Deep Learning for Text Mining

Part 3. Convolutional Neural Networks (CNN)

Yiming Yang
Language Technologies Institute
Carnegie Mellon University
www.cs.cmu.edu/~yiming

Outline

- Motivation
- CNN Building Blocks
- Optimization Algorithms
- Applications in Text Mining

Outline

- CNN Building Blocks
  - Convolution
  - Striding
  - Padding
  - Pooling
  - Dilation
- Optimization Algorithms
- Application to Text Mining

3

What is Convolution?

Dictionary

Search for a word

con·vo·lu·tion

/ˌkänvəˈlo͞oSHən/

noun

1. a thing that is complex and difficult to follow.
   "the convolutions of farm policy"
   synonyms: complexity, intricacy, complication, twist, turn, entanglement, contortion; More
2. a coil or twist, especially one of many.
   "crosses adorned with elaborate convolutions"
   synonyms: twist, turn, coil, spiral, twirl, curl, helix, whorl, loop, curlicue, kink, sinuosity; More

4

What is Convolution?

- Wikipedia https://en.wikipedia.org/wiki/Convolution
  - **convolution** is a mathematical operation on two functions (*f* and *g*) to produce a third function that expresses how the shape of one is modified by the other.

$$\boxed{(f * g)(t) \triangleq \int_{-\infty}^{\infty} f(\tau) g(t-\tau)\, d\tau.}$$

An equivalent definition is (see commutativity):

$$(f * g)(t) \triangleq \int_{-\infty}^{\infty} f(t-\tau) g(\tau)\, d\tau.$$

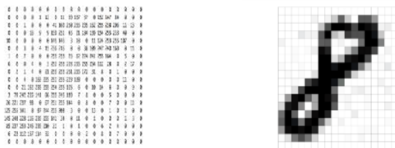
  - Essentially, it is an operation over f and g by taking a weighted sum of f (or g) using g (or f) as the weights.

Convolutional Neural Networks (CNNs)

- A type of neural networks
- Popular in image/video recognition, text mining, recommendation, etc.
- Inspired by biological processes
  - The *receptive (activated) filed* of cortical neurons can be approximated mathematically by a *convolution operation.*

Convolution Operations over an Image

- Ujjwal Karn https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/
  - Fig 4. Every image is an input matrix of pixel values (corresponding to function *f*)

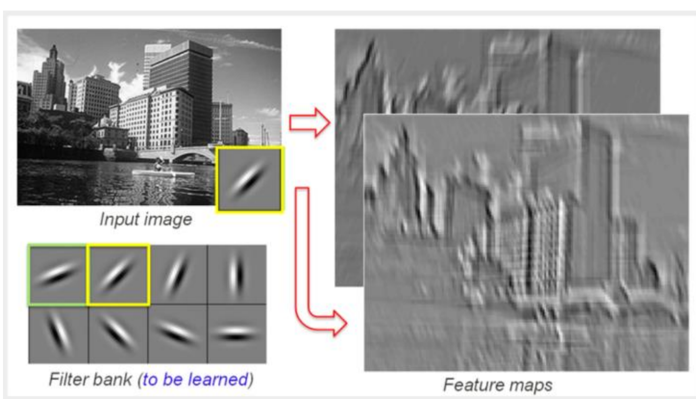


  - Fig 6. Each filter (corresponding to function *g*) applies to a local region over the entire input.

7

Convolution Operations over an Image (cont'd)

- Ujjwal Karn https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/



8

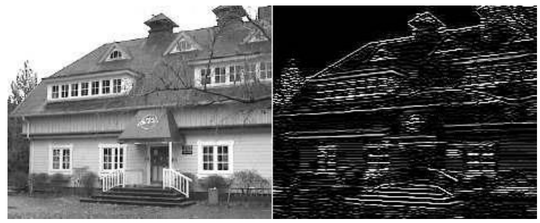
Convolution Operations over an Image (cont'd)

- AI Shack http://aishack.in/tutorials/image-convolution-examples/
- Blur Filter (local average)

