第2节 基本文字处理：正则表达式

From Languages to Information CS124
—— Lecture 2: Basic Text Processing
https://web.stanfhttps://web.stanford.edu/class/cs124/lec/textprocessingboth.pdford.edu/class/cs224n/lectures/cs224n-2017-lecture2.pdf

正则表达式 (Regular Expressions)

- 正则表达式是对字符串操作的一种逻辑公式，就是用事先定义好的一些特定字符、及这些特定字符的组合，组成一个**规定字符串**。
- 如何搜索这些词？
  - **woodchuck（土拨鼠）**
  - **woodchucks**
  - **Woodchuck**
  - **Woodchucks**

正则表达式：析取

- 字符集合 []

Pattern	Matches
[wW]oodchuck	Woodchuck, woodchuck
[1234567890]	任一数字

- 范围表示 [A-Z]

Pattern	Matches	
[A-Z]	单个大写字母	Drenched Blossoms
[a-z]	单个小写字母	my beans were impatient
[0-9]	单个数字	Chapter 1: Down the Hole

正则表达式：否定析取

- 否定 [^Ss]
  - 仅当“^”出现在字符集合模式（[]）的第一个字符时

Pattern	Matches	
[^A-Z]	非单个大写字母	Oyfn pripetchik
[^Ss]	既不是“S”也不是“s”	I have no exquisite reason
a^b	与字段相同	Look up a^b now

正则表达式：其他析取

- Woodchucks is another name for groundhog!
- 析取中的“|”，对匹配条件进行逻辑“或”运算

Pattern	Matches
`groundhog|woodchuck`	`groundhog` 或 `woodchuck`
`yours|mine`	`yours` 或 `mine`
`a|b|c`	`=[abc]`

正则表达式： ? * + .

Pattern	Matches	
colou?r	匹配前面的子表达式一次或零次	color colour
oo*h!	匹配前面的子表达式任意次	oh! ooh! oooooh!
o+h!	匹配前面的子表达式一次或多次	baa baaa baaaaaa
beg.n	匹配除“\n”和“\r”之外的任何单个字符	begin begun beg9n

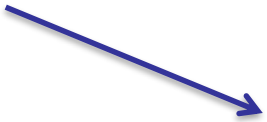
正则表达式： ^ $

Pattern	Matches	
^[A-Z]	匹配输入字行首	Palo Alto colour
^[^A-Za-z]	如果设置了RegExp对象的Multiline属性，^也匹配“\n”或“\r”之后的位置	1 “Hello”
\.$	匹配输入行尾	The end.
.$	如果设置了RegExp对象的Multiline属性，^也匹配“\n”或“\r”之前的位置	The end? The end!

举例

- 找到文本中的“**the**”的所有实例

the



遗漏了所有大写字母开头的实例

[tT]he

错误返回其他词汇（例：other, theology）

[^a-zA-Z][tT]he[^a-zA-Z] ✔

错误

- 我们刚刚经历的过程修复了两种错误
  - 匹配本不应被匹配的字符串（there，then，other）
    - **False positives**（误报 **Type I**）
  - 没有匹配到本应被匹配的字符串（The）
    - **False negatives**（漏报 **Type II**）

错误

- 在NLP中，我们经常处理此类错误
- 降低应用程序中的错误率通常涉及两种对立的工作：
  - 提高准确性或精确度（最小化False positives）
  - 提高覆盖率或召回度（最小化False negatives）

总结

- 正则表达式起着至关重要的作用
  - 对于任何的文本处理来说，正则表达式通常是复杂序列的首选模型
- 对于许多困难的任务，我们使用机器学习分类器
  - 但是正则表达式将作为分类器中的特征值
  - 在获取一般化时非常有用

第2节 基本文字处理：字标记

学习Natural Language Processing with Deep Learning CS224N/Ling284
—— Lecture 2: Word Vectors https://web.stanford.edu/class/cs224n/lectures/cs224n-2017-lecture2.pdf
Lecture Notes 1 https://web.stanford.edu/class/cs224n/lecture_notes/cs224n-2017-notes1.pdf

文本归一化 (Text Normalization)

- 每个NLP任务都需要做**文本归一化**处理：
  1. 在运行文本中对单词进行分段、标记
  2. 规范字格式
  3. 在运行文本中对句子进行分段

How many words?

