1314기 텍스트세미나

ToBig's 13기 이예진

CS224N

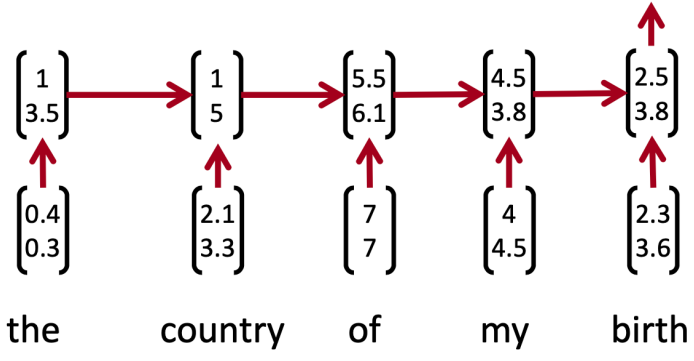
Lecture 11. ConvNets for NLP

Text CNN

Contents

RNNs 기존 RNN의 문제

- Prefix context 모두 포함 (the, of,…)
- 마지막 단어(final vector)에 영향을 많이 받음



CNN Main Idea : What if we compute vectors for every possible word subsequence of a certain length?

- 모든 가능한 단어 어절을 벡터로 계산하면 어떨까?

“tentative deal reached to keep government open” -> window size 3의 tri-gram

- tentative deal reached, deal reached to, reached to keep, to keep government, keep government open

- Text CNN의 필터는 텍스트의 지역적인 정보(단어의 등장순서/문맥정보) 보존
- 문법적으로 맞는지 확인 불가
- 언어학적으로 맞는 것 같지 않음

1d Convolution 위아래 밖에 없기 때문에 1d (cf. 비전처럼 위아래, 양옆으로 이동하면 2d)

"tentative deal reached to keep government open"

embedding vector 4-dim

tentative	0.2	0.1	−0.3	0.4
deal	0.5	0.2	−0.3	−0.1
reached	−0.1	−0.3	−0.2	0.4
to	0.3	−0.3	0.1	0.1
keep	0.2	−0.3	0.4	0.2
government	0.1	0.2	−0.1	−0.1
open	−0.4	−0.4	0.2	0.3

Dot product

t,d,r	−1.0
d,r,t	−0.5
r,t,k	−3.6
t,k,g	−0.2
k,g,o	0.3

7x4 -----(3x4 filter)----→ 5x1

Apply a **filter** (or **kernel**) of size 3

3	1	2	−3
−1	2	1	−3
1	1	−1	1

Unit 02 | 1d Convolution for Text

1d Convolution padding = 1

"tentative deal reached to keep government open"

ø	0.0	0.0	0.0	0.0
tentative	0.2	0.1	−0.3	0.4
deal	0.5	0.2	−0.3	−0.1
reached	−0.1	−0.3	−0.2	0.4
to	0.3	−0.3	0.1	0.1
keep	0.2	−0.3	0.4	0.2
government	0.1	0.2	−0.1	−0.1
open	−0.4	−0.4	0.2	0.3
ø	0.0	0.0	0.0	0.0

Apply 3 **filters** of size 3

3	1	2	−3
−1	2	1	−3
1	1	−1	1

1	0	0	1
1	0	−1	−1
0	1	0	1

1	−1	2	−1
1	0	−1	3
0	2	2	1

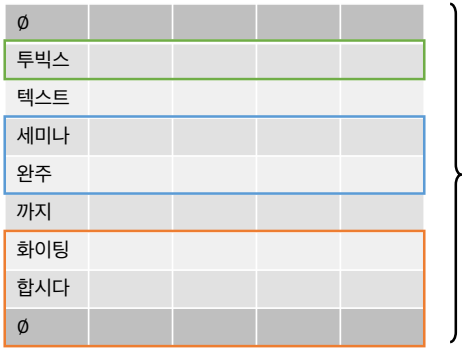
ø,t,d	−0.6	0.2	1.4
t,d,r	−1.0	1.6	−1.0
d,r,t	−0.5	−0.1	0.8
r,t,k	−3.6	0.3	0.3
t,k,g	−0.2	0.1	1.2
k,g,o	0.3	0.6	0.9
g,o,ø	−0.5	−0.9	0.1

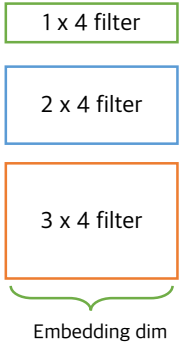
기존과 동일한 길이의 벡터

필터 갯수만큼

1d Convolution multiple filter

“tentative deal reached to keep government open”