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

- Horizontal Line Filter

-1	-1	-1
2	2	2
-1	-1	-1

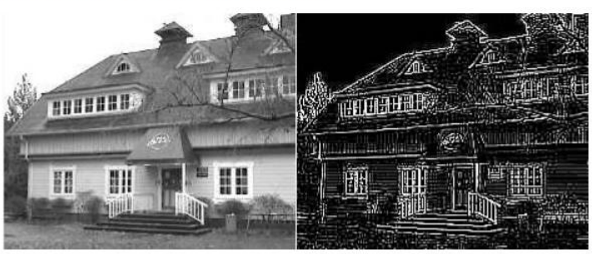


Convolution Operations over an Image (cont'd)

- AI Shack http://aishack.in/tutorials/image-convolution-examples/
- Edge Filter

-1	-1	-1
-1	8	-1
-1	-1	-1

Below result I got with edge detection:



Example of a Simple CNN [from ujjwalkarn, 2016]

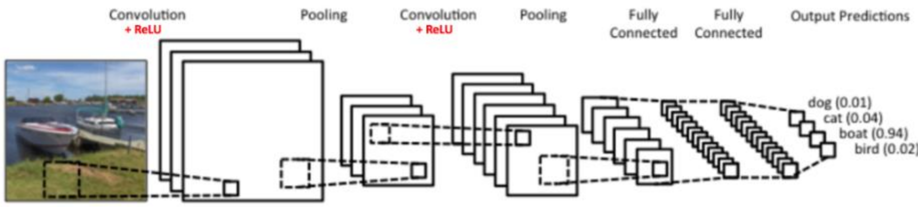


Figure 3: A simple ConvNet. Source [5]

- Automatically learn the filters (kernels) based on labeled training data
- Extracting local and lower-dimensional features from input data
- Computationally more efficient than MLPs or RNNs

11

Outline

- Motivation
- CNN Building Blocks
  - Convolution
  - Striding
  - Padding
  - Pooling
- Optimization Algorithms
- Applications in Text Mining

12

Terminology & Notation

- Input X (or function f in our introduction of convolution)
  - Numerical representation of an image, a sentence, a time series, etc.
- Filter (kernel) W (or function g in our introduction of convolution)
- Feature Map: the output of convolution

13

Image Convolution (a toy example)

- Apply a 3-by-3 filter to the 2-D input data of an image

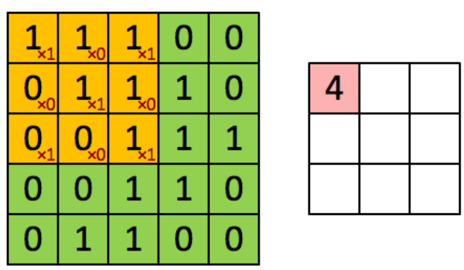
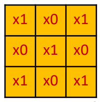


Image **Convoluted Features**

Animation: http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution
More references [here]()

14

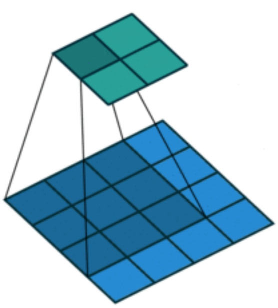
Image Convolution (toy example)

- The yellow patch 

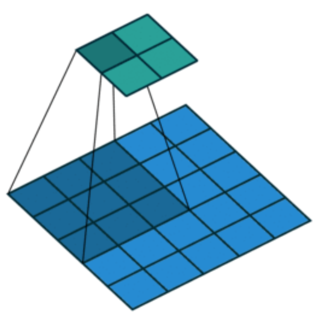
is called a **filter** (or **kernel** or **feature detector**).
- Denoting by n the input height/length, k the filter size, the convolution (a.k.a. **feature mapping**) reduces input volume $n^2 = 25$ to $k^2 = 9$ convoluted features.
- The filter has k^2 parameters which will be *automatically learned* by the system.
- Compared to a fully connected network for the $25 \rightarrow 9$ mapping, we will need to learn 25×9 parameters in the network.
- Therefore, convolution is a *computationally more feasible way* for extracting lower dimensional features from input data.