- I do uh main- mainly business data processing
  - Fragments（片段）, filled pauses（填充停顿）
- Seuss's cat in the hat is different from other cats!
  - **Lemma**（词元）: same stem（相同词干）, part of speech, rough word sense
    - cat and cats = same lemma
  - **Wordform**（词形）: the full inflected surface form（完全改变的表示）
    - cat and cats = different wordforms

How many words?

they lay back on the San Francisco grass and looked at the stars and their

- **Type**: an element of the vocabulary
- **Token**: an instance of that type in running text
- How many?
  - 15 tokens (or 14)
  - 13 types (or 12) (or 11?)

How many words?

N = number of tokens

V = vocabulary = set of types

Church and Gale (1990): |V| > O($N^{1/2}$)

$|V|$ is the size of the vocabulary

	Tokens = N	Types = |V|
Switchboard phone conversations	2.4 million	20 thousand
Shakespeare	884,000	31 thousand
Google N-grams	1 trillion	13 million

UNIX 中的简单标记

- (Inspired by Ken Church's UNIX for Poets.)
- 给定一个文本文件，输出单词标记及其频率

```
tr -sc 'A-Za-z' '\n' < shakes.txt
    | sort
    | uniq -c
```

Change all non-alpha to newlines

Sort in alphabetical order

Merge and count each type

```
1945 A            25 Aaron
  72 AARON         6 Abate
  19 ABBESS        1 Abates
   5 ABBOT         5 Abbess
 ... ...           6 Abbey
                   3 Abbot
                  ... ...
```

第一步：分词（tokenizing）

```
tr -sc 'A-Za-z' '\n' < shakes.txt | head
```

```
THE
SONNETS
by
William
Shakespeare
From
fairest
creatures
We
...
```

第二步：排序（sorting）

```
tr -sc 'A-Za-z' '\n' < shakes.txt | sort | head
```

```
A
A
A
A
A
A
A
A
A
...
```

更多的计算

- 合并大小写

```
tr 'A-Z' 'a-z' < shakes.txt | tr -sc 'A-Za-z' '\n' | sort |
uniq -c
```

- 对计数排序

```
tr 'A-Z' 'a-z' < shakes.txt | tr -sc 'A-Za-z' '\n' | sort |
uniq -c | sort -n -r
```

```
23243 the
22225 i
18618 and
16339 to
15687 of
12780 a
12163 you
10839 my
10005 in
8954  d
```

分词中的问题

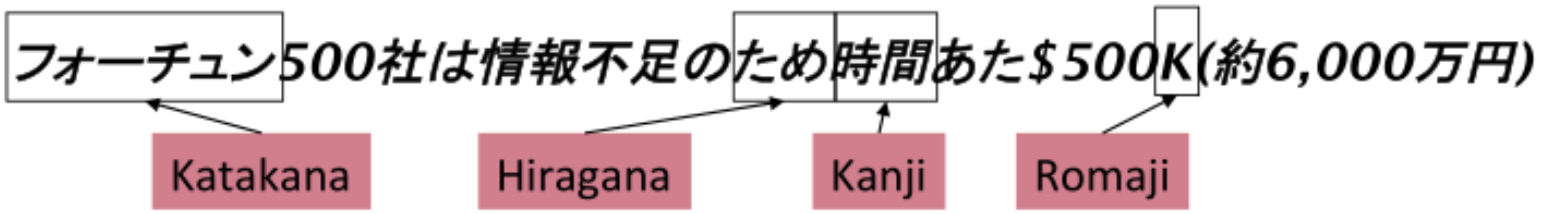
- Finland's capital → Finland Finlands Finland's ?
- What're, I'm, isn't → What are, I am, is not
- Hewlett-Packard → Hewlett Packard ?
- state-of-the-art → state of the art ?
- Lowercase → lower-case lowercase lower case ?
- San Francisco → one token or two ?
- m.p.h., PhD. → ??

