Deep contextualized word representations (ELMo)

2019.3.30

발표: 염혜원

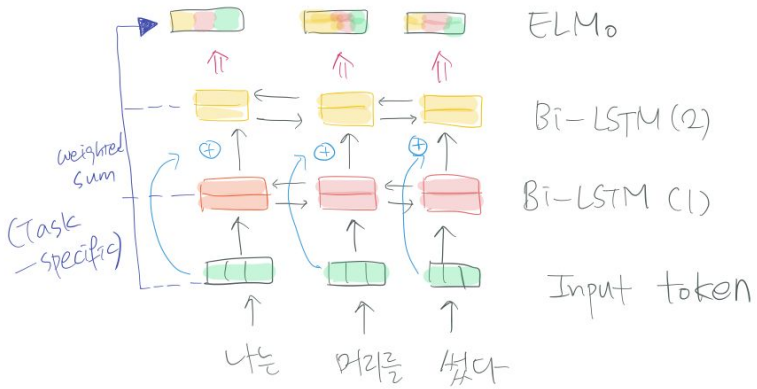
Abstract

- 새로운 형태의 deep contextualized word representation 소개
  (1) Syntactic, semantic 모두 고려
  (2) 여러 의미를 가지는 경우(polysemy; 다의어) 에 대한 고려

- 소개하는 워드 벡터들은 biLM 의 internal state들의 함수로 이루어지는데 각 다운스트림 과제에 따라 적절한 internal state mix를 가지도록 학습됨

1 Introduction

- Bi-LSTM으로 구현한 Language Model을 활용해 토큰 벡터들을 얻어서, **ELMo(Embeddings from Language Models)**라는 이름을 붙임
- 기존에는 Top LSTM layer만을 활용하는 경우가 많았는데, ELMo는 **biLM의 모든 internal layer를 활용**한다는 점에서, *deep* 하다고 볼 수 있음



1 Introduction

- Pretrained BiLM을 활용한 ELMo를 6개의 다른 NLP Task에 적용한 결과, 이전 state-of-the-art 성능을 뛰어넘는 결과를 보였음

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%

Table 1: Test set comparison of ELMo enhanced neural models with state-of-the-art single model baselines across six benchmark NLP tasks. The performance metric varies across tasks – accuracy for SNLI and SST-5; F_1 for SQuAD, SRL and NER; average F_1 for Coref. Due to the small test sizes for NER and SST-5, we report the mean and standard deviation across five runs with different random seeds. The "increase" column lists both the absolute and relative improvements over our baseline.

2 Related work

- Pretrained word vector의 활용이 표준화 되었지만, 하나의 단어에 하나의 벡터를 부여하다보니 **context-independent 한 이슈**가 있어왔음
- 워드 임베딩을 풍부하게 하기 위해, subword information을 활용하거나 다의어의 경우 의미별로 다른 벡터를 학습시키는 방법 등이 등장하였음 (본 연구에서도 **character embedding**을 통해 subword information을 활용함)
- context2vec, CoVe 등 최근 Context-dependent한 임베딩에 대한 연구가 이루어짐
- 이전 연구에 의하면 biRNN의 서로 다른 레이어가 다른 형태의 정보를 인코딩 한다고 알려져 있으며, 본 연구에서도 유사한 효과가 나타남

3 ELMo (1) Pretrained BiLM

- Bidirectional language models

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

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

- 두 방향으로부터 나오는 확률의 합을 최대화 하도록 학습시킴

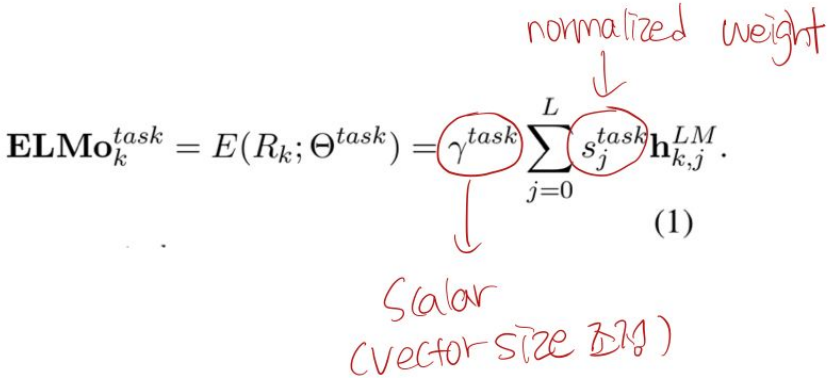
$$\sum_{k=1}^{N} \Big(\log p(t_k \mid t_1, \ldots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log p(t_k \mid t_{k+1}, \ldots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \Big).$$

** 인풋 벡터, 소프트맥스 레이어의 파라미터는 공유*

3 ELMo (2) Task-specific ELMo Embedding

- ELMo 임베딩은 **task 특화된 중간 레이어의 콤비네이션**을 가지게 됨
- 각 토큰별 pretrained biLM으로부터 representation set이 계산되고(토큰 레이어 + 각 biLSTM 레이어), 해당 representation 들의 조합을 학습하게 됨