Stride size: a hyper parameter of convolution

- The number of units to shift by the filter at each step



Striding of size 1 ("no striding")

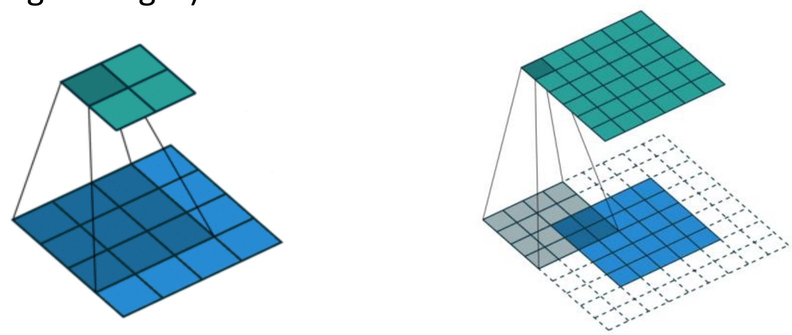


Striding of size 2

Animation: https://github.com/vdumoulin/conv_arithmetic

Padding

- adding zeros around the input image (for desirable output size, or focusing on edges)



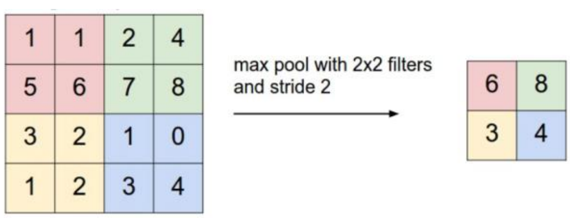
No padding *Padding*

Source: https://github.com/vdumoulin/conv_arithmetic

17

Image Pooling

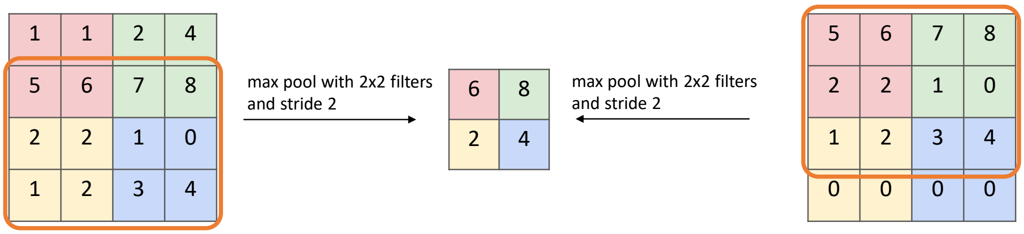
- Fixed operation (average or max) over each local region
- Typically used after convolution for further dimensionality reduction



Max pooling with 2×2 Filter.
Source: [*http://cs231n.github.io/convolutional-networks/#pool*](http://cs231n.github.io/convolutional-networks/#pool)

18

Image Pooling: Capturing the Local Invariance

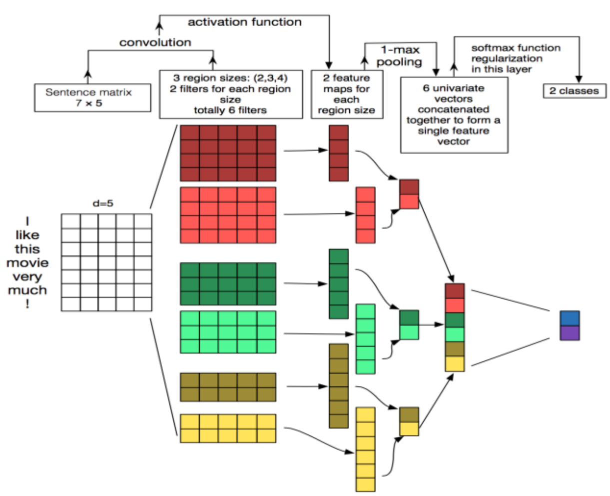


Max pooling with 2×2 Filter.
Source: http://cs231n.github.io/convolutional-networks/#pool

