ELMo : Deep contextualized word representations

Assign	Yujin Kim 상민 상민 이 유 유경 한 KM S 예진 문
Date	@2021년 3월 18일 → 2021년 3월 28일
Status	Completed

엘모 너무해!

1. Introduction

Word2Vec, Glove 등 사전 훈련된 언어 모델(pre-trained word representations)은 기존 NLU 모델에서 아주 중요했음. 하지만 구문, 언어별 문맥에 따른 다양한 의미를 모두 고려하기는 어려움.

ELMo

- 사전 훈련 + 문맥 = 문맥 반영 언어 모델(deep contextualized word representations)
- 입력된 전체 문장에 대해서 구성 단어들의 임베딩 제공 → bidirectional LSTM에서 만들어진 vector 사용

(1) biLM의 모든 internal layer 사용

- *linear combination*으로 생성 → LSTM top layer만 쓴 것보다 꽤 좋은 성능
- intrinsic evaluation으로 본 결과, 높은 LSTM층(higher-level)은 문맥에서 단어의 의미를 학습하게 되고, 낮은 층(low-level)은 단어의 문법적인 측면(POS 등)을

학습하게 된다.

(2) 기존 모델에 쉽게 추가될 수 있음

- 6개의 다양한 언어 이해 문제 해결 모델에 쉽게 추가됨
- ELMo만 추가 해도 오차가 20% 감소

(3) CoVe보다 좋은 성능

- Neural machine translation encoder를 사용하는 CoVe, 그것보다 좋은 성능을 가짐
- ELMo, CoVe 둘 다 deep representations가 LSTM top layer만 쓴 것보다 좋다는 걸 보여줌

2. Related work

(1) Context-independent representations

사전 훈련 모델(Word2Vec, GloVe)

- question answering, textual entailment, semantic role labeling 같은 대부분의 NLP에서 기본적으로 사용
- 하지만 각 단어에 하나의 context-independent representation만을 허용함 [*단점*]

Context2vec

- bidirectional LSTM을 사용하여 중심 단어를 기준으로 주변 문맥 encode

CoVe

- 중심 단어를 포함해서 supervised neural Machine Translation나 unsupervised 언어 모델을 사용
- 두 접근 방식 모두 큰 데이터에서 좋은 성능, 하지만 MT는 둘 이상의 언어 단어들이 여서 사이즈 제한이 있음

(2) LSTM layers

deep biRNNs

- 서로 다른 층은 서로 다른 특성을 임베딩한다
- deep LSTM의 낮은 층에 Part-Of-Speech(POS) tag 같은 문법적인 측면을 학습하면, 그걸 기반으로 단어 의존 구조(dependency parsing)를 파악하거나 결합 범주 문

법(CCG super tagging)으로 억양 등을 확인하는 높은 수준의 task 성능을 전체적으로 높일 수 있음

RNN 기반의 encoder-decoder MT

- 2개 층인 LSTM의 첫 번째 layer에서는 POS 예측이 더 효율적
- LSTM의 top layer에서는 문맥에서의 단어 의미를 학습

ELMo

- subword information으로 접근 or 단어별로 다른 vector를 학습하면서 (1)에서의 [*단점*]을 극복
- subword unit의 정보를 글자 단위의 convolution을 사용해서 학습 → 다양한 의미를 사전에 부여된 class로 train하지 않고 원활하게 합체
- (2)와 비슷하게 LSTM layers 모두 사용, downstream task(POS, context에서의 word sense 등 뭔가 좀 더 구체적인 task)를 학습하는데 좋음
- biLM을 라벨링 되지 않은 데이터로 pretrain 후 모델을 이용해서 task에 맞게 가중치 조정 = 다른 언어모델보다 작은 supervised model

3. ELMo: Embeddings from Language Models

ELMo는..

