2018학년도 2학기 언어와 컴퓨터

제11강 텍스트 처리

박수지

서울대학교 인문대학 언어학과

2018년 10월 29일 월요일

오늘의 목표

1. 정규표현식 패턴을 읽고 쓸 수 있다.
2. 1종 오류와 2종 오류의 차이를 설명할 수 있다.
3. 코퍼스가 무엇이고 어떻게 사용하는지 설명할 수 있다.
4. 문장을 보고 타입과 토큰의 수를 계산할 수 있다.
5. 자연언어처리에 필요한 텍스트 처리의 과정과 난점을 설명할 수 있다.

참조

에디. (집필 중). 딥 러닝을 이용한 자연어 처리 입문. 위키독스. https://wikidocs.net/book/2155

후반부 개괄

자연언어처리

책

주교재 Jurafsky and Martin. (In progress). Speech and Language Processing (3rd ed. draft)

- 책·PPT `https://web.stanford.edu/~jurafsky/slp3`

보조 교재 조엘 그루스 지음. 박은정 · 김한결 · 하성주 옮김. (2016). 《밑바닥부터 시작하는 데이터 과학: 데이터 분석을 위한 파이썬 프로그래밍과 수학 · 통계 기초》. 인사이트.

후반부 개괄

자연언어처리

내용

1. 데이터 전처리
2. 수학·통계 기초
3. 기계학습
    1. 나이브 베이즈 분류
    2. 로지스틱 회귀분석 (최대 엔트로피 모델)

응용 분야

- 언어 모델링
    - 문장 생성
- 감정 분석
    - 긍정-부정 분류

정규표현식(Regular Expression)

특정한 규칙을 가지는 문자열의 집합을 표현하는 형식언어

- 특정한 규칙을 가지는 문자열 $\Rightarrow$ **패턴**에 **매치**되는 문자열

예시: 학번을 매치시키는 패턴

XXXX-XXXXX

- `20[0-9][0-9]-[1-3][0-9][0-9][0-9][0-9]`
- `20\d{2}-[1-3]\d{4}`

참조

패턴과 텍스트가 주어졌을 때 매치되는 문자열을 표시해 주는 `https://regexr.com/`

패턴 구성

단순 문자 문자 그대로 매치시킨다.

메타 문자 집합의 규칙을 표현한다.

- 기본 메타 문자 매치시킬 문자들의 합집합
- 반복 메타 문자 패턴 매치 반복
- 특수 메타 문자 특정 범주의 문자들

오류의 두 가지 유형

1. 찾지 말아야 할 것을 찾았다. ⇒ 규칙을 강화하자!
2. 찾아야 할 것을 찾지 못했다. ⇒ 규칙을 완화하자!

더불어 말할 만한 사람과 말하지 않는 것은 사람을 잃는 것이다. [2종 오류]
더불어 말하지 않을 만한 사람과 말하는 것은 말을 잃는 것이다. [1종 오류]

《논어》〈위령공〉

어디에서 찾을 것인가?

코퍼스(Corpus, 말뭉치)

컴퓨터가 읽을 수 있는 언어 자료 모음

- 문자 언어: 텍스트, 이미지+전사
- 음성 언어: 오디오+전사
- 시각 언어: 비디오+전사

예시

"아무도 —기 전에"가 자연스러운 표현인지를 어떻게 검증하는가?

이론언어학자 직관 테스트를 해 보자.

코퍼스언어학자 코퍼스에서 얼마나 나오는지 세어 보자.

이론언어학에서도 사용하는 코퍼스

To see how this works, consider the forms from (7) above. A query for these forms using the Google search engine (12 May 2004) yielded the hit counts shown in (8).

(8)

		hits	
[mutɔgeːn-nɔk]	*mutagénnak*	0	0%
[mutɔgeːn-nɛk]	*mutagénnek*	32	100%
[ɔrzeːn-nɔk]	*arzénnak*	32	73%
[ɔrzeːn-nɛk]	*arzénnek*	12	27%
[pɔlleːr-nɔk]	*pallérnak*	15	100%
[pɔlleːr-nɛk]	*pallérnek*	0	0%

Hayes, B., & Londe, Z. (2006). Stochastic phonological knowledge: The case of Hungarian vowel harmony. Phonology, 23(1), 59-104. doi:10.1017/S0952675706000765

(어쨌든 찾고 세면 된다.)

텍스트 정제(Text normalization)

- 단어 토큰화(Word tokenization)
  예시 `get_word_list()`, `remove_punct()` ([숙제04], 7강)
- 어형 정제: 표제어 추출(Lemmatization), 어간 추출(Stemming)
- 문장 토큰화(Sentence tokenization)

주의

- 컴퓨터는 사람이 생각하는 단어의 정의를 모른다.
  단어 분리하여 자립적으로 쓸 수 있는 말
  어절 띄어쓰기의 단위
- 사람이 생각하는 단어의 정의는 목적에 따라 다르다.

