soyspacing

Heuristic Algorithm for Korean Spacing Correction

Hyunjoong Kim

soy.lovit@gmail.com

https://github.com/lovit/soyspacing/

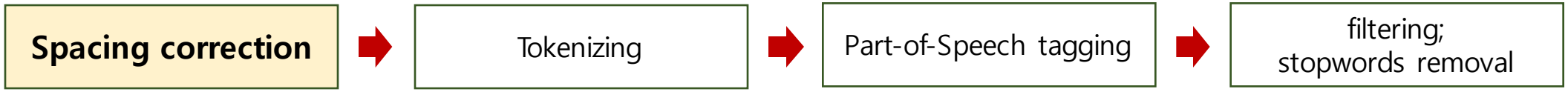
한국어 띄어쓰기 교정

- 한국어는 띄어쓰기가 잘 되지 않아도 가독성에 큰 영향이 없습니다.
  - 특히 온라인에서 작성되는 텍스트에는 띄어쓰기 오류가 자주 존재합니다.

실제 문장: 폰에 넣어다니면서 대사까지 다외움 아진짜

정답 문장: 폰에 넣어 다니면서 대사까지 다 외움 아 진짜

한국어 띄어쓰기 교정



- 띄어쓰기의 경우에는 tokenization / tagging에 영향을 줄 수 있습니다.
  - 띄어쓰기 오류를 교정하면 이후의 자연어처리 성능이 향상될 수 있습니다.

한국어 띄어쓰기 교정

- 한국어의 띄어쓰기 교정 문제는 “주어진 문장에서 띄어지지 않는 글자들 중, 본래 띄어써야 하는 부분을 교정하는 것“으로 한정합니다.

  - 사람은 붙여써야 하는 부분을 실수로 띄우는 경우가 드뭅니다.
    (사람도 해석하지 못합니다)

(예시) 아 예완 전히 다 무 시하고 임 의로 띄어 쓰 면해 독이 정 말어 려 움

한국어 띄어쓰기 교정

- 한국어 띄어쓰기 교정은 연속된 글자들에서 각 글자 다음에 공백이 올지 판단하는 binary classification 문제입니다.

- 띄어쓰기가 잘 되어있는 데이터로부터 띄어쓰기 패턴을 학습하여 오류를 수정할 수 있습니다.

한국어 띄어쓰기 교정 데이터 선택의 어려움

- 하지만 띄어쓰기가 잘 되어 있는 외부 데이터를 구하기는 어렵습니다.
  - 띄어쓰기 교정용 데이터의 핵심은, vocabulary distribution 입니다.
  - 띄어쓰기 교정은 '학습데이터에 존재한 단어열 사이를 인식하여 띄는 것'
  - 학습데이터에 존재하지 않은 어절이 등장하면 잘못된 교정을 할 수 있습니다.

- 뉴스 데이터 / 위키피디아 데이터를 이용하여 모델을 학습하여도
  영화 댓글 / 대화 텍스트의 띄어쓰기를 교정할 수는 없습니다.

한국어 띄어쓰기 교정 데이터 선택의 어려움

- 교정해야 하는 데이터의 vocabulary distribution 을 가장 잘 반영하는 데이터는, 교정해야 하는 데이터 그 자체입니다.

- 교정할 데이터에서 띄어쓰기가 잘 되어 있는 일부 데이터를 학습데이터로 이용한 뒤, 교정할 데이터에 다시 적용하여 띄어쓰기 오류를 교정합니다.

오류에 대한 관점

- 오류는 동일한 input에 대하여 소수의 output pattern 입니다.
  - “어서가요” (20%) / “어서 가요” (80%) 로 등장한다면

    “어서가요” → “어서 가요“ 로 교정되어야 합니다.

- 즉, 오류가 적다고 판단되는 데이터를 선택한 뒤, 전체 데이터에서 조금 등장한 패턴을 자주 등장한 패턴으로 수정

Related Works

Related Works

- 한국어 띄어쓰기 오류 교정은 두 가지 방법으로 분류할 수 있습니다.
  - Rule based 방식은 형태소분석, 규칙사전을 기반으로 교정을 합니다.
    - 규칙 기반은 새로운 단어에 의한 예외가 발생하기 쉽습니다.
    - 형태소분석 과정 자체가 비싼 계산량이 필요합니다.
  - Statistical model based 방식은 머신러닝 알고리즘을 이용합니다.
    - 데이터가 있다면 다양한 도메인에 확장성이 좋습니다.
    - HMM[1,2], CRF[3], structural SVM[4,5,6], LSTM[7] 등이 이용되었습니다.
    - Heuristic algorithms[8,9,10,11]도 이용되었습니다.

Related Works

- 중국어와 일본어에서도 word segmentation이 연구되었습니다.
  - 중국어/일본어는 본래 띄어쓰기를 하지 않기 때문에 연속된 글자열에서 단어를 나누는 과정 자체가 자연어처리의 가장 중요한 부분입니다.
  - CRF 기반의 알고리즘들이 HMM, MEMM보다 좋은 성능을 보였습니다[12,13].

Maximum Entropy Markov Model

- Machine learning 기반 알고리즘이나 heuristic algorithms이 띄어쓰기 교정을 위하여 이용하는 features는 앞/뒤에 등장하는 글자열입니다.

- CRF / MEMM을 이용한 띄어쓰기 교정기는 character feature를 이용하는 sequential labeling 알고리즘입니다.

Sequential labeling using Logistic Regression

- 주어진 sequence $x_{1:n} = [x_1, x_2, \dots, x_n]$에 대하여, 동일한 길이의 $y_{1:n} = [y_1, y_2, \dots, y_n]$를 output으로 출력하는 문제입니다.

- 간단한 방법으로 $P(y_{1:n}) = \prod_{i=1\ to\ n} P(y_i|x_{1:n})$ 처럼 n 개의 독립적인 판별 문제로 생각할 수도 있습니다.

Sequential labeling using Logistic Regression

- 이를 확률모형으로 표현하면 $argmax_y P(y_{1:n}|x_{1:n})$
  - $x_{1:n}$가 주어졌을 때, 확률값이 가장 큰 $y_{1:n}$를 찾는 것

- 이전의 label y_{i-1}을 고려하며 순차적으로 labeling할 수도 있습니다.
  - $P(y_{1:n}|x_{1:n}) \coloneqq P(y_1|x_{1:n}) \times (\prod_{i=2\ to\ n} P(y_i|x_{1:n}, y_{i-1}))$

Sequential labeling using Logistic Regression

- 이는 다음의 순차적인 classification의 연속으로 생각할 수 있습니다.
  - $y_i = f(x_{i-1}, x_i, y_{i-1})$의 classifier로 logistic regression를 이용할 수 있습니다.

$$y_1 = f(x_1)$$

$$y_2 = f(x_1, x_2, y_1)$$

$$\dots$$

$$y_n = f(x_{n-1}, x_n, y_{n-1})$$

Potential function as Representation

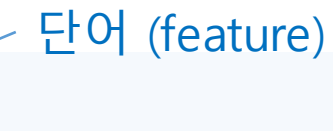
- Maximum Entropy Markov Model (MEMM)은 Logistic Regression을 이용한 sequential labeling을 위하여 feature representation을 변형한 것

  - [이것, 은, 예문, 입니다]와 같이 문장이 들어올 경우 단어열을 벡터로 표현해야 합니다.

