



Lecture 11-ConvNets for NLP

☰ Lecture	lecture 11
☰ 속성	

[ConvNets for NLP]

[ConvNets for NLP]

From RNNs to CNN

RNN의 단점

CNN의 장점 및 단점

CNNs

1D convolution for text

Other notion

용어

Simple CNN for sentence Classification

Toolkit & ideas for NLP task

Model comparison

Toolkit

Ideas

Deep CNN for sentence classification

Structure

Result

Quasi-recurrent Neural Networks

From RNNs to CNN

RNN의 단점

"the country of my birth"에서 birth라는 단어를 알기 위해서는 처음부터 birth 앞에까지 모든 단어들이 필요함

often capture too much of last words in final vector

시간이 오래 걸리고 bottleneck problem

CNN의 장점 및 단점

- 장점

N-gram features를 얻을 수 있음

국소적인 패턴을 얻을 수 있음

RNN보다 빠르며 bottleneck problem을 해결

- 단점

특정 길이의 단어들로 벡터를 계산하는 것이 CNN으로

구가 문법적이든 아니든 언어적이거나 타당하지는 않다

CNNs

1D convolution for text

1D convolution for text

(zero) padding

θ	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
θ	0.0	0.0	0.0	0.0

Apply a filter (or kernel) of size 3

3	1	2	-3
-1	2	1	-3
1	1	-1	1

- Adjust the size of output sequence
- Preserve dimensions of data flowing across the layer

Multiple filters

θ	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
θ	0.0	0.0	0.0	0.0


Apply 3 filters of size 3

3	1	2	-3	1	0	0	1	1	-1	2	-1
-1	2	1	-3	1	0	-1	-1	1	0	-1	1
1	1	-1	1	0	1	0	1	0	2	2	1

- Increase output channels
- Capture various features
- The filters to be specialized for different domain

pooling

θ	-0.6	0.2	1.4
$t_{d,r}$	-1.0	1.6	-1.0
$d_{r,t}$	-0.5	-0.1	0.8
α_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
$g_{o,\theta}$	-0.5	-0.9	0.1



max p	0.3	1.6	1.4
ave p	-0.87	0.26	0.53



- Summarize the output of convolution
- Capture sparse signal
- Subsampling(down-sampling)
- Give some invariance to fluctuation of inputs

7

- 더 넓은 범위를 한번에 보고 싶다면

1. 더 큰 filter를 사용
2. 더 넓게 dilated convolution을 사용
3. CNN을 더 깊게 만들기

Other notion

Other notions

increase stride

0	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	0.0

Compress data

0,0,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,0	-0.5	-0.9	0.1

Apply 3 filters of size 3

3 1 2 -1	1 0 0 1	1 -1 2 -1
-1 2 1 -3	1 0 -1 -1	1 0 -1 3
1 1 -1 1	0 1 0 1	0 2 2 1

local max pooling

0,0,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	-0.5	-0.9	0.1
0	-inf	-inf	-inf

Compress representation

Focus on more local characteristics

0,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,0	-0.5	-0.9	0.1

k-max pooling

0,0,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	-0.5	-0.9	0.1
2-max p	-0.2	1.6	1.4
	0.3	0.6	1.2

Compress representation

Keep more information than 1-max pooling

dilated convolution

0	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	0.0

Compress data

0,0,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	-1.6	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	-0.5	-0.9	0.1
1,1,2	0.3	0.6	0.9
1,1,2	0.3	0.6	0.9
1,1,2	0.3	0.6	0.9

