Layer Normalized LSTM CRF를 이용한 한국어 의미역 결정

박광현, 나승훈
전북대학교
khpark231@gmail.com, nash@jbnu.ac.kr

Layer Normalized LSTM CRFs for Korean Semantic Role Labeling

Kwang-Hyeon Park, Seung-Hoon Na
Chonbuk National University

요 약

딥러닝은 모델이 복잡해질수록 Train 시간이 오래 걸리는 작업이다. Layer Normalization은 Train 시간을 줄이고, layer를 정규화 함으로써 성능을 개선할 수 있는 방법이다. 본 논문에서는 한국어 의미역 결정을 위해 Layer Normalization이 적용 된 Bidirectional LSTM CRF 모델을 제안한다. 실험 결과, Layer Normalization이 적용 된 Bidirectional LSTM CRF 모델은 한국어 의미역 결정 논항 인식 및 분류(AIC)에서 성능을 개선시켰다.

1. 서 론

딥러닝이 뛰어난 성능을 보인다는 것은 증명이 되었지만, 여전히 모델이 복잡해질수록 Train 시간이 오래 걸리는 작업이다.

Train 속도를 높이는 방법에는 learning rate를 높이는 방법이 있지만, Gradient vanishing 혹은 Gradient exploding 문제를 야기한다.

learning rate의 값이 벗어나지 않을 정도로 크면 속도가 향상되기 때문에, Gradient vanishing 혹은 Gradient exploding 문제가 발생하지 않으면서 learning rate 값을 크게 설정할 수 있는 모델을 design 할 수 있도록 Internal covariate shift 개념을 이용한 Batch Normalization방법이 소개 되었다[1].

그러나 Mini-batch의 mean/variance를 이용하는 Batch Normalization의 효과는 mini-batch 크기에 따라 다르며, Train할 때 와 Test할때의 계산량이 변하는 문제가 발생하는데, 이를 해결한 방법으로 Layer Normalization 방법이 소개 되었다[2].

본 논문에서는 한국어 의미역 결정을 Layer Normalization이 적용 된 Bidirectional LSTM CRF를 이용해 성능이 개선 됨을 보인다.

2. 관련 연구

텍스트의 의미를 이해하는 것은 기계번역, 정보 추출, 정서 감지, 요약 등과 같은 많은 실제 응용 프로그램에서 중요한 역할을 한다. 의미론적 역할 레이블(SRL)은 각 동사의 의미론적 역할을 할당하는 중요한 NLP 작업이다. PropBank 및 FrameNet과 같은 리소스의 출현으로 SRL은 상당한 발전을 이루었지만, 여전히 SRL 시스템은 많은 작업과 사전 지식이 필요하였다. 이를 해결하기 위해 수작업으로 feature를 만드는 대신, 깊은 신경망을 이용해 자동으로 feature를 학습하는 방법에 대해서 많은 연구가 이루어 지고 있다[3-5]. 딥러닝 기법 중 순차 입력열에 특화되어 있는 LSTM기반 방법에 출력 노드간의 의존성을 모델링 하는 CRF를 결합한 방식인 LSTM CRF는 품사태깅, 개체명 인식 등에서 가장 우수한 성능을 보여주고 있다[8-9].

의미역 결정에서 FFNN(Feed Forward Neural Network)을 이용해 feature를 설계하는데 많은 시간과 노력이 들어가는 기계학습과 비교해 비슷한 성능을 보인다는 연구 결과를 보였고[3], [4]는 술어-논항 사이의 의존 관계 정보를 포함하고 있기 때문에 성능 향상에 큰 도움이 되는 구문 분석 정보 없이 Bidirectional LSTM CRF를 성능을 개선 시켰고, [5]는 Bidirectional LSTM으로 구성된 hidden layer를 한층 더 쌓은 Stacked Bidirectional LSTM-CRFs를 이용해 성능이 개선 됨을 보였다. 이 밖에, [6]은 동일 어휘임에도 분석 결과가 달라지는 경우를 해결하기 위해 의미역을 결정하는데 중요한 역할을 하는 술어의 의미 정보를 보다 명확하게 하기 위해 FrameNet의 의미 그룹 정보와 PropBank의 predicate senses 정보를 함께 사용하여 성능이 개선 됨을 보였고, [11]에서는 문장 구조가 달라도 동일 의미 정보를 지니는 경우가 있기 때문에 구문정보만으로는 한계가 있어 이를 해결 하기 위해 능동태와 수동태 정보, 자동사와 타동사 정보 등을 자질로 추가하여 성능이 개선 됨을 보였다. 또한, [7]에서는 서술어와 논항 사이의 dependency path를 이용해 성능이 개선 됨을 보였고, [12]에서는 의미 정보를 활용하는 방안으로 동형이의어 수준의 의미 애매성 해소, 고유 명사에 대한 개체명 인식 등의 정보를 사용하여 성능이 개선 됨을

보였다.