(x_{i-1}, x_i, y_{i-1})의 벡터 표현

- 띄어쓰기를 한다면, 한 문장을 아래처럼 sparse vector로 표현할 수 있음

  - "예문 입니다" 를 다음의 template을 이용
    - X[-1:0] : 앞글자와 현재글자
    - X[-1:0] & y[-1] : 앞글자와 현재글자, 앞글자의 띄어쓰기 정보
    - Y[-1] : 앞글자의 띄어쓰기 정보



- [[('x[0]=예', 1)],
  [('x[-1:0]=예문', 1), ('x[-1:0]=예문 & y[-1]=0', 1), ('y[-1]=0', 1)],
  [('x[-1:0]=문입', 1), ('x[-1:0]=문입 & y[-1]=1', 1), ('y[-1]=1', 1)],
  [('x[-1:0]=입니', 1), ('x[-1:0]=입니 & y[-1]=0', 1), ('y[-1]=0', 1)],
  [('x[-1:0]=니다', 1), ('x[-1:0]=니다 & y[-1]=0', 1), ('y[-1]=0', 1)]]

(x_{i-1}, x_i, y_{i-1})의 벡터 표현

- 한 문장을 sparse vector 형식으로 표현할 수 있습니다.

```
[ [ ('x[0]=예', 1) ],
  [ ('x[-1:0]=예문', 1), ('x[-1:0]=예문 & y[-1]=0', 1), ('y[-1]=0', 1) ],
  [ ('x[-1:0]=문입', 1), ('x[-1:0]=문입 & y[-1]=1', 1), ('y[-1]=1', 1) ],
  [ ('x[-1:0]=입니', 1), ('x[-1:0]=입니 & y[-1]=0', 1), ('y[-1]=0', 1) ],
  [ ('x[-1:0]=니다', 1), ('x[-1:0]=니다 & y[-1]=0', 1), ('y[-1]=0', 1) ] ]
```

char	Y	x[-1:0]=예문	x[-1:0]=예문 & y[-1]=0	'x[-1:0]=문입	x[-1:0]=문입 & y[-1]=1	..	y[-1]=0	y[-1]=1
예	0	0	0	0	0	..	0	0
문	1	1	1	0	0		1	0
입	0	0	0	1	1		0	1
니	0	0	0	0	0		1	0
다	1	0	0	0	0		1	0

Potential function as Representation

- Potential function, 혹은 filter F를 통하여 (x_{i-1}, x_i, y_{i-1})를 매우 큰 차원의 Boolean vector로 표현합니다.

- y_i를 판별하기 위한 F_i 의 각 차원 F_{ij}는 다음처럼 구성됩니다.

$F_{i0} = \mathbf{1}$ if (x_{i-1} = '이것', x_i = '은', y_{i-1} = '명사') else **0**

$F_{i1} = \mathbf{1}$ if (x_{i-1} = '은', x_i = '예문', y_{i-1} = '조사') else **0**

…

Maximum Entropy Markov Model (MEMM)

$$(x_i, x_{i-1}, y_{i-1}) = [\, F_1(x_i, x_{i-1}, y_{i-1}), \ldots, F_N(x_i, x_{i-1}, y_{i-1})]$$

e.g.) $h_i = F(x_i, x_{i-1}, y_{i-1})$ = [1, 0, 0, …, 1, 0, …]

$$(x_2, x_1, y_1) \rightarrow h_2$$

$$(x_3, x_2, y_2) \rightarrow h_3$$

…

$$(x_n, x_{n-1}, y_{h-1}) \rightarrow h_n$$

$$h_\theta(x_i, x_{i-1}, y_{i-1}) = \begin{bmatrix} P(y = 1|h_i; \boldsymbol{\lambda}) \\ \ldots \\ P(y = K|h_i; \boldsymbol{\lambda}) \end{bmatrix} = \frac{1}{\sum_{j=1}^{K} \exp(\boldsymbol{\lambda}^{(j)T} h_i)} \begin{bmatrix} \boldsymbol{\lambda}^{(1)T} h_i \\ \ldots \\ \boldsymbol{\lambda}^{(K)T} h_i \end{bmatrix}$$

Maximum Entropy Markov Model (MEMM)

- Maximum Entropy Markov Model (MEMM)은 Logistic Regression을 이용한 sequential labeling을 위하여 feature representation을 변형한 것

$$P(y|x) = \prod_{i=1}^{n} \frac{\exp\left(\sum_{j=1}^{m} \lambda_j f_j(x, i, y_i, y_{i-1})\right)}{\sum_{y_i'} \exp\left(\sum_{i=1}^{n} \lambda_j f_j(x, i, y_i, y_{i-1})\right)}$$

per each word

Logistic Regression

* McCallum, A., Freitag, D., & Pereira, F. C. (2000, June). Maximum ntropy Markov Models for Information Extraction and Segmentation. In Icml (Vol. 17, pp. 591-598).

Label bias problem

- MEMM는 바로 앞의 label, y_{i-1}의 정보만을 살펴본 뒤, y_i를 결정합니다.
  - 이 때, (y_i, y_{i-1})의 출현 횟수가 큰 label 로 y_i가 편향
  - 즉, local 하게만 의사결정을 하다보니 y_{i-1}가 infrequent 할 때는 $\mathrm{P}(y_i, |y_{i-1})$ 가 커져 확률이 왜곡됩니다 (좀 더 자세한 설명은 아래의 material 참고)
  - CRF는 이 문제를 해결하기 위하여 MEMM의 구조를 바꿉니다.

$$P(y|x) = \prod_{i=1}^{n} \frac{\exp\left(\sum_{j=1}^{m} \lambda_j f_j(x, i, y_i, y_{i-1})\right)}{\sum_{y_i`} \exp\left(\sum_{i=1}^{n} \lambda_j f_j(x, i, y_i, y_{i-1})\right)}$$

http://web.donga.ac.kr/yjko/usefulthings/MEMM&CRF.pdf

MEMM to CRF

- CRF는 i=1 to n으로 나눠서 Logistic Regression을 하지 않는 대신에, 전체 $y_{[1:n]}$의 경향까지 고려해서 한 번의 Logistic Regression을 수행

