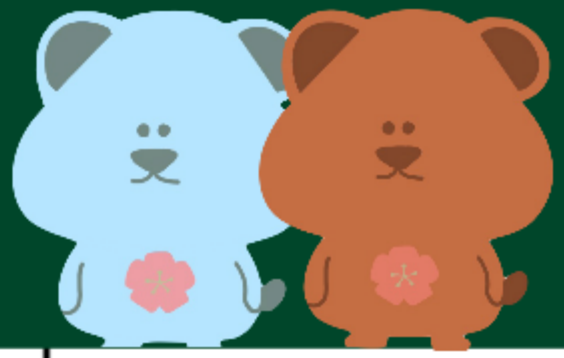


Week 12. Lecture 11 - Convolutional Networks for NLP

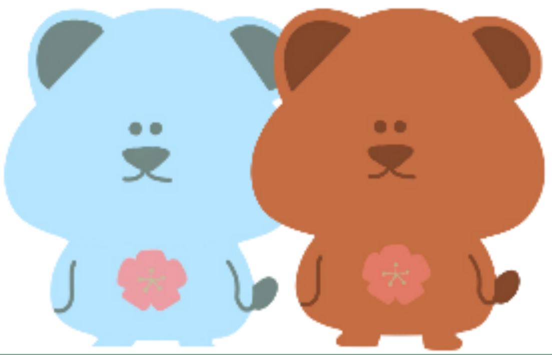
발표자: 조서영, 김소민

EWHA EURON

목차

EWHA EURON

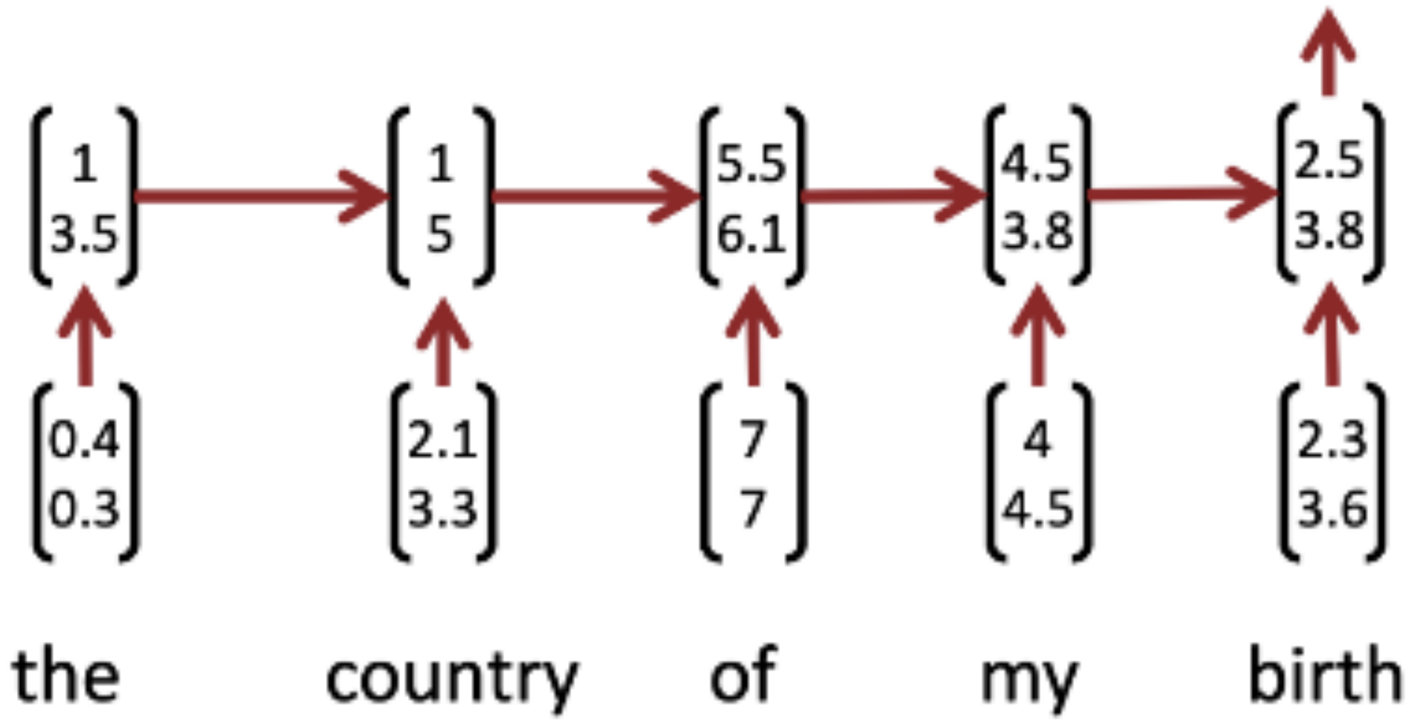
Why CNN?