Filter 크기를 조정해서 uni-gram, bi-gram, tri-gram 만들 수 있음

더 많은 필터를 사용해서 output 의 차원을 늘릴 수록 기존 문장에 대한 정보량 커짐

Unit 02 | 1d Convolution for Text

1d Convolution pooling

Max pooling

Ø,t,d	−0.6	0.2	1.4
t,d,r	−1.0	1.6	−1.0
d,r,t	−0.5	−0.1	0.8
r,t,k	−3.6	0.3	0.3
t,k,g	−0.2	0.1	1.2
k,g,o	0.3	0.6	0.9
g,o,Ø	−0.5	−0.9	0.1
max p	0.3	1.6	1.4

average pooling

Ø,t,d	−0.6	0.2	1.4
t,d,r	−1.0	1.6	−1.0
d,r,t	−0.5	−0.1	0.8
r,t,k	−3.6	0.3	0.3
t,k,g	−0.2	0.1	1.2
k,g,o	0.3	0.6	0.9
g,o,Ø	−0.5	−0.9	0.1
ave p	−0.87	0.26	0.53

K-max pooling, k=2

Ø,t,d	−0.6	0.2	1.4
t,d,r	−1.0	1.6	−1.0
d,r,t	−0.5	−0.1	0.8
r,t,k	−3.6	0.3	0.3
t,k,g	−0.2	0.1	1.2
k,g,o	0.3	0.6	0.9
g,o,Ø	−0.5	−0.9	0.1
2-max p	−0.2	1.6	1.4
	0.3	0.6	1.2

나온 순서도 반영

NLP에서는 max pooling을 선호

정보가 모든 token에 있는 것이 아니라 sparse하게 있음

Unit 02 | 1d Convolution for Text

1d Convolution stride, local max pooling, dilation

Ø	0.0	0.0	0.0	0.0
tentative	0.2	0.1	−0.3	0.4
deal	0.5	0.2	−0.3	−0.1
reached	−0.1	−0.3	−0.2	0.4
to	0.3	−0.3	0.1	0.1
keep	0.2	−0.3	0.4	0.2
government	0.1	0.2	−0.1	−0.1
open	−0.4	−0.4	0.2	0.3
Ø	0.0	0.0	0.0	0.0

Stride = 2

Ø,t,d	−0.6	0.2	1.4
d,r,t	−0.5	−0.1	0.8
t,k,g	−0.2	0.1	1.2
g,o,Ø	−0.5	−0.9	0.1

Stride = 2, local max pooling (2개씩 봄)

Ø,t,d	−0.6	0.2	1.4
t,d,r	−1.0	1.6	−1.0
d,r,t	−0.5	−0.1	0.8
r,t,k	−3.6	0.3	0.3
t,k,g	−0.2	0.1	1.2
k,g,o	0.3	0.6	0.9
g,o,Ø	−0.5	−0.9	0.1
Ø	−Inf	−Inf	−Inf

Ø,t,d,r	−0.6	1.6	1.4
d,r,t,k	−0.5	0.3	0.8
t,k,g,o	0.3	0.6	1.2
g,o,Ø,Ø	−0.5	−0.9	0.1

Striding을 하면서 pooling을 한다

이미지에 많이 사용되고, nlp에는 잘 사용 하지 않음

Dilation = 2

Ø,t,d	−0.6	0.2	1.4	1
t,d,r	−1.0	1.6	−1.0	2
d,r,t	−0.5	−0.1	0.8	3
r,t,k	−3.6	0.3	0.3	4
t,k,g	−0.2	0.1	1.2	5
k,g,o	0.3	0.6	0.9	6
g,o,Ø	−0.5	−0.9	0.1	7

1,3,5	0.3	0.0
2,4,6		
3,5,7		

넓은 범위를 적은 파라미터로 커버할 수 있음

레이어 깊을 수록 효과 증대

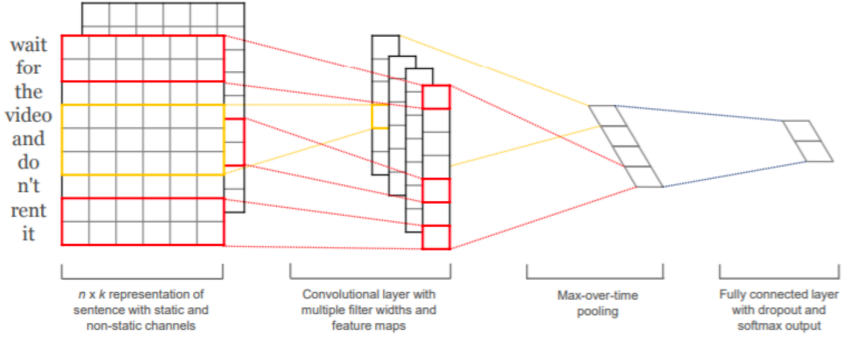
Single Layer CNN for Sentence Classification

- Yoon Kim (2014): Convolutional Neural Networks for Sentence Classification. EMNLP 2014.

https://arxiv.org/pdf/1408.5882.pdf Code: https://arxiv.org/pdf/1408.5882.pdf [Theano!, etc.]