  - (MEMM) $P(y|x) = \prod_{i=1}^{n} \frac{\exp\left(\sum_{j=1}^{m} \lambda_j f_j(x,i,y_i,y_{i-1})\right)}{\sum_{y_i`} \exp\left(\sum_{i=1}^{n} \lambda_j f_j(x,i,y_i,y_{i-1})\right)}$

  - ➔ (CRF) $P(y|x) = \frac{\exp\left(\sum_{j=1}^{m} \sum_{i=1}^{n} \lambda_j f_j(x,i,y_i,y_{i-1})\right)}{\sum_{y`} \exp\left(\sum_{j=1}^{m} \sum_{i=1}^{n} \lambda_j f_j(x,i,y_i,y_{i-1})\right)}$

Decision of feature design

Character sub-sequence features

- Machine learning 기반 알고리즘이나 heuristic algorithms이 띄어쓰기 교정을 위하여 이용하는 features는 앞/뒤에 등장하는 글자열입니다.

- 적절한 features를 설계할 수 있도록 character features 의 길이 및 위치에 따른 정보량을 확인하여 봅시다.

Character sub-sequence features

- Features를 세 종류로 나누겠습니다.
  - L (left-side) : x_i 의 앞에 등장한 글자들, $x_{i-p:i}$
  - C (central) : x_i 의 앞/뒤에 등장한 글자들, $x_{i-p, i+q}$
  - R (right-side): x_i의 뒤에 등장한 글자들, $x_{i:i+q}$

X = '테스트문장입니다'

$L_{i-2:i}(i=2)$ = '테스트' // $C_{i-2:i+2}(i=2)$ = '테스트문장' // $R_{i:i+2}(i=2)$ = '트문장'

Purity of character features

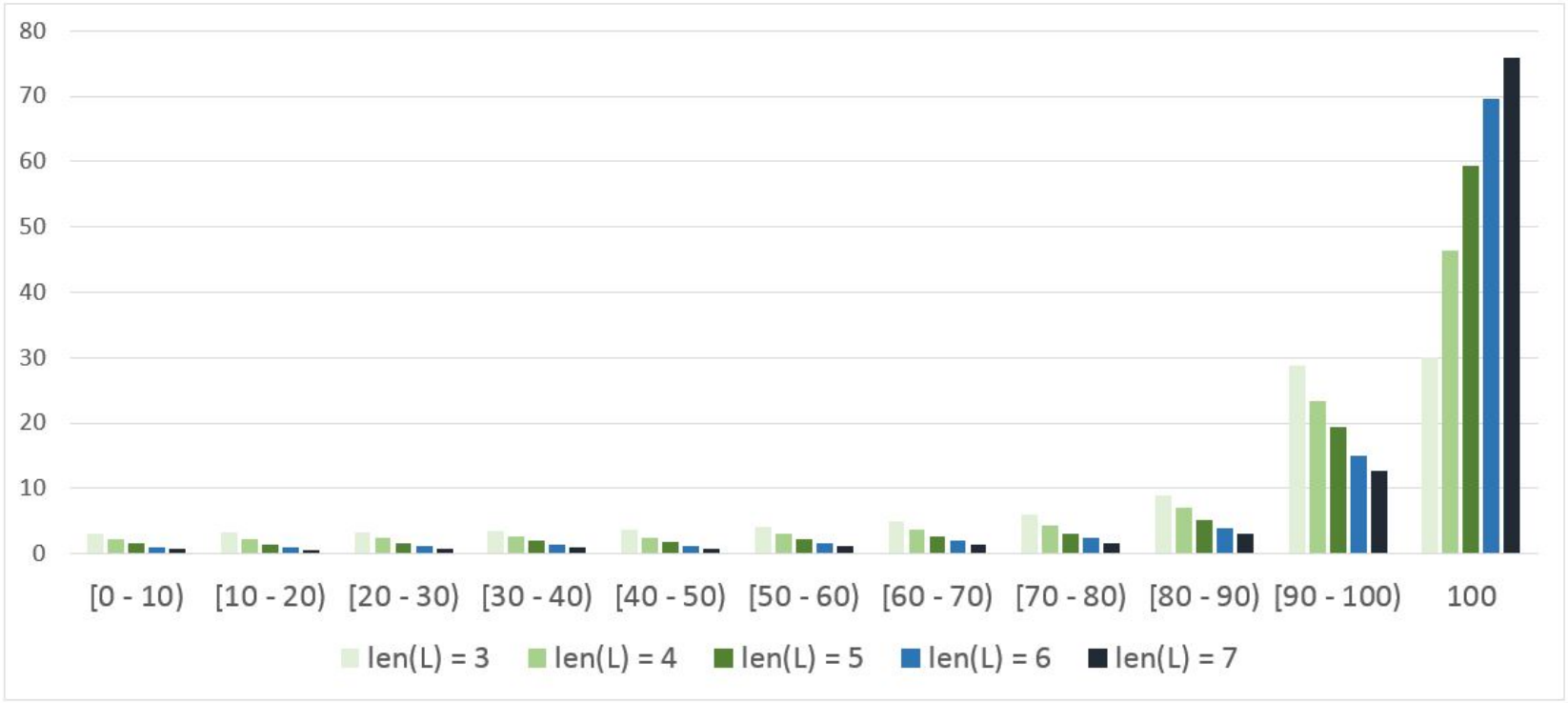
- L, C, R features의 길이별로 purity를 측정합니다.
  - Purity 는 $L_{i-k,i}(i)$에 대하여, i 글자의 띄어쓰기 태그의 일관성입니다.
  - Purity of $L_{i-2:i}(i=2)$ 는 '테스트'라는 글자 다음에 {띄어쓰기: 90, 붙여쓰기: 10} 이 등장하였다면, | 90 – 10 | / (90 + 10) = 0.8 로 정의합니다.