3. Layer Normalizaed LSTM CRF를 이용한 한국어 의미역 결정

그림 1은 본 논문에서 제안하는 Layer Normalized LSTM CRF기반 의미역 태깅의 뉴럴 모델을 도식화하여 보여준다.

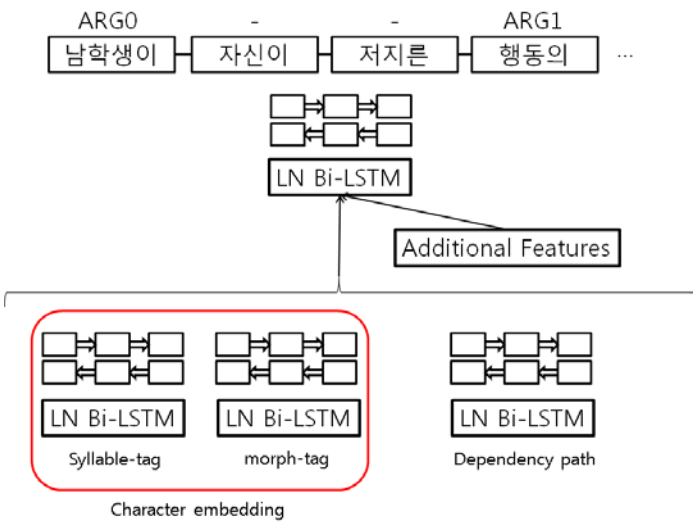

Figure 1. Layer Normalized LSTM CRF를 이용한 의미역 결정을 위한 뉴럴 모델 구조

그림 1에서 보다시피, 각 단어마다 LSTM의 입력표상 representation)이 정의되는데, 이 때 입력표상은 1) 문자 LSTM기반 단어표상 (word representation), 2) Dependency path, 3) additional features로 구성된다.

3.1 Batch Normalization & Layer Normalization

현재 layer의 입력은 이전 layer들의 변화에 영향을 받게 되는데 이전 layer의 파라미터 변화로 인해 현재 layer의 입력분포가 변하는 현상을 Covariate shift라고 한다. Covariate shift를 줄이는 방법 중 하나는 입력을 mean 0, variance 1로 바꿔주는 것(Whitening)이다. 하지만 전체 데이터를 기준으로 mean/variance를 학습시마다 계산하면 계산량이 많이 필요한데, 이때 나온 방법이 Batch Normalization 이다. Batch Normalization은 신경망에서 학습시 평균과 분산을 조정하여 Covariate shift를 줄이는 방법이다. 그림 2는 Batch Normalization 방법을 보여준다.

Input: mini-batch에 대한 입력값 x:
$B = \{x_{1...m}\}$; 및 학습 할 parameters γ, β
Output: $\{y_i = BN_{\gamma,\beta}(x_i)\}$

$\mu_B \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$ //mini-batch mean

${\sigma_B}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$ //mini-batch variance

$\hat{x_i} \leftarrow \frac{x_i - \mu_B}{\sqrt{{\sigma_B}^2 + \varepsilon}}$ //normalize

$y_i \leftarrow \gamma\hat{x_i} + \beta \equiv BN_{\gamma,\beta}(x_i)$ //scale 및 shift

Figure 2. Batch Normalizing Transform[1]

Batch Normalization은 전체 데이터에 대해 mean/variance를 계산하는 대신 지금 계산하고 있는 mini-batch에 대해서만 mean/variance를 구한 다음 inference를 할 때에는 data 전체에 대해서 mean/variance를 계산한 후, 정규화를 시키게 된다. 이 정규화가 입출력 값의 범위를 제한할 수 있기 때문에 linear transform을 사용하는데 이 transform에 있는 scale과 shift 파라미터 γ, β를 학습하면서 더욱 정교해지게 된다.