Why CNN?

기존 RNN의 한계

- The, of 같은 prefix까지 포함한 context를 학습한다
- Final vector에 마지막 단어의 영향이 너무 커질 수 있다



EWHA EURON

Why CNN?

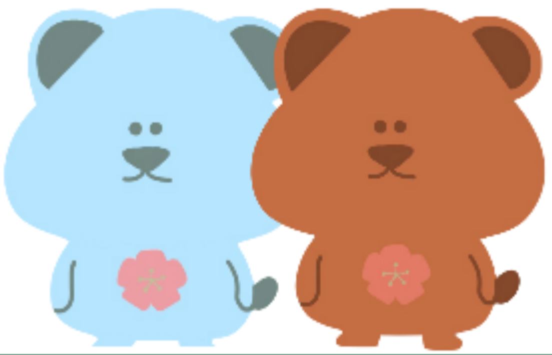
CNN

- Main idea: 모든 가능한 어절에 대해 vector를 계산하면 어떨까?

- Ex) “tentative deal reached to keep government open”에서 벡터를 계산
  - ‘tentative deal reached’, ‘deal reached to’, ‘reached to keep’, ‘to keep government’, ‘keep government open’

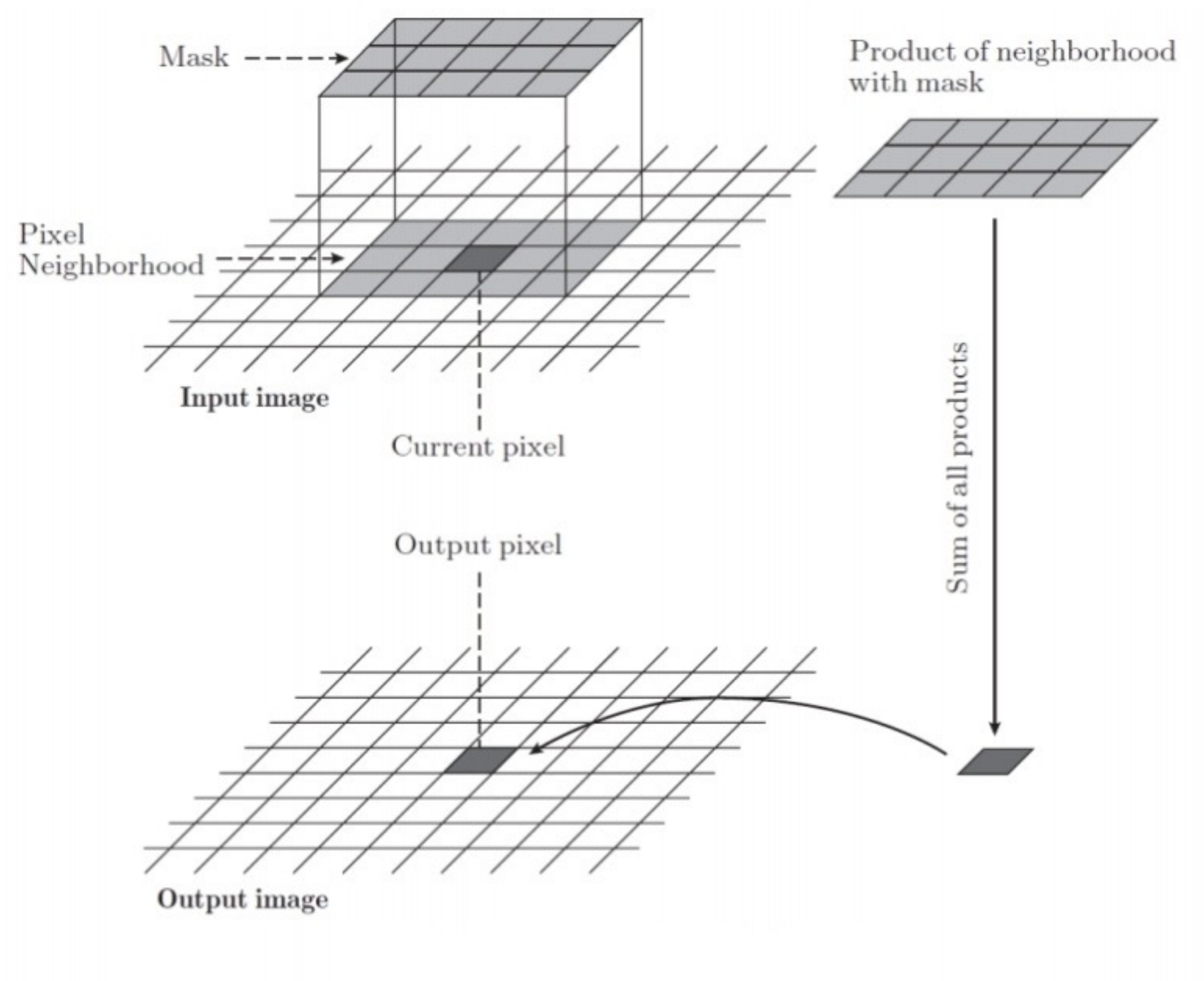
- 어절이 문법에 맞는지는 신경쓰지 않음

CNN



CNN

- Convolutional Neural Net - what is convolution?



CNN

- 1D convolution for text
  - 한 방향으로만 이동하므로 1D

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

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

3	1	2	−3
−1	2	1	−3
1	1	−1	1

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 -----------→ 5x1

3x4 filter

CNN

- 1D convolution for text

∅	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
∅	0.0	0.0	0.0	0.0

∅,t,d	−0.6
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
g,o,∅	−0.5

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

3	1	2	−3
−1	2	1	−3
1	1	−1	1

11

Convolution 적용 후에도 크기 유지하기 위해 zero-padding 적용

CNN

- 1D convolution for text

∅	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
∅	0.0	0.0	0.0	0.0

