VDCNN

Very Deep Convolutional Networks for Text Classification

전은영

배경

Computer vision 분야에서 deep CNN으로 엄청 성공!

NLP의 CNN은 다소 "얕다(shallow)"

NLP에도 deep CNN 써보자!

A sequence of words (e.g. 문장)

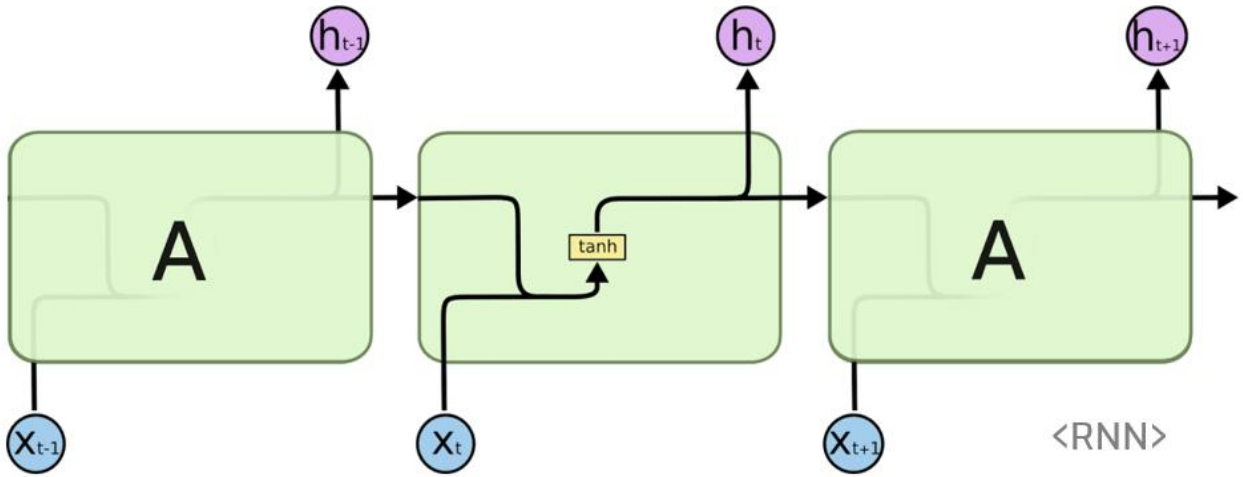
특징

- 복잡한 문법적(syntax), 의미적(semantic) 관계
- local & long-range dependencies

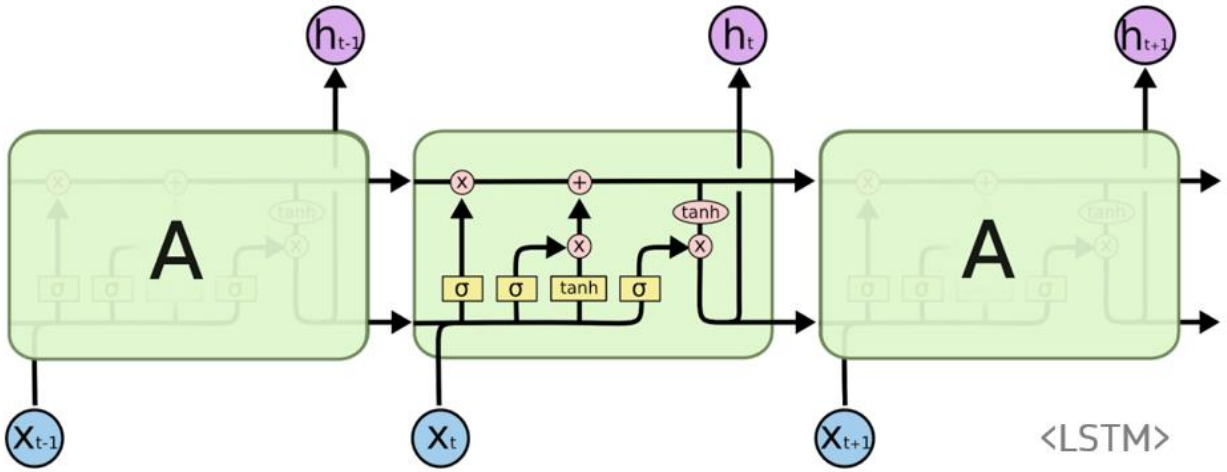
주된 방법

- 문장을 토큰의 시퀀스로 여기고
  토큰들을 RNN으로 처리

RNN for text processing : 장점



<RNN>



<LSTM>

- Tokens이 순서대로 처리

- 내부 states로 문장 전체를 기억

⇒ long-range dependencies 모델링 가능

RNN for text processing : 단점

- 시퀀스 처리를 위한 “generic” 학습 기계

⇒ lacking task-specific structure

논문 제시 방안 : Deep-CNN

이미지의 compositional 구조 덕분에 CNN 잘 됐다.