X = '테스트문장입니다'

$L_{i-2:i}(i=2)$ = '테스트' // $C_{i-2:i+2}(i=2)$ = '테스트문장' // $R_{i:i+2}(i=2)$ = '트문장'

Purity of character features

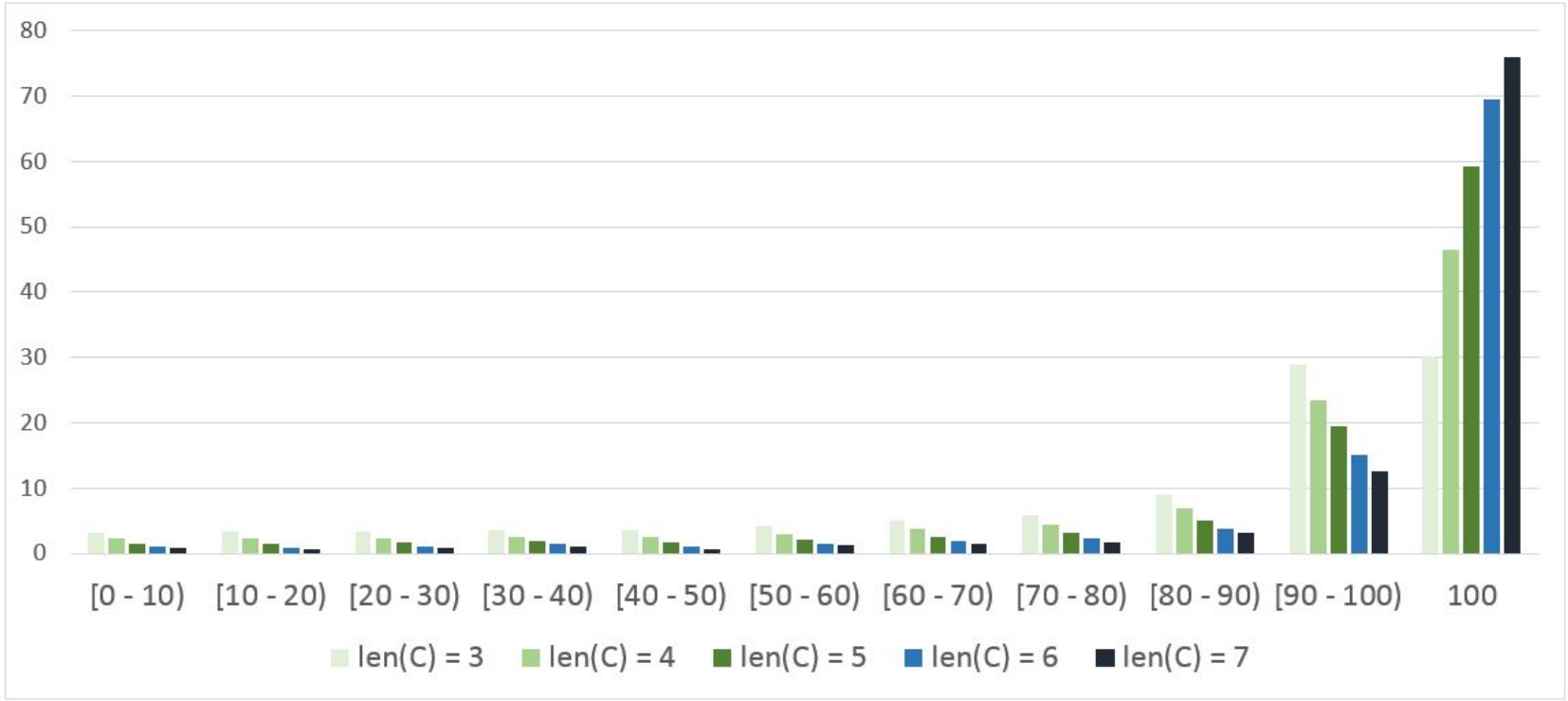
- L, C, R features의 길이별로 purity를 측정합니다.
  - 30,092 건의 뉴스데이터에서 길이별로 purity를 측정합니다.



뉴스 기사의 길이별 L 의 purity

Purity of character features

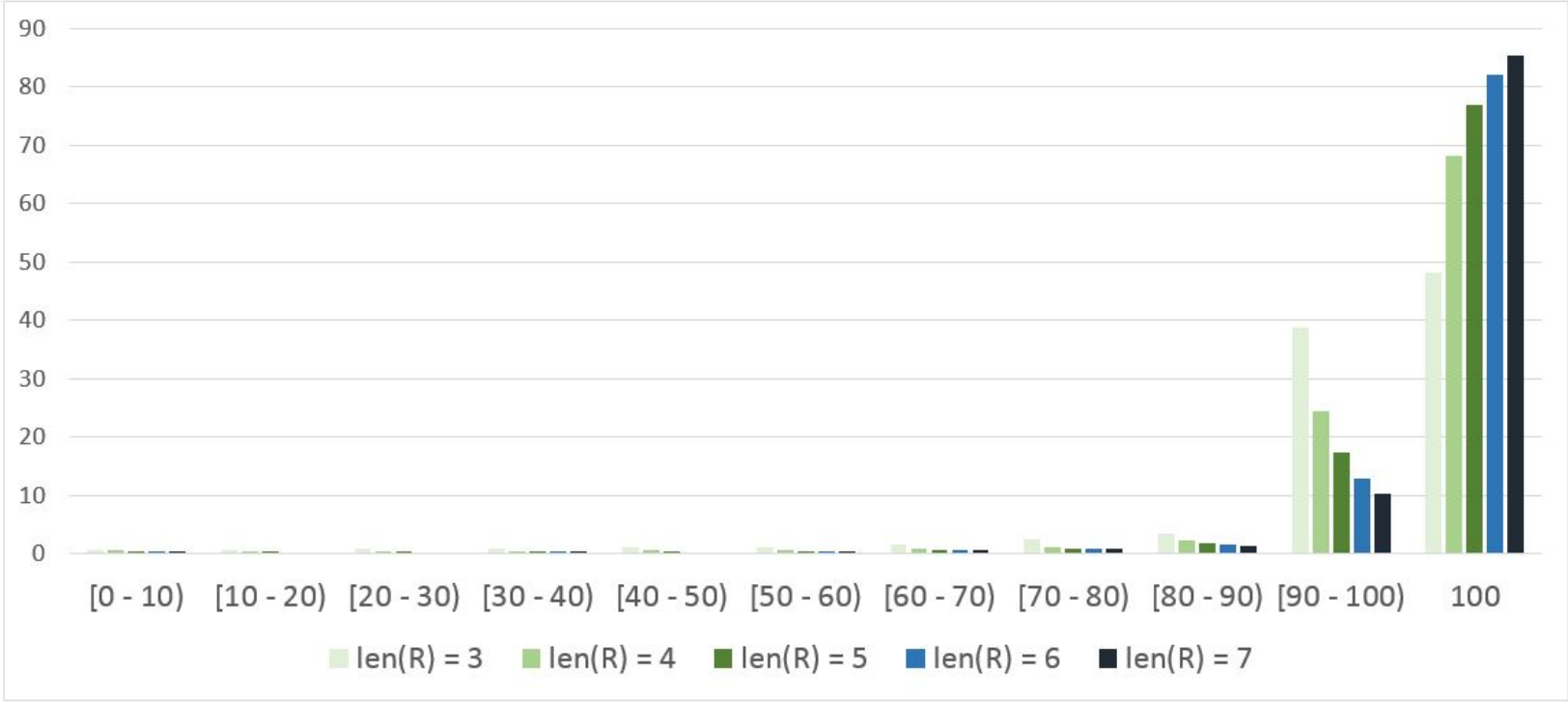
- L, C, R features의 길이별로 purity를 측정합니다.
  - 30,092 건의 뉴스데이터에서 길이별로 purity를 측정합니다.



