Brain neuro
Language
Processing

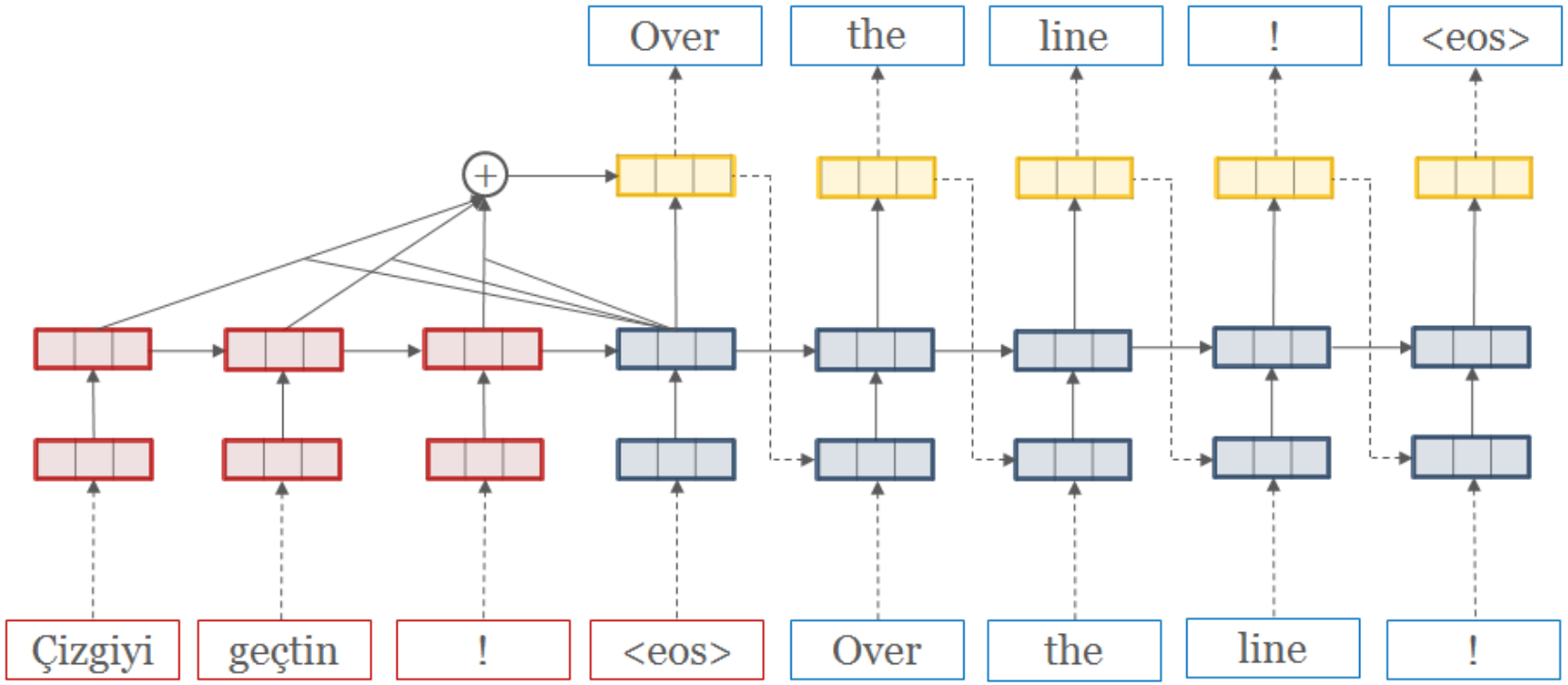
Overcoming Vocabulary Limit

고려대학교 컴퓨터학과
Brain-neuro Language Processing Lab
이찬희
E-mail : chanhee0222@korea.ac.kr

Index

Motivation

Neural Network과 언어



인간의 언어 - 문자 ⟷ NN 입출력 - 숫자

Neural Network 단어 입력

“… … … … …. The fat cat sat on the mat. … … … … …“

“… … … … …. 32 832 561 634 6132 32 565 … … … … …”