더 구체적인 문제

- 단어의 경계는 무엇인가? — It's, San Francisco, …
- 어떤 토큰이 같은 타입인가? — The vs. the, am vs. are, …

원인

표기법 대소문자, 띄어쓰기, ...

언어 합성어, 축약형, 두문자어, …

Joy to the word, though linguists can not tell you what one is.
It's somehow phonological, syntactic and morphological,
But one thing is for sure,
Yes one thing is for sure:
We're pretty sure it's listed in the lexicon.

http://specgram.com/CLXVI.1/03.linguistick.hymnary.html

단어 분리 (Word segmentation)

단어의 경계를 결정하는 작업

어디에 필요한가?

- 띄어쓰기가 없는 언어: 중국어, 일본어 등
- 음성언어처리: 연속적인 음성 자료

기법

- 규칙 기반: MaxMatch 등
- 통계적: 단어가 분리된 대량의 데이터에서 패턴을 학습하기

실습 `maxmatch.py`

```
function MaxMatch(sentence, dictionary) returns word sequence W

    if sentence is empty
        return empty list
    for i ← length(sentence) downto 1
        firstword = first i chars of sentence
        remainder = rest of sentence
        if InDictionary(firstword, dictionary)
            return list(firstword, MaxMatch(remainder,dictionary) )

    # no word was found, so make a one-character word
    firstword = first char of sentence
    remainder = rest of sentence
    return list(firstword, MaxMatch(remainder,dictionary) )
```

Figure 2.11 The MaxMatch algorithm for word segmentation.

단어 정제

표제어 추출 am → be, the going → the going, having → have

어간 추출 am → be, the going → the go, having → hav

형태론에 대한 지식이 있어야 한다.

