

自然语言处理

Basic Text Processing

几个基本的文本处理实例

- 例：检索包含“面向对象编程”和“stro*rup”的文档
 - 其中，“*”是通配符
- 例：如何理解和处理字符串：“can’t,” “\$22.50,” “New York,” and “so-called”
- 例：如何比较两个字符串的相似程度：“chopsticks” and “chosticks”

Topics for Today

- 正则表达 (Regular Expression)
- 形符化 (Tokenization)
 - 词的形符化
 - 词的归一化 (Normalization)
 - 句子的形符化
- 最小编辑距离 (Minimum Edit Distance)
 - 字符串的比较
 - 最小编辑距离
 - 加权的最小编辑距离

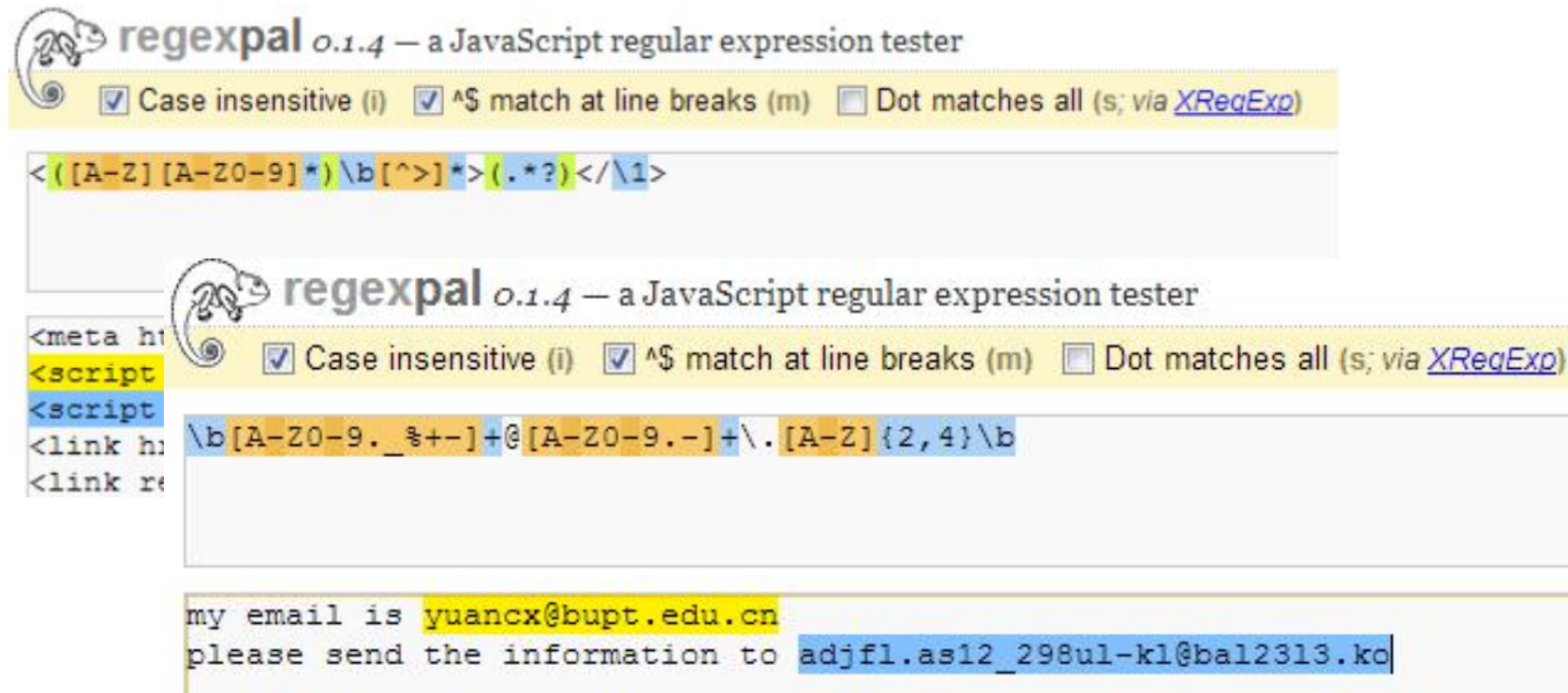
Ref. J & M, 2009 Ch3.