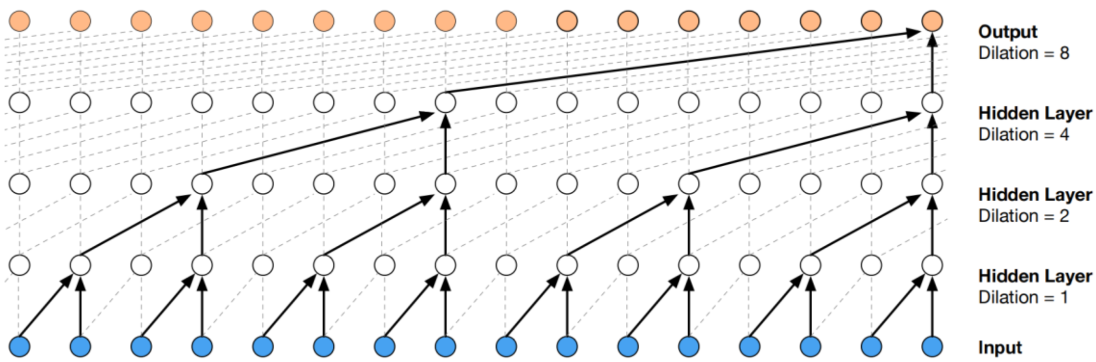
Comparing the images on the left and the right: The local pattern (orange box) is shifted upwards by 1 unit on the right, but the output of pooling operation remain unchanged.

Applying convolution/pooling to text

- Use word embedding to obtain the input "image" (1-D)
- Use convolution filters of **size $m \times d$**
  - **m is the number of words a filter takes into account(usually 1-5, like n-gram)**
  - **d is the size of word embedding**



Dilated convolution [Oord, et al., 2016]



- Use dilation to model input granularity or to skip certain input dimension (dilation=1 equivalent to conventional convolution)
- Stack multiple dilated convolutions with different dilations enlarges the receptive field without exploding the number of parameters

The receptive field

- http://blog.christianperone.com/2017/11/the-effective-receptive-field-on-cnns/

  The receptive field in Convolutional Neural Networks (CNN) is the region of the input space that affects a particular unit of the network. Note that this input region can be not only the input of the network but also output from other units in the network, therefore this receptive field can be calculated relative to the input that we consider and also relative to the unit that we are taking into consideration as the “receiver” of this input region.

Outline

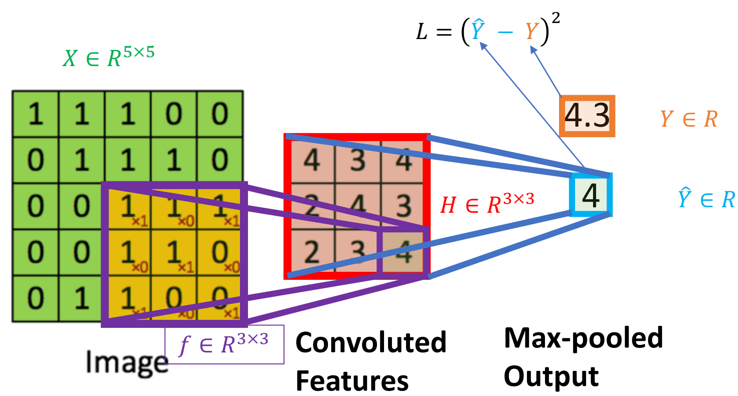
- Motivation
- CNN Building Blocks
- Optimization Algorithms
- Applications in Text Mining

Optimization Algorithms

- Loss function for CNN:
  - $L(y, \hat{y}) = L(y, f_W(x))$
  - f_W: contains operations of convolution, padding, pooling, …
  - W: all model parameters
- Optimization
  - $\frac{\partial f_W}{\partial W}$ is differentiable for convolution, padding, pooling, …
  - Optimize through backpropagation and gradient descent
  - Acceleration on GPU

Optimization for CNN

- A toy system
- $X \in R^{5\times5}$: the input matrix
- $H \in R^{3\times3}$: the convoluted feature matrix;
  - Indexed by (i, j)
- $\hat{Y} \in R$: the model output
- $Y \in R$: ground truth
- $f \in R^{3\times3}$: the filter matrix, model parameter to be learned
  - Indexed by (k, l)