- 전체 문장을 input으로 받고 그에 대한 각 단어들의 representation 생산
- [3.1] 2층짜리 biLM들의 꼭대기에서 Character convolution 계산
- [3.2] word representations? 내부 layer들의 선형 결합(?).
- [3.4] 그래서 큰 사이즈로 biLM pretrain 시켰을 때 semi-supervised learning 가능
- [3.3] 이건 지금 있는 NLP랑 합체 쌉가능 ;)

3.1 Bidirectional language models

(1) Forward LM

N개의 순차적인 token이 주어졌을 때, 포워드 언어 모델(Forward LM)을 다음과 같이 계산

- $(t_1, \ldots, t_{k-1})$이 주어졌을 때 token t_k가 나타날 확률을 모두 곱해서 전체 입력 확률을 구함

$$p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_1, t_2, \ldots, t_{k-1})$$

- 문맥에 의존적이지 않은 k번째 단어의 representation을 x_k^{LM} 라고 하면, 이걸 forward LSTM의 L개 층에 통과시킴
- 입력 위치 = k, 현재 layer = $j(1, \ldots, L)$ 일 때, output? 문맥의존 representation, $\overrightarrow{h}_{k,j}^{LM}$
- 꼭대기층의 $\overrightarrow{h}_{k,L}^{LM}$ 은 Softmax layer를 사용해서 다음 token t_{k+1} 을 예측

(2) Backward LM

Forward LM과 비슷하지만, 반대로 흘러감(미래의 representation을 기반으로 현재 representation 유추)

$$p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_{k+1}, t_{k+2}, \ldots, t_N)$$

- $(t_{k+1}, \ldots, t_N)$이 주어졌을 때 token t_k에 대해서 $\overleftarrow{h}_{k,j}^{LM}$ 를 j층에서 output으로 만듦

(3) 합체

biLM은 forward & backward LM을 합침. 두 방향의 log likelihood를 최대화 시키는 방향으로 학습 진행

$$\sum_{k=1}^{N} \Big(\log p(t_k \mid t_1, \ldots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log p(t_k \mid t_{k+1}, \ldots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \Big)$$

- token representations = Θ_x
- Softmax layer = Θ_S
- Forward LSTM parameter = $\overrightarrow{\Theta}_{LSTM}$, Backward LSTM parameter = $\overleftarrow{\Theta}_{LSTM}$
- 두 방향 간 가중치를 공유함

3.2 ELMo

(노란색 배경 → 발표 자료에 꼭 넣어야 하는 핵심, 흰색 배경 → 설명을 돕기 위한 문장)

ELMo는 biLM에서 등장하는 중간 매체 layer의 표현들을 특별하게 합친 것이다. 각 토큰 tk에 대하여, L개의 layer인 BiLM은 2L+1개의 표현을 계산한다.

$$\begin{aligned} R_k &= \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\ &= \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\}, \end{aligned}$$

이 때, hLMk,0은 token layer를 뜻하고, hLMk,j=[h→LMk,j;h←LMk,j]는 biLSTM layer를 의미한다. 그래서 모든 layer에 존재하는 representation을 R인 single vector로 혼합하는 과정을 거친다:

$$\mathbf{ELMo}_k = E(R_k; \mathbf{\Theta}_e)$$

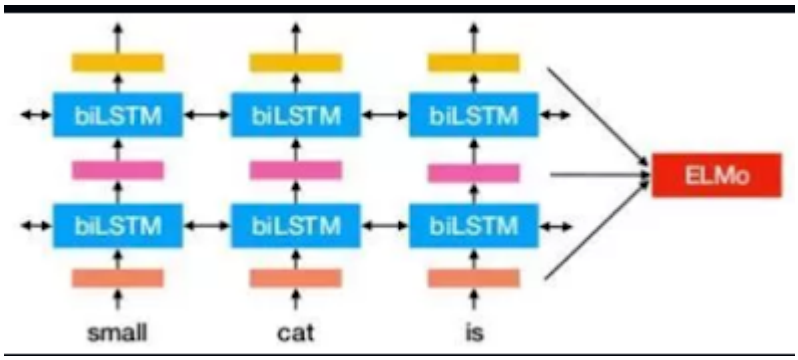
예시로, 가장 간단한 ELMo 버전은 가장 높은 layer를 취하는 방법이 있다:

E(Rk)=hLMk,L이 ELMo는 task에 맞게 또 변형될 수 있다.

$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}.$$

stask는 softmax-normalized weight를 의미, γtask는 task model을 전체 ELMo vector의 크기를 조절하는 역할을 맡는다.