text도 비슷한 특징 있다.

ex. 문자 -> 단어, 문구, 문장, …

History of CNN for NLP

- (dynamic) k-max pooling
- 문자 단위(character-level) 입력
- CNN+RNN

최대 6 conv. layer,

다양한 크기의 필터 사용 for short & long-span relations.

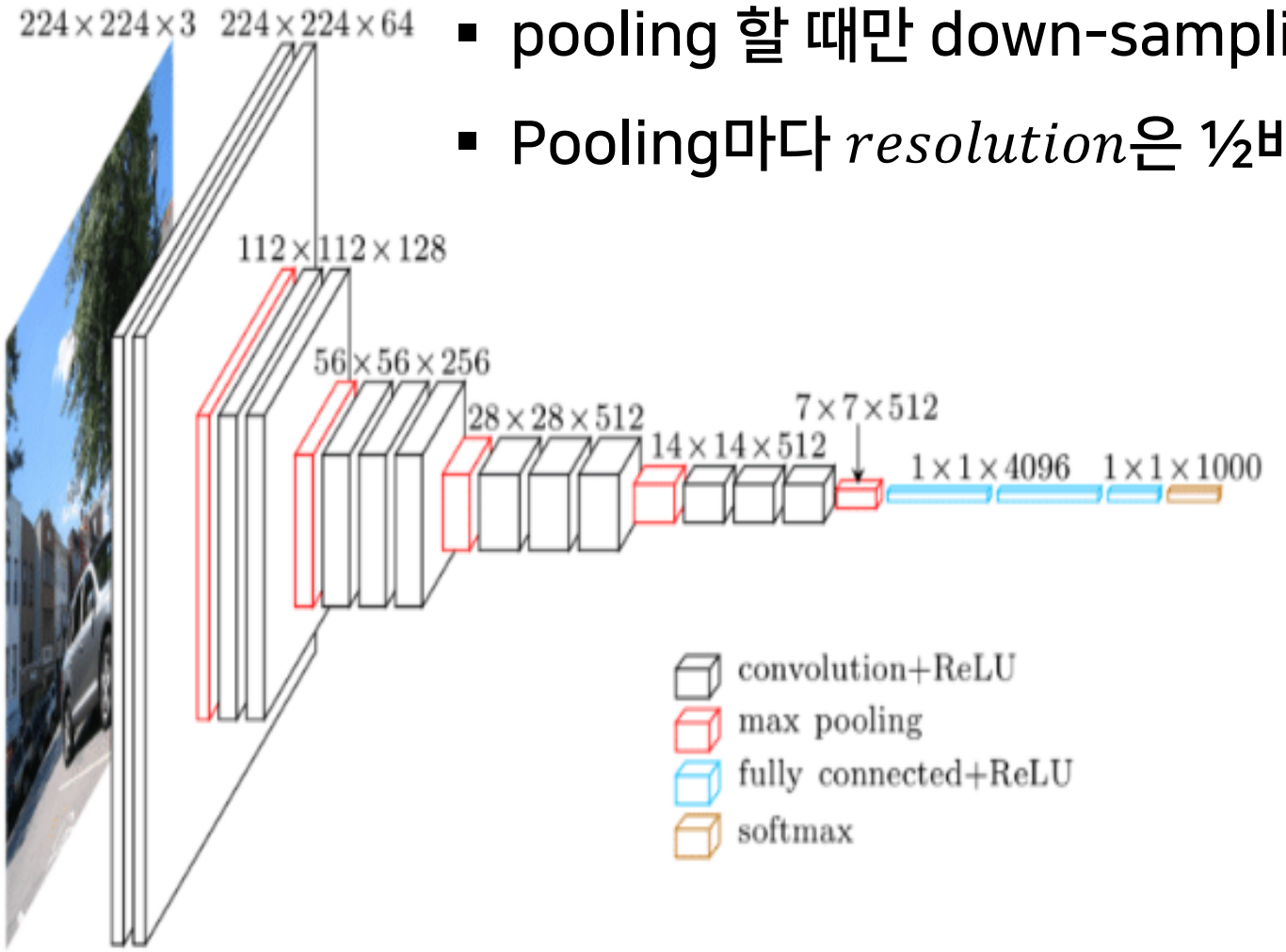
(kernel size 3,5,7)

