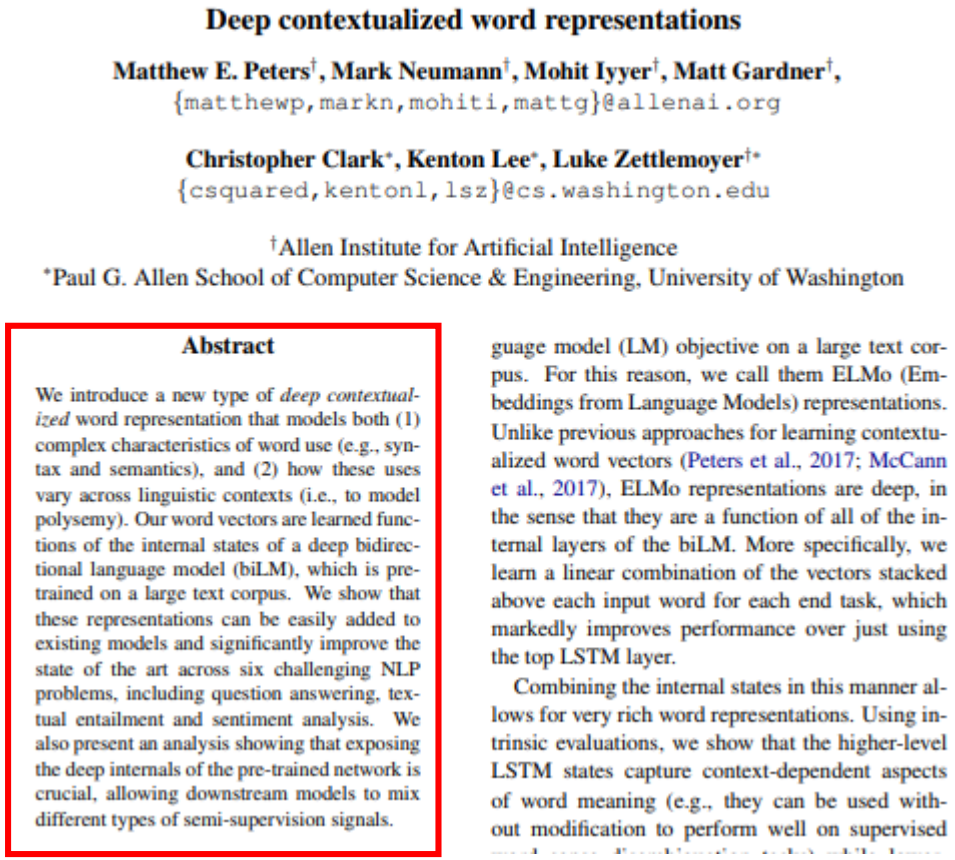
Sentence Embeddings Using Korean Corpora

ELMO

5.4 ELMo: Embeddings from Language Models

Deep contextualized word representations – Peters et al., 2018



Deep contextualized word representations

Matthew E. Peters[†], **Mark Neumann**[†], **Mohit Iyyer**[†], **Matt Gardner**[†],
{matthewp,markn,mohiti,mattg}@allenai.org

Christopher Clark[*], **Kenton Lee**[*], **Luke Zettlemoyer**[†*]
{csquared,kentonl,lsz}@cs.washington.edu

[†]Allen Institute for Artificial Intelligence
[*]Paul G. Allen School of Computer Science & Engineering, University of Washington

Abstract

We introduce a new type of *deep contextualized* word representation that models both (1) complex characteristics of word use (e.g., syntax and semantics), and (2) how these uses vary across linguistic contexts (i.e., to model polysemy). Our word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-trained on a large text corpus. We show that these representations can be easily added to existing models and significantly improve the state of the art across six challenging NLP problems, including question answering, textual entailment and sentiment analysis. We also present an analysis showing that exposing the deep internals of the pre-trained network is crucial, allowing downstream models to mix different types of semi-supervision signals.