단어 단위 처리

- 단어/형태소를 서로 독립적인 최소 단위로 취급
- 단어/형태소를 최소 단위로 했을 때의 한계점
  - 단어/형태소의 목록이 미리 정의되어 있어야 함 - vocabulary
  - 신경망에서 학습시켜야 할 파라미터의 수가 단어/형태소의 개수와 비례하여 증가
  - 정의되지 않은(OOV, Out Of Vocabulary) 단어 발생

Out-Of-Vocabulary Word

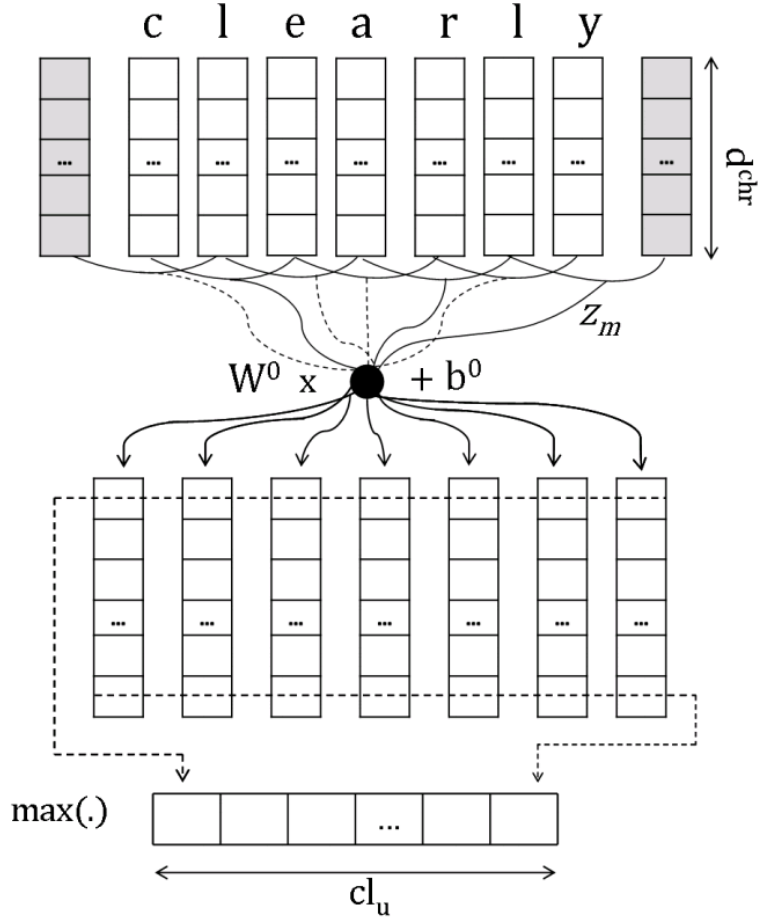
- 모든 단어를 vocabulary에 등록할 수는 없음
  - 컴퓨팅 자원의 한계
  - 새로운 단어의 생성
- Vocabulary에 포함되지 못한 단어는 “알수없음(Unknown)”으로 대체
  - “새 몬스터는 기존의 <UNK>처럼 알을 부화시키는 방식이 아니라 야생에서 등장할 전망이다. 업데이트 후에는 <UNK>지방에서 발견된 일부 <UNK>을 <UNK>지방에 사는 <UNK>으로 진화시킬 수 있게 되고 성별 구분도 늘어난다. <UNK>에서 얻을 수 있는 아이템도 늘어난다. <UNK>을 잡을 때 사용할 수 있는 열매는 기존의 <UNK>에서 <UNK>의 움직임을 둔화시켜주는 <UNK>열매와 사탕을 두 배로 얻을 수 있는 <UNK>열매가 추가된다.“

문자 단위 처리

- 단어/형태소가 아닌 문자를 최소 처리 단위로 사용
- 단어/형태소의 수 ≫ 문자의 수
- 한 언어 내에서 사용되는 문자의 집합은 변화가 매우 적음
- 크게 두 가지 접근 방법이 존재
  - Convolution 사용
  - 문자 단위 Recurrent Neural Network 사용