Batch Normalization을 사용함으로써, 더 큰 learning rate를 사용하여 학습속도를 향상시키고, covariate shift문제를 줄이고, 더 큰 weight가 더 작은 gradient를 유도하기 때문에 parameter growth가 안정화 되는 효과를 볼 수 있다. 하지만, batch size에 따라 Batch Normalization의 효과가 변하고, Train할때와 Test할때의 계산량이 다르다는 문제가 생기는데, 이를 해결한 방법이 Layer Normalization이다.

그림 3은 Layer Normalization 방법을 보여준다.

$$\mu = \frac{1}{H}\sum_{i=1}^{H} z_i \quad (1)$$

$$\sigma = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(z_i - \mu)^2} \quad (2)$$

$$LN(\mathbf{z};\boldsymbol{\alpha},\boldsymbol{\beta}) = \frac{(\mathbf{z}-\mu)}{\sigma} \odot \boldsymbol{\alpha} + \boldsymbol{\beta} \quad (3)$$

$$\mathbf{f}_t = LN(\mathbf{W}_{xh_f}\mathbf{x}_t) + LN(\mathbf{W}_{hh_f}\mathbf{h}_{t-1}) + \mathrm{b}_{h_f} \quad (4)$$

$$\mathbf{i}_t = LN(\mathbf{W}_{xh_i}\mathbf{x}_t) + LN(\mathbf{W}_{hh_i}\mathbf{h}_{t-1}) + \mathrm{b}_{h_i} \quad (5)$$

$$\mathbf{o}_t = LN(\mathbf{W}_{xh_o}\mathbf{x}_t) + LN(\mathbf{W}_{hh_o}\mathbf{h}_{t-1}) + \mathrm{b}_{h_o} \quad (6)$$

$$\mathbf{g}_t = LN(\mathbf{W}_{xh_g}\mathbf{x}_t) + LN\left(\mathbf{W}_{hh_g}\mathbf{h}_{t-1}\right) + \mathrm{b}_{h_g} \quad (7)$$

$$\mathbf{c}_t = \sigma(\mathbf{f}_t) \odot \mathbf{c}_{t-1} + \sigma(\mathbf{i}_t) \odot \tanh(\mathbf{g}_t) \quad (8)$$

$$\mathbf{h}_t = \sigma(\mathbf{o}_t) \odot \tanh(\mathbf{c}_t) \quad (9)$$

위 식(1) 과 식(2)는 Layer Normalization에서 mean과 variance를 구하는 식이며 z_i 는 벡터 $\mathbf{z}$ 의 i 번째 원소를 나타낸다. Batch Normalization에서는 batch size인 m이 사용되었지만, Layer Normalization에서는 한 layer 에서의 hidden units수를 나타내는 H가 사용 되었다. Batch Normalization과 달리 Layer Normalization mini-batch 크기에 어떠한 제약도 받지 않는다. 또한, RNN에서 Batch Normalization을 적용하면 sequence 각 시간 단계에 대해 별도의 statistics를 계산하고 저장해야 하고, Test sequence가 Train sequence보다 긴 경우 문제가 발생하는데 Layer Normalization은 정규화 조건이 현재 시간 단계에서 Layer에 대한 합계 입력에만 의존하기 때문에 이러한 문제가 없다.

식(3)-(9)는 LSTM에서의 Layer Normalization 수식을 나타내는데, 식(3)에서 gain $\boldsymbol{\alpha}$와 bias $\boldsymbol{\beta}$는 scale과 shift를 위한 파라미터로, batch normalization과 마찬가지로 non-linearity 이전에 적용되며, $\boldsymbol{\alpha}$는 1, $\boldsymbol{\beta}$는 0으로 초기화 하였다.

$\odot$는 두 벡터 사이의 element-wise 곱셈을 나타내며, 식(4)-(8)에서 $\mathbf{W}_{hh}$ 는 hidden 과 hidden 사이의 가중치,

$\mathbf{W}_{xh}$ 는 x와 hidden 사이의 가중치를 나타내고 식(4)-(8)의 LN 함수에는 생략되었지만 각각 gains α_i 와 biases β_i 파라미터도 포함되어 있으며, 식(8)-(9)의 σ 는 sigmoid 함수를 의미한다.