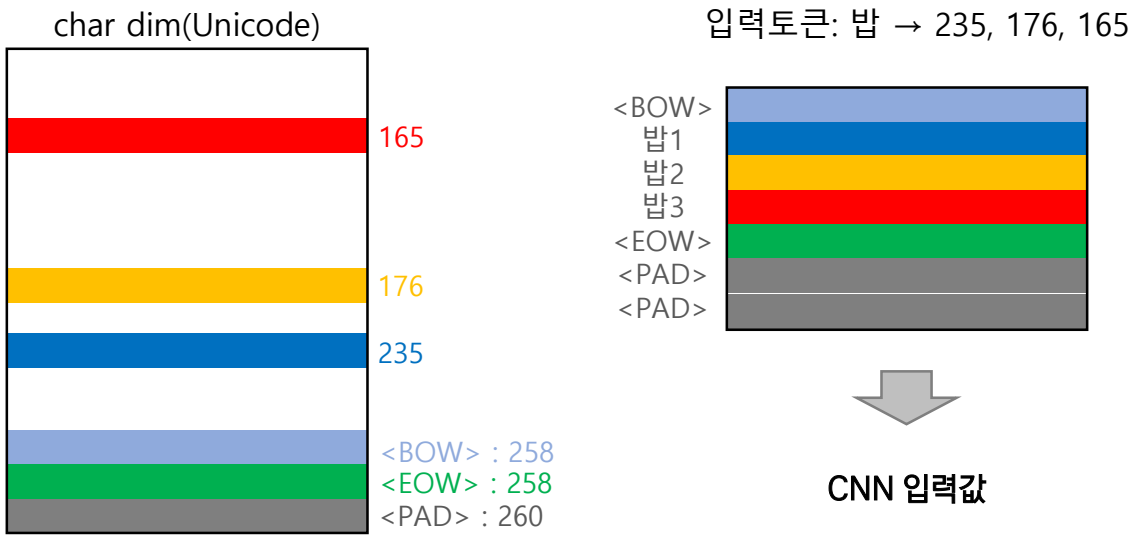
guage model (LM) objective on a large text corpus. For this reason, we call them ELMo (Embeddings from Language Models) representations. Unlike previous approaches for learning contextualized word vectors (Peters et al., 2017; McCann et al., 2017), ELMo representations are deep, in the sense that they are a function of all of the internal layers of the biLM. More specifically, we learn a linear combination of the vectors stacked above each input word for each end task, which markedly improves performance over just using the top LSTM layer.

Combining the internal states in this manner allows for very rich word representations. Using intrinsic evaluations, we show that the higher-level LSTM states capture context-dependent aspects of word meaning (e.g., they can be used without modification to perform well on supervised

- 2018년 발표된 문장 임베딩 기법
- 문맥 반영(contextualized) 임베딩을 통해 다의어 처리가 가능
  - Characteristics of word use와
  - 문맥 별 word usage의 상이함(다의어 등)을 모두 고려
  - river “bank”와 “bank” account를 다르게 임베딩
- Computer vision 분야의 transfer learning을 nlp에 적용
- Six challenging NLP tasks에 대해 최고 성능을 뛰어넘음
  - Q&A: SQuAD 데이터셋 사용, acc 4.7% 향상
  - Textual Entailment(의미추론): acc 0.7% 향상
  - Semantic Role Labelling: acc 3.2% 향상
  - Conference Resolution: acc 3.2% 향상
  - Named Entity Recognition: acc 3.2% 향상
  - Sentiment Analysis: 3.3% 향상