∅,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,∅	−0.5	−0.9	0.1

Apply 3 **filters** of size 3

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

Could also use (zero) padding = 2

Also called "wide convolution"

CNN

- 1D convolution for text

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

- Convolution 결과를 요약하는 방법

- Max pooling: 가장 강조되는 부분
- Average pooling: 문장 전체에 대한 맥락

- 보통 NLP에서는 max pooling이 선호됨
- 정보가 모든 token에 있는 게 아니라 sparse하게 있음

CNN

- 1D convolution for text
  - Stride = 2

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

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

3	1	2	−3
−1	2	1	−3
1	1	−1	1

- Local 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
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

EWHA EURON

CNN

- 1D convolution for text
  - Dilation = 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

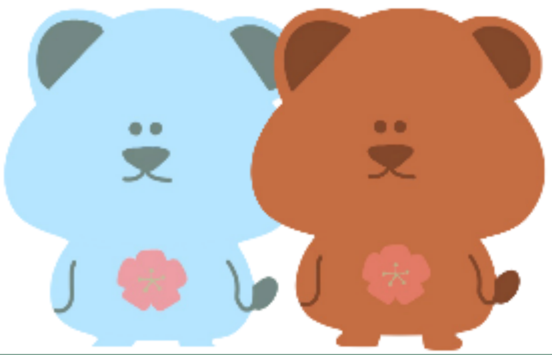
1,3,5	0.3	0.0
2,4,6		
3,5,7		

2	3	1
1	−1	−1
3	1	0

1	3	1
1	−1	−1
3	1	−1

- 많은 파라미터 없이 문장의 넓은 범위 볼 수 있음
- 레이어 깊을수록 효과 좋음

Single Layer CNN for Sentence Classification



Single Layer CNN for Sentence Classification

- Yoon Kim (2014): Convolutional Neural Networks for Sentence Classificaton
- A variant of convolutional NNs of Collobert, Weston et al. (2011)