正则表达

- 使用一个字符串来描述、匹配一系列符合某个句法规则的字符串
- 正则表达式通常被用来检索、替换那些符合某个模式的文本
- 去除HTML Tags
 - `<TAG\b[^>]*>(.*?)</TAG>` : 匹配一组HTML标签对
- 去除空格
 - `^\s+` : 删除开头的空格 (spaces and tabs)
 - `\s+$` : 删除结尾的空格 (spaces and tabs)
 - 验证Email地址:
 - `\b[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}\b`
-

正则表达

- Examples:



The image displays two screenshots of the **regexpal** 0.1.4 website, which is a JavaScript regular expression tester. The interface includes a search bar, a text area for the regular expression, and a text area for the test string. The first screenshot shows a regular expression `<([A-Z][A-Z0-9]*)\b[^>]*>(.*?)</\1>` being tested against the string `<meta http-equiv="script"><script><link href="mailto:yuancx@bupt.edu.cn"><link href="mailto:adjfl.as12_298ul-k1@bal2313.ko">`. The second screenshot shows the same regular expression being tested against the string `my email is yuancx@bupt.edu.cn please send the information to adjfl.as12_298ul-k1@bal2313.ko`. The matches are highlighted in yellow and blue.

regexpal 0.1.4 — a JavaScript regular expression tester

☒ Case insensitive (i) ☒ ^\$ match at line breaks (m) ☐ Dot matches all (s; via [XRegExp](#))

`<([A-Z][A-Z0-9]*)\b[^>]*>(.*?)</\1>`

regexpal 0.1.4 — a JavaScript regular expression tester

☒ Case insensitive (i) ☒ ^\$ match at line breaks (m) ☐ Dot matches all (s; via [XRegExp](#))

`\b[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}\b`

my email is yuancx@bupt.edu.cn
please send the information to adjfl.as12_298ul-k1@bal2313.ko

- See more interesting tests at <http://regexpal.com/>

形符化：几个实例

- Finland' s capital
 - Finland? Finlands? Finland' s
- What' re, I' m, isn' t
 - What are, I am, is not
- Jurafsky-Martin
 - Jurafsky and Martin as two tokens?
- State-of-the-art: break up?
 - San Francisco, New York: one token or two?
- Words with punctuation
 - U.S.A., PhD.

形符化

- 形符化：将文本分解成词、短语、符号或其它有意义的形符（token）元素的过程
 - 形符化可以发生在不同程度的颗粒度：一个文本可以分解为段落、句子、词、音节（syllables）、音素（phonemes）
 - 形符序列作为其它进一步处理的输入（如句法分析、文本分类等）

形符化

- 形符化：将文本分解成词、短语、符号或其它有意义的形符（token）元素的过程
 - 形符化可以发生在不同程度的颗粒度：一个文本可以分解为段落、句子、词、音节（syllables）、音素（phonemes）
- 形符化可用于：
 - 信息检索（Information retrieval）
 - 信息抽取（Information extraction）
 - 拼写检查
 -
- 三个基本任务
 - 行文中的词切分及词的形符化
 - 词形的归一化
 - 行文中的句子切分

什么是词？

- **Cat** 和 **cats** 是同一个词吗？
- 几个术语：
 - Lemma(词元): 具有相同词干、主要词性及相似词义的词汇集合 (type, 词典大小)
 - Cat and cats = same lemma
 - run, ran, running: run
 - Wordform(词形): 词在形式上的曲折变化 (token, 词语规模)
 - Cat and cats = different wordforms
- 共有多少个词？
 - 现代汉语常用词表: 常用词56008个, 常用字3500个
 - 英文:

	tokens	types
Shakespeare	884000	31000
Google N-gram	1 trillion	13 million

分词

- 语言因素：
- 例如中文、日文等词汇中间没有空格
 - 莎拉波娃现在居住在美国东南部的佛罗里达
 - 莎拉波娃/ 现在/ 居住/ 在/ 美国/ 东南部/ 的/ 佛罗里达
 - シャラポワ/ は/ 現在/ アメリカ合衆国/ 南東部/ の/ フロリダ/ に/ 住ん/ で/ いる/
- 日语更加复杂，包含了平假名、片假名及阿拉伯字符
 - 20代女子が選んだ人気スキンケアランキング
- 两个挑战：
 - 新词：
 - 未登录词（Out-of-vocabulary）：水立方 肿么 给力 然并卵
 - 歧义：
 - 网球拍/ 卖完了 or 网球/ 拍卖/ 完了
 - 你要考虑你自己的/ 将来/ vs 市长/ 将/ 来/ 我们学校考察工作