Apply 3 filters of size 3

3 1 2 -1	1 0 0 1	1 -1 2 -1
-1 2 1 -3	1 0 -1 -1	1 0 -1 3
1 1 -1 1	0 1 0 1	0 2 2 1

Use a relatively small number of parameters to have a larger receptive field

8

용어

Channel, Filter, Kernel, Stride, Padding, Pooling, Feature Map, Activation Map

Simple CNN for sentence Classification

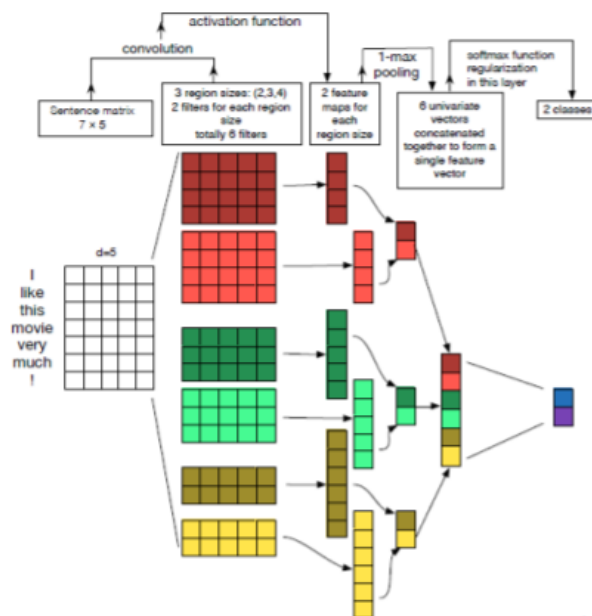
- 목표 : Sentence Classification
- 성능 : 하나의 CNN layer와 pooling을 사용해서 꽤 높은 성능을 보임
- 구조

Dropout과 Max-norm regularization을 사용

multiple filters(100) with various widths(various n-grams)

Multi-channel input idea(word vector와 word vector 복사본을 만들어 그 중 하나는 고정, 다른 하나는 fine-tuned)

Model Architecture



<https://arxiv.org/pdf/1510.03820.pdf> 14

Toolkit & ideas for NLP task

Model comparison

Good baseline Model



Bag of vectors : 문장의 word embedding의 평균을 이용해 text classification



Window Model : good for single word classification for problems that do not need wide context

Advanced Model



CNNs : good for representing sentence meaning(sentence classification)



RNNs : good for sequence tagging/classification/language models, but slower than CNNs

Toolkit

- Gated units used vertically

LSTM과 GRU에서 gate를 연결해주는 개념에서 차용한 Skipping

Residual Net과 Highway Net이 있음

- 1*1 convolutions

먼저 1*1 Convolution을 사용하면 필터의 개수가 몇 개 인지에 따라 output의 dimension은 달라지지만, 원래 가로 세로의 사이즈는 그대로 유지된다.

그래서 filter 의 개수를 원래 input의 dimension 보다 작게 하면, dimension reduction의 효과가 난다.

원래 image 쪽에서 Convolution layer는 "Spatial Relation"을 고려하여 이 image가 어떤 image인지 패턴을 통해 파악하는 용도인데, 1*1 사이즈를 사용한다는 것은 한 픽셀만 고려하기 때문에 패턴 인식보다는 dimension reduction이라는 전처리 용도로 생각해야 한다.

Dimension reduction을 이용하기 위해 1*1 conv 의 개수를 줄이면, activation의 depth가 작아져서 filter의 총 parameter의 개수가 감소한다.

이 형태를 bottleneck 구조라고 하는데, dimension reduction을 한 뒤 모든 연산을 하고 다시 filter의 갯수를 늘려서 고차원으로 늘리는 방법을 이용하기 때문에 bottleneck이라고 부른다.

출처: <https://yunmap.tistory.com/entry/전산학특강-CS231n-1X1-Convolution-이란>

- Batch Normalization

internal covariate shift 문제를 해결하기 위해 배치 단위로 정규화를 시켜주는 것

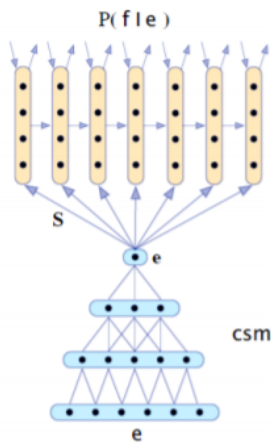
학습속도가 빨라지고 초기화에 덜 민감해진다

과적합을 예방

복잡도(계산량)을 늘림

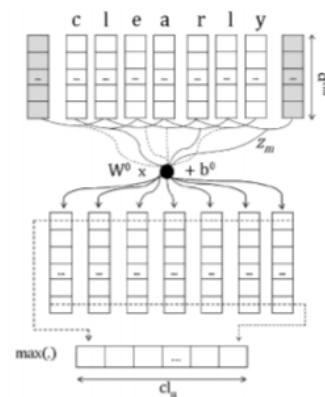
Ideas

Encoder(CNN)-Decoder(RNN) networks



- Shrink the input data continuously by stacking up convolutional layers
- Take the final pulled vector as a sentence representation