뉴스 기사의 길이별 C 의 purity

Purity of character features

- L, C, R features의 길이별로 purity를 측정합니다.
  - 30,092 건의 뉴스데이터에서 길이별로 purity를 측정합니다.



뉴스 기사의 길이별 R 의 purity

Purity of character features

- Purity를 살펴보면 대부분의 경우 purity 가 90보다 큽니다.
  - Character features 만 이용하여도 (=앞/뒤 문맥만 보아도) 띄어쓰기 문제는 어렵지 않음을 의미합니다.
  - 하지만 일부 features는 purity가 0에 가깝습니다. 앞/뒤 글자만 봐서는 혼동되는 부분이 존재한다는 의미입니다.

Purity of character features

- 길이가 길수록 purity는 올라갑니다.
  - Feature 길이는 "사용하는 문맥의 범위"입니다.
  - 더 넓은 문맥을 살펴볼수록 더 정확한 판단을 할 수 있습니다.

False space의 원인

- 앞이나 뒤만 봐서는 혼동되는 features가 있습니다.
  - $L_{i-1:i}$ = '이다' 는 x[i]='다'의 1개 앞글자를 이용하는 feature입니다.
  - '-이다'는 대표적인 조사이기 때문에 띄어쓰기를 할 경우가 많습니다.
  - 하지만 $C_{i-1:i+2}$ = '이다음에' 에서 x[i]='다'는 띄어쓰지 않습니다.
  - 앞/뒤 한쪽의 문맥만 보면 잘못된 판단을 할 수 있는 features가 존재합니다.

False space의 원인

- 앞이나 뒤만 봐서는 혼동되는 features가 있습니다.
  - 주로 '-이[다]' vs '이[다]음에' 와 같이, 자주 이용되는 조사/어미가 다른 어절의 왼쪽에 존재할 때 발생하는 현상입니다.
  - 뉴스 데이터에서 이처럼 앞의 문맥만 볼 때와 앞/뒤를 모두 볼 때 띄어쓰기 정보가 달라지는 경우는 4.757% 에 해당합니다.

False space의 원인

- Features를 문맥에 맞춰서 선택할 수 있어야 합니다.
  - CRF는 모든 경우에 동일한 potential functions을 이용합니다.
  - '이[다]음에'의 경우에도 $L_{i-1:i}$='이다'를 feature로 이용하기 때문에 잘못된 parameters가 학습될 수 있습니다.

- 문맥에 맞춰 적절한 features를 이용한다면 "공격적인 띄어쓰기"를 방지할 수 있습니다.

soyspacing

proposed heuristic algorithm

Scoring

"C[-1:3] = 이**다**음에"

"T[-1:3] = 0**0**01": 100번, "T[-1:3] = 0**0**00": 50번

T[0] = 0이므로 (뛴다 = 0번, **안뛴다 150번**) → "**-1점** = -150 / (150 + 0)"

"C[-1:0] = 이**다**"

"T[-1:0] = 0**0**": 200번, "T[-1:0] = 0**1**": 500번,

T[0] = 0이므로 (**뛴다 = 500번**, 안뛴다 200번) → **3/7점** = +500 - 200 / (500 + 200)

Scoring

띄어쓰기 태그 판별 규칙: 띄어쓰기를 해야하는지 아닌지 확실한 부분부터 labeling을 수행

1. 세 종류의 **L, C, R의 빈도수가 0보다 크고**, C[0] 부분에 대하여 **동일한 label**을 보이면
   L, C, R 점수의 평균을 띄어쓰기 점수로 이용
2. L, R의 빈도수가 0이더라도 **C의 빈도수가 0 이상**이면 C의 점수를 띄어쓰기 점수로 이용
   (L이나 R의 빈도수가 0보다 크면 이와 C의 평균 점수를 띄어쓰기 점수로 이용)
3. C의 빈도수가 0이면, **L과 R의 빈도수가 0보다 크고, 동일한 label**을 보이면
   L과 R의 평균 점수를 띄어쓰기 점수로 이용
4. **그 외**, L이나 R이 단독으로만 빈도수가 0 이상이면 띄어쓰기 **점수를 0으로** 설정.
   (단, 첫글자는 제외)

- 점수가 높은 글자부터 태그 판별. 다음 라운드에서 판별된 태그를 반영하여 점수 재계산
- 더 이상 판별할 태그가 없을 때 까지 반복

Illustration

글자	태그	L 점수	C 점수	R 점수	띄어쓰기 점수
데	?				
이	?				
터	?				
과	?				
학	**1**				
시	?				
대	**1**				

입력어: 데이터과학 시대

띄어쓰기 점수 하한: 0.6
붙여쓰기 점수 상한: -0.4
입력어: 데이터과학 시대

"과학_시대"로 띄어졌기 때문에 학은 1로 태깅

Illustration

글자	태그	L 점수	C 점수	R 점수	띄어쓰기 점수
데	**0**	0	0	-1	**-1**
이	?	0	-1	0	-1
터	?	1	-0.33	0	0.33
과	?	0	-1	-1	-1
학	**1**	.	.	.	.
시	?	0	-1	0	-1
대	**1**	.	.	.	.

입력어: 데이터과학 시대

띄어쓰기 점수 하한: 0.6
붙여쓰기 점수 상한: -0.4
입력어: 데이터과학 시대

첫글자이며, 띄어쓰기 점수 크기가 가장 크므로 태그

Illustration

글자	태그	L 점수	C 점수	R 점수	띄어쓰기 점수
데	?	0	0	-1	-1
이	**0**	0	-1	0	**-1**
터	?	1	-0.33	0	0.33
과	?	0	-1	-1	-1
학	**1**	.	.	.	.
시	?	0	-1	0	-1
대	**1**	.	.	.	.

입력어: 데이터과학 시대

띄어쓰기 점수 하한: 0.6
붙여쓰기 점수 상한: -0.4
입력어: 데이터과학 시대

띄어쓰기 점수가 같으므로
"x[0]=데" 대신에 "x[1]=이"를
먼저 태깅할 수도 있음

Illustration

글자	태그	L 점수	C 점수	R 점수	띄어쓰기 점수
데	**0**	.	.	.	.
이	**0**	0	-1	0	-1
터	?	1	-0.33	0	0.33
과	?	0	-1	-1	-1
학	**1**	.	.	.	.
시	?	0	-1	0	-1
대	**1**	.	.	.	.