위의 수식을 도식화를 통해 살펴보자면



해당 그림과 같이 각 Bi-LSTM Layer들을 통해 나오는 hidden representation을 task의 비율에 맞추어 더해 ELMo vector를 만드는 것이다.

즉 과정을 다시 정리해보면

(1) BiLSTM layer의 꼭대기층의 token이 softmax layer를 통해 다음 token을 예측하도록 훈련시킨다.

(2) 훈련된 BiLSTM layer에 input sentence를 넣고 각 layer의 representation 합을 가중치를 통해 합한다.

(3) input sentence length만큼 single vector가 생성된다.

3.3 Using biLMs for supervised NLP tasks

이렇게 pre-trained된 biLM과 NLP task를 위한 supervised architecture를 결합하여 task model의 성능을 향상시켰다. 이 때, 이 논문에서는 이 representation의 선형 결합을 다음과 같이 학습시켰다.

- 추가 개선점

(1) 먼저 biLM없이 supervised model의 가장 낮은 layer를 고려했다. 대부분의 supervised NLP model은 가장 낮은 층에서 공통적인 architecture를 공유한다. 따라서 이는 ELMo를 쉽게 붙일 수 있는 계기가 되었다. 주어진 sequence (t1,...tN)에 대해 pre-trained token representation인 xk를 사용했으며 때때로 문자 기반 representation을 사용하는 것이 NLP task의 대부분이었다. 이 때, bi-RNN이나 CNN, feed forward network를 통해 context에 의존적인 representation hk를 만드는 것이 NLP task에서 주로 하는 작업이였다.

(2) 따라서 supervised model에 ELMo를 부착하기 위해 biLM의 가중치값을 고정시키고 ELMo vector인 ELMotaskk를 token representation xk와 결합시켜 다음과 같은 [xk;ELMotaskk]representation을 생성하였다. 이 representation은 context-sensitive representation hk를 만들기 위한 input으로 사용된다.

(3) 마지막으로 ELMo에 dropout을 설정하는 것이 더 좋다는 것을 알았으며, 때때로 λ||w||22와 같은 regularization factor를 더하는 게 좋아 몇몇 케이스에서는 regularization을 생성하였다.

3.4 Pre-trained bidirectional language model architecture

- 언어 모델의 혼잡도(perplexity)와 모델의 크기 및 계산량을 고려하여 기존의 CNN-BIG-LSTM의 은닉 계층과 임베딩 가중치의 차원을 절반으로 축소하였다.
- 최종 모델은 4096 units, 512 차원사영과 첫번째 층에서 두번째 층으로 residual connection으로 이루어진 2개의 biLSTM 층이 있다.
- 그 이후 2048개의 character n-gram convolutional filter로 이루어진 층이 있어 총 3개의 층으로 구성된다.
- 사전 훈련이 진행된 이후에는 특정 영역의 데이터를 처리하기 위해 미세조정 될 수 있다.

→ Downstream task에서 미세조정된 biLM을 사용할 경우 상당한 혼잡도(perplexity)를 줄일 수 있다.

4. Evaluation

6개의 NLP task 밴치마크에 대하여 단지 ELMo를 추가하는 것만으로도 SOTA(최고성능) 달성

Task	Previous SOTA		Our baseline	ELMo + baseline	Increase (absolute/ relative)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

- “increase” 열은 baseline에 비해 절대적 – 상대적 향상을 나타냄.

1. **Question answering(질의응답)**
    - Dataset : Stanford Question Answering Dataset(SQuAD) : 10만개 이상의 위키피디아 질문-답 쌍을 크라우드 소싱한 데이터셋
    - Baseline : Bidirectional Attention Flow model
    - ELMo를 추가 : F1 4.7% 증가 (81.8 → 85.8) / 에러 9% 감소 / 단일 모델 SOTA 1.4% 향상 / CoVe를 추가한 것보다 1.8% 더 높은 성능을 보임
2. **Textual entailmen**t = 주어진 전제에서 가설이 참인지 판단하는 문제
    - Dataset : Stanford Natural Language Inference (SNLI) corpus : 55만개 이상의 가설-전제 쌍
    - BaseLine : ESIM sequence model
    - ELMo 추가 : 5개 앙상블 모델에서 평균적으로 7%의 정확도 향상 (89.3%)
3. **Semantic role labeling(의미역 결정)**
    - SRL system : "누가 누구에게 무엇을 했는지"에 대답하는 모델
    - Baseline : 8-layer deep biLSTM
    - ELMo 추가 : F1 3.2% 향상 (81.4% → 84.6%)
4. **Coreference resolution(상호참조)**

