 'addressed': 1, 'crisis': 2, 'tweeting': 1, 'picture': 1, 'signing': 1, 'already': 1, 'passed': 1, 'bills': 1,
1, 'which': 4, 'concerned': 1, 'naming': 1, 'post': 1, 'offices': 1, 'while': 4, 'also': 3, 'complaining': 1, 'was': 9, 'staying': 1, 'instead': 1, 'heading': 1, '16': 1, 'Florida': 1, 'golf': 1, 'vacation': 1, 'planned': 1,
1, 'senator': 1, 'Democrat': 1, 'Brian': 1, 'Schatz': 1, 'from': 6, 'Hawaii': 1, 'flew': 1, 'all': 3, 'way': 1, 'what': 6, 'tweeted': 5, '17': 1, 'minute': 1, 'visit': 1, 'family': 1, 'before': 2, 'hopping': 1, 'plane': 1,
1, 'Senate': 10, 'latest': 2, 'round': 1, 'brinksmanship': 1, 'Wheels': 1, 'down': 1, 'IAD': 1, 'ready': 1, 'vote': 2, 'stupid': 1, 'when': 5, 'got': 2, 'At': 1, 'midnight': 1, 'Friday': 5, 'slipped': 1, 'partial': 2, 'decli
'bow': 1, 'demands': 2, '5': 1, 'billion': 1, 'taxpayer': 1, 'cash': 1, 'repeatedly': 1, 'promised': 1, 'Mexico': 1, 'would': 5, 'pay': 1, 'While': 2, 'negotiations': 2, 'expected': 1, 'continue': 2, 'adjourned': 1, 'Saturda
'afternoon': 1, 'likely': 1, 'assuring': 1, 'until': 1, 'least': 2, 'Thursday': 1, 'scheduled': 1, 'reconvene': 1, 'OUR': 1, 'GREAT': 1, 'COUNTRY': 1, 'MUST': 1, 'HAVE': 1, 'BORDER': 1, 'SECURITY': 1, 'night': 1, 'along': 2,
1, 'demanded': 1, 'great': 1, 'barrier': 2, 'stop': 1, 'were': 3, 'gangs': 1, 'criminals': 1, 'pouring': 1, 'across': 2, 'White': 6, 'official': 3, 'told': 2, 'Pamela': 2, 'Brown': 3, 'staffers': 2, 'used': 1, 'around': 2,
'feels': 1, 'different': 1, 'Craziness': 1, 'going': 2, 'onShock': 1, 'dread': 1, 'hung': 1, 'four': 1, 'mile': 1, 'corridor': 1, 'between': 2, 'Pentagon': 1, 'Capitol': 1, 'Hill': 1, 'sensational': 1, 'resignation': 2, 'lon
'seen': 1, 'crucial': 1, 'restraint': 1, 'erratic': 1, 'commander': 1, 'reported': 2, 'be': 2, 'angry': 1, 'letter': 1, 'But': 5, 'plans': 1, 'pull': 1, 'Afghanistan': 1, 'pushed': 1, 'quit': 1, 'still': 2, 'reverberating':
'Turkish': 1, 'officials': 2, 'threatening': 1, 'retaliation': 1, 'Kurds': 1, 'according': 2, 'Post': 1, 'In': 1, 'another': 2, 'example': 1, 'apparent': 1, 'desire': 1, 'flex': 1, 'executive': 1, 'asking': 1, 'legal': 2, 'a
1, 'Two': 1, 'people': 2, 'familiar': 2, 'matter': 3, 'described': 1, 'furious': 2, 'Fed': 1, 'interest': 1, 'rate': 1, 'hike': 1, 'tumbling': 1, 'So': 1, 'far': 1, 'not': 5, 'made': 3, 'final': 1, 'determination': 1, 'story
'first': 2, 'Bloomberg': 1, 'revealed': 1, 'acting': 1, 'Attorney': 1, 'General': 1, 'Matt': 1, 'Whitaker': 4, 'prosecutors': 2, 'referenced': 1, 'actions': 1, 'crimes': 1, 'former': 3, 'lawyer': 1, 'Michael': 2, 'Cohen': 2,
1, 'guilty': 1, 'sources': 1, 'Amid': 1, 'uproar': 1, 'went': 1, 'almost': 2, 'unnoticed': 1, 'decided': 1, 'ignore': 1, 'recommendation': 1, 'senior': 1, 'Department': 1, 'ethics': 1, 'recuse': 1, 'Russia': 1, 'probe': 2,
```

Python Console | Terminal | 4: Run | 6: TODO | Event Log

Word Cloud using Python

Before the most important step:

1. *Build a folder called 'static' in your computer.*
2. *Install some packages for word cloud.*

static

Word Cloud using Python

Data Visualization using Python

- Necessity:
  - NumPy (Computing Package)
  - Scipy (Scientific Computing Package)
  - Pillow(Image)
  - Matplotlib (Diagram Package)
  - wordcloud (Word Cloud Package)

Installation Sequence

- Some packages also need some other required packages

Word Cloud using Python

Step 4: employ the word cloud code.