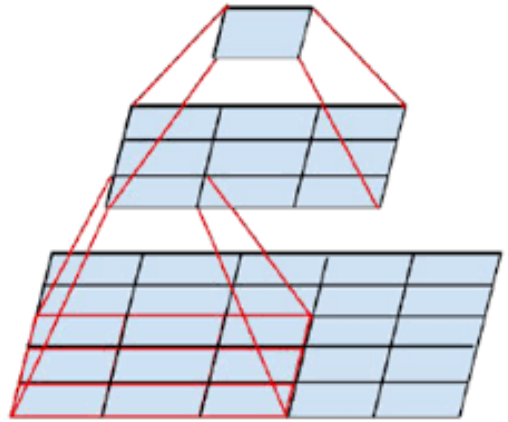
VDCNN

- 문자 단위(character-level) 입력
- VGG와 ResNet 참고
- text processing에서 처음으로 very deep CNN (최대 29 conv.)
- “깊이의 이점(benefit of depth)”를 처음으로 보여줌

VDCNN : VGG

- (동일한 조건의) 작은 conv. 와 pooling을 깊게 쌓은 구조
- pooling 할 때만 down-sampling (conv는 same으로)
- Pooling마다 $resolution$은 ½배, 필터 수는 2,3배





- **동일한 조건의 conv?**

⇒ 작은 conv 쌓아서

큰 conv와 같은 효과 볼 수 있다

by convolution factorization

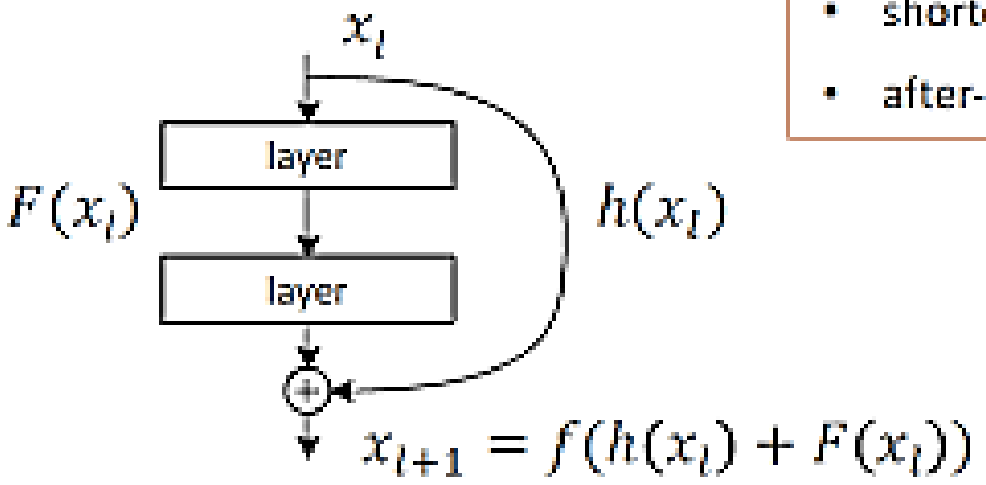
파라미터 수: n개의 작은 conv < 큰 conv

VDCNN : ResNet

- 깊은 신경망 학습하지 못하는 이유?