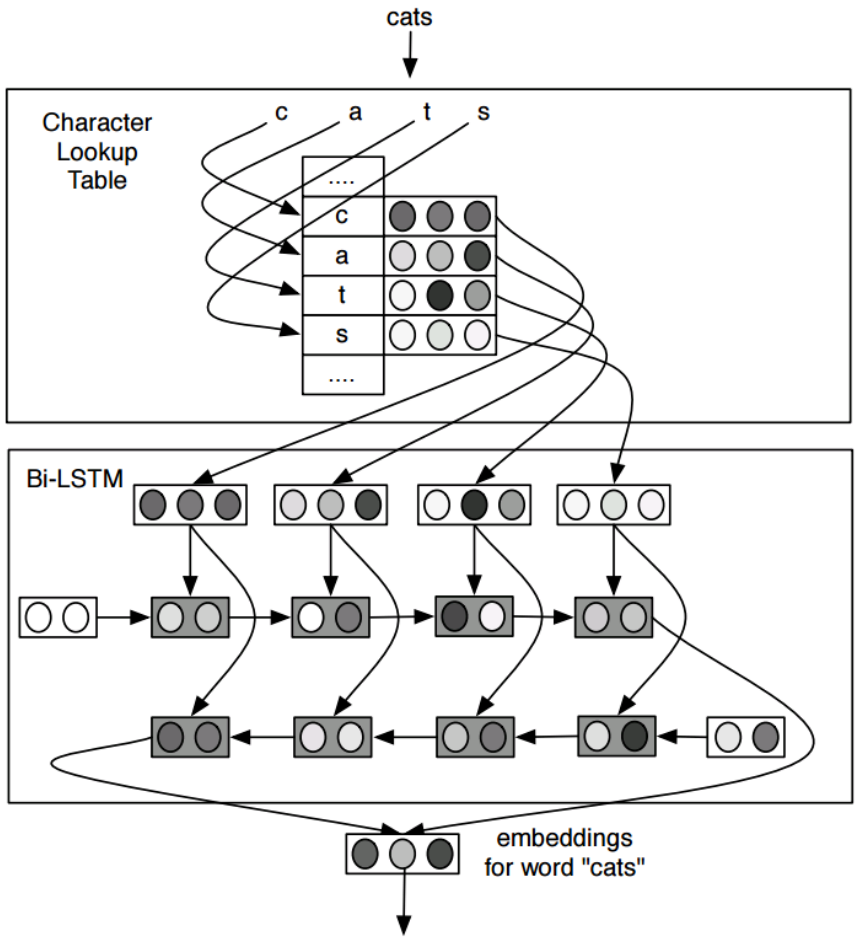
Convolution

- Word embedding과 같은 방법으로 각 문자를 character embedding로 변환
- 단어를 구성하는 문자들의 character embedding에 convolution 적용 후, pooling을 사용하여 단어 벡터 생성