```
import urllib.request
from bs4 import BeautifulSoup
from DataWordCloud import word_cloud_generate
```

```
response = urllib.request.urlopen('https://edition.cnn.com/2018/12/22/politics/shutdown-mattis-whitaker-trump/index.html')
HTMLText = response.read()
BSobj = BeautifulSoup(HTMLText, "html.parser")
Content = BSobj.find("section", {"class": "zn zn-body-text zn-body zn--idx-0 zn--ordinary zn-has-multiple-containers zn-has-48-containers"})

# print(WordFrequency(Content.get_text()))

picurl=word_cloud_generate(WordFrequency(Content.get_text()))

print("word cloud saved in "+picurl)
```

Word Cloud using Python

Results:

Disadvantages:

1. Too many words

→to set a threshold

2. Lots of unmeaning words

→to employ a list for stop words

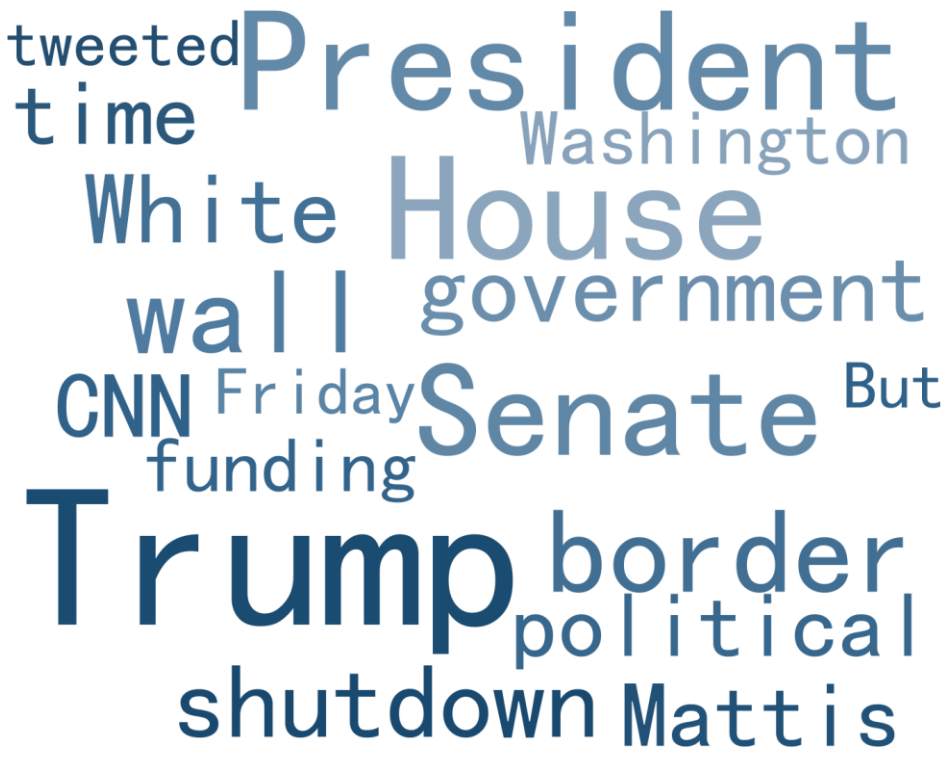
http://www.lextek.com/manuals/onix/stopwords1.html

Word Cloud using Python

Optimization