입력어: 데이터과학 시대

띄어쓰기 점수 하한: 0.6
붙여쓰기 점수 상한: -0.4
입력어: 데이터과학 시대

첫글자부터 태깅했다면,
다음으로 점수가 높음 임의의 글자에 대하여
띄어쓰기 태깅을 수행

Illustration

글자	태그	L 점수	C 점수	R 점수	띄어쓰기 점수
데	**0**	.	.	.	.
이	**0**	.	.	.	.
터	?	1	-0.33	0	0.33
과	?	0	-1	-1	-1
학	**1**	.	.	.	.
시	**0**	0	-1	0	**-1**
대	**1**	.	.	.	.

입력어: 데이터과학 시대

띄어쓰기 점수 하한: 0.6
붙여쓰기 점수 상한: -0.4
입력어: 데이터과학 시대

Illustration

글자	태그	L 점수	C 점수	R 점수	띄어쓰기 점수
데	**0**	.	.	.	.
이	**0**	.	.	.	.
터	?	1	-0.33	0	0.33
과	**0**	0	-1	-1	**-1**
학	**1**	.	.	.	.
시	**0**	.	.	.	.
대	**1**	.	.	.	.

입력어: 데이터과학 시대

띄어쓰기 점수 하한: 0.6
붙여쓰기 점수 상한: -0.4
입력어: 데이터과학 시대

Illustration

글자	태그	L 점수	C 점수	R 점수	띄어쓰기 점수
데	0	.	.	.	.
이	0	.	.	.	.
터	?	1	-0.33	0	0.33
과	0	.	.	.	.
학	1	.	.	.	.
시	0	.	.	.	.
대	1	.	.	.	.

입력어: 데이터과학 시대

띄어쓰기 점수 하한: 0.6
붙여쓰기 점수 상한: -0.4
입력어: 데이터과학 시대

띄어쓰기 점수가 하한보다 작기 때문에 태깅을 하지 않음
태그가 달라지지 않기 때문에 오류 교정을 중지

Performance

Dataset

- 띄어쓰기 알고리즘의 정확한 precision / recall을 측정하기 위해서는 띄어쓰기 오류가 없는 데이터가 필요합니다.

- 뉴스 데이터는 온라인에서 구할 수 있는 깨끗한 데이터 중 하나입니다.

Dataset

- 2016-10-20 뉴스를 성능 평가용 데이터로 이용합니다.
  - 30,091 건의 뉴스 기사
  - 뉴스에 노이즈를 추가하여 학습 / 테스트 데이터를 만듭니다.
  - 띄어쓰기 교정 알고리즘을 이용하여 추가된 노이즈가 얼마나 제거되는지 확인합니다.

Performance measure

- 우리는 띄어쓰기 교정 문제를 "본래 띄어써야 하는 부분을 붙여쓴 경우"로 정의하였습니다.

- 띄어쓰기 교정은 두 가지 관점에서 성능이 측정되어야 합니다.
  - 본래 띄어써야 하는 부분이 제대로 띄어졌는가? → remain noise
  - 본래 붙여써야 하는 부분이 띄어지진 않았는가? → false spacing

Model setting

- 제안된 모델과 CRF 기반 모델은 동일한 character features를 이용합니다.
  - Character features 길이: 3 ~ 7
  - Feature min-count : 5

- CRF feature templates
  - bulabula

Remain noise: Proposed algorithm

remain noise		Test data noise level									
		1	2	3	5	10	15	20	30	40	50
Train data noise level	1	0.203	0.341	0.520	0.854	1.731	2.581	3.424	5.125	6.807	8.475
	2	0.175	0.410	0.527	0.866	1.753	2.615	3.470	5.197	6.901	8.591
	3	0.177	0.351	0.638	0.880	1.785	2.665	3.534	5.294	7.028	8.750
	5	0.183	0.363	0.554	1.097	1.844	2.752	3.649	5.467	7.259	9.027
	10	0.203	0.400	0.610	1.009	2.508	3.044	4.033	6.039	8.020	9.978
	15	0.231	0.454	0.692	1.144	2.308	4.323	4.572	6.843	9.089	11.314
	20	0.269	0.534	0.810	1.341	2.701	4.041	6.805	8.024	10.656	13.270
	30	0.433	0.861	1.312	2.166	4.351	6.513	8.670	16.217	17.277	21.554
	40	0.798	1.581	2.391	3.969	7.945	11.927	15.908	23.861	34.821	39.839
	50	0.896	1.777	2.691	4.465	8.921	13.413	17.880	26.829	35.773	46.123

Remain noise: Proposed algorithm

- 5 % 노이즈 데이터로 학습한 모델로 30 % 노이즈를 교정하면 24.533%가 교정되고, 5.467% 가 남는다는 의미입니다.

remain noise		Test data noise level									
		1	2	3	5	10	15	20	30	40	50
Train data noise level	1	0.203	0.341	0.520	0.854	1.731	2.581	3.424	5.125	6.807	8.475
	2	0.175	0.410	0.527	0.866	1.753	2.615	3.470	5.197	6.901	8.591
	3	0.177	0.351	0.638	0.880	1.785	2.665	3.534	5.294	7.028	8.750
	5	0.183	0.363	0.554	1.097	1.844	2.752	3.649	5.467	7.259	9.027
	10	0.203	0.400	0.610	1.009	2.508	3.044	4.033	6.039	8.020	9.978
	15	0.231	0.454	0.692	1.144	2.308	4.323	4.572	6.843	9.089	11.314
	20	0.269	0.534	0.810	1.341	2.701	4.041	6.805	8.024	10.656	13.270
	30	0.433	0.861	1.312	2.166	4.351	6.513	8.670	16.217	17.277	21.554
	40	0.798	1.581	2.391	3.969	7.945	11.927	15.908	23.861	34.821	39.839
	50	0.896	1.777	2.691	4.465	8.921	13.413	17.880	26.829	35.773	46.123