    : 텍스트 안에서 같은 real world entities를 가리키는 언급을 찾아내는 것

예) **한글은 조선의 4대** 왕인 **세종대왕**이 **창제한** 한국어**의** 문자이다. → 여기서 조선 4대 왕 = 세종대왕이 같은 것을 지칭한다는 것을 모델이 알 수 있도록!

- Baseline : end-to-end span-based neural model
- ELMo 추가 : F1 3.2% 향상(67.2 → 70.4)

5. **Named entity extraction(개체명 인식)**
    - Datasets : The CoNLL 2003 NER task
    - Baseline : pre-trained word embedding 사용 (Character-based CNN representation + 2개의 biLSTM layer와 CRF loss)

    ↔ 해당 모델은 가장 상위의 biLM 층의 가중치만 학습하는데 비해, ELMo는 모든 biLM 층의 가중치 평균을 학습한다는 점이 다름.

    - ELMo 추가 : F1 91.93% → 92.22%
6. **Sentiment analysis(감성 분석)**
    - Datasets : The fine-grained sentiment classification task in the Stanford Sentiment Tree bank(SST-5)
    - Baseline : Biattentive classification network (BCN)
    - CoVe를 ELMo로 대체 : 기존 모델에 비해 절대적인 정확도가 1.0% 향상

5. Analysis

5.1 Alternate layer weighting schemes

- Regularization parameter λ 의 역할

λ 매우 큰 경우 (λ = 1) ： 전체 weight function이 단순히 평균을 내는 역할을 수행

λ 값이 매우 작은 경우 (λ = 0.001) : 각 층에 대한 weight 값이 다양하게 적용됨

(the regularization parameter is also important, as large values such as = 1 effectively reduce the weighting function to a simple average over the layers, while smaller values (e.g., = 0.001) allow the layer weights to vary.)

Task	Baseline	Last Only	All layers	
			λ=1	λ=0.001
SQuAD	80.8	84.7	85.0	**85.2**
SNLI	88.1	89.1	89.3	**89.5**
SRL	81.6	84.1	84.6	**84.8**

[마지막 층의 결과만 사용(Last Only) Vs 모든 층의 결과 사용(All layers)]

1) 모든 층의 representation을 사용한 경우의 성능이 가장 우수함

2) λ 값을 작게 해서 서로 다른 각 층의 weight을 학습하도록 하는 것이 더 좋은 성능을 보임

- 대부분의 경우 더 작은 λ 값을 사용하는 것이 효과적

(NER 과 같이 작은 크기의 학습 데이터를 사용하는 경우 λ 크기에 큰 영향을 받지 않음)

5.2 Where to include ELMo?

현재 논문에서 사용한 모든 구조에서는 가장 낮은 층(lowest layer)의 입력에만 ELMo representation을 적용하지만, 특정 task에 대해서는 output 에도 ELMo representation을 추가하는 것이 성능을 향상시킬 수 있음

Task	Input Only	Input & Output	Output Only
SQuAD	85.1	**85.6**	84.8
SNLI	88.9	**89.5**	88.7
SRL	**84.7**	84.3	80.9

1) SQuAD와 SNLI의 경우, input과 output 모두에 ELMo representation을 추가했을 때 성능이 우수

-SNLI, SQuAD 모두 biRNN 이후에 Attention을 사용하기 때문에, output에 ELMo representation을 추가하는 것이 성능을 향상

2) SRL은 입력 계층에만 ELMo representation를 추가했을 때 가장 성능이 우수

5.3 What information is capture by the biLM's representations?

- biLM의 문맥을 고려한 representation을 추가

  - 다양한 NLP task에 유용한 정보를 표현 가능 → 문맥을 고려한 representation은 특정한 단어를 사용되는 문맥에 따라서 서로 다르게 표현함

	Source	Nearest Neighbors
GloVe	play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular play on Alusik 's grounder {...}	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway play for Garson {...}	{...} they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

Table 4: Nearest neighbors to "play" using GloVe and the context embeddings from a biLM.