1단계: pre-training 단계 학습

1-1. 문자 단위 컨볼루션 레이어

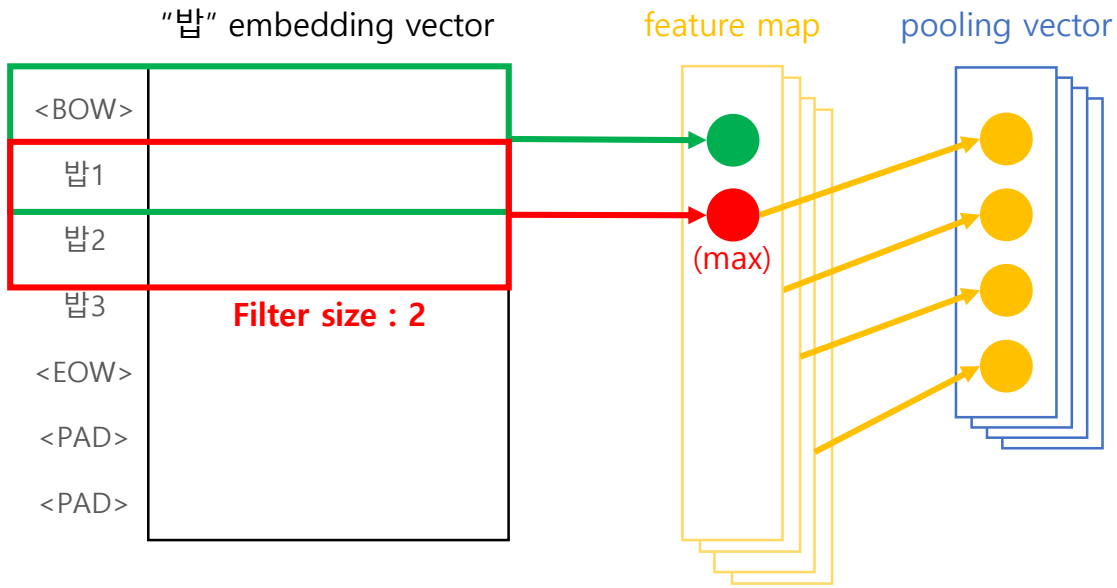


ELMo의 CNN 입력값 생성

- 문자 단위 컨볼루션 레이어는 각 단어 내 문자들 사이의 의미적·문법적 관계를 도출
  - Elmo의 입력 단위는 문자(해당 문자에 대응하는 유니코드)임
  - 〈BOW〉 + 〈unicode〉 + … + 〈 unicode 〉 + 〈EOW〉 + 〈PAD〉 + …

1단계: pre-training 단계 학습

1-1. 문자 단위 convolution layer



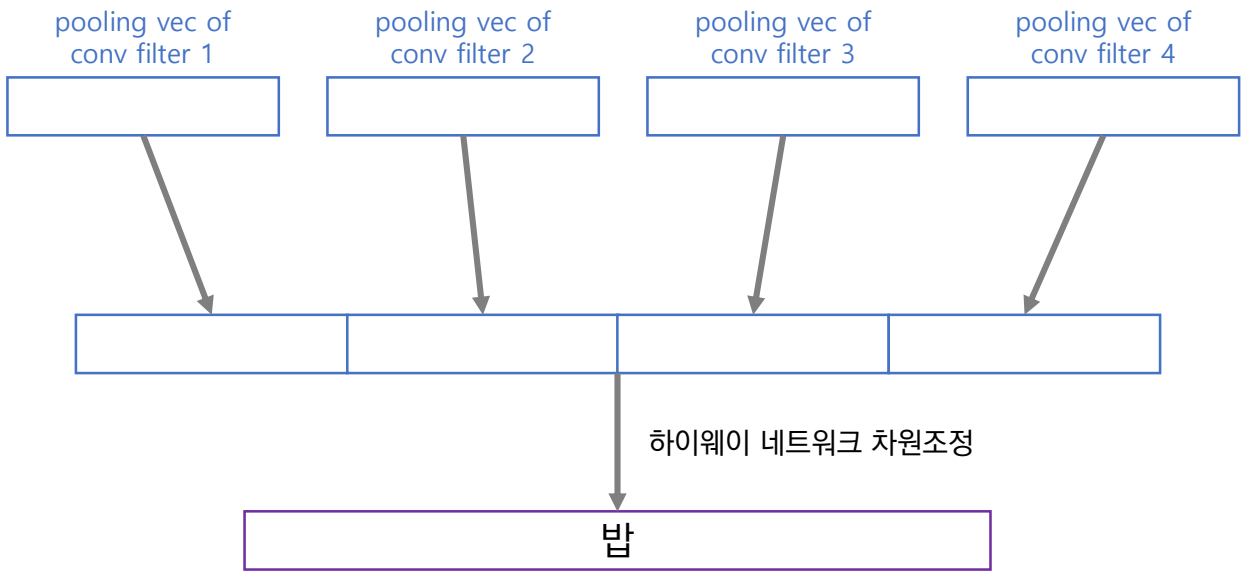
ELMo의 CNN 입력값 생성

- 문자 단위 convolution layer는 각 단어 내 문자들 사이의 의미적·문법적 관계를 도출
  - Elmo의 입력 단위는 문자(해당 문자에 대응하는 유니코드)임
  - 〈BOW〉 + 〈unicode〉 + … + 〈 unicode 〉 + 〈EOW〉 + 〈PAD〉 + …
- 생성된 단어 벡터는 크기 n의 convolution filter을 거쳐 feature map 생성
- Feature map상 가장 큰 값을 pooling vector 인자로 선정
- 사용자 지정 크기만큼 feature map 생성
  (feature map 개수 = pooling vector 크기)