```
Get the Word Dict
Loop words in Word Dict:
   if word threshold > n:
         if  word not a stop word:
               save it in Selected Word Dict
         else:
               continue
   else:
         continue
```

Know the content without reading: President Trump, White House, and Washington Government shutdown this Friday!



Word Cloud using Python

We use ‘ ’(a space) to divide words one by one in English, how to segment words in Chinese or Japanese?

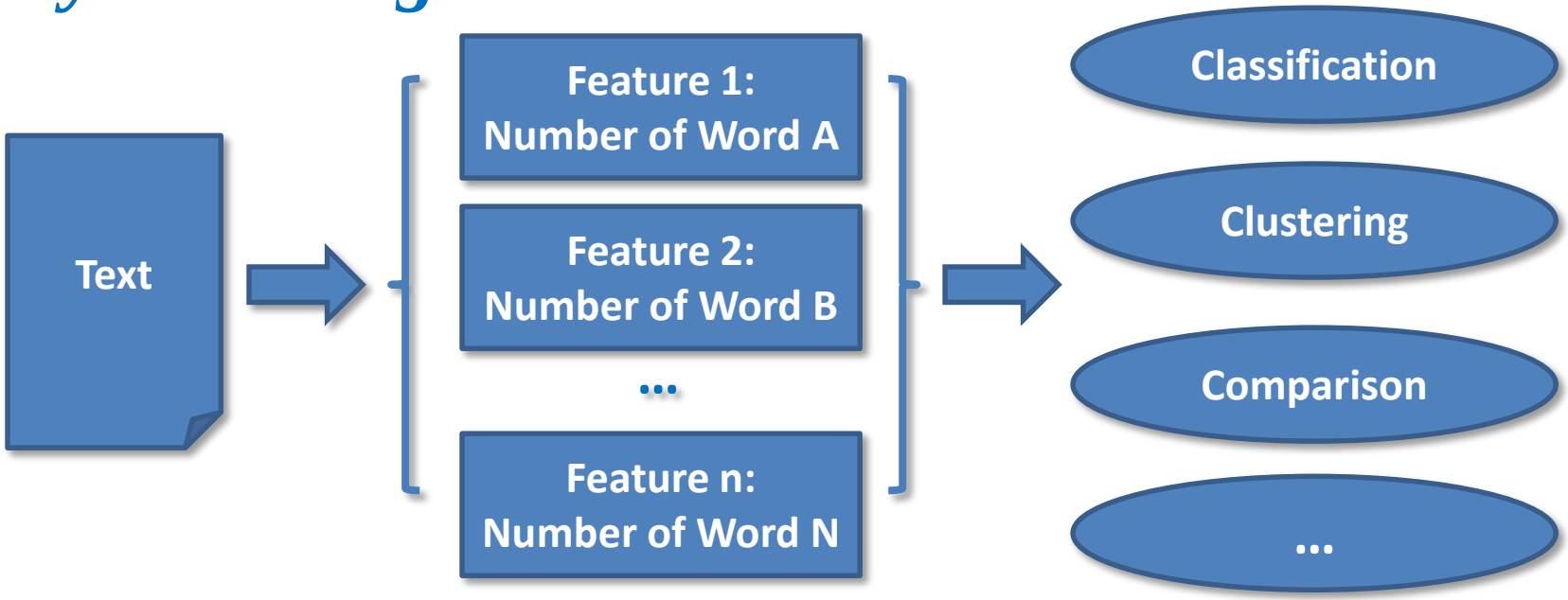
上海外国语大学
SHANGHAI INTERNATIONAL STUDIES UNIVERSITY

the first step for Chinese information processing

Chinese Word Segmentation

Chinese Word Segmentation

Why Word Segmentation?



However, it is difficult to extract words from Chinese text.

Chinese Word Segmentation

Difficulties: Disambiguation

乒乓球拍卖完了

乒乓|球拍|卖完了

乒乓球|拍卖|完了

N-gram

- Chinese Word Segmentation 分词
  - Forward Max. matching method 正向最大匹配
  - Backward Max. matching method 逆向最大匹配
  - Statistical matching method 统计学方法

Forward Max. matching method, FMM

正向最大匹配

准备工作：需要分词词典D
设MaxLen表示最大词长度
算法：

1. 从生语料N中取长度为MaxLen的字串str，令Len= MaxLen
2. 把str与D中的词相匹配
3. 若匹配成功，则认为str为词，N中去掉str(指针前移Len个单位)，返回1
4. 若匹配不成功，

- 若Len>1则Len--，从生语料N中取长度为Len的字串str返回2；
- 否则，得到单字词， N中去掉str(指针前移1个单位)，返回1

若4中得到的单字不是词，则要进行未登录词处理
若待切分的语料字串长度小于MaxLen，则取str为待切分语料

Chinese Word Segmentation

Backward Max. matching method, BMM

逆向最大匹配

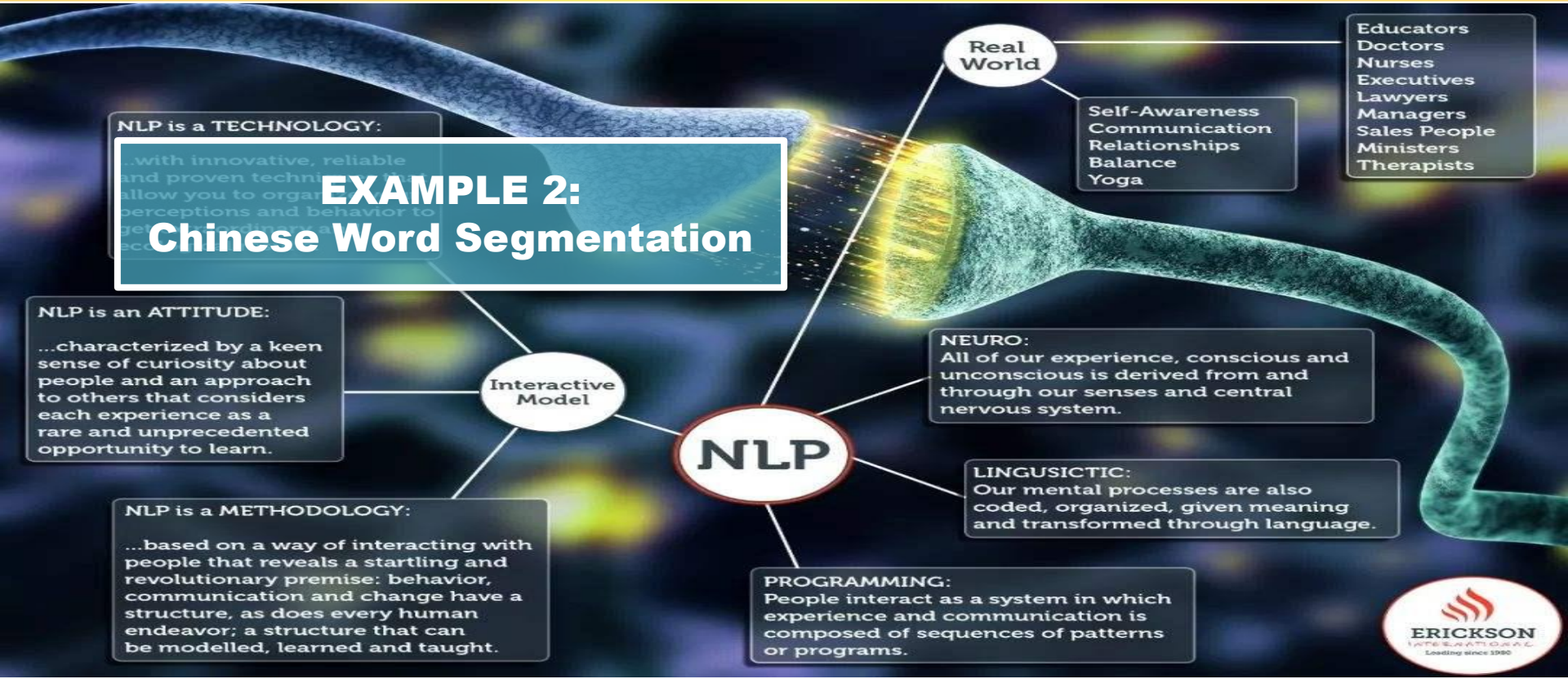
1. Similar to FMM, but the text is scanned from the right side
2. Often jointly use with FMM

Chinese Word Segmentation

- **Statistical Matching Method**