- 목표: 문장 분류

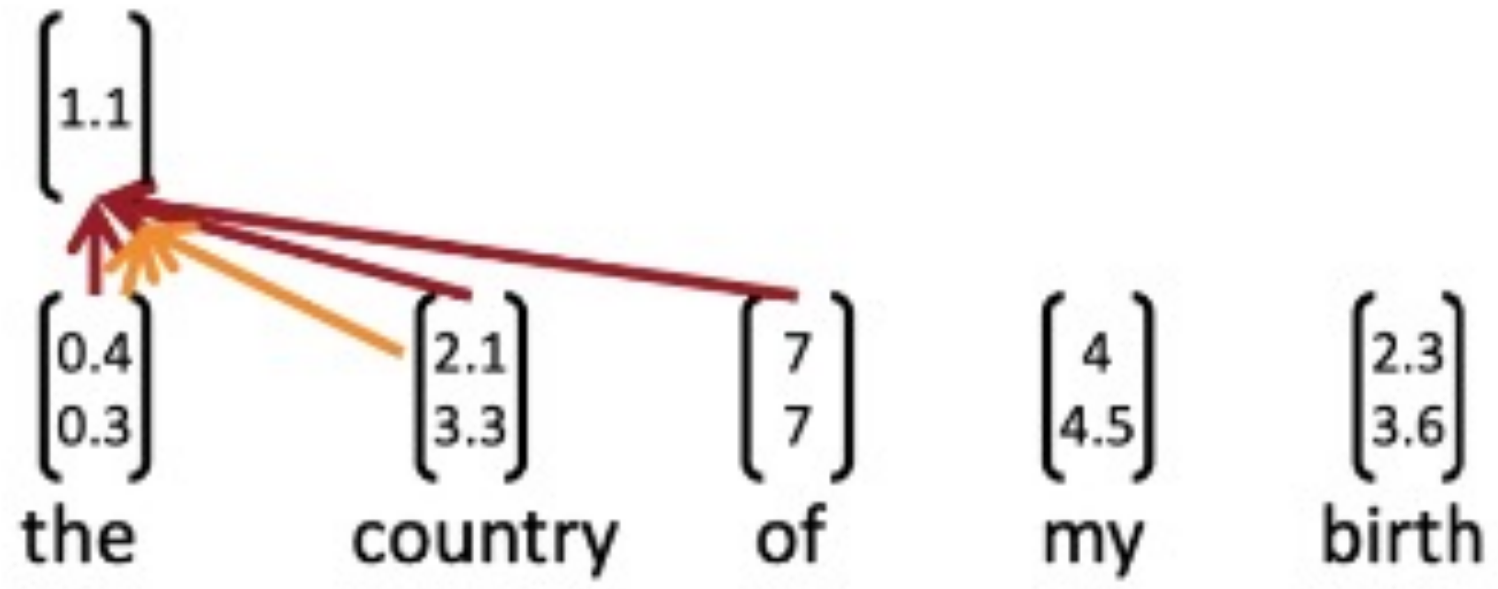
  - 주로 문장이 긍정적인지 부정적인지를 분류

Single Layer CNN for Sentence Classification

- Yoon Kim (2014): Convolutional Neural Networks for Sentence Classificaton

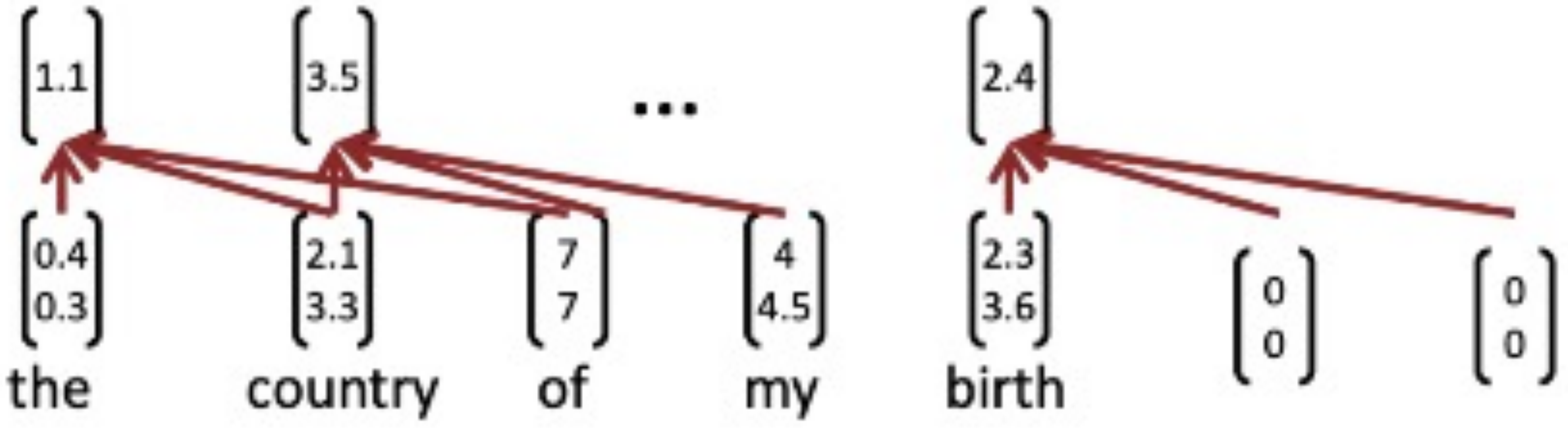
- Convolution 연산
  - Matrix 구조가 아니라 vector들을 이어붙여서 연산

  - 미리 학습된 단어 벡터
  - 문장: 벡터들을 이어붙인 것
  - Convolutional filter: h개의 단어들을 커버
    - Filter도 벡터다!