⇒ Vanishing or exploding gradients

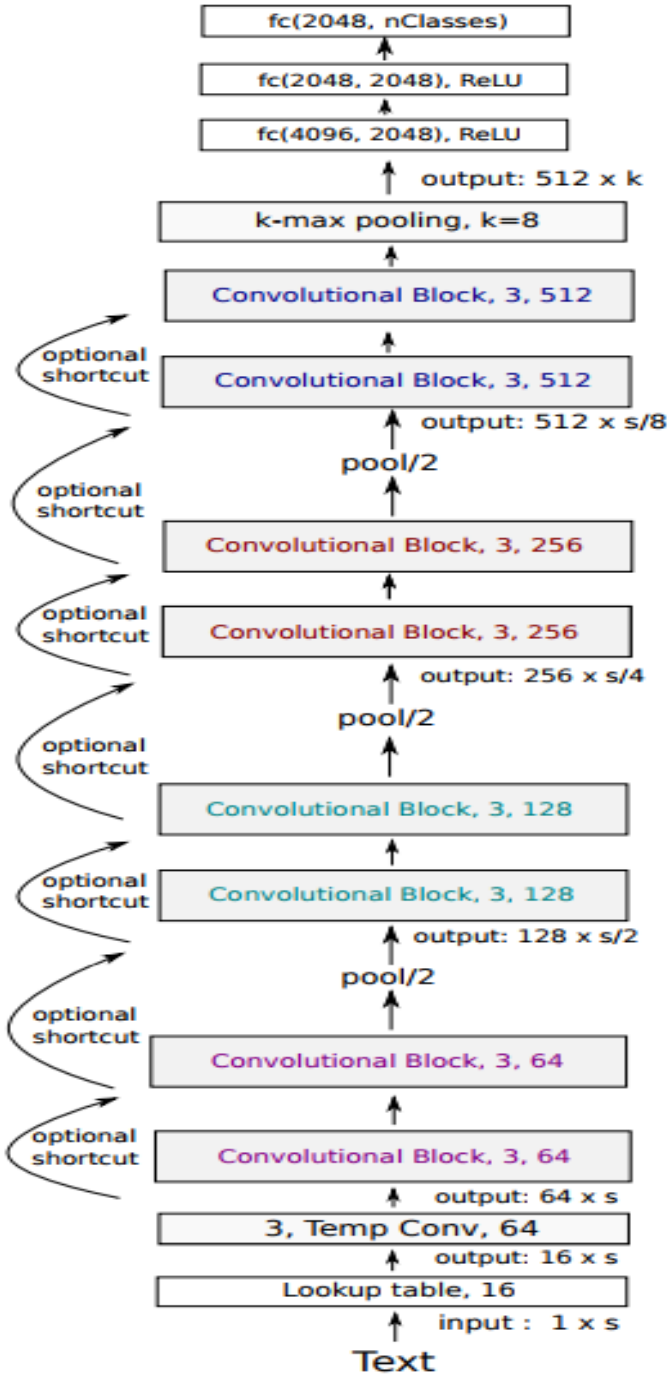
ResNet "Skip Connection(Short Cut)"으로 보완



- shortcut mapping: h = identity
- after-add mapping: f = ReLU

VDCNN Architecture

- Lookup table (embedding)
  - Embedding size =16
  - s : 고정 길이의 input character 수
- Conv. Blocks & 3 pooling(down-sampling)
  - Kernel size=3
- K-max pooling
  - 처음으로 고정된 dimension으로 down-sample
- F.C with ReLU
- Temporal batch normalization for regularization (drop-out 사용 x)