```
FMM and BMM
Begin initialize Path←{},AmbiguousString,SubString←{}
   While (AmbiguousString.Length>0)
   {
       //只考虑以当前HMM第一个状态开始的匹配序列
       SubString←以AmbiguousString中的第一个字为基准，取出所有可能的匹配字符串
       Foreach SubString
       {
           //提供当前情况下所有的概率，为判断歧义作参考
               计算当前每一种可能情况的概率P(SubString)  //unigram, bigram, trigram with smoothing
       }
       //选择概率最大的SubString添加到Path
       将argmax(P(SubString))添加到Path
       //准备考察除去最大概率的SubString后的AmbiguousString，从HMM序列首部开始，除去所有的匹配状态
       AmbiguousString.Remove(0, argmax(P(SubString)).Length)
    }
    Return Path
End
```

Chinese Word Segmentation



新华每日电讯

新华每日电讯

4.2万 文章 | 2.2亿 总阅读

查看TA的文章>

创新是改革开放的生命

2018-12-23 01:08

新华社评论员“创新是改革开放的生命。”习近平总书记在庆祝改革开放40周年大会重要讲话中作出的这一重要论断,深刻指明创新对于推进改革开放事业的极端重要性,为进一步解放思想、开拓创新注入了强大精神动力。

“苟日新,日日新,又日新”,创新是一个民族进步的灵魂,是一个国家兴旺发达的不竭动力,也是中华民族最深沉的民族禀赋。改革开放40年来,党带领人民大胆地试、勇敢地改,闯出了一片新天地,探索出前无古人的中国特色社会主义道路。从实行家庭联产承包到农村承包地“三权”分置,从兴办经济特区到设立自贸区,改革开放事业就是在不断解放思想、开拓创新中从无到有发展起来的。

创新,是破局开路的利器;创新,是点亮未来的希望。没有创新就没有中国的今天,也就没有中国的明天。推进新时代的改革开放,必须把创新置于更加突出位置,不断推进理论创新、制度创新、科技创新、文化创新等各方面创新,让创新在全社会蔚然成风。

创新源泉的涌流,来自思想解放的破冰。“真理标准大讨论”如一声春雷,打破“两个凡是”的禁锢;建立社会主义市场经济体制的提出,解开束缚人们思想和行动多年的绳索。40年实践历程启示我们,没有思想的大解放,就不会有创新活力的大迸发,就不会有改革开放的大突破。实践发展永无止境,解放思想永无止境。新征程上,面对新形势、新任务、新挑战,更加需要打开

Chinese Word Segmentation

Step 1: get the text

```
import urllib.request
from bs4 import BeautifulSoup

response = urllib.request.urlopen('http://www.sohu.com/a/283822245_117503')
HTMLText = response.read()
BSobj = BeautifulSoup(HTMLText, "html.parser")
Content = BSobj.find("article", {"class": "article"})
print(Content.get_text())
```

Run: ChineseWordSegmentation01

```
"C:\Program Files\Python36\python.exe" "D:/SISU/Courses/2018计算新闻学（研）/Lecture 7/ChineseWordSegmentation01.py"

原标题：创新是改革开放的生命
    新华社评论员“创新是改革开放的生命。”习近平总书记在庆祝改革开放40周年大会重要讲话中作出的这一重要论断，深刻指明创新对于推进改革开放事业的极端重要性，为进一步解放思想、开拓创新注入了强大精神动力。
“苟日新，日日新，又日新”，创新是一个民族进步的灵魂，是一个国家兴旺发达的不竭动力，也是中华民族最深沉的民族禀赋。改革开放40年来，党带领人民大胆地试、勇敢地改，闯出了一片新天地，探索出前无古人的中国特色社会主义道路。从实行家庭联产承包到农村承包地“三权”分置，从兴办经济特区到设立自贸区，改革开放事业就是在不断解放思想、开拓创新中从无到有发展起来的。
创新，是破局开路的利器；创新，是点亮未来的希望。没有创新就没有中国的今天，也就没有中国的明天。推进新时代的改革开放，必须把创新置于更加突出位置，不断推进理论创新、制度创新、科技创新、文化创新等各方面创新，让创新在全社会蔚然成风。
创新源泉的涌流，来自思想解放的破冰。“真理标准大讨论”如一声春雷，打破“两个凡是”的禁锢；建立社会主义市场经济体制的提出，解开束缚人们思想和行动多年的绳索。40年实践历程启示我们，没有思想的大解放，就不会有创新活力的大迸发，就不会有改革开放的大突破。实践发展永无止境，解放思想永无止境。新征程上，面对新形势、新任务、新挑战，更加需要打开解放思想这个“总开关”，继续鼓起闯的勇气、迈开试的步子，及时回答时代之问、人民之问，廓清困扰和束缚实践发展的思想迷雾，不断推动实践基础上的理论创新，推动改革开放取得新成就。
当今世界，变革创新的潮流滚滚向前。谁排斥变革，谁拒绝创新，谁就会落后于时代，谁就会被历史淘汰。从这个意义上说，抓创新就是抓发展，谋创新就是谋未来。
“问渠那得清如许？为有源头活水来。”激发创新活力，要靠全面深化改革、扩大开放；随着改革开放不断向前推进，更强劲的创新动能必将喷涌而出。这是一个相互促进、相得益彰的过程。要通过深化各领域改革，破除一切制约创新的思想障碍和制度藩篱，在全社会大力营造勇于创新、鼓励成功、宽容失败的良好氛围，培育并用好各类创新人才，进一步激发全社会创新活力和创造潜能。
惟改革者进，惟创新者强，惟改革创新者胜。新时代的改革开放，呼唤我们弘扬创新精神、汇聚创新力量、推动创新发展，在更加广袤的历史时空创造新的更大奇迹。
新华社北京12月22日电返回搜狐，查看更多  责任编辑：
```

Python Console | Terminal | 4: Run | 6: TODO | Event Log

Chinese Word Segmentation

Step2: Cut words using FMM

Step2-1: build the dictionary of Chinese Words

准备工作：需要分词词典D
设MaxLen表示最大词长度
算法：
1. 从生语料N中取长度为MaxLen的字串str，令Len= MaxLen
2. 把str与D中的词相匹配
3. 若匹配成功，则认为str为词，N中去掉str(指针前移Len个单位)，返回1
4. 若匹配不成功，
- 若Len>1则Len--，从生语料N中取长度为Len的字串str返回2；
- 否则，得到单字词， N中去掉str(指针前移1个单位)，返回1

若4中得到的单字不是词，则要进行未登录词处理
若待切分的语料字串长度小于MaxLen，则取str为待切分语料

Chinese Word Segmentation

Step2: Cut words using FMM

Step2-1: build the dictionary of Chinese Words