Remain noise: CRF based algorithm

remain noise		Test data noise level									
		1	2	3	5	10	15	20	30	40	50
Train data noise level	1	0.164	0.334	0.508	0.833	1.681	2.493	3.290	4.871	6.390	7.850
	2	0.122	0.245	0.361	0.592	1.178	1.747	2.280	3.302	4.241	5.093
	3	0.143	0.280	0.435	0.696	1.401	2.072	2.716	3.987	5.184	6.321
	5	0.192	0.373	0.567	0.936	1.870	2.784	3.669	5.416	7.097	8.706
	10	0.272	0.532	0.812	1.332	2.731	3.972	5.245	7.765	10.195	12.527
	15	0.363	0.715	1.092	1.792	3.584	5.488	7.110	10.591	13.996	17.328
	20	0.355	0.701	1.070	1.759	3.511	5.261	7.198	10.348	13.648	16.868
	30	0.420	0.827	1.260	2.074	4.137	6.200	8.209	12.754	16.133	19.970
	40	0.550	1.092	1.653	2.728	5.428	8.138	10.786	16.078	22.158	26.369
	50	0.685	1.362	2.055	3.406	6.789	10.178	13.535	20.214	26.796	34.296

Remain noise

- 전반적으로 제안된 모델이 CRF 기반 모델들보다 노이즈를 덜 제거합니다.
- 특히 제안된 모델은 학습데이터의 noise level 이 50% 에 가까워지면 거의 작동하지 않습니다.
  - 제안된 모델은 다수결의 원칙을 따르기 때문입니다.

False spacing: Proposed algorithm

remain noise		Test data noise level									
		1	2	3	5	10	15	20	30	40	50
Train data noise level	1	0.406	0.407	0.408	0.410	0.414	0.418	0.422	0.431	0.439	0.449
	2	0.394	0.394	0.396	0.397	0.402	0.406	0.409	0.418	0.426	0.435
	3	0.380	0.381	0.381	0.383	0.388	0.392	0.395	0.403	0.411	0.420
	5	0.350	0.350	0.351	0.351	0.356	0.360	0.363	0.371	0.378	0.387
	10	0.284	0.284	0.285	0.286	0.288	0.292	0.295	0.302	0.309	0.316
	15	0.223	0.223	0.224	0.225	0.228	0.227	0.233	0.239	0.244	0.250
	20	0.177	0.177	0.178	0.179	0.181	0.183	0.180	0.190	0.194	0.199
	30	0.084	0.084	0.084	0.084	0.085	0.086	0.087	0.081	0.089	0.091
	40	0.028	0.028	0.028	0.028	0.029	0.029	0.029	0.028	0.024	0.030
	50	0.015	0.015	0.015	0.015	0.015	0.015	0.015	0.015	0.015	0.012

False spacing: CRF based algorithm

remain noise		Test data noise level									
		1	2	3	5	10	15	20	30	40	50
Train data noise level	1	1.149	1.154	1.159	1.168	1.192	1.213	1.238	1.285	1.331	1.377
	2	1.726	1.735	1.745	1.764	1.815	1.863	1.916	2.017	2.117	2.224
	3	1.278	1.284	1.289	1.299	1.327	1.353	1.382	1.437	1.491	1.546
	5	0.941	0.946	0.949	0.957	0.978	0.996	1.017	1.058	1.097	1.139
	10	0.549	0.552	0.555	0.561	0.578	0.591	0.609	0.641	0.673	0.704
	15	0.345	0.347	0.349	0.353	0.363	0.373	0.383	0.405	0.425	0.444
	20	0.376	0.378	0.380	0.385	0.395	0.405	0.416	0.437	0.458	0.476
	30	0.230	0.231	0.232	0.235	0.242	0.248	0.255	0.269	0.281	0.293
	40	0.126	0.126	0.127	0.128	0.132	0.134	0.137	0.144	0.150	0.155
	50	0.059	0.059	0.059	0.060	0.062	0.063	0.065	0.069	0.073	0.076

False spacing

- 제안된 모델은 False spacing 의 경우가 적습니다.
  - 제안된 모델은 보수적으로 띄어쓰기를 수행합니다.
  - 정보가 불충분하거나, 띄어쓰기 점수가 애매한 경우에는 띄어쓰기를 하지 않습니다.

- 띄어쓰기 교정은 이후 자연어처리 엔진의 input을 정제하는 과정입니다.
  - 띄어쓰기를 하지 않더라도 품사 판별 / 토크나이징이 잘 될 수 있습니다.
  - 하지만 한 번 띄어진 단어가 제대로 인식되는 것은 매우 어렵습니다

F1 score: Proposed algorithm

F1-macro all		0.698
F1-macro >= 30	%	0.808

remain noise		Test data noise level									
		1	2	3	5	10	15	20	30	40	50
Train data noise level	1	0.608	0.739	0.786	0.831	0.865	0.879	0.886	0.892	0.896	0.898
	2	0.628	0.724	0.788	0.832	0.865	0.878	0.885	0.891	0.895	0.897
	3	0.634	0.746	0.770	0.832	0.865	0.877	0.884	0.890	0.893	0.896
	5	0.647	0.753	0.794	0.812	0.864	0.875	0.882	0.887	0.891	0.893
	10	0.672	0.766	0.801	0.834	0.829	0.868	0.873	0.878	0.881	0.883
	15	0.693	0.773	0.801	0.828	0.847	0.817	0.859	0.863	0.866	0.867
	20	0.700	0.766	0.789	0.811	0.826	0.832	0.786	0.839	0.842	0.843
	30	0.653	0.689	0.695	0.708	0.714	0.717	0.719	0.627	0.722	0.724
	40	0.324	0.340	0.333	0.339	0.340	0.339	0.339	0.339	0.229	0.337
	50	0.186	0.199	0.186	0.193	0.194	0.191	0.192	0.191	0.191	0.144

F1 score: CRF based algorithm

F1-macro all		0.693
F1-macro >= 30	%	0.731

remain noise		Test data noise level									
		1	2	3	5	10	15	20	30	40	50
Train data noise level	1	0.400	0.554	0.635	0.722	0.802	0.834	0.852	0.870	0.881	0.888
	2	0.326	0.481	0.574	0.678	0.783	0.827	0.851	0.878	0.893	0.903
	3	0.384	0.543	0.626	0.720	0.807	0.843	0.862	0.883	0.894	0.902
	5	0.434	0.586	0.660	0.738	0.808	0.835	0.850	0.867	0.876	0.882
	10	0.512	0.640	0.692	0.748	0.788	0.810	0.820	0.831	0.837	0.843
	15	0.545	0.644	0.679	0.719	0.748	0.753	0.766	0.773	0.778	0.782
	20	0.535	0.639	0.677	0.718	0.751	0.762	0.761	0.778	0.783	0.788
	30	0.567	0.644	0.668	0.696	0.717	0.724	0.730	0.722	0.741	0.745
	40	0.529	0.575	0.585	0.604	0.616	0.620	0.625	0.630	0.614	0.639
	50	0.441	0.464	0.465	0.475	0.482	0.484	0.486	0.490	0.495	0.477

F1 score

- 제안된 모델은 더 높은 F1 score를 보입니다.
  - Remain noise 가 더 있지만, False spacing을 훨씬 적게 하기 때문에 제안된 모델의 precision 이 높습니다.
  - 하지만 조금 낮은 수준의 recall을 보이기 때문에 F1 score는 제안된 모델이 더 좋습니다.