分词

- 汉语中的切分歧义：
- 1) 交集型切分歧义：如果汉字串有ABC，AB和BC同时成词，称之为交集型切分歧义
 - 网球场 网球/场 网/球场
 - 结合成 结合/成 结/合成
 - 网球拍卖完了 网球拍/卖完了 网球/拍卖/完了
- 2) 组合型切分歧义：如果汉字串有AB，A、B、AB同时成词，称之为组合型切分歧义
 - 他站/起/身/来 他明天/起身/去学校
- 3) 混合型切分歧义：即是交集型歧义又是组合型歧义
 - 这样的人才能成大器

分词

- 一个简单的切分算法（通常作为基线算法）：
 - 最大匹配（也称为贪婪算法）
- 给定一个中文词表，和一个输入的字串
 - (1) Start a pointer at the beginning of the string
 - (2) Find the longest word in dictionary that matched the string starting at pointer
 - (3) Move the pointer over the word in string
 - (4) Go to 2
- 最大匹配切分的性能：90%+
- 性能更好的算法：
概率模型 (e.g., 隐马尔可夫模型)

归一化

- 为什么要进行归一化：
 - USA 和 U.S.A. 指同一个词条
 - anti-discriminatory 和 antidiscriminatory 指同一个词条
 - 对于IR来说，文本及查询语句的索引需要具有相同的词形
- 包括：词元、词干及大小写归一化
- 通常可以通过一系列规则进行归一化
 - 例如：删除词条中的点号
- 另一种方案：做不对称扩展
 - 输入：window 检索：window, Window, windows
 - 输入：Window 检索：window, Window, windows
 - 输入：windows 检索：window, Window, windows

归一化

- 词元化将词的曲折变化/变形转化为词的基本型
- 这一操作将文本中的词映射到词典中的词
- e.g.,
 - am, is, are → be
 - car, cars, car's, cars' → car
 - the boy's cars are different colors → the boy car be different color

归一化

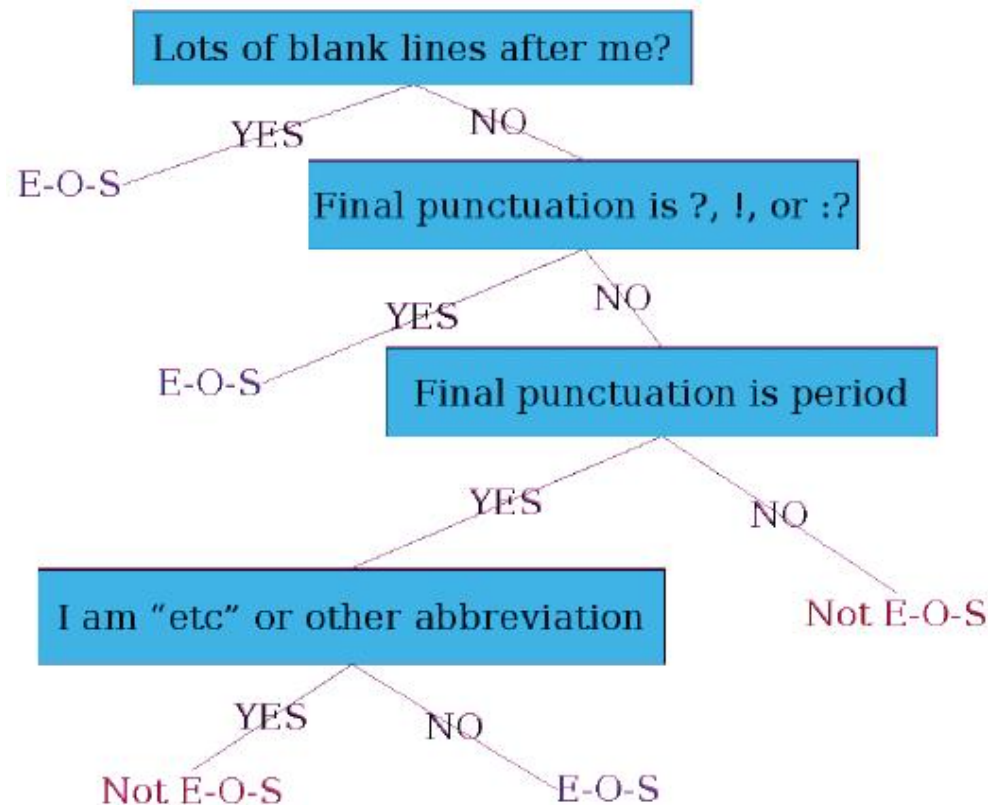
- 词干化为词的曲折变化/变形找到词根
- E.g.,
 - catlike, catty → cat
 - stemmer, stemming, stemmed → stem
 - fishing, fished, and fisher → fish
 - argue, argued, argues, arguing, and argus → argu
 - 但是: argument and arguments → argument
- 一些常用的规则:
 - sses → ss
 - ies → i
 - ational → ate
 - tional → tion