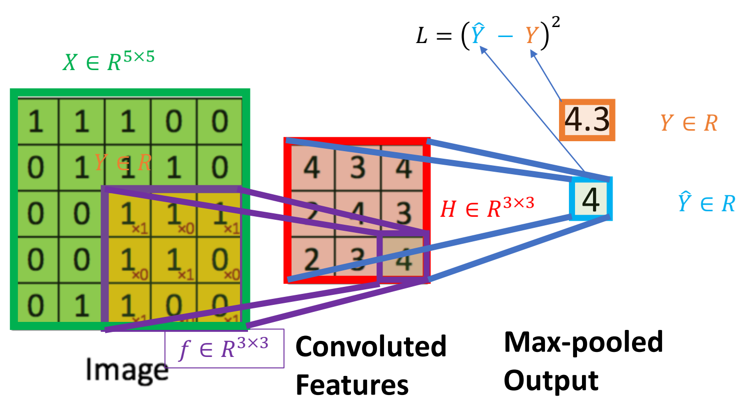
Optimization for CNN

- Train the system with stochastic gradient descent
- Updating rule at step k

$$f^{(k)} := f^{(k-1)} - \eta_k \nabla \left(\frac{1}{|B|} \sum_{i \in B} L_i(f^{(k-1)}) \right)$$

  - L_i: loss on i'th example
  - B: mini-batch of (x_i, y_i)
  - η_k: learning rate at step k
- Chain rule

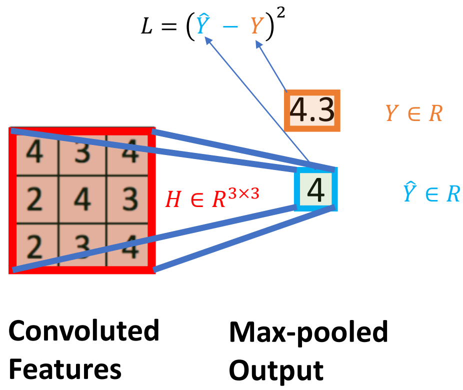
$$\frac{\partial L}{\partial f} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial H} \frac{\partial H}{\partial f}$$



Backpropagation for Max-pooling

- First, we need to compute $\frac{\partial L}{\partial H} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial H}$
- $\frac{\partial L}{\partial \hat{y}}$ is trivial, $\frac{\partial L}{\partial \hat{y}} = 2(\hat{Y} - Y)$
- $\frac{\partial \hat{y}}{\partial H_{11}} = 1$
- $\frac{\partial \hat{y}}{\partial H_{12}} = 0$
- $\frac{\partial \hat{y}}{\partial H_{21}} = 0$
- $\frac{\partial \hat{y}}{\partial H_{22}} = 1$
- …

We need to remember the index of the maximum in H



Max pooling with 2×2 Filter.
Source: http://cs231n.github.io/convolutional-networks/#pool

27

Backpropagation for convolutional

- We already have the gradient $\frac{\partial L}{\partial H}$, now we need the gradient of L w.r.t f_{kl} , i.e.,

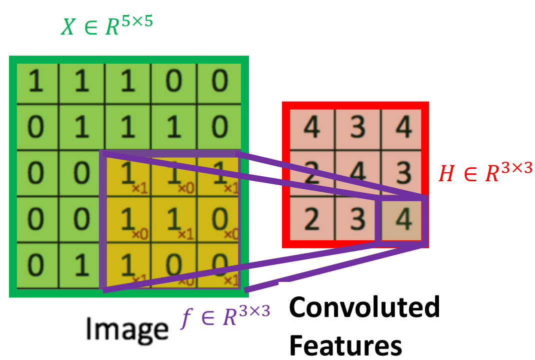
$$\frac{\partial L}{\partial f_{kl}} = \sum_{i,j\in\{1,2,3\}} \frac{\partial L}{\partial H_{ij}} \frac{\partial H_{ij}}{\partial f_{kl}}$$

$$= \sum_{i,j\in\{1,2,3\}} \boxed{\frac{\partial L}{\partial H_{ij}}} \boxed{X_{i+k-1,j+l-1}}$$

Act like the filter

Sliding the filter over X

Another convolutional operation



Source: http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution
More references [here]()