分词：语言问题

- French
  - ***L'ensemble*** → one token or two?
    - ***L*** ? ***L'*** ? ***Le*** ?
    - Want ***l'ensemble*** to match with ***un ensemble***
- German noun compounds are not segmented（德语中的名词复合词不分段）
  - ***Lebensversicherungsgesellschaftsangesteller***
    - 'life insurance company employee'
    - German information retrieval needs ***compound splitter***（德语的信息检索需要复合词分配器）

分词：语言问题

- 中文和日文的词汇间没有空格分隔
  - **莎拉波娃现在居住在美国东南部的弗罗里达。**
  - **莎拉波娃 现在 居住 在 美国 东南部 的 弗罗里达**
  - **Sharapova now lives in US southeastern Florida**
- 日文更复杂，有多个字母混杂
  - 多种格式的日期/数量表示



中文分词

- 中文词汇是由汉字组成的
  - 汉字通常含有一个音节和一个语素
  - 每个词汇平均由2.4个汉字组成
- 标准基线分割算法：
  - 最大匹配算法（也称贪心算法 - Greedy）

最大匹配 —— 分词算法

- 给定一个中文词汇表和一个字符串

1. 在字符串开头启动指针
2. 在字典中找到与从指针开始的字符串匹配的最长单词
3. 将指针移动至越过此单词
4. 跳至第 2 步重复此过程

最大匹配分割猜想

- Thecatinthehat　　the cat in the hat
- Thetabledownthere　　the table down there

  theta bled own there
- 对英语不适用！
- 但是对中文非常适用
  - **莎拉波娃现在居住在美国东南部的弗罗里达。**
  - **莎拉波娃 现在 居住 在 美国 东南部 的 弗罗里达**
- 现代概率分割算法的效果更好

第2节 基本文字处理：词语规范化和词干提取

学习Natural Language Processing with Deep Learning CS224N/Ling284

—— Lecture 2: Word Vectors https://web.stanford.edu/class/cs224n/lectures/cs224n-2017-lecture2.pdf

Lecture Notes 1 https://web.stanford.edu/class/cs224n/lecture_notes/cs224n-2017-notes1.pdf

规范化（Normalization）

- 需要规范化用语
  - 信息检索：索引文本 & 查询字词必须有相同的格式
    - We want to match ***U.S.A.*** and ***USA***
- 我们隐式定义了属于的等价类
  - e.g., 删除字词中的句号
- 替代方案：非对称性扩展（asymmetric expansion）
  - Enter: ***window*** Search: ***window, windows***
  - Enter: ***windows*** Search: ***Windows, windows, window***
  - Enter: ***Windows*** Search: ***Windows***
- 可能更强大，但效率更低

大写转换（Case folding）

- 像IR这样的应用：将所有字母转换为小写
  - 由于用户倾向于使用小写字母
  - 可能出现的例外：句中的大写字母？
    - e.g., ***General Motors***
    - ***Fed*** vs. ***fed***
    - ***SAIL*** vs. ***sail***
- 对于情感分析，MT，信息提取
  - 字母的大小写转换是非常有帮助的（***US*** versus ***us*** is important）

词形还原（Lemmatization）

- 将变形或变体形式转换为基本形式
  - *am, are, is → be*
  - *car, cars, car's, cars' → car*
- *The boy's cars are different colors → the boy car be different color*
- 词形还原：必须找到正确的字典首词（headword）形式
- 机器翻译（Machine translation）
  - Spanish quiero ('I want'), quieres ('you want') same lemma as querer 'want'

形态学（Morphology）

- 语素：
  - 构成单词的微小而有意义的单元
  - 词干（stems）：核心含义单位
  - 词缀（Affixes）：依附于词干上的单元
    - 通常具有语法功能

词干提取（Stemming）

- 在信息检索中，将词语缩短至其词干形式
- 词干提取即是对词缀的粗略删减
  - 语言依赖
  - e.g. ***automate(s), automatic, automation*** all reduced to ***automat***.