```
def word_map_initial(): #把词汇装入词典
    WordMap=set()
    conn = pymysql.connect(user='root', password='123456', database='nlp',charset="utf8")
    cursor = conn.cursor()
    sqlstr="SELECT DISTINCT WORD_NAME FROM NLP_WORD_MAP"
    cursor.execute(sqlstr)
    rows = cursor.fetchall()
    for row in rows:
        WordMap.add(str(row[0]))
    conn.commit()
    cursor.close()
    conn.close()
    return WordMap
```

Chinese Word Segmentation

Step2: Cut words using FMM

Step2-2: FMM Code

```
def word_seg_fmm(content): #正向匹配
    WordMap = word_map_initial()
    MaxLen=10 #最大词长
    Len=MaxLen #动态切割词长
    Seg_Content="" #返回的切割结果

    while len(content)>0:
        if content[0:Len] in WordMap: #词典中有匹配
            Seg_Content=Seg_Content+content[0:Len]+"|"
            content=content[Len:]
            Len=MaxLen
            #print("Seg_Content1:"+Seg_Content)
            continue
        else: #词典中无匹配
            Len=Len-1
            if Len==1:#仅剩一个词还没匹配到
                Seg_Content = Seg_Content + content[0:Len] + "|"
                content = content[Len:]
                Len = MaxLen
                #print("Seg_Content2:" + Seg_Content)
    return Seg_Content[:-1]
```

Chinese Word Segmentation

Step 3: Revise the head and the main body.

```
1   import pymysql
2   import urllib.request
3   from bs4 import BeautifulSoup
4
5   def word_map_initial():...
31
32  def word_seg_fmm(content):...
53
54  response = urllib.request.urlopen('http://www.sohu.com/a/283822245_117503')
55  HTMLText = response.read()
56  BSobj = BeautifulSoup(HTMLText, "html.parser")
57  Content = BSobj.find("article", {"class": "article"})
58  WordSegResult = word_seg_fmm(Content.get_text())
59  print(WordSegResult)
```

Chinese Word Segmentation

Results:

```
"C:\Program Files\Python36\python.exe" "D:/SISU/Courses/2018计算新闻学（研）/Lecture 7/ChineseWordSegmentation.py"

|原|标题|：|创新|是|改革|开放|的|生命|
|  |  |新华社|评论员|“|创新|是|改革|开放|的|生命|。|”|习|近平|总书记|在|庆祝|改革|开放|40|周年|大会|重要|讲话|中|作出|的|这|一|重要|论断|，|深刻|指明|创新|对于|推进|改革|开放|事业|的|极端|重要性|，|为|进一步|解放思想|、|开拓|创新|注入|了|强大|精神|动力|。|  |
|“|苟|日新|，|日日|新|，|又|日新|”|，|创新|是|一个|民族|进步|的|灵魂|，|是|一个|国家|兴旺发达|的|不竭|动力|，|也|是|中华民族|最|深沉|的|民族|禀赋|。|改革|开放|40|年|来|，|党|带领|人民|大胆|地|试|、|勇敢|地|改|，|闯|出|了|一|片|新|天地|，|探索|出|前无古人|的|中国|特色|社会主义|道路|。|从|实行|家庭|联产承包|到|农村|承包地|“|三|权|”|分|置|，|从|兴办|经济特区|到|设立|自贸区|，|改革|开放|事业|就是|在|不断|解放思想|、|开拓|创新|中|从无到有|发展|起来|的|。|  |
|创新|，|是|破|局|开路|的|利器|；|创新|，|是|点|亮|未来|的|希望|。|没有|创新|就|没有|中国|的|今天|，|也|就|没有|中国|的|明天|。|推进|新时|代|的|改革|开放|，|必须|把|创新|置于|更加|突出|位置|，|不断|推进|理论|创新|、|制度|创新|、|科技|创新|、|文化|创新|等|各方|面|创新|，|让|创新|在|全社|会|蔚然成风|。|  |
|创新|源泉|的|涌流|，|来自|思想解放|的|破|冰|。|“|真理|标准|大|讨论|”|如|一|声|春雷|，|打破|“|两个凡是|”|的|禁锢|；|建立|社会主义|市场经济|体制|的|提出|，|解开|束缚|人们|思想|和|行动|多年|的|绳索|。|40|年|实践|历程|启示|我们|，|没有|思想|的|大解|放|，|就|不会|有|创新|活力|的|大|迸发|，|就|不会|有|改革|开放|的|大|突破|。|实践|发展|永无止境|，|解放思想|永无止境|。|新|征程|上|，|面对|新|形势|、|新任|务|、|新|挑战|，|更加|需要|打开|解放思想|这个|“|总|开关|”|，|继续|鼓|起|闯|的|勇气|、|迈开|试|的|步子|，|及时|回答|时代|之|问|、|人民|之|问|，|廓清|困扰|和|束缚|实践|发展|的|思想|迷雾|，|不断|推动|实践|基础|上|的|理论|创新|，|推动|改革|开放|取得|新|成就|。|  |
|当今|世界|，|变革|创新|的|潮流|滚滚|向前|。|谁|排斥|变革|，|谁|拒绝|创新|，|谁|就|会|落后|于|时代|，|谁|就|会|被|历史|淘汰|。|从|这个|意义|上|说|，|抓|创新|就是|抓|发展|，|谋|创新|就是|谋|未来|。|  |
|“|问|渠|那|得|清|如|许|？|为|有|源头|活水|来|。|”|激发|创新|活力|，|要|靠|全面|深化|改革|、|扩大|开放|；|随着|改革|开放|不断|向前|推进|，|更|强劲|的|创新|动能|必将|喷涌|而|出|。|这|是|一个|相互|促进|、|相得益彰|的|过程|。|要|通过|深化|各|领域|改革|，|破除|一切|制约|创新|的|思想|障碍|和|制度|藩篱|，|在|全社|会|大力|营造|勇于|创新|、|鼓励|成功|、|宽容|失败|的|良好|氛围|，|培育|并用|好|各类|创新|人才|，|进一步|激发|全社|会|创新|活力|和|创造|潜能|。|  |
|惟|改革者|进|，|惟|创新者|强|，|惟|改革|创新者|胜|。|新时|代|的|改革|开放|，|呼唤|我们|弘扬|创新|精神|、|汇聚|创新力|量|、|推动|创新|发展|，|在|更加|广袤|的|历史|时空|创造|新|的|更|大|奇迹|。|  |
|新华社|北京|1|2|月|2|2|日|电|返回|搜狐|，|查看|更|多|  |责任|编辑|：|