归一化

- 大小写归一化
- 通常需要将所有字母的大写都转化为小写
 - **exception**: 句子中的大写除外
 - Genaral Motors, Fed vs. fed
- 对于信息检索、信息抽取、机器翻译、情感分析等任务来说，词的大小写很重要
 - US vs us
 - Oracle vs oracle
 - simplistic , silly and **TEDIOUS!!**
- Stanford 形符化工具: [Stanford Tokenizer](#)

句子切分

- 判断一个词是否是句子的结尾：
 - !, ? 通常作为句子的边界
 - “.” 却具有歧义
- 决策树：



句子切分

- 更复杂的特征：
 - Prob(word with "." occurs at end-of-s)
 - Prob(word after "." occurs at begin-of-s)
 - Length of word with "."
 - Length of word after "."
 - Case of word with ".": Upper, Lower, Cap, Number
 - Case of word after ".": Upper, Lower, Cap, Number
 - Punctuation after "." (if any)
 - Abbreviation class of word with "." (month name, unit-of-measure, title, address, name, etc.)

Topics for Today

- 正则表达 (Regular Expression)
- 形符化 (Tokenization)
 - 词的形符化
 - 词的归一化 (Normalization)
 - 句子的形符化
- 最小编辑距离 (Minimum Edit Distance)
 - 字符串的比较
 - 最小编辑距离
 - 加权的最小编辑距离

Ref. J & M, 2009 Ch3.

几个在token层处理文本的例子

- 拼写检查

- Non-word检测:

- 词典之外的词，假设为拼写错误：例如 “chostick”
 - 纠正：应为 "chopstick"

- 错误检测及纠正:

- 词典中收录的词，但仍然被假设为拼写错误：例如 “war and piece”
 - 纠正：应为 “war and peace”

几个在token层处理文本的例子

- 评估机器翻译或语音识别的质量
 - 新闻发言人**确认**本拉登**遇难**。
 - 新闻发言人**证实**本拉登**已被害**。
- 实体抽取及指代 (Entity Extraction and Coreference)
 - **IBM** Inc. announced today
 - **IBM'** s profits
 - 西安交通大学王树国昨天宣布
 - 西安交通大学**王校长**出席了本次会议

几个在token层处理文本的例子

- 搜索引擎中的查询优化：
 - How do I fix "chostick"?
 - Search through all words:
 - chosen
 - christ
 - chopstick
 - Pick the one that's **closest** to "chostick"
- **"closest"** 指什么意思？
 - 两个字符串的相似程度？
- 需要一个距离测度 (**distance metric**)
- **一个最简单的距离测度：编辑距离**
 - (其它更复杂的概率模型：noisy channel, etc.)