Character RNN

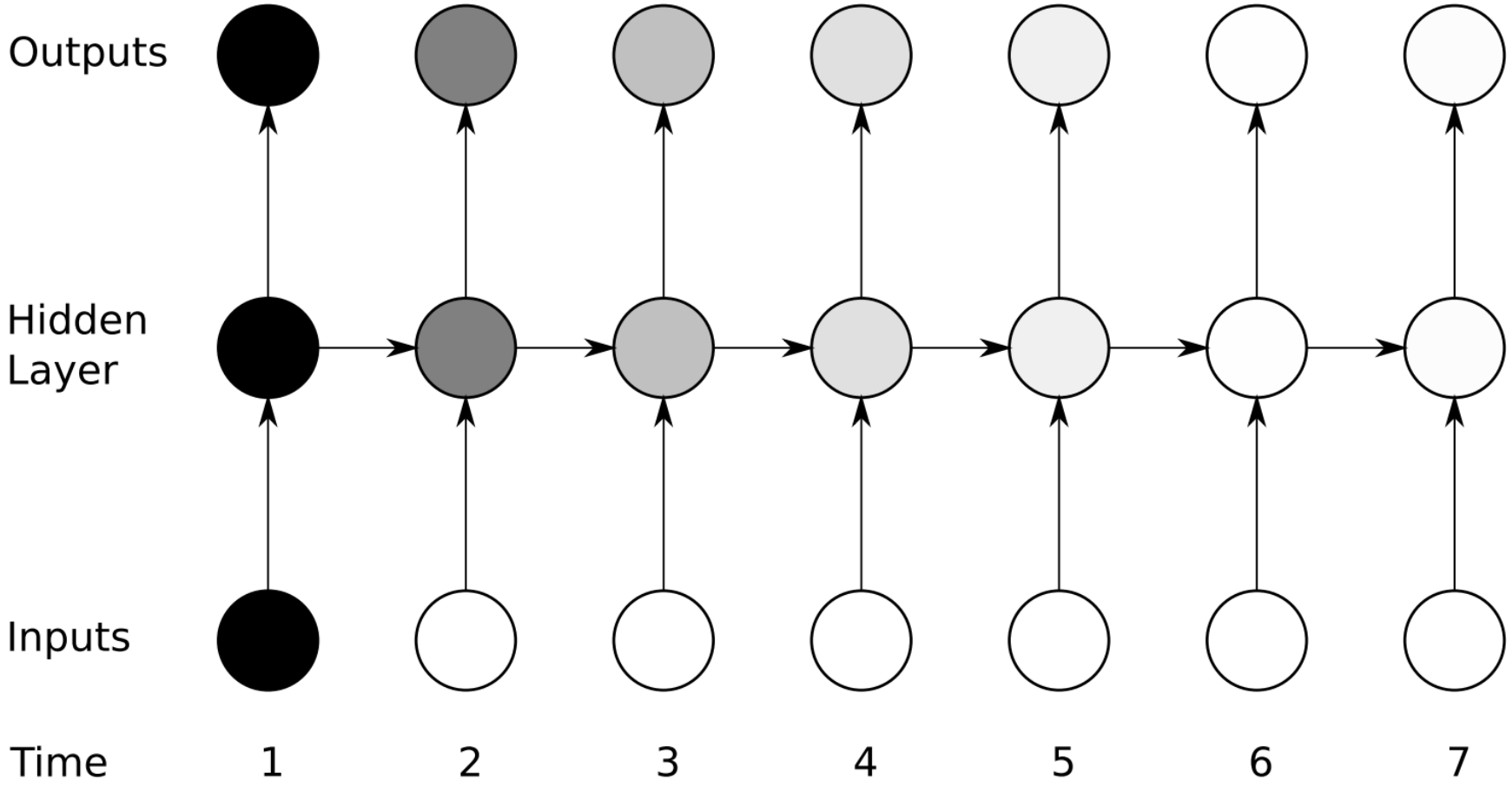
- 단어를 구성하는 문자들의 character embedding에 RNN을 적용하여 단어 벡터 생성
- Word embedding을 보조하는 역할로 사용, 혹은 word embedding 없이 character RNN만을 사용



문제점

- Convolution - 단어 내 문자의 순서와 단어 벡터가 독립적: anagram 구분에 취약 (altitude/latitude, silent/listen)
- RNN - 입력 token이 단어의 수에서 문자의 수로 증가: long term dependency에 취약
- Convolution, RNN을 사용하여 문자 단위 처리를 할 경우, 추가적인 신경망 구조가 필요: 파라미터 수, 계산량 증가