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



3 ELMo (3) How to use

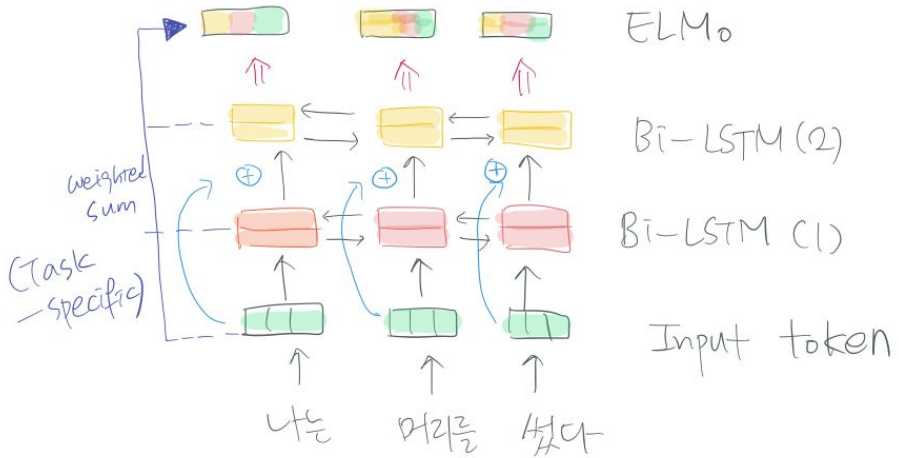
- ELMo를 supervised model에 추가하기 위해서는 biLM의 weight를 고정시키고, 토큰 벡터와 도출된 ELMo 벡터를 task RNN의 Input으로 활용하면 됨
- 일부 task 에서는 (SNLI, SQuAD) ELMo 벡터를 **task RNN의 Output에 다시 concat** 시키는 것이 효과가 있었음

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

Table 3: Development set performance for SQuAD, SNLI and SRL when including ELMo at different locations in the supervised model.

3 ELMo (4) Etc.

- Residual connection, character based input representation(Character CNN) 활용



4 Evaluation

- 단순히 ELMo를 끼워넣는 것으로만으로도 새로운 SOTA 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%

Table 1: Test set comparison of ELMo enhanced neural models with state-of-the-art single model baselines across six benchmark NLP tasks. The performance metric varies across tasks – accuracy for SNLI and SST-5; F_1 for SQuAD, SRL and NER; average F_1 for Coref. Due to the small test sizes for NER and SST-5, we report the mean and standard deviation across five runs with different random seeds. The "increase" column lists both the absolute and relative improvements over our baseline.

5 Analysis

- 기존에 top layer output만 사용 한 것 대비 성능 향상을 검증함
- 대부분의 경우 Regularization parameter λ 가 작을수록 성능이 좋은 경향이 있음 (why??)

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**

Table 2: Development set performance for SQuAD, SNLI and SRL comparing using all layers of the biLM (with different choices of regularization strength λ) to just the top layer.

5 Analysis

- biLM의 **첫 번째 레이어는 syntactic 정보**를, **두 번째 레이어는 semantic 정보**를 더 잘 인코딩 하는 것으로 나타남
- 이는 biLM의 **모든 레이어를 사용**하는 것이 성능향상에 도움이 된다는 것을 증명함

(Semantic)

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*

Table 5: All-words fine grained WSD F_1. For CoVe and the biLM, we report scores for both the first and second layer biLSTMs.

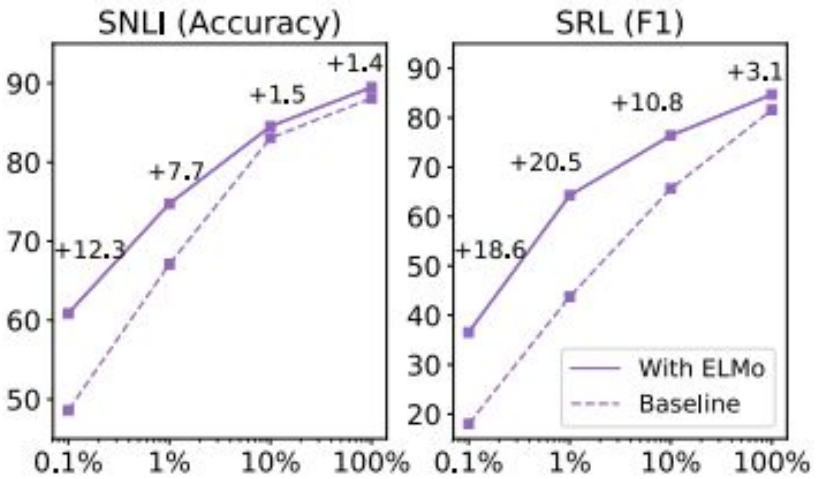
(Syntactic)

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

Table 6: Test set POS tagging accuracies for PTB. For CoVe and the biLM, we report scores for both the first and second layer biLSTMs.

5 Analysis

- ELMo 를 활용하면 같은 성능을 내는 데에 있어 훨씬 **학습이 효율적**임
  : 더 작은 트레이닝셋, 더 작은 에폭 수



6 Conclusion

- We have introduced a general approach for learning **high-quality deep context-dependent representations from biLMs**, and shown large improvements when applying ELMo to a broad range of NLP tasks.
- Through ablations and other controlled experiments, we have also confirmed that the **biLM layers efficiently encode different types of syntactic and semantic information** about words in-context, and that using all layers improves overall task performance.

참고할만한 자료

- 텐서플로 구현: Deep Dive The ELMo Implementation
-