- A variant of convolutional NNs of Collobert, Weston et al. (2011)

➢ One convolutional layer and pooling

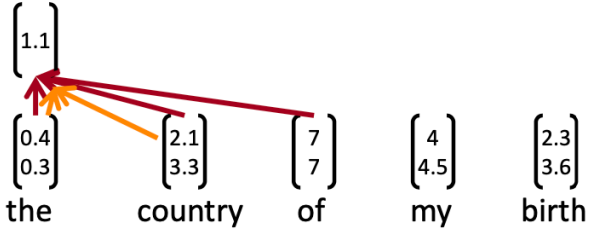


Yoon Kim (2014): Convolutional Neural Networks for Sentence Classification. EMNLP 2014.

Convolution 연산

- Matrix 구조가 아닌 단어 벡터들을 concat 해서 연산

- Word vectors : $X_i \in R^k$ (pre-trained word vectors)
- Sentence : $X_{1:n} = [X_1 ; X_2 ; \dots ; X_n]$ (vectors concatenated)
- Convolutional filter : $W \in R^{hk}$ (over window of **h** words)

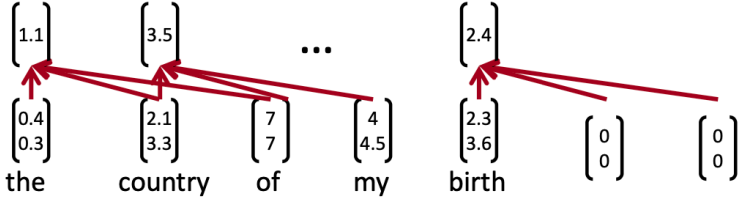


Yoon Kim (2014): Convolutional Neural Networks for Sentence Classification. EMNLP 2014.

Convolution 연산

$$c_i = f(\mathbf{w}^T \mathbf{x}_{i:i+h-1} + b)$$

- Sentence $x_{1:n} = x_1 \oplus x_2 \oplus x_3 \oplus, \cdots, \oplus x_n$
- 함수 f : tanh
- 피쳐맵 결과 $\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}] \in \mathbb{R}^{n-h+1}$



Yoon Kim (2014): Convolutional Neural Networks for Sentence Classification. EMNLP 2014.

Pooling 연산

- 피쳐맵 결과 $\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}] \in \mathbb{R}^{n-h+1}$ 에서 pooling 연산

$$\hat{c} = \max\{\mathbf{c}\}$$

- Multiple filter 이용
- Max 연산으로 filter size 변화 & 문장 길이에 대처 가능 Filter size : 3, 4, 5 각각 100 feature maps 사용

Yoon Kim (2014): Convolutional Neural Networks for Sentence Classification. EMNLP 2014.

Multi-channel input idea

- (워드임베딩) Initialize with pre-trained word vectors (word2vec or Glove)
- 'static' / 'non - static' 으로 사용

Classification after CNN layer

- Convolution 레이어를 한 번 통과한 후, pooling 후에 최종 벡터를 얻음
- 필터 m개를 사용한 최종 벡터 $\mathbf{z} = [\hat{c}_1, \ldots, \hat{c}_m]$
- soft max 통해 classification 수행 $y = softmax\left(W^{(S)}z + b\right)$

Yoon Kim (2014): Convolutional Neural Networks for Sentence Classification. EMNLP 2014.