Long Term Dependency 문제



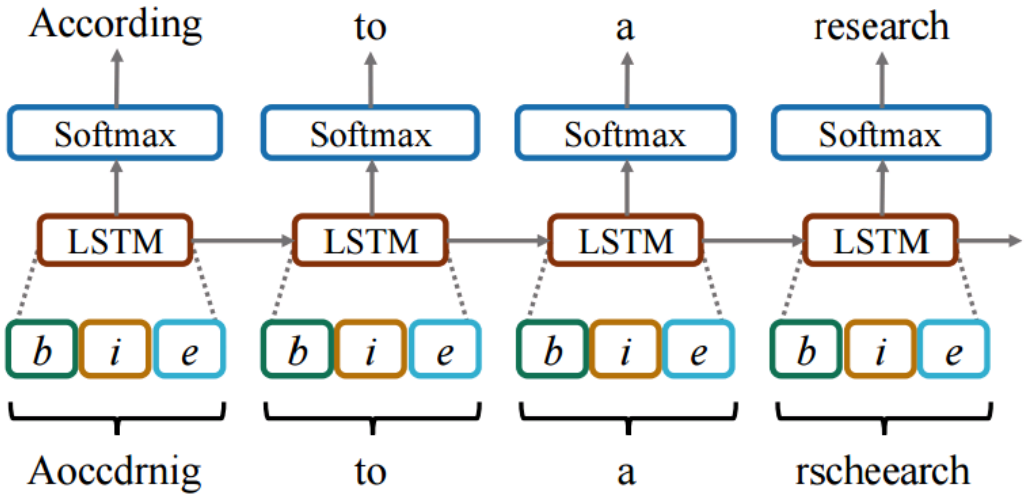
Background

Word Superiority Effect

- 캠릿브지 대학의 연결구과에 따르면, 한 단어 안에서 글자가 어떤 순서로 배되열어 있는가 하것는은 중하요지 않고, 첫째번와 마지막 글자가 올바른 위치에 있것는이 중하요다고 한다. 나머지 글들자은 완전히 엉진창망의 순서로 되어 있지을라도 당신은 아무 문없제이 이것을 읽을 수 있다. 왜하냐면 인간의 두뇌는 모든 글자를 하나 하나 읽것는이 아니라 단어 하나를 전체로 인하식기 때이문다.

Semi-character RNN

- 이 현상에 착안하여 시도된 단어 표현
  - Sakaguchi, Keisuke, et al. "Robsut Wrod Reocginiton via semi-Character Recurrent Neural Network." arXiv preprint arXiv:1608.02214 (2016).
- 단어의 첫 글자와 마지막 글자만 보존하고, 사이 글자들은 순서가 없는 bag of characters로 처리
- 이를 이용하여 철자 교정 실험 수행