3.2 입력

본 논문에서는 문자 LSTM기반 단어표상, dependency path, additional feature를 사용하였다. 문자 LSTM기반 단어표상으로는 형태소-태그 단위 문자, 음절-태그 단위 문자를 사용하였다. 형태소-태그 단위 문자는 단어와 품사태그를 합친 프랑스/NNP 형태로 구성 하였고 음절-태그 단위 문자는 띄어쓰기 정보를 활용하기 위해 프/B-NNP 랑/I-NNP 스/I-NNP 형태로 구성하였다. 이 문자들로부터 입력 단어 표상을 얻기 위해 형태소-태그 단위 문자 와 음절-태그 단위 문자 각각을 Bi-LSTM을 적용하여 마지막 상태 벡터를 결합한 후 MLP를 적용하여 입력 단어 표상을 얻어내었다.

Dependency path는 서술어-논항 사이의 dependency 관계로 "부시 검사는 남학생이 자신이 저지른 행동의 중대성을 인식하지 못하고 있는 것 같다고 말했다." 에서 서술어 "말했다." 와 논항 "같다고" 사이의 dependency path는 [quot, aux] 가 된다. Dependency path의 표상은 Bi-LSTM을 적용 하고, 마지막 상태 벡터와 입력 단어 표상과 결합한 뒤 MLP를 적용하여 사용하였다. 다음 그림 5는 dependency path의 예시를 보여준다.

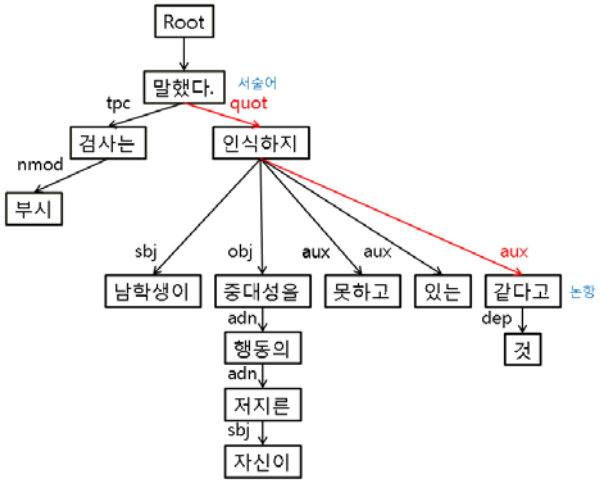

Figure 5. 말했다(서술어)와 같다고(논항) 사이의 dependency path

Additional feautre는 어절태그, 서술어의 어휘 및 품사정보, 서술어와 현재 어절 사이의 위치 정보, head의 위치 가 사용되었다. 다음 표 1은 additional feautre의 예시를 보여준다.

Table 1. Additional feature

Feature	Example
어절태그	남학생/NNG|이/JKS ➔ NNG~JKS
서술어의 어휘 및 품사정보	인식/NNG|하/XSV|지/EC ➔ 인식/NNC
서술어와 현재 어절 사이의 위치 정보	문장 "자신이 저지른 행동의 중대성을 인식하지" 에서 서술어 "인식하지" 와 현재 어절 "자신이" 의 위치 정보는 PREV 4 가 된다.
head의 위치	tree상의 parent(head가 문장내에 없으면 root로 표시)

3.2 출력

문자들의 임베딩 벡터(character embedding vector)와 자질 임베딩 벡터(feature embedding vector)를 결합하여 Layer Normalized Bidirectional LSTM을 적용하여 은닉 상태 벡터를 구하고, 출력층에 전달되어 각 태그별로 확률을 계산하게 되는데 출력층의 인접 노드들간의 의존성 모델링을 위해 CRF를 추가하였다.

4. 실험

4.1 실험 셋팅

본 논문에서는 의미역 결정 평가를 위해 Korean Propbank의 Newswire 말뭉치만을 사용하였고, Tree 구조의 말뭉치를 변환하는 도중 말뭉치에 오류가 있거나, 변환에 실패하는 문장은 제외 하였다.

전체 문장 23659 문장 중 20110 문장은 학습데이터로, 1183 문장은 개발셋, 2366 문장은 평가셋으로 사용하였다.