Process finished with exit code 0
```

Chinese Word Segmentation Algorithms

- FMM
- BMM

```
def word_seg_fmm(content): #正向匹配
    MaxLen=10 #最大词长
    Len=MaxLen #动态切割词长
    Seg_Content="" #返回的切割结果

    while len(content)>0:
        if content[0:Len] in WordMap: #词典中有匹配
            Seg_Content=Seg_Content+content[0:Len]+"|"
            content=content[Len:]
            Len=MaxLen
            #print("Seg_Content1:"+Seg_Content)
            continue
        else: #词典中无匹配
            Len=Len-1
            if Len==1:#仅剩一个词还没匹配到
                Seg_Content = Seg_Content + content[0:Len] + "|"
                content = content[Len:]
                Len = MaxLen
                #print("Seg_Content2:" + Seg_Content)
    return Seg_Content[:-1]
```

```
def word_seg_bmm(content): #逆向匹配
    MaxLen=10 #最大词长
    Len=MaxLen #动态切割词长
    Seg_Content="" #返回的切割结果

    while len(content)>0:
        if content[-Len:] in WordMap: #词典中有匹配
            Seg_Content=content[-Len:]+"|"+Seg_Content
            content=content[:-Len]
            Len=MaxLen
            #print("Seg_Content1:"+Seg_Content)
            continue
        else: #词典中无匹配
            Len=Len-1
            if Len==1:#仅剩一个词还没匹配到
                Seg_Content = content[-Len:] + "|" + Seg_Content
                content = content[:-Len]
                Len = MaxLen
                #print("Seg_Content2:" + Seg_Content)
    return Seg_Content[:-1]
```

Chinese Word Segmentation

Tips for Chinese Word Segmentation

- Initialization is very important
- Segment in the memory (not hard disk or data bases) to accelerate the segmentation speed
- Using “set” to store the dictionary, and “dict” for segmented words in Python
- For Tag Analysis, a precise word segmentation is unnecessary

上海外国语大学
SHANGHAI INTERNATIONAL STUDIES UNIVERSITY

data visualization with word cloud for Chinese text

Chinese Word Cloud

Chinese Word Cloud

Steps:

1. Download a news report
2. Word segmentation
3. Word tag extraction and statistical computing
4. Data visualization and news summarization

Chinese Word Cloud

Database Preparation

- Word Dictionary (required)
- Stop Word Dictionary (required)
- Dictionaries of Terms (optional)
- Word Chains (required if using N-gram)
- Part of Speech (optional)
- Word Sentiment (optional for Sentiment Analysis)

Chinese Word Cloud

Word Frequency Computing for Chinese

- str.split() for all tags
- Discarding One-Char tags
- Discarding Stop-Word tags
- Select tags whose term frequencies are larger than a threshold (for example >2)
- Other statistical computing

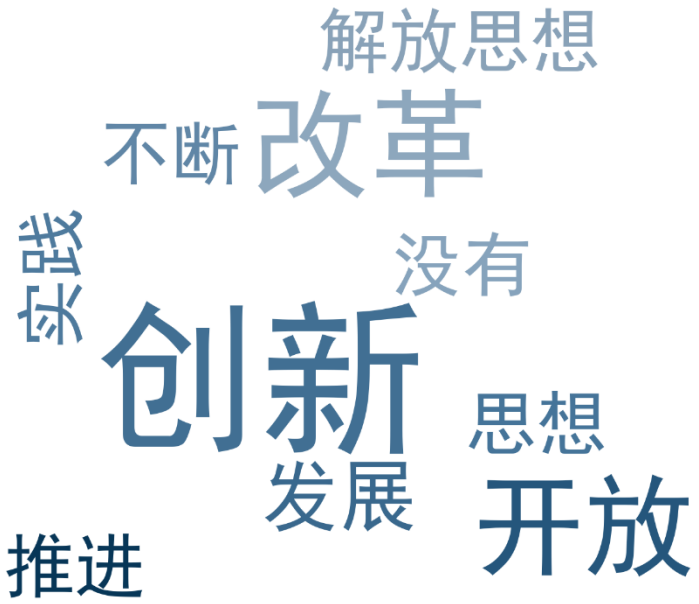
Chinese Word Cloud

Chinese Word Frequency