28

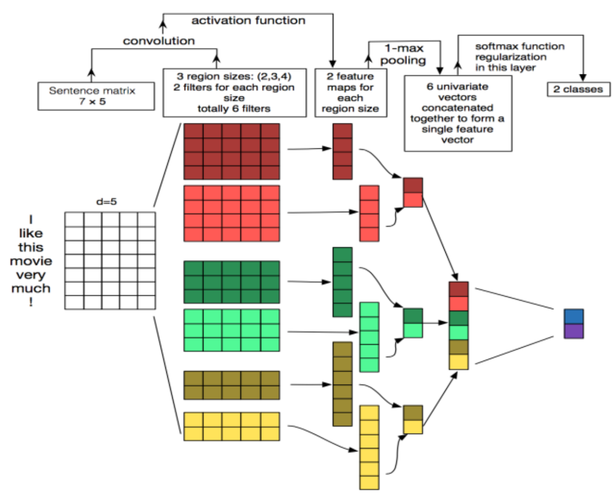
Outline

- Motivation
- CNN Building Blocks
- Optimization Algorithms
- Applications in Text Mining
  - Text Classification
  - Language Modeling

29

Text Classification

- Why CNN?
  - TF-IDF features assume independency among words.
  - CNN filters/pooling capture local dependencies and shift-invariant features.



Typical CNN structure for text classification [Kim 2014]

30

Benchmark evaluation results in error (smaller is better)

Table 4: Testing errors of all the models. Numbers are in percentage. “Lg” stands for “large” and “Sm” stands for “small”. “w2v” is an abbreviation for “word2vec”, and “Lk” for “lookup table”. “Th” stands for thesaurus. ConvNets labeled “Full” are those that distinguish between lower and upper letters

Model	AG	Sogou	DBP.	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
BoW	11.19	7.15	3.39	7.76	42.01	31.11	45.36	9.60
BoW TFIDF	10.36	6.55	2.63	6.34	40.14	28.96	44.74	9.00
ngrams	7.96	2.92	1.37	**4.36**	43.74	31.53	45.73	7.98
ngrams TFIDF	**7.64**	**2.81**	**1.31**	4.56	45.20	31.49	47.56	8.46
Bag-of-means	**16.91**	**10.79**	**9.55**	**12.67**	**47.46**	**39.45**	**55.87**	**18.39**
LSTM	13.94	4.82	1.45	5.26	41.83	29.16	40.57	6.10
Lg. w2v Conv.	9.92	4.39	1.42	4.60	40.16	31.97	44.40	5.88
Sm. w2v Conv.	11.35	4.54	1.71	5.56	42.13	31.50	42.59	6.00
Lg. w2v Conv. Th.	9.91	-	1.37	4.63	39.58	31.23	43.75	5.80
Sm. w2v Conv. Th.	10.88	-	1.53	5.36	41.09	29.86	42.50	5.63
Lg. Lk. Conv.	8.55	4.95	1.72	4.89	40.52	29.06	45.95	5.84
Sm. Lk. Conv.	10.87	4.93	1.85	5.54	41.41	30.02	43.66	5.85
Lg. Lk. Conv. Th.	8.93	-	1.58	5.03	40.52	28.84	42.39	5.52
Sm. Lk. Conv. Th.	9.12	-	1.77	5.37	41.17	28.92	43.19	5.51
Lg. Full Conv.	9.85	8.80	1.66	5.25	38.40	29.90	40.89	5.78
Sm. Full Conv.	11.59	8.95	1.89	5.67	38.82	30.01	40.88	5.78
Lg. Full Conv. Th.	9.51	-	1.55	4.88	38.04	29.58	40.54	5.51
Sm. Full Conv. Th.	10.89	-	1.69	5.42	**37.95**	29.90	40.53	5.66
Lg. Conv.	12.82	4.88	1.73	5.89	39.62	29.55	41.31	5.51
Sm. Conv.	15.65	8.65	1.98	6.53	40.84	29.84	40.53	5.50
Lg. Conv. Th.	13.39	-	1.60	5.82	39.30	**28.80**	40.45	**4.93**
Sm. Conv. Th.	14.80	-	1.85	6.49	40.16	29.84	**40.43**	5.67

Red: worst performance on each dataset

Parameters/hyperparameters have a large influence!