사용 된 의미역의 수는 None label 'O'를 포함하여 27개로 다음 표 2는 사용 된 의미역 태그를 나타낸다.

Table 2. 의미역 태그

labels	ARG0, ARG1, ARG2, ARG3, ARG5, ARGA, ARGM, ARGM-TMP, ARGM-LOC, ARGM-EXT, ARGM-CAU, AUX, ARGM-DIS, ARGM-INS, ARGM-MNR, ARGM-PRD, ARGM-ADV, ARGM-PRP, ARGM-CND, ARGM-DIR, ARG0-INS, ARGM-NEG, ARG0-DIS, AUX-DIS, ARG1-EXT, ARG0-TMP, O

4.1 실험 결과

Layer Normalization이 적용 된 LSTM CRF의 성능 향상을 알아보기 위해 LSTM CRF모델과 비교 하였다.

다음 표 3은 기존의 연구 결과와 LN Bi-LSTM CRF 모델의

실험 결과를 나타낸다. 표 3의 실험결과는 기존연구와 평가셋이 다르기 때문에 완전하지 않은 비교이다.

Table 3. 실험 결과 (AIC, F1)

	Dev	Test
Bi-LSTM CRF Model[4]		78.17%
Stacked Bidirectional LSTM-CRF Model[5]		78.57%
Our Bi-LSTM CRF Model	79.98%	77.86%
Our LN Bi-LSTM CRF Model	80.55%	78.46%

LN: Layer Normalization

표 4에 나온 결과는 F1으로 실험한 결과 중 제일 높게 나온 성능을 표기 하였다. Layer Normalization이 성능에 영향을 주는지 더욱 정밀하게 확인 해 보기 위해, 파라미터를 동일하게 하고 실험 한 5번의 결과의 평균을 낸 결과는 다음과 같다.

Table 4. 실험 결과 평균 (AIC, F1)

	Dev	Test
Bi-LSTM CRF Model	80.14%	77.57%
LN Bi-LSTM CRF Model	80.56%	78.10%

5. 결론

본 논문에서는 Layer Normalized LSTM CRF를 이용해 Layer Normalization이 의미역 결정 성능 향상에 도움이 되는지 실험 하였다. 실험 결과, 제안 방법을 이용한 Layer Normalized LSTM CRF모델은 한국어 의미역 결정 테스크상에서 성능이 개선됨을 보였다.

향후 연구로는 의미역 결정에 Attention mechanism을 적용하여 Attention mechanism이 의미역 결정에서 성능을 개선 시킬 수 있는지 평가 하고자 한다.

6. 참고문헌

[1] Sergey Ioffe, Christian Szegedy, Accelerating Deep Network Training by Reducing Internal Covariate Shift, 2015, arXiv

[2] Jimmy Lei Ba, Jamie Ryan Kiros, Geoffrey E. Hinton, Layer Normalization, 2016, arXiv

[3] 배장성, 이창기, 임수종, 딥 러닝을 이용한 한국어 의미역 결정, 2015, KCC

[4] 배장성, 이창기, Bidirectional LSTM CRF를 이용한 End-to-end 한국어 의미역 결정, 2015, KIISE

[5] 배장성, 이창기, Stacked Bidirectional LSTM-CRFs를 이용한 한국어 의미역 결정, 2017, KIISE

[6] 박태호, 차정원, 형태 의미 정보를 이용한 한국어 의미역 결정, 2017, KCC

[7] Michael Roth, Mirella Lapata, Neural Semantic Role Labeling with Dependency Path Embeddings, 2016, arXiv

[8] 나승훈, 민진우, 문자 기반 LSTM CRF를 이용한 개체명 인식, 2016, KCC

[9] 민진우, 오효정, 나승훈, 식품 도메인 개체명 인식을 위한 문자 기반 LSTM CRF, 2016, KCC

[10] 박광현, 나승훈, 문자 기반 LSTM CRF를 이용한 한국어 의미역 결정, 2017, KCC

[11] 박태호, 차정원, CRFs 기반의 한국어 의미역 부착 성능 향상을 위한 자질 선택, 2016, 정보과학회지

[12] 임수종, 임준호, 이충희, 김현기, 의미 프레임과 유의어 클러스터를 이용한 한국어 의미역 인식, 2016, 정보과학회논문지