```
def WordFrequency(InputText):
    s_array = str(InputText)
    s_array = s_array.replace(',', '|')
    s_array = s_array.replace('.', '|')
    s_array = s_array.replace('\'', '|')
    s_array = s_array.replace('\"', '|')
    s_array = s_array.replace(':', '|')
    s_array = s_array.replace(';', '|')
    s_array = s_array.replace('?', '|')
    s_array = s_array.replace('`', '|')
    s_array = s_array.replace('~', '|')
    s_array = s_array.replace('!', '|')
    s_array = s_array.replace('@', '|')
    s_array = s_array.replace('#', '|')
    s_array = s_array.replace('%', '|')
    s_array = s_array.replace('$', '|')
    s_array = s_array.replace('^', '|')
    s_array = s_array.replace('&', '|')
    s_array = s_array.replace('*', '|')
    s_array = s_array.replace('(', '|')
    s_array = s_array.replace(')', '|')
    s_array = s_array.replace('-', '|')
    s_array = s_array.replace('+', '|')
    s_array = s_array.replace('_', '|')
    s_array = s_array.replace('=', '|')
    s_array = s_array.replace('{', '|')
    s_array = s_array.replace('}', '|')
    s_array = s_array.replace('[', '|')
    s_array = s_array.replace(']', '|')
    s_array = s_array.replace('|', '|')
    s_array = s_array.replace('\\', '|')
    s_array = s_array.replace('<', '|')
    s_array = s_array.replace('>', '|')

    s_array = s_array.replace('。', '|')
    s_array = s_array.replace('“', '|')
    s_array = s_array.replace('”', '|')
    s_array = s_array.replace('；', '|')
    s_array = s_array.replace('，', '|')
    s_array = s_array.replace('——', '|')
    s_array = s_array.replace('……', '|')
    s_array = s_array.replace('？', '|')
    s_array = s_array.replace('！', '|')
    s_array = s_array.replace('￥', '|')
    s_array = s_array.replace('、', '|')
    s_array = s_array.replace('《', '|')
    s_array = s_array.replace('》', '|')
    s_array = s_array.replace('\u3000', '|')
    s_array = s_array.replace('\n', '|')
    s_array = s_array.split('|')
    word_dict = {}
    for i in range(len(s_array)):
        if s_array[i] not in word_dict:
            word_dict[s_array[i]] = 1
        else:
            word_dict[s_array[i]] += 1
    return word_dict
```

Chinese Word Cloud

Results:

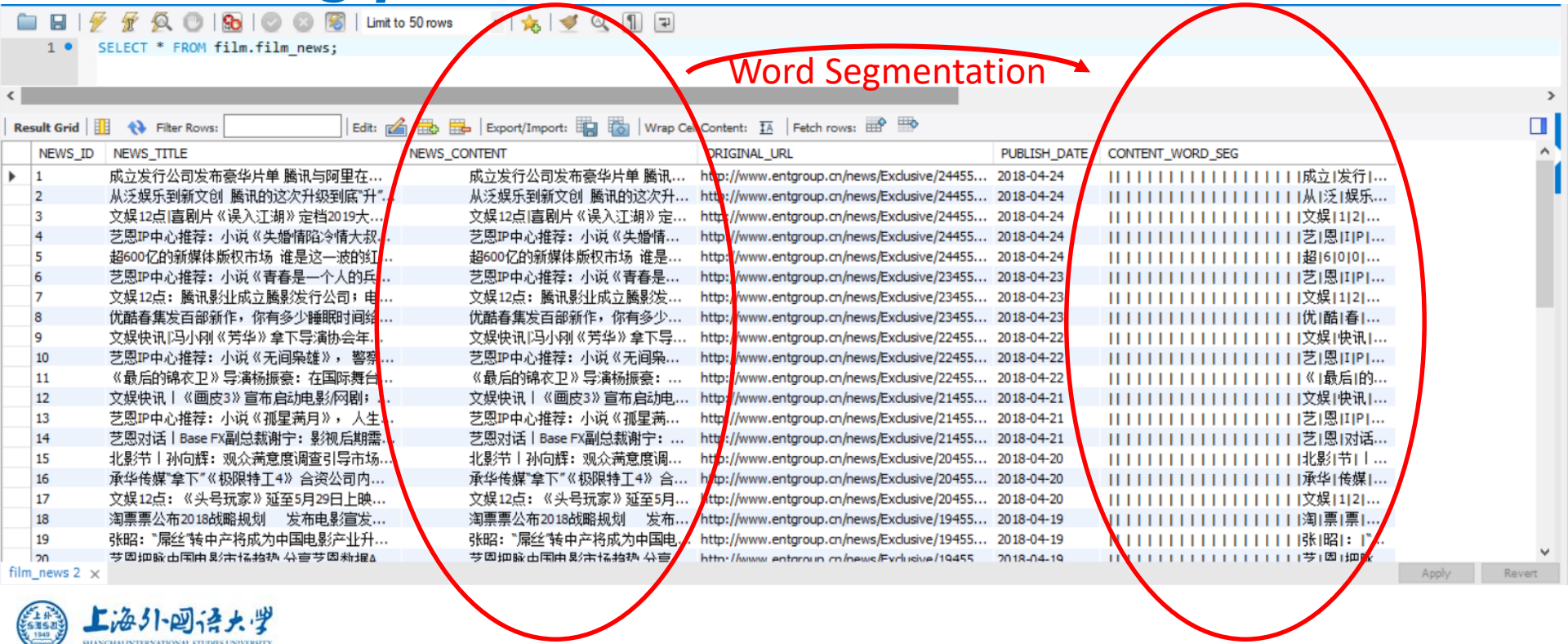



```python
import urllib.request
from bs4 import BeautifulSoup
import DataWordCloud
from ChineseWordSeg import word_seg_fmm,WordFilter

def WordFrequency(InputText):...

response = urllib.request.urlopen('http://www.sohu.com/a/283822245_117503')
HTMLText = response.read()
BSobj = BeautifulSoup(HTMLText, "html.parser")
Content = BSobj.find("article", {"class": "article"})

# print(WordFrequency(Content.get_text()))

WordSegResult = word_seg_fmm(Content.get_text())

WordDict = WordFrequency(WordSegResult)