- 특히 학습데이터의 노이즈 수준을 적절히만 잡을 수 있다면 (under 30%) F1 score 의 차이는 명확해집니다.

Training cost

- 제안된 모델은 substring counting 이 학습의 전부입니다.
- CRF 기반 모델은 긴 학습 시간이 필요합니다.
  - substring counting을 하여 features의 min count cutting을 수행합니다.
    - 이 과정은 제안된 모델의 학습과정입니다.
  - 그 뒤, potential function을 이용하여 feature 를 변환한 뒤,
  - Logistic regression을 위한 gradient descent 기반 학습을 수행합니다.
  - Parameter가 수렴할 때까지 이 과정을 반복해야만 합니다.

Discussion

학습데이터 설정

- 성능 평가로부터 일부 노이즈가 있어도 띄어쓰기 교정기는 학습이 될 수 있음을 확인하였습니다.
- 하지만 지나치게 많은 노이즈는 띄어쓰기 교정기 학습을 방해합니다.

- 가능한 적은 노이즈의 문장들을 학습데이터에 추가하는 것이 좋습니다.

학습데이터 설정

- 주어진 데이터에서 평균 어절의 길이가 짧은 순서대로 문장을 고릅니다
  - 한국어 평균 어절의 길이는 약 3.4입니다.
  - 매우 긴 어절들만 포함되어 있는 문장이라면 띄어쓰기 교정용으로는 적절하지 않습니다.
  - Vocabulary distribution이 맞을 수 있도록 평균 어절 길이가 짧은 순서대로, 미리 설정한 data volume 까지 문장을 골라서 학습데이터를 구성합니다.

띄어쓰기 정보의 유용성

- 교정해야 할 문장에는 일부 띄어쓰기가 되어 있습니다.
  - 띄어쓰기 부분을 모두 제거한 뒤, 알고리즘을 적용할 수도 있습니다.
  - 하지만 일부 존재하는 띄어쓰기는, 그것마저 없으며 이해되기 어렵기 때문에 사용자가 달아둔 것입니다.
  - 붙여써야 하는 부분을 굳이 띄어쓰는 경우는 없으니, 이 정보를 활용하는 것이 좋습니다.

띄어쓰기 정보의 유용성

- 일부 띄어쓰기 정보에 대한 유용성을 검증합니다.
  - 10 % noise level test 데이터에 대하여 교정 알고리즘을 적용합니다.
  - 모든 경우에 존재하는 띄어쓰기는 유지하는 것이 더 좋습니다.

Proposed model

Remain noise (%)		With space (10 % noise)	Without space
Train data noise level	3	1.785	19.591
	10	2.508	21.996
	20	2.701	28.384

CRF based model

Remain noise (%)		With space (10 % noise)	Without space
Train data noise level	3	1.401	11.116
	10	2.731	23.241
	20	3.511	32.052

Conclusion

Conclusion

- 명확한 context를 이용할수록 정확한 tagging을 할 수 있습니다.

  - 띄어쓰기가 혼동되는 문맥은 그리 많지 않습니다.
    Character features만 잘 이용하여도 띄어쓰기는 충분히 교정할 수 있습니다.

  - 앞/뒤, 한 방향의 문맥만 이용하면 false spacing이 일어날 수 있습니다.
    양방향의 문맥 정보를 이용해야 안전한 띄어쓰기를 할 수 있습니다.

Conclusion

- 노이즈가 없는 완벽한 학습데이터를 구할 필요가 없습니다.

  - 학습데이터에 어느 정도 노이즈가 있어도 모델은 잘 학습됩니다.
  - 지나치지 않는 노이즈라면 그대로 학습데이터로 이용할 수 있습니다.

  - 더 중요한 것은 단어의 분포이니, 띄어쓰기를 교정하려는 데이터에서 노이즈가 적은 문장들을 골라 학습데이터를 구성하면 됩니다.

References

[1] D.-G. Lee, S.-Z. Lee, H.-C. Rim, H.-S. Lim, Automatic word spacing using hidden markov model for refining korean text corpora, in: Proceedings of the 3rd workshop on Asian language resources and international standardization-Volume 12, Association for Computational Linguistics, pp. 1–7.

[2] D.-G. Lee, H.-C. Rim, D. Yook, Automatic word spacing using probabilistic models based on character n-grams, IEEE Intelligent Systems 22 (2007).

[3] K. Shim, Automatic word spacing based on conditional random fields, Korean Journal of Cognitive Science 22 (2011) 217–233.

[4] C. Lee, H. Kim, Automatic korean word spacing using pegasos algorithm, Information Processing & Management 49 (2013) 370–379.

[5] C. Lee, J. Kim, J. Kim, H. Kim, Joint models for korean word spacing and pos tagging using 425 structural svm, Journal of KIISE: Software and Applications 40 (2013) 826–832. 26

[6] C. Lee, E. Choi, H. Kim, Balanced korean word spacing with structural svm., in: EMNLP, pp. 875–879.

[7] C. Lee, S. Lee, K. Kim, H. Lim, A data-driven spacing correction system using character-level unidirectional lstm, Proceedings of the Korean Information Science Society Conference (2017) 660–430 662

References

[8] G. Hong, H.-C. Rim, Korean spacing by improving viterbi segmentation, in: Advanced Language Processing and Web Information Technology, 2007. ALPIT 2007. Sixth International Conference on, IEEE, pp. 75–80.

[9] M.-y. Kang, S.-w. Jung, H.-c. Kwon, Category-pattern-based korean word-spacing, Lecture notes in 435 computer science 4285 (2006) 288.

[10] M.-y. Kang, S.-w. Choi, H.-c. Kwon, A hybrid approach to automatic word-spacing in korean, in: IEA/AIE, Springer, pp. 284–294.

[11] S.-S. Kang, Eojeol-block bidirectional algorithm for automatic word spacing of hangul sentences, Journal of KIISE: Software and Applications 27 (2000) 441–447.

[12] H. Tseng, P. Chang, G. Andrew, D. Jurafsky, C. Manning, A conditional random field word segmenter for sighan bakeoff 2005, in: Proceedings of the fourth SIGHAN workshop on Chinese language Processing, volume 171.

[13] T. Kudo, K. Yamamoto, Y. Matsumoto, Applying conditional random fields to japanese morphological analysis., in: EMNLP, volume 4, pp. 230–237.