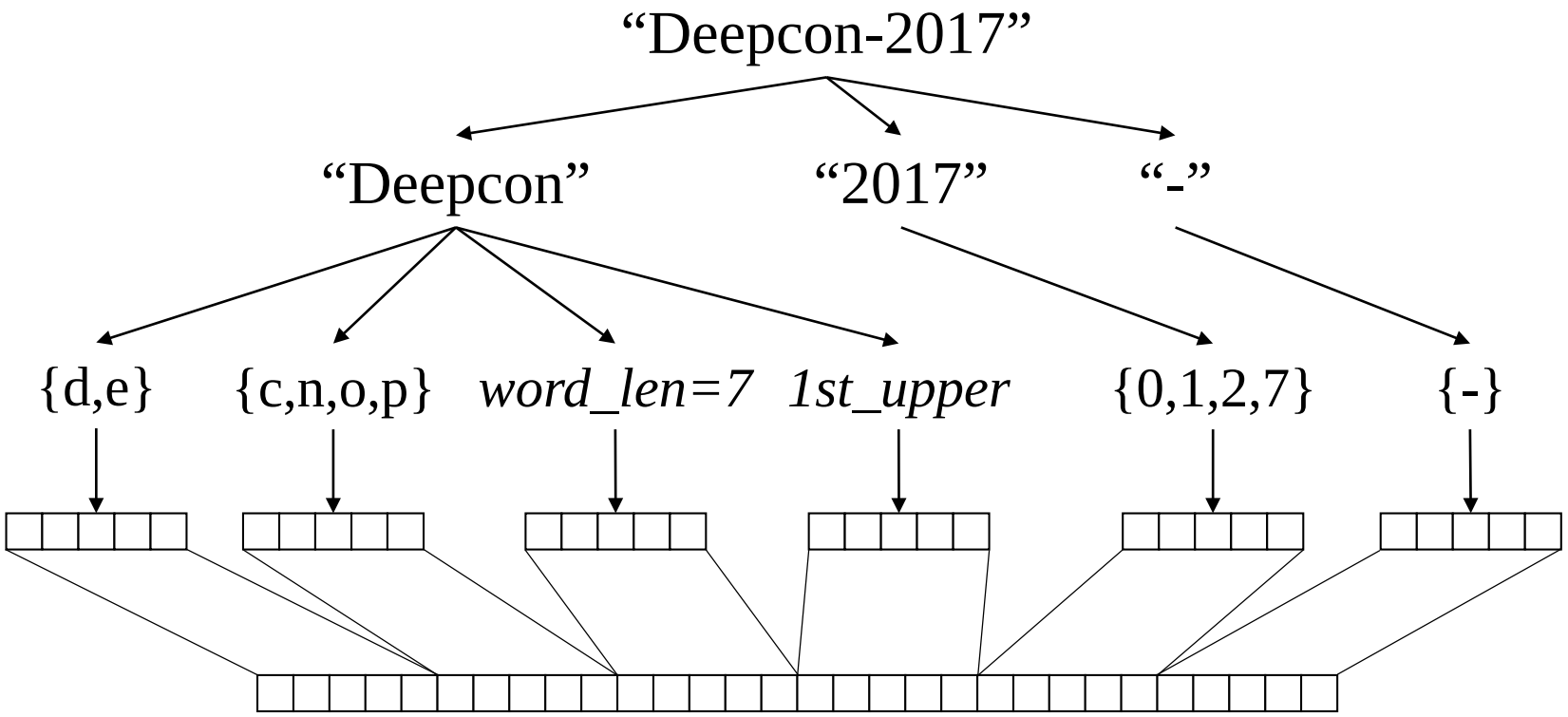


Bags of Features

- 단어를 구성하는 문자는 bag of characters로 저장하되, 손실되는 문자 순서 정보를 보충할 수 있는 정보를 추가
  - 단어를 분할하여 문자들의 정보 손실 감소
  - 문자들의 순서가 무시되면서 손실된 정보를 벡터에 추가
  - 서로 다른 단어가 같은 벡터 표현을 갖는 현상을 최소화

Representing Word with Bags of Features

Bags of Features

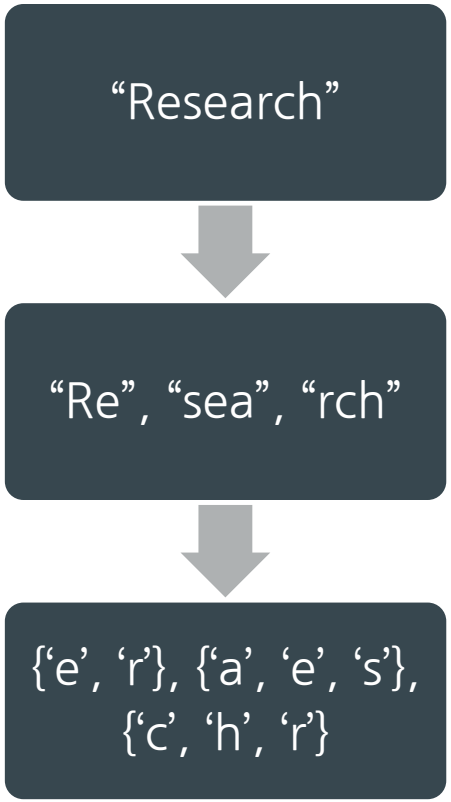


Bags of Features 구성

- Bags of Features에는 아래 feature들이 사용됨 (영어 기준)
  - Alphabet Feature
  - Digits Feature
  - Special Characters Feature
  - Capitalization Feature
  - Word Length Feature

Alphabet Feature

- 단어 순서 정보 손실을 최소화하기 위해 단어를 여러 조각으로 분할
  - 각 조각 당 문자의 수가 최대한 균일 하도록 분할함
  - 품사 태깅 실험에서는 한 단어를 3 부분으로 분할
- 모든 문자는 소문자로 변환
- 한 조각 내에 동일한 문자가 2회 이상 등장할 경우 ‘중복‘ 표시
- 각 조각은 bag of characters를 사용하여 벡터화



Capitalization Feature

- Alphabet feature에서 모든 문자를 소문자로 변환하였기 때문에 대소문자 정보는 별도로 저장
- 단어를 아래 4가지 중 하나로 구분하여 one-hot encoding으로 벡터화
  - 모두 소문자
  - 모두 대문자
  - 첫 글자만 대문자
  - 기타

Special Characters Feature

- 학습에 사용할 코퍼스에서 등장 빈도가 가장 높은 15개 특수 문자 선정
- 위 15개 외의 특수 문자는 ‘unknown’으로 대체
- Bag of characters를 사용하여 벡터화

Other Features

- Digits Feature
  - 숫자는 bag of characters를 사용하여 벡터화
  - Alphabet feature에서와는 달리 중복 표시 사용 안함, 여러 조각으로 분할하지 않음
- Word Length Feature
  - 단어의 문자 수를 0~20, 20이상으로 구분하여 one-hot encoding으로 벡터화