SelectedWordDict={}
SelectedWordDict = WordFilter(WordDict)

threshold = 4
for key, value in SelectedWordDict.items():
    if value >= threshold:
        continue
    else:
        SelectedWordDict.pop(key)

picurl=DataWordCloud.word_cloud_generate(SelectedWordDict)

print("word cloud saved in "+picurl)
```

Tips for your Final Project

Tips for your Final Project

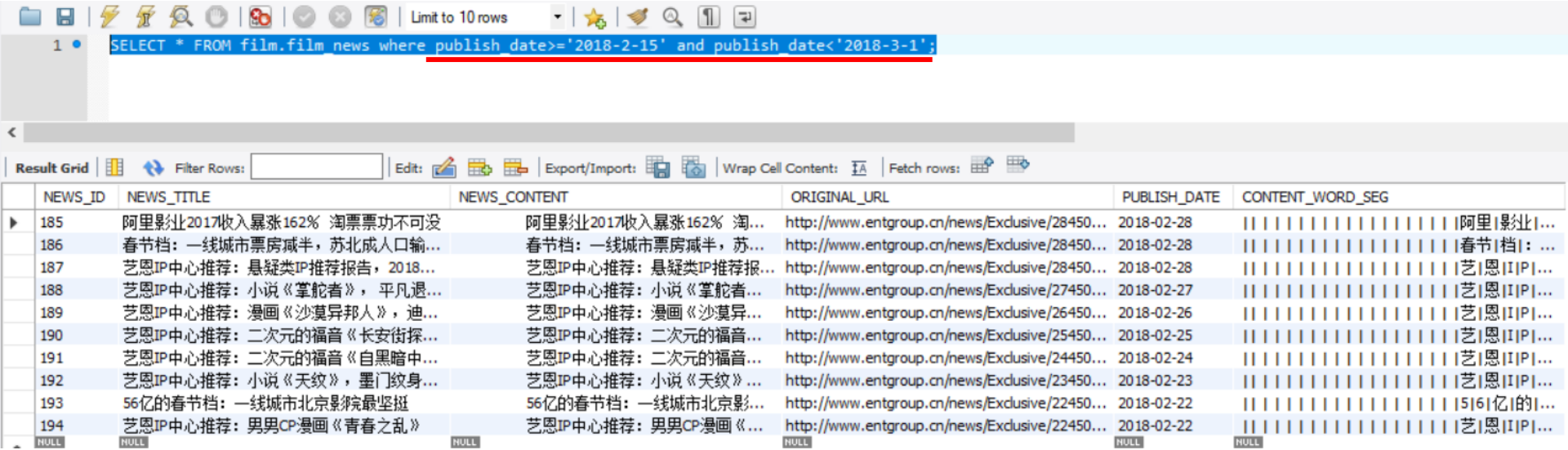
Word seg for NEWS_CONTENT



Tips for your Final Project

Date/Time limitations for Word Cloud

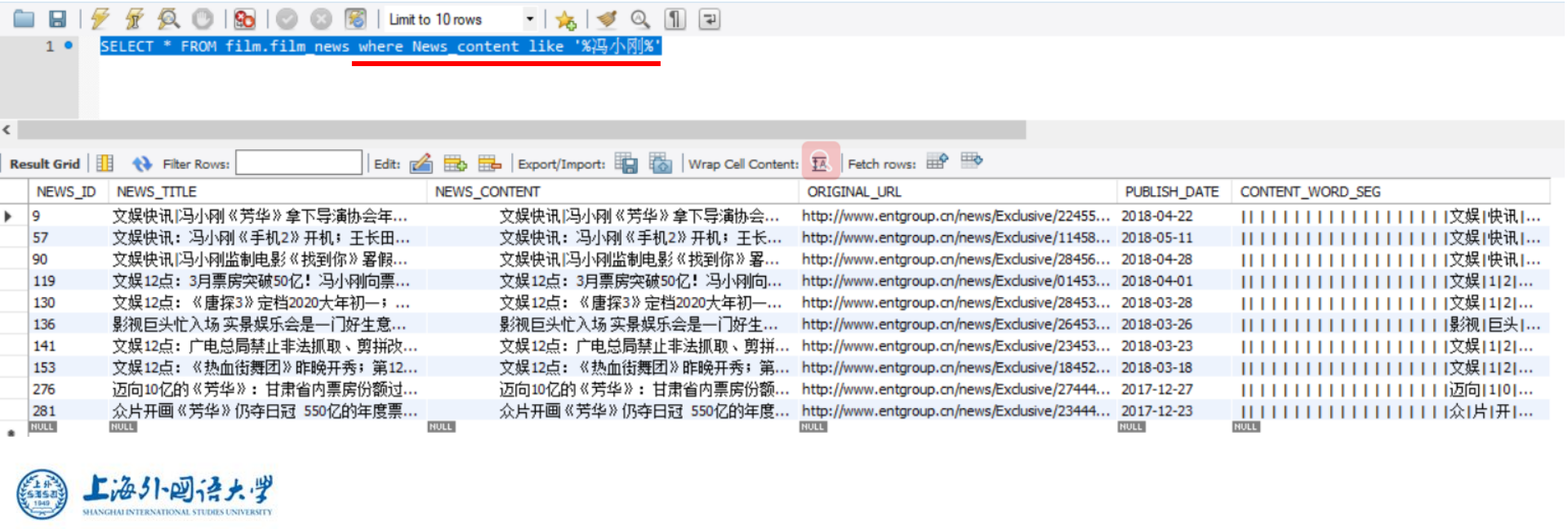
- **Condition: Where …**



Tips for your Final Project

Content limitations for Word Cloud

– Condition: Where …



上海外国语大学
SHANGHAI INTERNATIONAL STUDIES UNIVERSITY

The End of Lecture 7

Thank You

http://www.wangting.ac.cn