Blue: best performance on each dataset

07/23/2017 @Yiming Yang, lecture on Deep Learning for Text Mining 31

31

Parameters

- Parameter **matters**!
- Model parameters include
  - Convolution/pooling size
  - Striding/padding size
  - Dilation scope
- Optimization Algorithms
  - SGD/Ada-grad
  - Mini-batch size in SGE
- In text classification
  - Embedding methods

07/23/2017 @Yiming Yang, lecture on Deep Learning for Text Mining 32

32

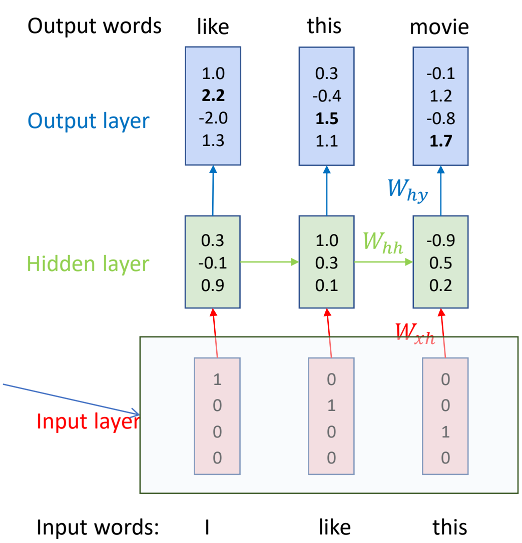
Embedding Strategies

- Pre-training vs. No pre-training
  - If an unlabeled large corpus is available, pretraining of word embedding can be used for faster fine-tuning and better generalization ability
- Word-level/Character-level embedding
  - Word embedding can be initialized with pretrained embedding, missing words are randomly initiated
  - Too many words are unknown in pretraining, character-level embedding may be used

Language modeling with RNN

- Issue: Word embedding (input vectors) cannot handle out-of-vocabulary words

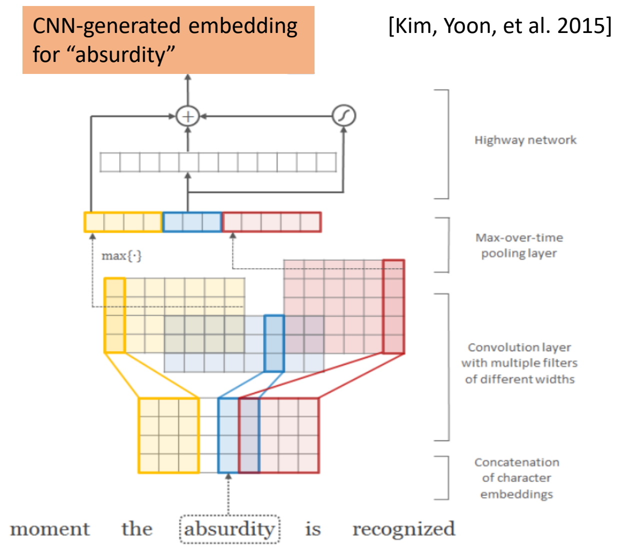
Word-level embedding: The parameter are stored in a look-up table of size $|V| \times d$, where $|V|$ is the vocabulary size, and d is embedding dimension



Language modeling with character-level CNN

[Kim, Yoon, et al. 2015]

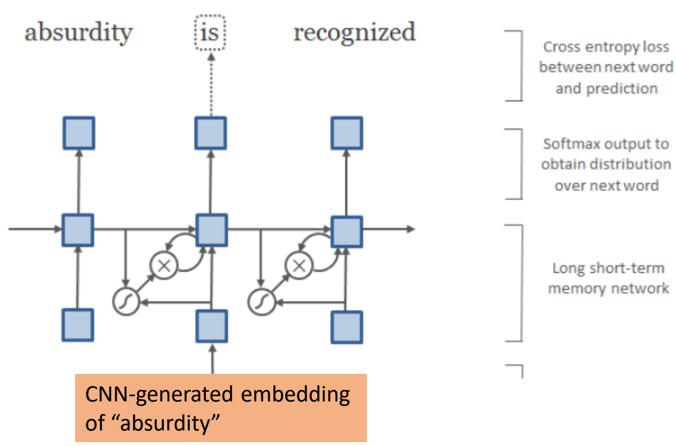
- Using CNN to learn a mapping
  - from the input vectors of character-level embedding
  - to the output vector of the embedded word



[Kim, Yoon, et al. 2015]

Language Modelling with CNN+LSTM

- The CNN-generated embedding is then fed into a LSTM for predicting the current word given the past context.