[문맥을 고려하지 않는 단어 임베딩인 GloVe와 ELMo를 비교]

1) GloVe :

- play라는 단어에 고정된 의미를 부여 → 유사한 단어에 한정적인 표현만 제공

(단어의 representation이 고정되어 있기 때문에 play처럼 여러 뜻을 갖고 있는 단어의 경우, 여러 의미를 단어 representation에 담기 어려움)

2) ELMo :

-"어떠한 역할을 수행한다." 는 의미의 play와 "희곡/연극" 의 의미의 play를 구분

(context에 따라 얻어지는 "play"에 대한 벡터가 다름)

- WSD (Word-Sense Disambiguation) task에 따른 performance

Model	F_1
WordNet 1st Sense Baseline	65.9
Raganato et al. (2017a)	69.9
Iacobacci et al. (2016)	**70.1**
CoVe, First Layer	59.4
CoVe, Second Layer	*64.7*
biLM, First layer	67.4
biLM, Second layer	*69.0*

1) BiLM이 CoVe biLSTM보다 성능이 우수함

2) BiLM의 first layer를 이용하는 것보다 Second layer를 이용하는 것이 성능이 우수

→ 단어 중의성 해소에 더 적합

→ 두번째 layer를 통해 얻어진 벡터들이 semantic한 정보를 잘 알아냄

- POS(Part-Of-Speech) task로 token의 품사를 확인하는 task

Model	Acc.
Collobert et al. (2011)	97.3
Ma and Hovy (2016)	97.6
Ling et al. (2015)	**97.8**
CoVe, First Layer	*93.3*
CoVe, Second Layer	92.8
biLM, First Layer	*97.3*
biLM, Second Layer	96.8