Single Layer CNN for Sentence Classification

- Yoon Kim (2014): Convolutional Neural Networks for Sentence Classificaton



- Convolution 연산: $c_i = f(\mathbf{w}^T \mathbf{x}_{i:i+h-1} + b)$
  - 함수 f: tanh
  - 모든 가능한 h 길이에 대해: $\{\mathbf{x}_{1:h}, \mathbf{x}_{2:h+1}, \ldots, \mathbf{x}_{n-h+1:n}\}$
  - 피쳐맵 결과: $\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}] \in \mathbb{R}^{n-h+1}$

Single Layer CNN for Sentence Classification

- Yoon Kim (2014): Convolutional Neural Networks for Sentence Classificaton

  - Pooling 연산: 피쳐맵 결과에서 max pooling
    - Idea: 가장 중요한 activation을 찾아내는 것

From feature map $\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}] \in \mathbb{R}^{n-h+1}$

Pooled single number: $\hat{c} = \max\{\mathbf{c}\}$

- Filter 여러 개 사용
- 다양한 h 길이를 사용
- Max 연산 → c의 길이는 상관없어짐

Single Layer CNN for Sentence Classification

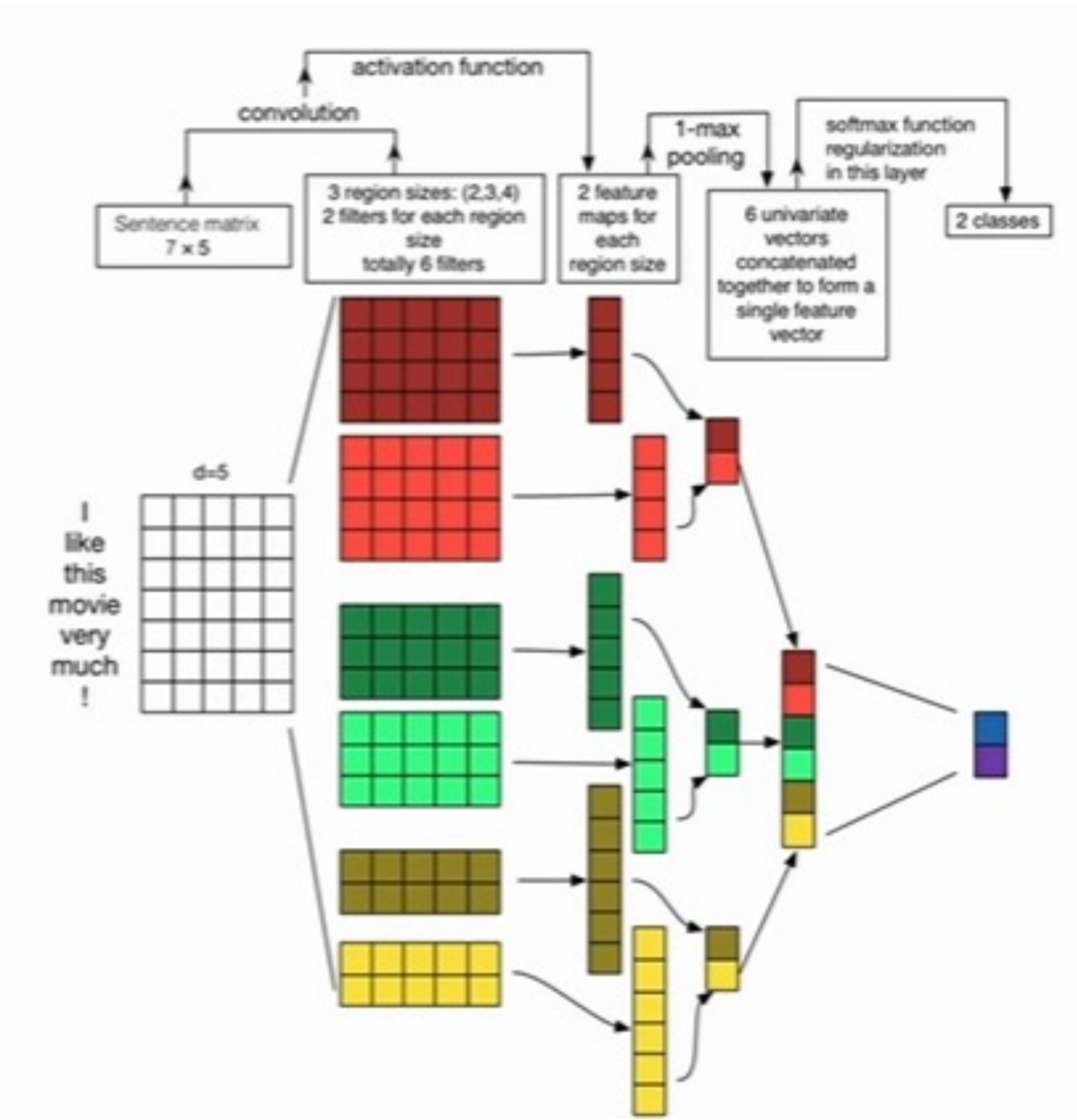
- Yoon Kim (2014): Convolutional Neural Networks for Sentence Classificaton