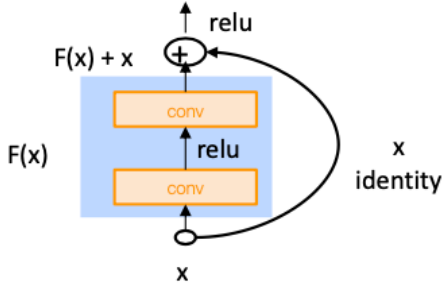
Dropout

- train $y = softmax\left(W^{(S)}(r \circ z) + b\right)$
- test $\hat{W}^{(S)} = pW^{(S)}$
- P = 0.5 로 train에 적용, test에는 p 값만 존재하고 no dropout & scaling final vector
- 2-4% accuracy 향상

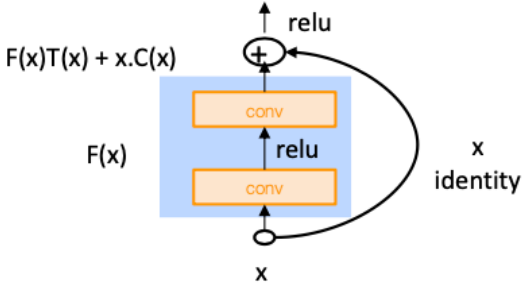
L2 norm

- 각 class weight vectors 에 적용
- $\|W_{c\cdot}^{(S)}\| > s$ 이면 rescaling $\|W_{c\cdot}^{(S)}\| = s$
- 자주 사용하지 않음

Gated units used vertically



Residual block
(He et al. ECCV 2016)



Highway block
(Srivistava et al. NeurIPS 2015)

둘은 결국 같은 역할 -> x에 대한 정보를 얼마나 넘길지

T(transform gate), C(carry gate)

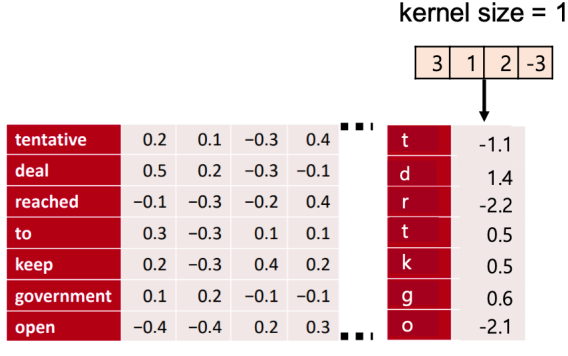
Unit 04 | 추가&정리

Batch Normalization (BatchNorm)

- 보통 CNN에 쓰임
- Convolution 연산의 output 을 배치별로 정규화시킴, (평균 0, 분산 1)
- Z-transform 같이 모델의 안정적인 학습을 가능하게 함
- 배치 별 업데이트는 변동이 클 수 있기 때문에 많이 사용하면 안됨
- Parameter initialization 에 덜 민감
- Learning rate에 대한 tunning 이 쉬워짐

1 x 1 Convolutions

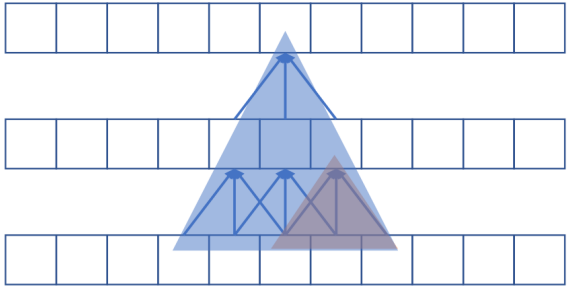
- 1d convolution, kerner_size = 1
- 적은 파라미터로 channel 축소 가능 (작은 필터 사용) -> from many channels to fewer channels
- Fully connected layer 의 input 으로 사용 가능



Unit 04 | 추가&정리

CNN for NLP

- **tokens** -> **multi-word** -> **expressions** -> **phrases** -> **sentence** 순서로 포착
- 구현이 굉장히 잘 되어있고 사용하기 편함
- 중요한 것이 첫번째 단어와 마지막 단어에 있으면, 포착하기 위해서 많은 convolution layer 가 필요함
- RN(Relation Network, 너무 전체를 봄) 과 CNN(너무 local로 봄) 의 장점을 합쳐서 나온 것이 Self-Attention



Captures k-grams hierarchically

Unit 04 | 추가&정리

Bag of Vectors

- CBoW, Bag of words 모델들..