5.4 ELMo: Embeddings from Language Models

1단계: pre-training 단계 학습

1-1. 문자 단위 convolution layer

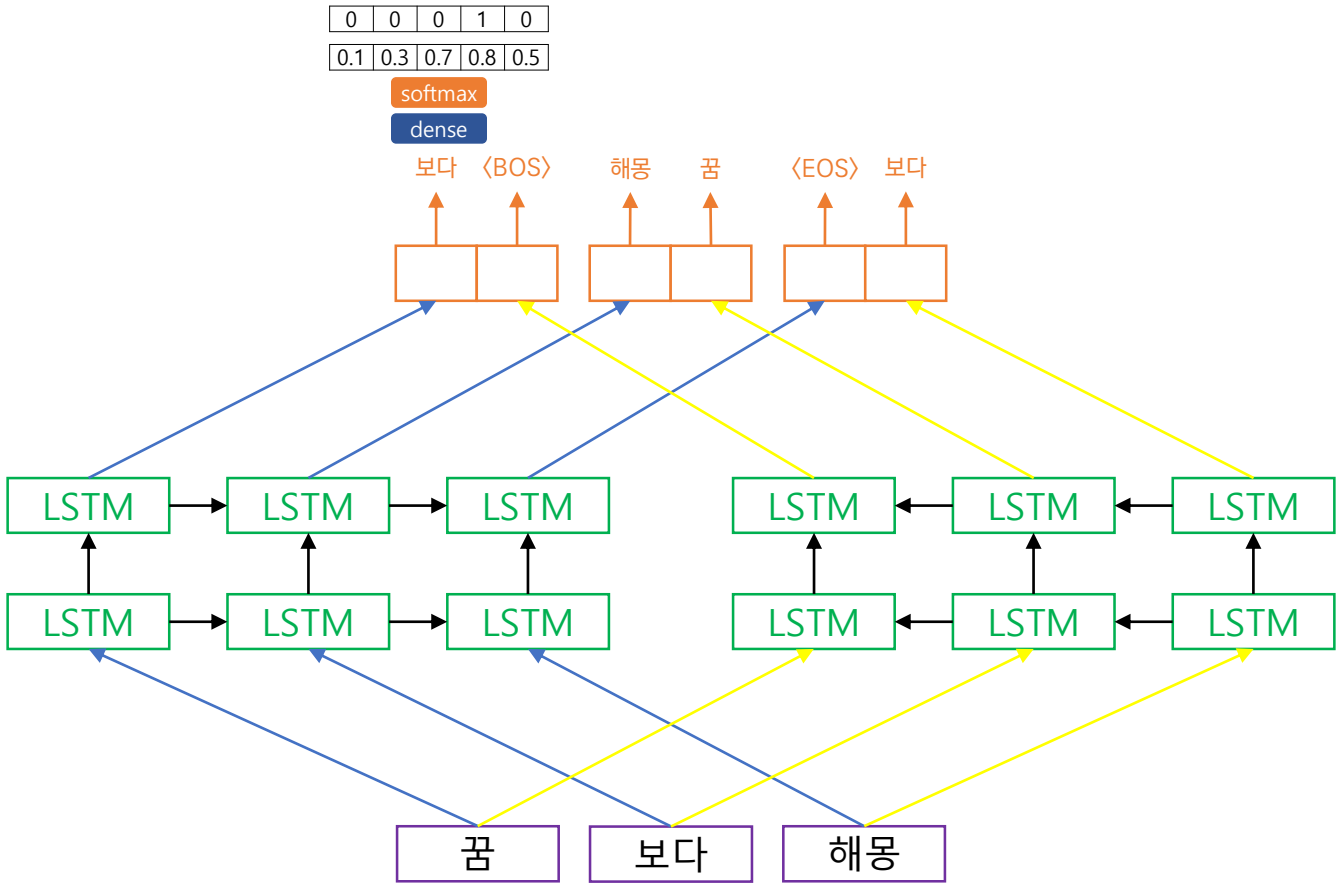


ELMo 양방향 LSTM 입력값 생성

- 문자 단위 convolution layer는 각 단어 내 문자들 사이의 의미적·문법적 관계를 도출
  - Elmo의 입력 단위는 문자(해당 문자에 대응하는 유니코드)임
  - 〈BOW〉 + 〈unicode〉 + … + 〈 unicode 〉 + 〈EOW〉 + 〈PAD〉 + …
- 생성된 단어 벡터는 크기 n의 convolution filter을 거쳐 feature map 생성
- Feature map상 가장 큰 값을 pooling vector 인자로 선정
- 사용자 지정 크기만큼 feature map 생성
  (feature map 개수 = pooling vector 크기)
- 생성된 pooling vector들을 concat하여 양방향 LSTM의 입력값 완성

1단계: pre-training 단계 학습

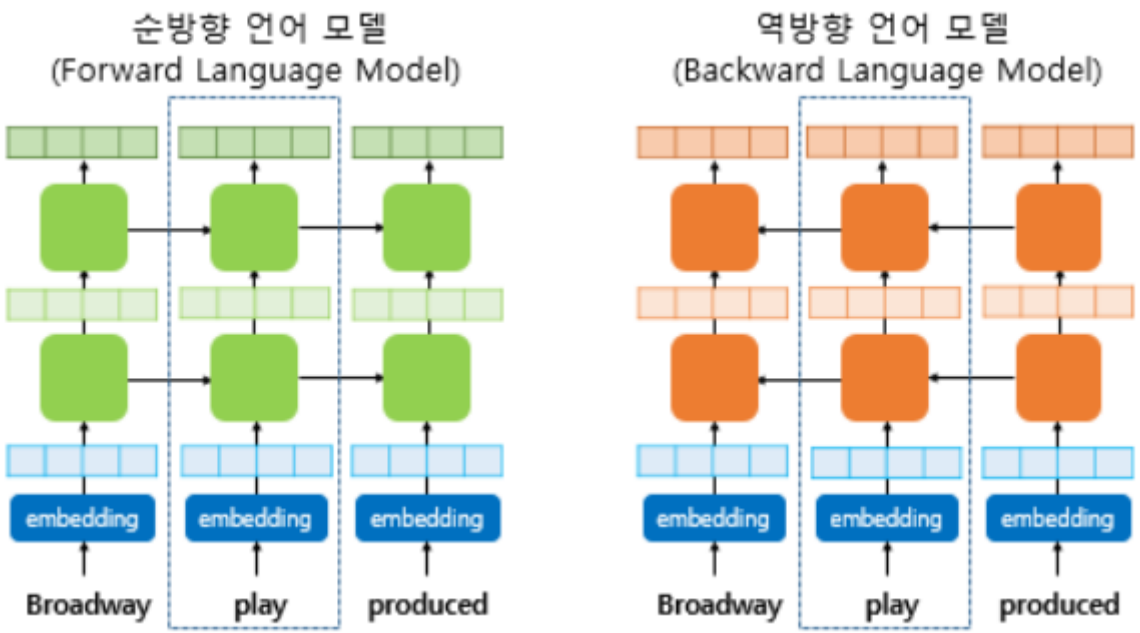
1-2. Bidirectional Language Model