Performance on language modeling [Yoon Kim et al. AAAI 2016]

	PPL	Size
LSTM-Word-Small	97.6	5 m
LSTM-Char-Small	92.3	5 m
LSTM-Word-Large	85.4	20 m
LSTM-Char-Large	78.9	19 m
KN-5 (Mikolov et al. 2012)	141.2	2 m
RNN† (Mikolov et al. 2012)	124.7	6 m
RNN-LDA† (Mikolov et al. 2012)	113.7	7 m
genCNN† (Wang et al. 2015)	116.4	8 m
FOFE-FNNLM† (Zhang et al. 2015)	108.0	6 m
Deep RNN (Pascanu et al. 2013)	107.5	6 m
Sum-Prod Net† (Cheng et al. 2014)	100.0	5 m
LSTM-1† (Zaremba et al. 2014)	82.7	20 m
LSTM-2† (Zaremba et al. 2014)	78.4	52 m

Table 3: Performance of our model versus other neural language models on the English Penn Treebank test set. PPL refers to perplexity (lower is better) and size refers to the approximate number of parameters in the model. KN-5 is a Kneser-Ney 5-gram language model which serves as a non-neural baseline. †For these models the authors did not explicitly state the number of parameters, and hence sizes shown here are estimates based on our understanding of their papers or private correspondence with the respective authors.

07/23/2017 @Yiming Yang, lecture on Deep Learning for Text Mining 37

37

[Kim, Yoon, et al. 2015]

CNN-generated embedding

	In Vocabulary					Out-of-Vocabulary		
	while	*his*	*you*	*richard*	*trading*	*computer-aided*	*misinformed*	*looooook*
LSTM-Word	*although* *letting* *though* *minute*	*your* *her* *my* *their*	*conservatives* *we* *guys* *i*	*jonathan* *robert* *neil* *nancy*	*advertised* *advertising* *turnover* *turnover*	– – – –	– – – –	– – – –
LSTM-Char (before highway)	*chile* *whole* *meanwhile* *white*	*this* *hhs* *is* *has*	*your* *young* *four* *youth*	*hard* *rich* *richer* *richter*	*heading* *training* *reading* *leading*	*computer-guided* *computerized* *disk-drive* *computer*	*informed* *performed* *transformed* *inform*	*look* *cook* *looks* *shook*
LSTM-Char (after highway)	*meanwhile* *whole* *though* *nevertheless*	*hhs* *this* *their* *your*	*we* *your* *doug* *i*	*eduard* *gerard* *edward* *carl*	*trade* *training* *traded* *trader*	*computer-guided* *computer-driven* *computerized* *computer*	*informed* *performed* *outperformed* *transformed*	*look* *looks* *looked* *looking*

Table 6: Nearest neighbor words (based on cosine similarity) of word representations from the large word-level and character-level (before and after highway layers) models trained on the PTB. Last three words are OOV words, and therefore they do not have representations in the word-level model.

07/23/2017 @Yiming Yang, lecture on Deep Learning for Text Mining 38

38

Comparison with RNN-based models

- RNN
  - Long-term dependency
  - Sequential computation (slow)
- CNN
  - N-gram-like models
  - Computation is easier to be parallelized (faster)
- Can be jointly used in combination, or by each alone.

References

- An Intuitive Explanation of Convolutional Neural Networks, ujjwalkarn, 2016 https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/
- AI Shack http://aishack.in/tutorials/image-convolution-examples/
- http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution
- https://github.com/vdumoulin/conv_arithmetic
- http://cs231n.github.io/convolutional-networks/#pool
- http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/
- Convolutional Neural Networks for Sentence Classification. [Kim 2014]
- Character-Aware Neural Language Models. [Kim 2015]
- http://papers.nips.cc/paper/5782-character-level-convolutional-networks-for-text-classification.pdf [Zhang 2015]