형태소 분리

형태론적으로 복잡한 언어 (굴절어 < 교착어 < 포합어) 에 특히 필요

고립어 even to the male cats / even to the female cats

굴절어 até aos gatos / até às gatas

교착어 수코양이들에게조차 / 암고양이들에게조차

문장 분리

문장의 경계를 찾는 작업

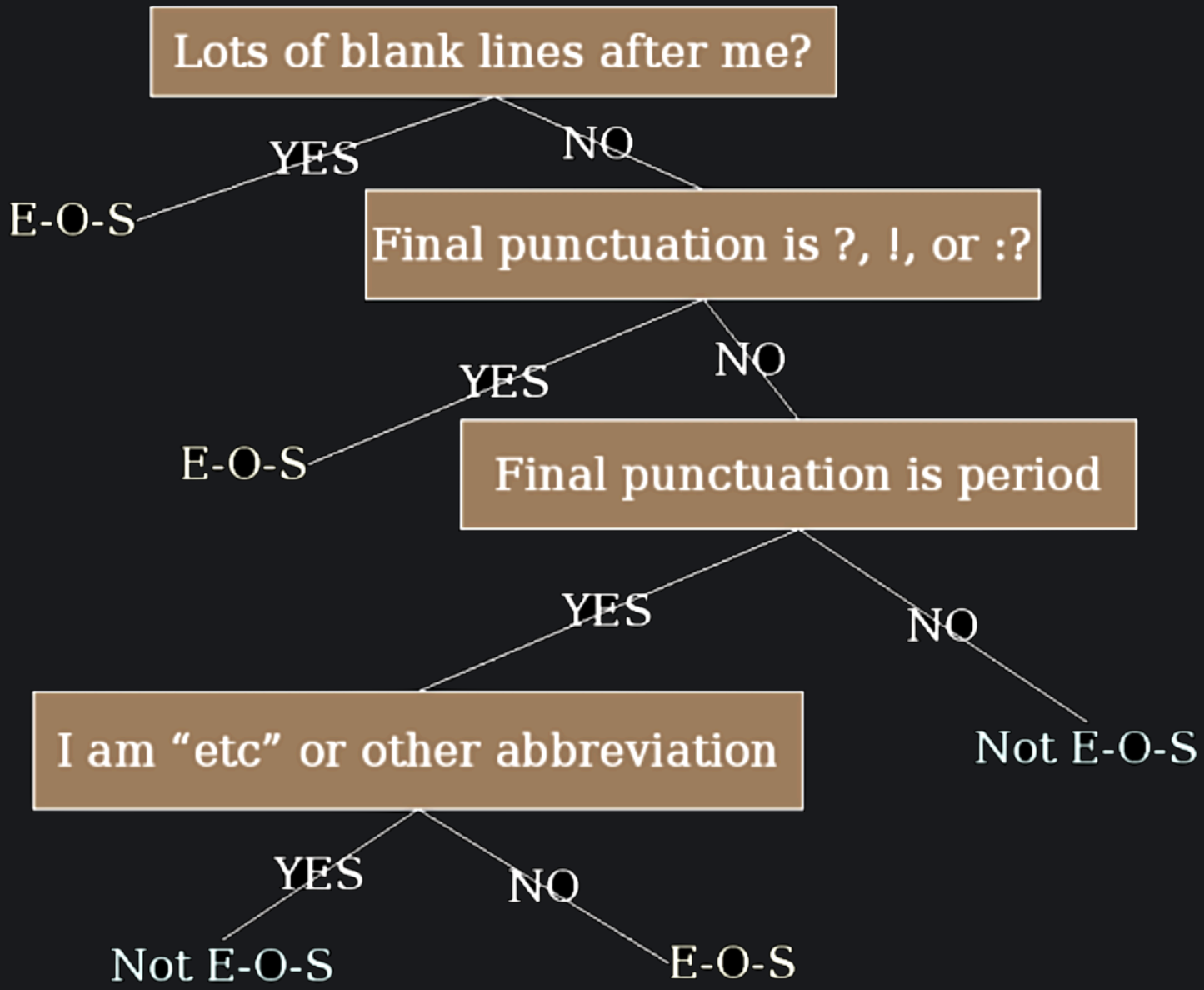
문제

문장 부호(., ?, !)가 항상 문장의 경계인가?

- (?), (!), 3!
- Mr., U.S.A., 3.14159, 255.255.0.0, ...

⇒ 문장 부호가 어떤 조건을 만족할 때 문장의 경계가 되는가?

실습 `decision_tree.py`


Lots of blank lines after me?
YES
NO
E-O-S
Final punctuation is ?, !, or :?
YES
NO
E-O-S
Final punctuation is period
YES
NO
I am “etc” or other abbreviation
Not E-O-S
YES
NO
Not E-O-S
E-O-S

최소 편집 거리(Minimun Edit Distance)

두 문자열이 서로 얼마나 비슷한지를 나타내는 척도

- 문자 하나를 편집할 때마다 “멀어지는” 방식

활용

- 철자 교정
- 단어 정제
  원인 고유명사, 외래어 표기 등
  응용 기계 번역, 정보 검색, 음성 인식 등
- …

연구 사례: 최소 편집 거리로 음성적 유사도 측정하기

이상아 (2017) 단어 임베딩과 음성적 유사도를 이용한 트위터 '서치 방지 단어' 의 자동 예측

먼저 최소 편집 거리는 두 문자열이 서로 얼마나 비슷한지를 나타내는 척도 중 하나로, 한 단어의 철자가 다른 단어와 같아지도록 수정하는 과정(철자의 삽입, 삭제, 대체) 각각에 비용을 부여하는 방식이다. 본 연구에서는 먼저 각각의 단어를 자소 단위로 분해하고, 최소 편집 거리 알고리즘을 적용하되 자소들의 특성에 따라 자소의 대체 비용에 각각 다른 값을 부여하는 규칙을 정의하였다. 기본 대체 비용을 1로 설정하고, 아래 [표2]의 기준들에 해당하는 자음, 모음의 그룹 내에서 발생하는 대체에는 0과 1 사이의 값을 정의하여 이용하였다.

[표2] 편집 대체 비용의 판단 기준

기준	자소 그룹	대체 비용
자음의 조음 위치	{ㄱ, ㄲ, ㅋ}, {ㄷ, ㄸ, ㅌ}, {ㅂ, ㅃ, ㅍ}, {ㅅ, ㅆ}, {ㅈ, ㅉ, ㅊ}	0.5
모음 발음상의 유사성	{ㅏ, ㅐ}, {ㅗ, ㅜ}, {ㅓ, ㅗ} 등	0.5
	{ㅔ, ㅐ}, {ㅖ, ㅒ} 등	0.2

이상아. (2017). 〈단어 임베딩과 음성적 유사도를 이용한 트위터 '서치 방지 단어' 의 자동 예측〉. 《한글 및 한국어 정보처리 학술대회 논문집》 (pp. 190–193).