  - Classification after one CNN layer
    - Convolution – Max pooling → 최종 벡터

    - 필터 m 개 사용한 최종 벡터: $\mathbf{z} = [\hat{c}_1, \ldots, \hat{c}_m]$

    - Softmax 통해 최종 classification 수행: $y = softmax\left(W^{(S)}z + b\right)$

Single Layer CNN for Sentence Classification

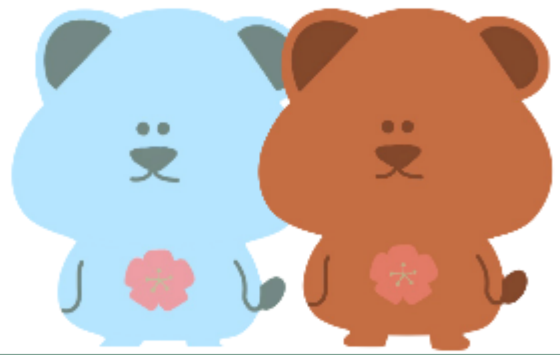


Single Layer CNN for Sentence Classification

- Yoon Kim (2014): Convolutional Neural Networks for Sentence Classificaton
  - Dropout: 2-4% accuracy 향상
  - L2 norm

If $\|W_{c\cdot}^{(S)}\| > s$, then rescale it so that: $\|W_{c\cdot}^{(S)}\| = s$