Bags of Features의 장점

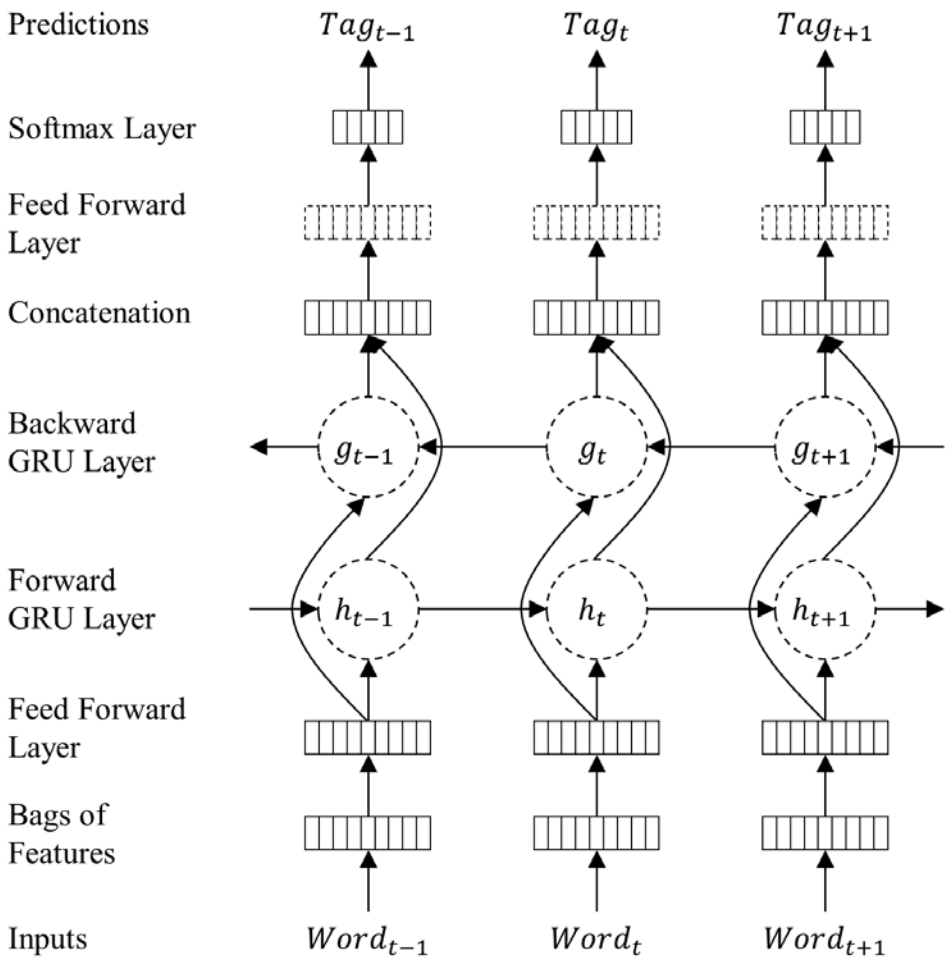
- 단어의 문자들로부터 단어 벡터를 생성하기 때문에 문자 단위 처리의 장점을 계승
  - 사용할 단어/형태소를 미리 정의해 둘 필요가 없음
  - OOV 단어의 처리 성능 향상
  - 신경망의 파라미터 수가 단어/형태소의 개수와 독립적
- Convolution 혹은 character-level RNN의 단점 개선
  - 신경망 입력이 단어 단위 - long term dependency 문제 완화
  - 단순한 규칙 기반 벡터화 - 추가적인 신경망 구조가 필요 없음
  - Word embedding을 보조하지 않고 독립적으로 사용 가능