‘스타벅스’와 가장 가까운 단어 결정하기

후보: 스따벆스, 스타렉스, 스티븐스, 스틸녹스

1. 최소 편집 거리 계산: ‘스타렉스’ 1승

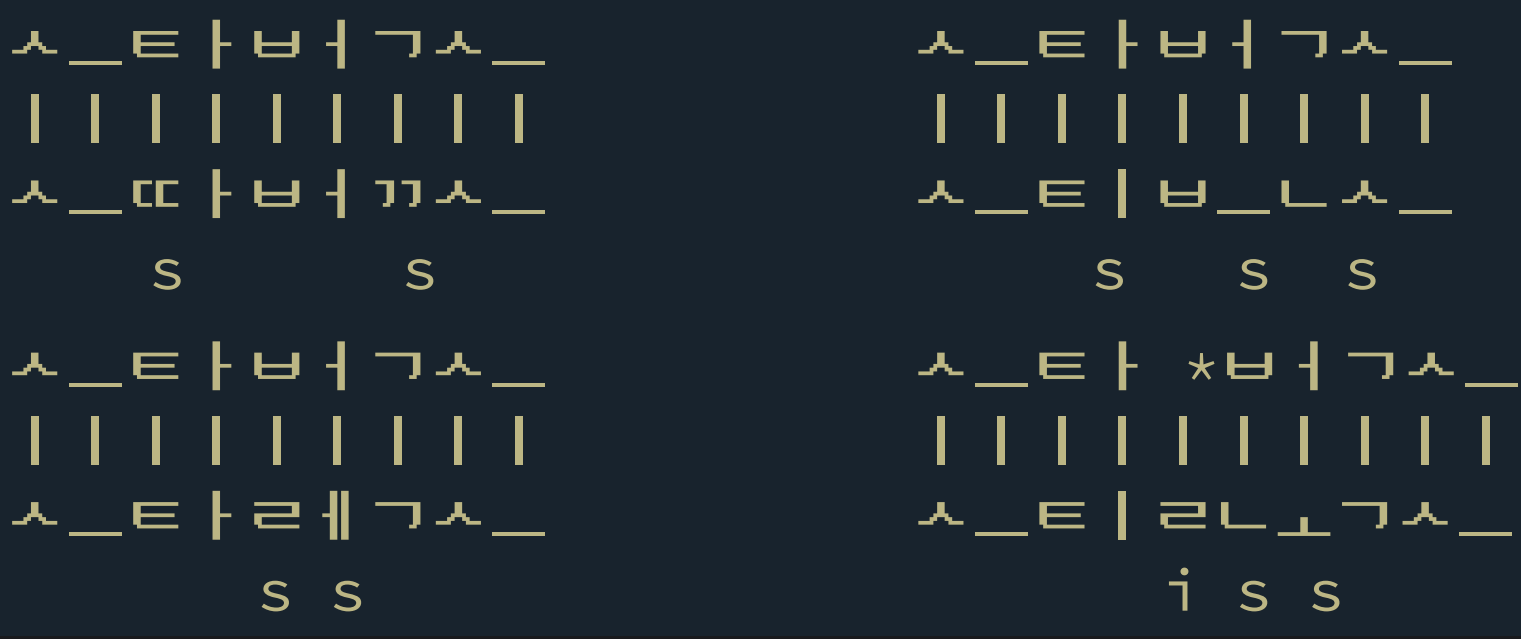
스타벅스	스타벅스	스타벅스	스타벅스
| | | |	| | | |	| | | |	| | | |
스따벆스	스타렉스	스티븐스	스틸녹스
s s	s	s s	s s

s substitution

‘스타벅스’와 가장 가까운 단어 결정하기

후보: 스따벆스, 스타렉스, 스티븐스, 스틸녹스

2. 자모 분해 후 최소 편집 거리 계산: ‘스따벆스’, ‘스타렉스’ 1승



i insertion

s substitution

‘스타벅스’와 가장 가까운 단어 결정하기

후보: 스따벆스, 스타렉스, 스티븐스, 스틸녹스

3. 자모 분해 후 음성 특징을 반영한 최소 편집 거리 계산: ‘스따벆스’ 1승

ㅅ_ㅌㅏㅂㅓㄱㅅ_
ㅣ ㅣ ㅣ ㅣ ㅣ ㅣ ㅣ ㅣ ㅣ
ㅅ_ㄸㅏㅂㅓㄲㅅ_
s s

$$D(\text{ㅌ}, \text{ㄸ}) + D(\text{ㄱ}, \text{ㄲ}) = 0.5 + 0.5 = 1$$

ㅅ_ㅌㅏㅂㅓㄱㅅ_
ㅣ ㅣ ㅣ ㅣ ㅣ ㅣ ㅣ ㅣ ㅣ
ㅅ_ㅌㅏㄹㅐㄱㅅ_
s s

$$D(\text{ㅂ}, \text{ㄹ}) + D(\text{ㅓ}, \text{ㅐ}) = 1 + 1 = 2$$

s substitution

다음 시간에 할 일

정규표현식과 단어·문장 분리를 파이썬에서 구현해 보기

- `re` 모듈 사용법 익히기
- ELIZA(교재 2.1.6) 만들기
- NLTK 설치하기
- (시간이 남으면 konlpy 설치하기)