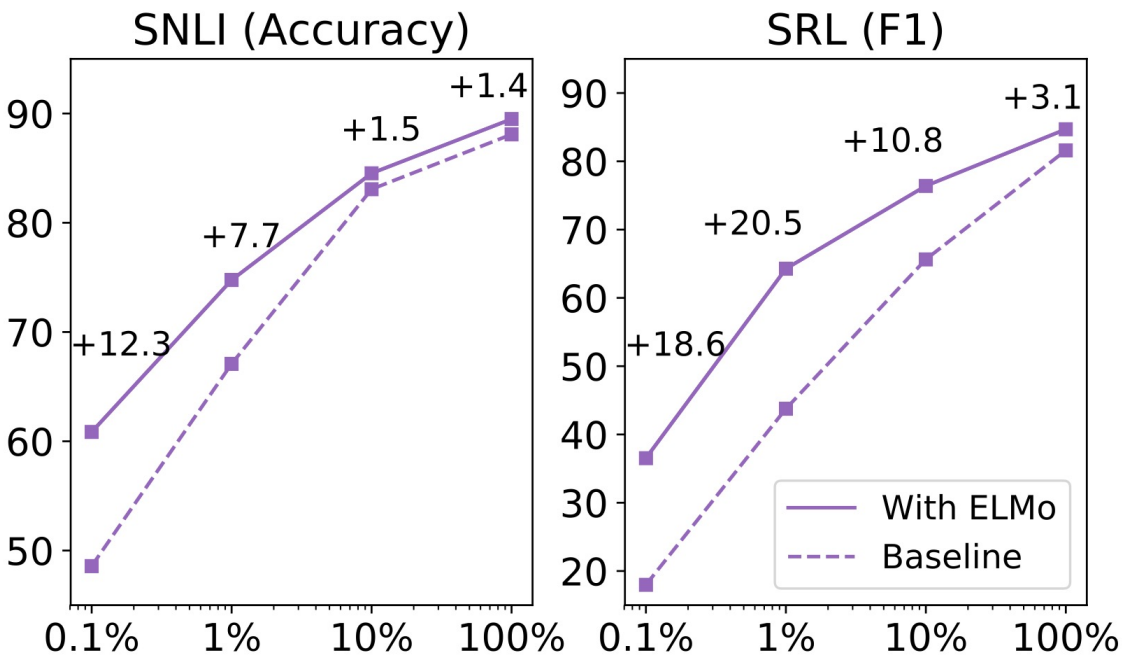
1) BiLM이 CoVe 인코더보다 정확도 더 높음

2) BiLM의 first layer가 second layer보다 성능이 우수

→ 첫번째 projection layer를 통해 얻어진 벡터들이 syntactic한 정보를 더 내포하고 있음

⇒biLM의 표현은 CoVe에 있는 표현보다 WSD와 POS 태깅으로 전달이 더 용이 (ELMo가 downstream tasks에서 CoVe를 능가하는 이유)

5.4 Sample efficiency

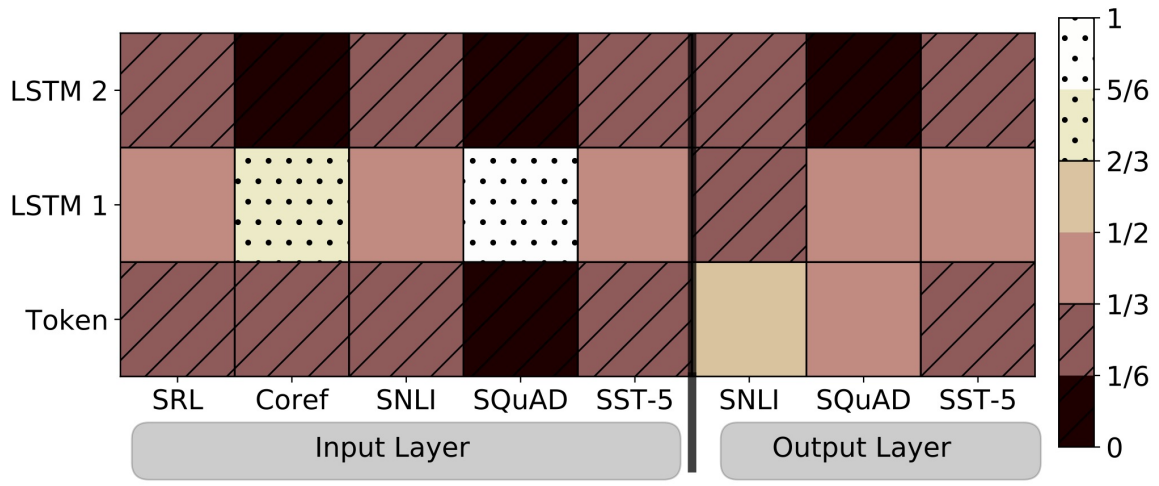


모델에 ELMo를 추가하면 매개변수 업데이트 횟수와 전체적인 훈련 세트 크기 측면에서 모두 효율성이 증가함.

더 작은 훈련 세트를 더 효율적으로 훈련함.

위 그림을 보면 같은 크기의 dataset에서 ELMo를 사용하는 경우가 훨씬 더 좋은 성능을 보임.
SRL의 경우에서는 ELMo를 사용한 model이 training dataset의 단 1%을 학습했을 때 달성한 수치와 baseline model이 training dataset의 10%를 학습했을 때의 수치가 동일한 것을 확인할 수 있음

5.5 Visualization of learned weights



소프트맥스 정규화된 학습 계층 가중치를 시각화한 것임.
Input Layer에서는 첫번째 biLSTM layer를 선호함.
Output Layer에서는 weight가 균형있게 분배되었지만, 낮은 layer를 좀 더 선호함.

6. Conclusion

biLM으로부터 고품질의 깊은 문맥의존 representation을 학습하는 일반적인 접근법을 도입함.

광범위한 NLP작업들에서 ELMo를 적용했을 때 큰 향상이 있는 것을 볼 수 있음.

biLM 계층이 문맥 내 단어들에 대한 다른 유형의 구문 및 의미 정보를 효율적으로 인코딩하고, 모든 계층 사용 시 전반적인 작업 향상이 있음.