最小编辑距离

- 最小编辑距离 (minimum edit distance) 指两个字符串之间进行如下编辑操作的最小次数：
 - Insertion(插入)
 - deletion (删除)
 - substitution(替换)
- 一个字符串通过以上编辑操作转化到另一个字符串：

e.g., I N T E * N T I O N
 * E X E C U T I O N

最小编辑距离

INTENTION

删除I

NTENTION

将N替换为E

ETENTION

将T替换为X

EXENTION

插入U

EXENUATION

将N替换为C

EXECUTION

- 若每个操作的代价为1
 - 编辑距离为5
- 若替换操作的代价为2
 - 编辑距离为8

最小编辑距离

INTENTION

将I替换为E

ENTENTION

将N替换为X

EXTENTION

删除T

EXENTION

删除N

EXETION

插入C

EXECTION

插入C

EXECUTION

- 若每个操作的代价为1
 - 距离为6
- 若替换操作的代价为2
 - 距离为8

最小编辑距离

- 对于两个字符串 S_1 和 S_2 ，设其长度分别为 n 和 m
 - Distance(i, j) or $D(i, j)$ 表示 $S_1[1..i]$ 和 $S_2[1..j]$ 的最小编辑距离
 - i.e., 即将 S_1 中的前 i 个字符转化为 S_2 中的前 j 个字符所需的最小编辑次数
 - S_1, S_2 的最小编辑距离则为: $D(n, m)$
 - $D(n, m)$ 可以分解为 $D(i, j)$ for all i ($0 < i < n$) and j ($0 < j < m$)
- 如何找到最小编辑距离？

动态规划 (Dynamic Programming)

- Base conditions:

- $D(i,0) = i$
- $D(j,0) = j$

- Recurrence Relation

For each $i=0, \dots, n$,
for each $j=0, \dots, m$

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

- Termination: $D(n, m)$ is distance
- Bottom-up
 - We compute $D(i, j)$ for small i, j
 - And compute increase $D(i, j)$ based on previously computed smaller values

编辑距离表

N	9									
O	8									
I	7									
T	6									
N	5	$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$								
E	4									
T	3									
N	2									
I	1									
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

N	9	8	9	10	11	12	11	10	9	8
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

回退 (backtrace)

- 保留一个回退指针
- 每计算完一个格子，记录其最小编辑路径
- 所有格子计算完后，从右上角开始回溯，得到两个字符串的最小编辑路径

回退 (backtrace)

- Backtrace

N	9	8	9	10	11	12	11	10	9	8
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

回退 (backtrace)

- Base conditions:
 - $D(i,0) = i$
 - $D(j,0) = j$
- Recurrence Relation

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases} \quad \text{Case 3}$$

$$\text{Ptr}(i, j) = \begin{cases} \text{left} & \text{Case 1} \\ \text{own} & \text{Case 2} \\ \text{diag} & \text{Case 3} \end{cases}$$

回退 (backtrace)

n	9	↓ 8	↙←↓ 9	↙←↓ 10	↙←↓ 11	↙←↓ 12	↓ 11	↓ 10	↓ 9	↙ 8	
o	8	↓ 7	↙←↓ 8	↙←↓ 9	↙←↓ 10	↙←↓ 11	↓ 10	↓ 9	↙ 8	← 9	
i	7	↓ 6	↙←↓ 7	↙←↓ 8	↙←↓ 9	↙←↓ 10	↓ 9	↙ 8	← 9	← 10	
t	6	↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↙←↓ 9	↙ 8	← 9	← 10	←↓ 11	
n	5	↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↙←↓ 9	↙←↓ 10	↙←↓ 11	↙↓ 10	
e	4	↙ 3	← 4	↙← 5	← 6	← 7	←↓ 8	↙←↓ 9	↙←↓ 10	↓ 9	
t	3	↙←↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↙ 7	←↓ 8	↙←↓ 9	↓ 8	
n	2	↙←↓ 3	↙←↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↓ 7	↙←↓ 8	↙ 7	
i	1	↙←↓ 2	↙←↓ 3	↙←↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙ 6	← 7	← 8	
#	0	1	2	3	4	5	6	7	8	9	
	#	e	x	e	c	u	t	i	o	n	

Topics for Today

- 正则表达 (Regular Expression)
- 形符化 (Tokenization)
 - 词的形符化
 - 词的归一化 (Normalization)
 - 词元和词干 (Lemmatization and stemming)
 - 大小写 (Case folding)
 - 句子的形符化
- 最小编辑距离 (Minimum Edit Distance)
 - 字符串的比较
 - 最小编辑距离
 - 加权的最小编辑距离

Ref. J & M, 2009 Ch3.