VDCNN Architecture : Design rules

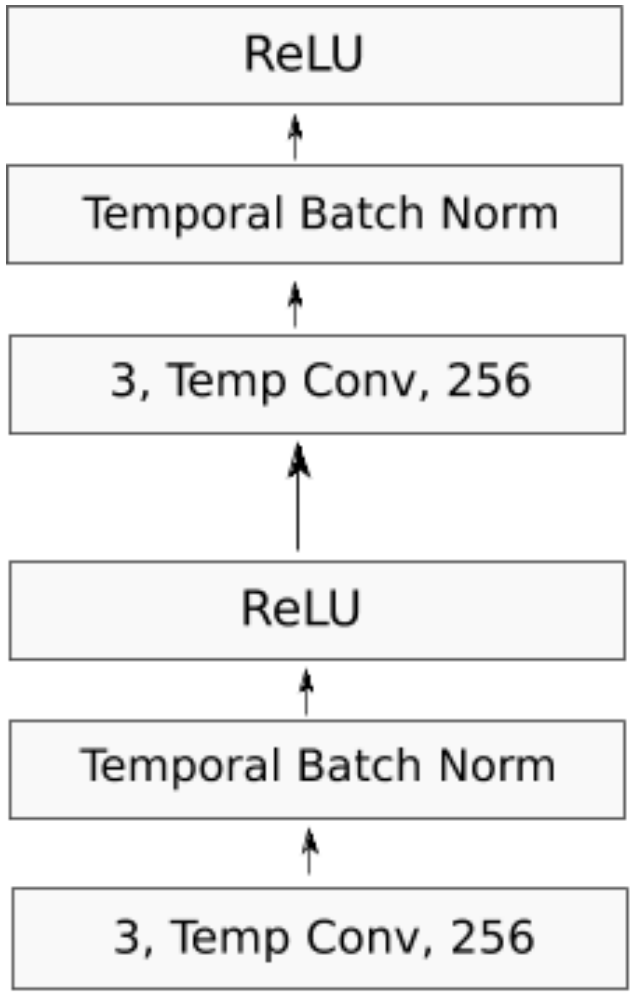
- 두 가지의 디자인 규칙

① 같은 output temporal resolution을 가지면,
그 layers의 feature maps 수도 같다

② Temporal resolution이 $\frac{1}{2}$이 되면(pooling),
feature maps의 수를 2배한다.

output of conv. : 512 x S_d , S_d=$\frac{s}{2^p}$, p : pooling 수

VDCNN Architecture : Convolutional block



VDCNN Architecture : Down-sampling

- 목적 : resolution을 $\frac{1}{2}$ 배
- 3 가지 방법으로 실험
  - Between K_i and K_{i+1}
  - K_{i+1}의 첫 번째 conv. layer stride를 2로 하기 (ResNet-like)
  - K_i 뒤에 k-max pooling (dynamic k-max pooling)
  - K_i 뒤에 kernel size가 3이고 stride가 2인 max-pooling (VGG-like)

VDCNN Architecture : Best configurations

Depth:	9	17	29	49
conv block 512	2	4	4	6
conv block 256	2	4	4	10
conv block 128	2	4	10	16
conv block 64	2	4	10	16
First conv. layer	1	1	1	1
#params [in M]	2.2	4.3	4.6	7.8

Experimental evaluation : Tasks and Data

Data set	#Train	#Test	#Classes	Classification Task
AG's news	120k	7.6k	4	English news categorization
Sogou news	450k	60k	5	Chinese news categorization
DBPedia	560k	70k	14	Ontology classification
Yelp Review Polarity	560k	38k	2	Sentiment analysis
Yelp Review Full	650k	50k	5	Sentiment analysis
Yahoo! Answers	1 400k	60k	10	Topic classification
Amazon Review Full	3 000k	650k	5	Sentiment analysis
Amazon Review Polarity	3 600k	400k	2	Sentiment analysis

Table 3: Large-scale text classification data sets used in our experiments. See (Zhang et al., 2015) for a detailed description.

data augmentation x , preprocessing x (lower-casing 제외)

Experimental evaluation : Model settings

- character-level
  - 69 tokens(66 + special padding + space + unknown token)

```
abcdefghijklmnopqrstuvwxyz0123456
789-,;.!?:'"/|_#$%^&*~`+=<>()[]{}
```

- size of input text
  - s=1024
  - 부족하면 채우고(padding), 넘어가면 자른다(truncated)

- Training
  - initializer : He
  - optimizer : SGD
  - mini-batch : 128
  - initial learning rate : 0.01, momentum : 0.9
  - temporal batch norm without drop-out

Experimental evaluation : Results

- baseline

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 adapted to datasets whose samples contain multiple sentences.

Experimental evaluation : Results

▪ 3 가지 depth와 , 3 가지 pooling 비교

- 깊을 수록 성능 향상, max pooling이 잘 됨

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.

Experimental evaluation : Results

- 49 conv. layer??

=> sota는 아님, 단 short cut으로 error는 줄임

depth	without shortcut	with shortcut
9	37.63	40.27
17	36.10	39.18
29	35.28	36.01
49	37.41	36.15

Table 6: Test error on the Yelp Full data set for all depths, with or without residual connections.

Experimental evaluation : Results

- 데이터 많을 때 특히 잘 된다.
- 깊을 수록 잘 된다 (최대 29 conv. layer).
- Max-pooling이 다른 pooling보다 잘 된다.
- sota CNN보다 잘 된다
- 너무 깊으면 accuracy degradation => short-cut이 감소시킨다!
- 더 많은 class를 가진 tasks도 유망하다

Thank You