- 문자 단위 CNN을 통해 뽑힌 단어 벡터들(=문장)은 2개의 LSTM Layer(BiLM)에 동일하게 입력값으로 활용됨
  - Forward LSTM Layer
  - Backward LSTM Layer
  - 〈BOS〉, 〈EOS〉토큰 활용 문장 임베딩
- Forward LSTM Layer의 경우 “꿈“을 통해 “보다“를, “해몽”을 통해 “〈EOS〉”를 예측
- Backward LSTM Layer의 경우 “꿈“을 통해 “〈BOS〉“를, “해몽”을 통해 “보다”를 예측
- 거대한 말뭉치를 단어 1개씩 슬라이딩해 가며 다음 단어를 예측하는 과정을 통해 문장 내 단어들 사이의 의미적 · 문법적 관계를 도출 가능함
- 출력 히든 벡터에 softmax를 적용한 확률 백터와, 정답 단어 인덱스만 1인 one-hot-vector 사이의 cross entropy를 최소화하며 학습 진행

2단계: downstream task 단계 학습

2-1. ELMo Layer



출처: https://wikidocs.net/33930

- 문자 단위 CNN을 통해 뽑힌 단어 벡터들(=문장)은 BiLM의 LSTM Layer 각각에 동일한 입력값으로 활용. BiLM 각 층의 산출값은 ELMo Layer 입력값으로 활용.