Character-level representations



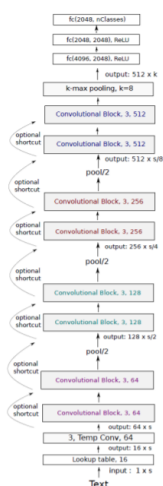
NEXT CHAPTER

20

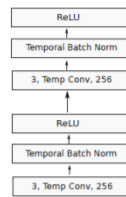
Deep CNN for sentence classification

Structure

- Starting point : sequence models have been very dominant in NLP but all the models are basically not very deep
- Works from the character-level
- Result is constant size, since text is truncated or padded
- Local max pooling

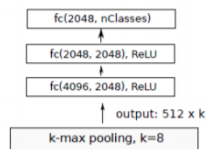


Convolutional block



- Two conv. Layers, each one followed by a temporal BN and an ReLU activation
- Padding for preservation of temporal resolution
- Use small size filters (3) so that networks can be deep and get more expressive in a few parameters

Classification tasks



- Temporal resolution of the output is first **down-sampled** to a fixed dimension using **k-max pooling (extracts the k most important features indep. position)**
- the 512 x k resulting features are transformed into a single vector (Flatten) which is the input to a three FC layer with ReLU
- The number of output neurons in there depends on clf task

23

Result

- ✓ deeper networks are better
- ✓ However, if it is too deep, its performance will be degraded
- ✓ Shrinkage method – “MaxPooling” is easier

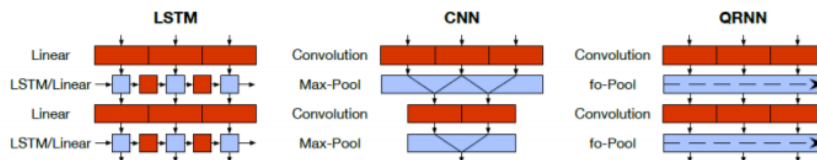
Quasi-recurrent Neural Networks

Quasi-recurrent Neural Networks

<https://arxiv.org/abs/1611.01576>

QRNNs : have the advantages of both CNN and RNN

- Parallel computation across both timestep & minibatch dimensions
- Output depend on the overall order of elements in the sequence



LSTM notation

$$\begin{aligned} f_t &= \sigma(W_{xf}^T \cdot x_t + W_{hf}^T \cdot h_{t-1} + b_f) \\ i_t &= \sigma(W_{xi}^T \cdot x_t + W_{hi}^T \cdot h_{t-1} + b_i) \\ o_t &= \sigma(W_{xo}^T \cdot x_t + W_{ho}^T \cdot h_{t-1} + b_o) \\ g_t &= \tanh(W_{xg}^T \cdot x_t + W_{hg}^T \cdot h_{t-1} + b_g) \\ c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\ h_t &= o_t \odot \tanh(c_t) \end{aligned}$$

QRNN notation

$$\begin{aligned} z_t &= \tanh(W_z^1 x_{t-k+1} + W_z^2 x_{t-k+2} + \dots + W_z^k x_t) \\ f_t &= \sigma(W_f^1 x_{t-k+1} + W_f^2 x_{t-k+2} + \dots + W_f^k x_t) \\ o_t &= \sigma(W_o^1 x_{t-k+1} + W_o^2 x_{t-k+2} + \dots + W_o^k x_t) \end{aligned}$$

$$\begin{aligned} Z &= \tanh(W_z * X) \\ F &= \sigma(W_f * X) \\ O &= \sigma(W_o * X) \end{aligned}$$

(*) Denotes a **masked convolution** along the timestep dimension

"Dynamic average pooling"

$$h_t = f_t \odot h_{t-1} + (1 - f_t) \odot z_t$$

function controlled by gates that can **mix states across timesteps**, but which **acts independently on each channel** of the state vector

Parallelism across channels

Parallelism across timesteps