Various Toolkits



Various Toolkits

- Bag of Vectors

→ 간단한 classification problems에 유용

- Window Model

→ Single word classification에 유용 Ex) POS, NER

- CNNs

→ Classification 에 유용, GPU로 연산하기에 유용

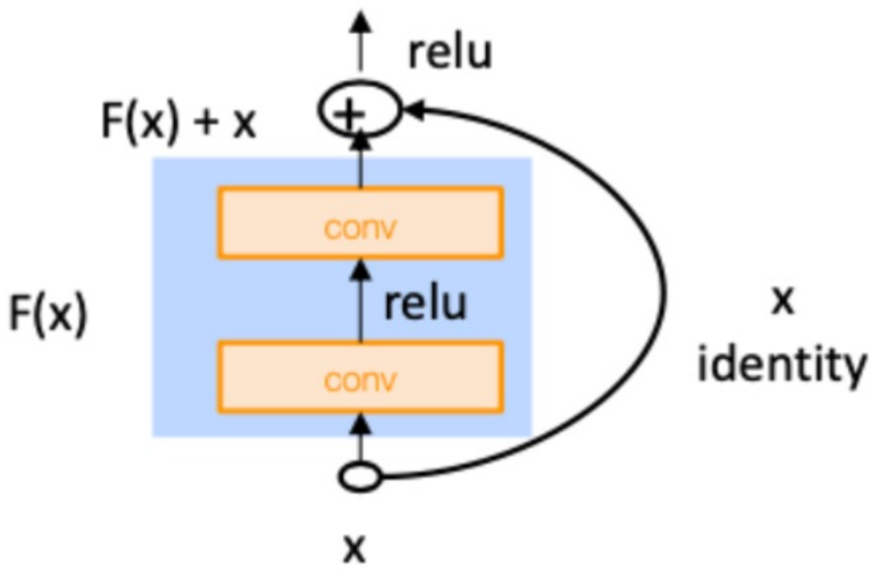
- RNNs

→ sequence tagging, classification , language model 에 유용

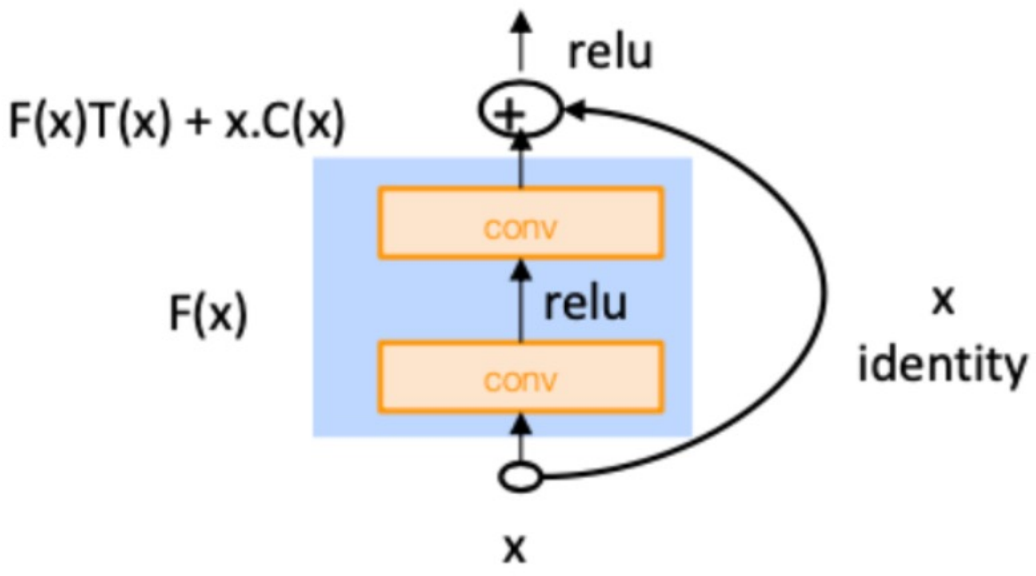
Additional Useful theories

Gated Units used Vertically

Key idea:
summing candidate update with shortcut (highway) connection



Residual block
(He et al. ECCV 2016)



Highway block
(Srivistava et al. NeurIPS 2015)

Additional Useful theories

Batch Normalization

- CNN에서 자주 사용됨
- 특정 batch에서의 convolution output을 mean = 0, unit variance로 rescale 해주는 작업

+ 모델들이 초기 parameter initialization에 덜 민감하도록 도움
+ learning rate을 tune하는데 더 쉬움

Additional Useful theories

1x1 (1) Convolutions

- Kernel size가 1인 Convolution Kernel
- 여러개의 Channel을 적은 수의 Channel로 map해줌

- Fully connected linear layer의 역할
  - Fully Connected Layer와 다르게 parameter 도 적어서 효율적임

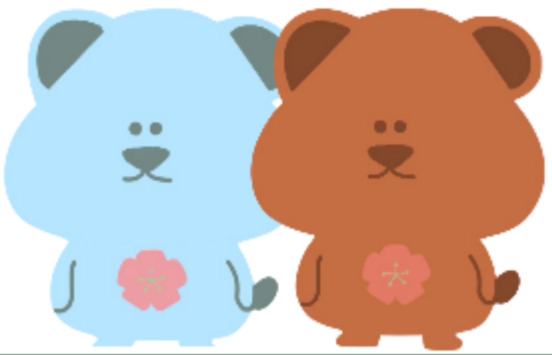
CNN의 쓰임 - NLP

번역 (Machine Translation)에 사용

- Kalchbrenner and Blunsom (2013)
- NMT의 가장 초기 성공적인 모델 중 하나
- Encoding시 CNN, Decoding 시 RNN 사용

\+ 단어 말고 Character 단위 CNN을 통해 LM 모델 만드는 시도도 있음
(다음주 세미나에 다뤄짐)