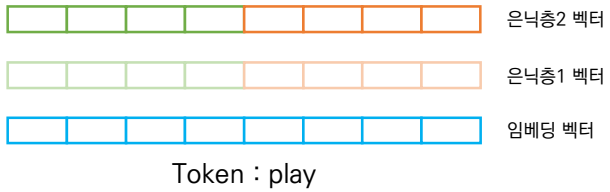
- BiLM 2개의 Layer는 1개의 임베딩층과 2개의 LSTM 은닉층으로 구성되어 있음(default). 각 단어의 층(임베딩층, 은닉층, 은닉층)별 신출값을 연결하여 ELMo Layer의 입력값으로 사용
  - 양방향 RNN은 이전 층의 출력값을 합친 후 다음 층의 입력값으로 보냄
  - BiLM은 1) 학습시 각 신경망은 독립적으로 학습 진행
    2) 학습 후 ELMo Layer에 사용키 위해 concatenate

- 이러한 정보흐름의 직관적 아이디어는 각 층의 출력값이 가진 정보는 서로 다른 정보일 것이며, 이를 모두 활용한다는 것

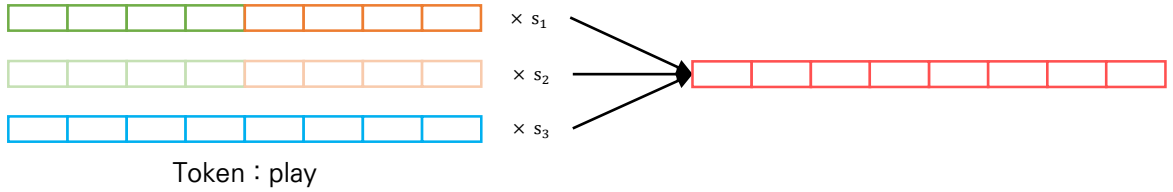
2단계: downstream task 단계 학습

2-1. ELMo Layer

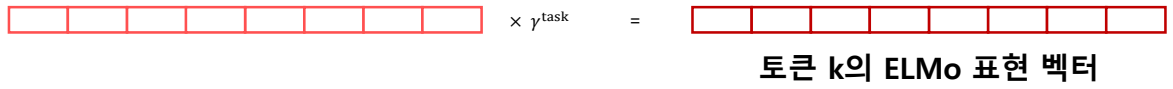
1단계: 각 층의 출력값을 연결(concatenate) → $h_{k,j}^{LM}$



2단계: 각 층의 출력값 별로 가중치 부여 → s_j^{task} , 그리고 sum



3단계: 벡터의 크기를 결정하는 스칼라 매개변수를 곱연산 → γ^{task}



- 1단계 : k번째 토큰의 각 LSTM Layer의 j번째 층 산출값을 연결
- 2단계: 현재의 downstream task에서 j번째 Layer의 중요도를 의미하는 스칼라 가중치 s_j^{task}를 j번째 층마다 곱연산
  - 그리고 각 층의 벡터를 합침(sum)
  - 각 층의 가중치 s_j^{task}는 downstream task 학습 과정에서 loss 최소화 방향으로 업데이트되는 학습 파라미터
- 3단계: 현재 학습중인 특정 downstream task의 중요도 γ를 곱해 ELMo 표현(representation) 벡터가 최종적으로 산출됨
  - γ 역시 downstream task 훈련 과정에서 업데이트
- 문자 단위 convolution layer, BiLM은 대규모 corpus를 활용한 pretrain 단계에서 학습, ELMO Layer은 특정 테스크를 수행하는 fine tuning 단계에서 학습됨(6장 상세).