Window Model

- Relation Network: Skip Bigrams

CNNs

RNNs

'How to represent a sentence'

average 하는 cbow 모델을 제외하면, 나머지 개념들은 stack을 한다. 각 token별로 토큰위치에 맞는 lookup을 보고 벡터들을 뽑아냄

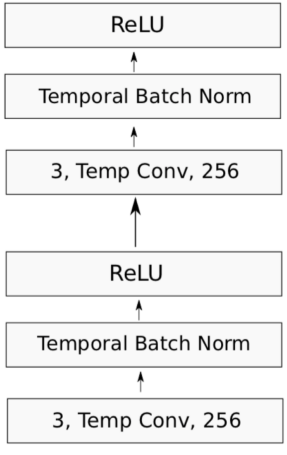
지금까지 배운 token representation 방법들은, combine 해서 같이 쓰임

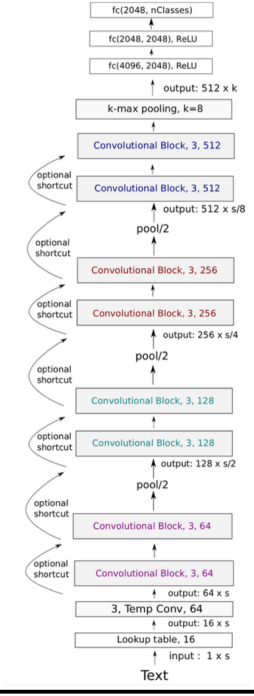
+ Task가 classification이면 궁극적으로는 average 가 가장 일반적

Unit 05 | Deep CNN for Text

Alexis Conneau (2017) : Very Deep Convolutional Networks for Text Classification

- Nlp에서는 이미지처럼 신경망이 매우 깊지 않다
- Vision system의 VGGnet or ResNet을 닮은 것이 **VD-CNN**
- Batch norm & relu 사용해서, Convolution layer 2개씩 사용
- Kernel_size = 3, padding 으로 input 길이 고정





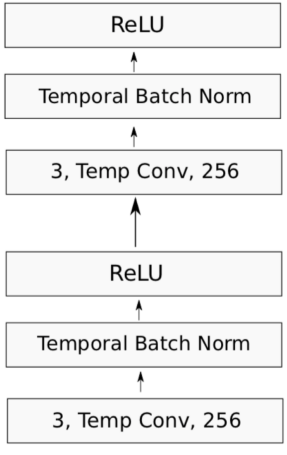
Alexis Conneau (2017) : Very Deep Convolutional Networks for Text Classification

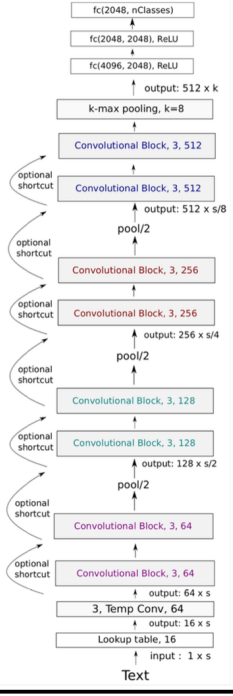
Corpus:	AG	Sogou	DBP.	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
Method	n-TFIDF	n-TFIDF	n-TFIDF	ngrams	Conv	Conv+RNN	Conv	Conv
Author	[Zhang]	[Zhang]	[Zhang]	[Zhang]	[Zhang]	[Xiao]	[Zhang]	[Zhang]
Error	7.64	2.81	1.31	4.36	37.95*	28.26	40.43*	4.93*
[Yang]	-	-	-	-	-	24.2	36.4	-

Table 4: Best published results from previous work. Zhang et al. (2015) best results use a Thesaurus data augmentation technique (marked with an *). Yang et al. (2016)'s hierarchical methods is particularly

Depth	Pooling	AG	Sogou	DBP.	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
9	Convolution	10.17	4.22	1.64	5.01	37.63	28.10	38.52	4.94
9	KMaxPooling	9.83	3.58	1.56	5.27	38.04	28.24	39.19	5.69
9	MaxPooling	9.17	3.70	1.35	4.88	36.73	27.60	37.95	4.70
17	Convolution	9.29	3.94	1.42	4.96	36.10	27.35	37.50	4.53
17	KMaxPooling	9.39	3.51	1.61	5.05	37.41	28.25	38.81	5.43
17	MaxPooling	8.88	3.54	1.40	4.50	36.07	27.51	37.39	4.41
29	Convolution	9.36	3.61	1.36	4.35	**35.28**	27.17	37.58	**4.28**
29	KMaxPooling	**8.67**	**3.18**	1.41	4.63	37.00	27.16	38.39	4.94
29	MaxPooling	8.73	3.36	**1.29**	**4.28**	35.74	**26.57**	**37.00**	4.31

Table 5: Testing error of our models on the 8 data sets. No data preprocessing or augmentation is used.





Unit 00 | reference

CS224n : Natural Language Processing with Deep Learning Stanford, Winter 2019

DSBA 연구실 CS224n 세미나 강의 자료

건국대학교 컴퓨터공학부 인공지능 수업

조경현 교수님의 딥러닝을 이용한 자연어 처리

들어주셔서 감사합니다.