For example compressed and compression are both accepted as equivalent to compress

For exampl compress and compress ar both accept as equival to compress

词干提取算法（Porter's algorithm）
最为普遍的英文词干提取算法

Step 1a

sses	→ ss	caresses	→ caress
ies	→ I	ponies	→ poni
ss	→ ss	caress	→ caress
s	→ Ø	cats	→ cat

Step 1b

(*V*)ing	→ Ø	walking	→ walk
		sing	→ sing
(*V*)ed	→ Ø	plastered	→ plaster

…

Step 1a

ational	→ ate	relational	→ relate
izer	→ ize	digitizer	→ digitize
ator	→ ate	operator	→ operate

…

Step 1a

sses	→ ss	caresses	→ caress
ies	→ I	ponies	→ poni
ss	→ ss	caress	→ caress
s	→ Ø	cats	→ cat

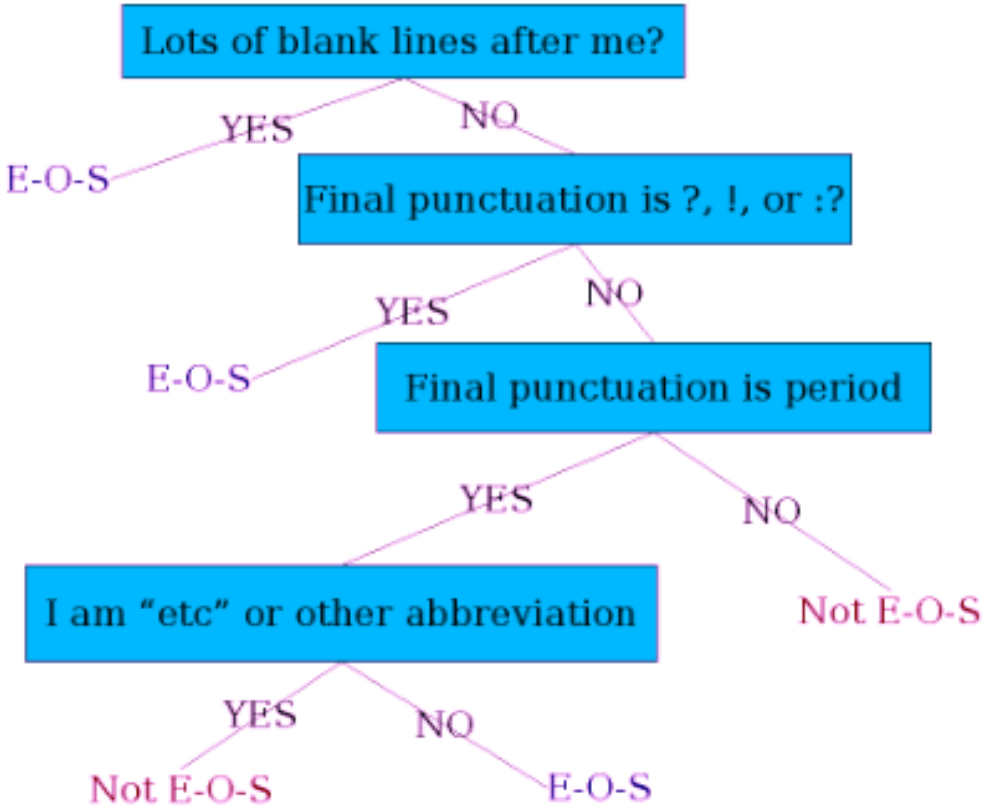
第2节 基本文字处理：句子划分与决策树

学习Natural Language Processing with Deep Learning CS224N/Ling284
—— Lecture 2: Word Vectors https://web.stanford.edu/class/cs224n/lectures/cs224n-2017-lecture2.pdf
Lecture Notes 1 https://web.stanford.edu/class/cs224n/lecture_notes/cs224n-2017-notes1.pdf

句子划分（Sentence Segmentation）

- ！，？是相对明确的标点符号
- 句号“.”的含义模糊
  - 句子边界
  - 缩写表示（Dr.）
  - 数学表示（.02%）
- 构建二元分类器
  - Looks at a “.”
  - Decides EndOfSentence/NotEndOfSentence
  - Classifiers: hand-written rules, regular expressions, or machine-learning

确定单词是否为句子结尾：决策树（Decision Tree）



决策树的实现

- 决策树仅仅是 if-then-else 的语句
- 有趣的是其选择功能
- 难以手动建立决策树结构
  - 手动构建至适用于简单的特征及领域
    - 对于数字要素，选择每个阈值太难
  - 相反，结构通常通过机器学习从训练语料库中得出

决策树和其他分类器

- 当特征可被任何类型的分类器利用时
  - 线性回归（Logistic regression）
  - SVM
  - 神经网络（Neural Nets）

Thank you !