Experiments

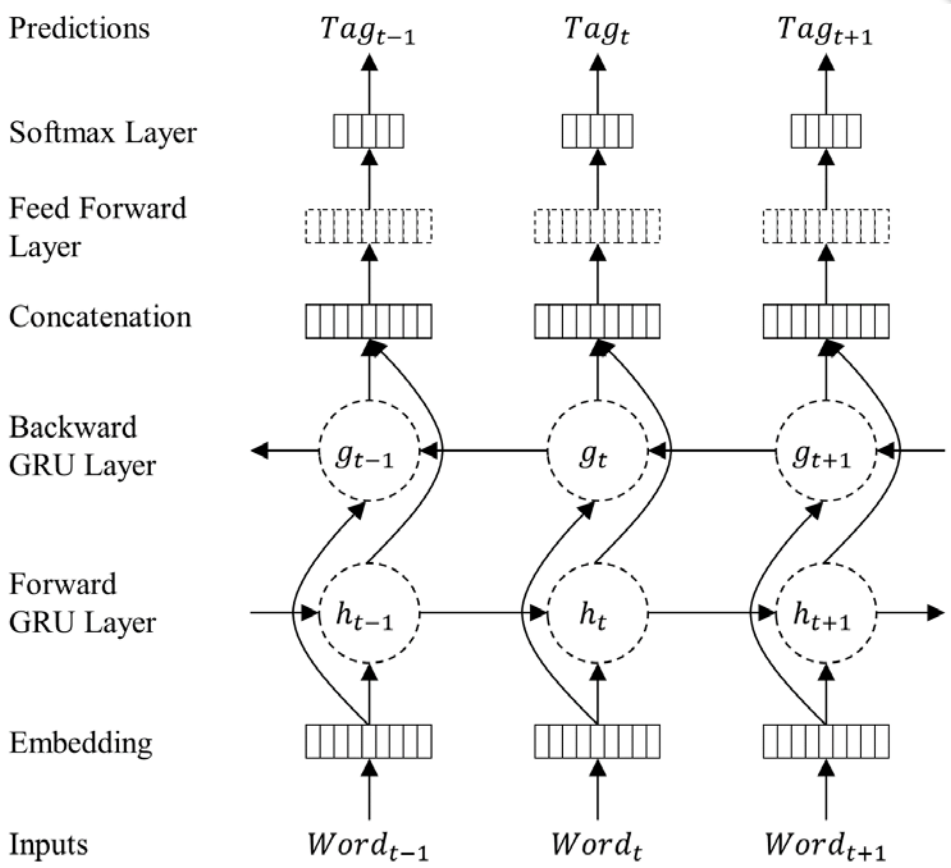
Bags of Features 효과 검증

- BOF를 사용하여 단어를 표현하는 것의 효과 검증을 위해 품사(Part of Speech) 태깅 실험 수행
- BOF가 단어를 효과적으로 벡터화한다면 품사 태깅에서 우수한 성능을 낼 것이라는 가정
- One-hot encoding으로 단어를 처리하는 모델을 baseline으로 제작

Model Architecture



(a) Bags of Features



(b) One-hot Encoding

실험 데이터

- Penn TreeBank 데이터의 Wall Street Journal 부분 사용
- 영어 품사 태깅 실험에서 가장 보편적으로 쓰이는 데이터
- Train data: section 0-18
- Development data: section 19-21
- Test data: section 22-24
- 총 45종류의 품사 태그 사용

결과

- One-hot encoding baseline 대비
  - OOV 단어 태깅 정확도 49.68% 향상
  - 전체 단어 태깅 정확도 0.96% 향상
  - 모델의 파라미터 수 약 3배 감소

Model	All	OOV	Param.
One-hotbaseline	96.16	57.71	32,119K
Bags of Features	**97.08**	**86.38**	**10,585K**

결과

- 기존 state-of-the-art 기술들 대비
  - 추가 데이터 활용하지 않는 모델들과 유사한 성능을 보임
  - 단순한 모델 사용 - 모델 구조 개선으로 성능 향상 기대

Approach	All	OOV	Extra
Manning (2011)	97.32	90.79	Yes
Shen et al. (2007)	97.33	89.61	No
Sun (2014)	97.36	-	No
Moore (2015)	97.36	91.09	Yes
Hajič et al. (2009)	97.44	-	Yes
Søgaard (2011)	97.50	-	Yes
Tsuboi (2014)	97.51	91.64	Yes
Huang et al. (2015)	97.55	-	Yes
Choi (2016)	97.64	92.03	Yes
This work	97.08	86.38	No

감사합니다.

이찬희
E-mail : chanhee0222@korea.ac.kr