Very Deep CNN



VD-CNN for Text Classification

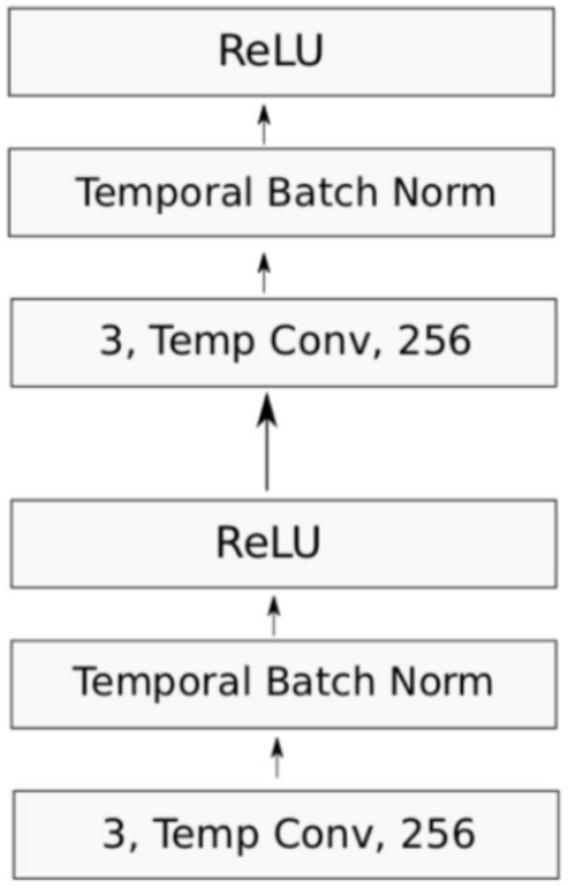
Key Question: Computer Vision 영역에서는 Layer를 깊게 쌓을수록 성공적인 것이 확인 되었는데, NLP에서도 똑같이 적용할 수 없을까?

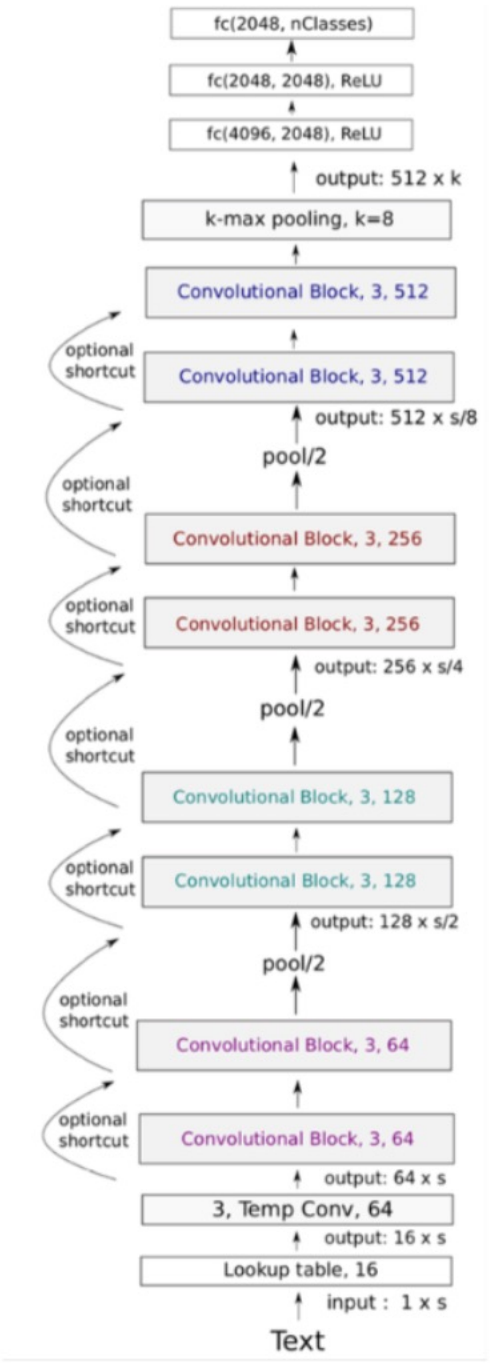
- VD-CNN (Very Deep Convolutional Networks)
- Conneau, Schwenk, Lecun, Barrault. EACL 2017
- Character 단위로 input을 받음

VD-CNN for Text Classification

- ResNet, VGGNet 와 형태가 유사

- 각 CONV 블록은 2개의 Batch Norm 과 ReLU를 적용한 Convolutional Layer 2개로 이루어짐
- CONV Kernel size = 3
- Padding으로 dimension 조절





VD-CNN Results

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.

Quasi-Recurrent Network

Q-RNN

Insight: RNNs are SLOW -> Make Faster?
Key Idea: Take the best and parallelizable parts of RNN & CNN

→ Q- RNN (Quasi-Recurrent Neural Networks)

+ Faster than RNN
+ Use depth as a substitute for true recurrence
- Need Deeper network to get as good performance as LSTM

→ Pseudo-recurrence

THANK YOU