Convolutional Neural Network for Natural Language Processing

Hyunjoong Kim

soy.lovit@gmail.com

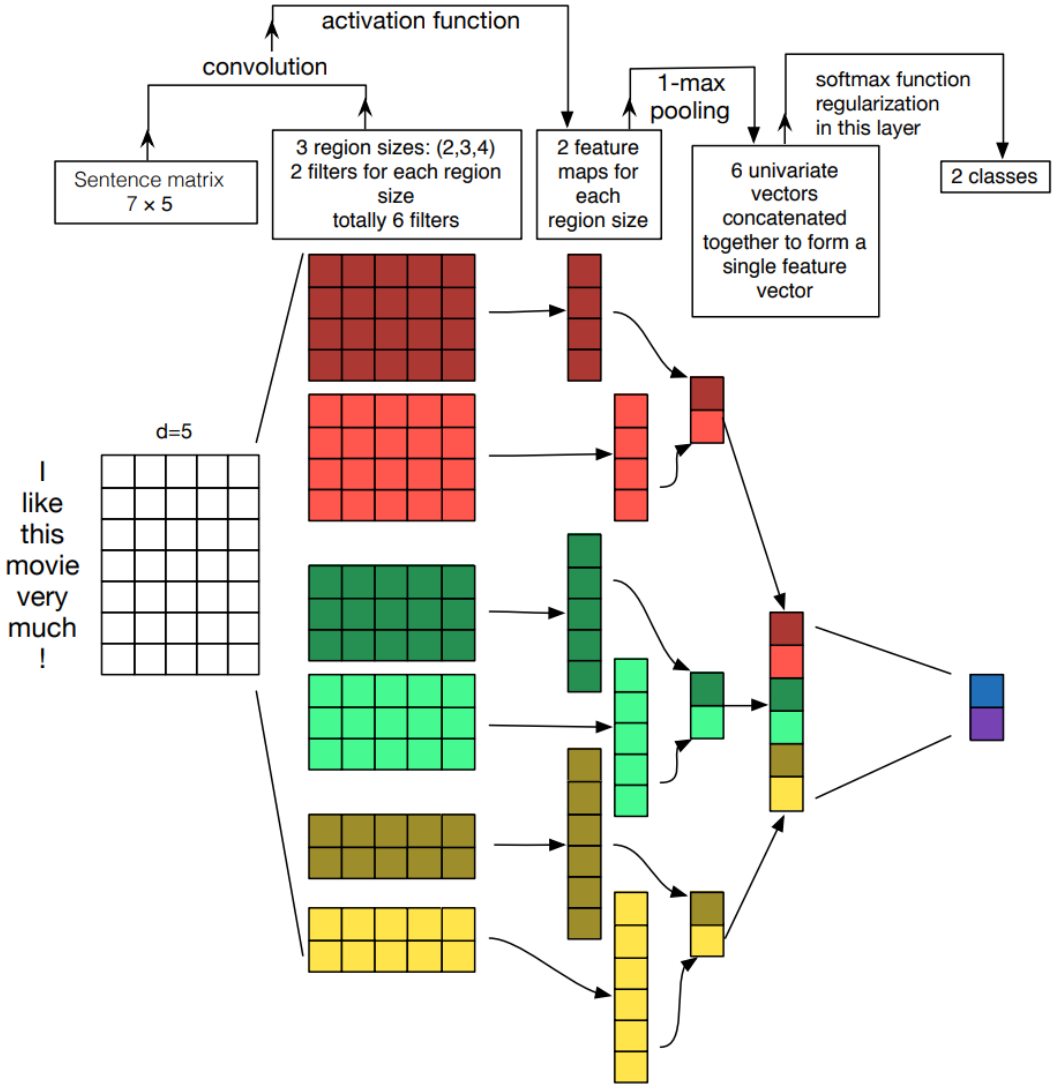
github.com/lovit

Convolutional Neural Network

- Sentence classification 을 위하여 CNN 이 이용되었습니다.
- CNN 은 (n, m, c) 형식의 이미지를 input 으로 가정합니다.
- Sentence 를 (n, m, c) 으로 표현할 수 있습니다.

Word – level convolutional neural network

- 5 차원으로 표현된 word embedding 벡터를 이어붙이면 $(n, d, 1)$ 의 이미지를 만들 수 있습니다.
  - n : 문장 길이
  - d : word embedding dimension
- Filters 는 정사각형이 아닌 $(k, d, 1)$ 의 모양으로 만듭니다.
  - n-gram 역할을 합니다.



Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*.

Word – level convolutional neural network

- 문장은 길이가 다르지만, CNN 의 input 은 고정된 크기의 이미지입니다.
  - 길이가 짧은 문장에 대하여, 뒷부분을 zero vector 로 처리합니다.
  - 크기가 같은 input image 형식이 됩니다.

I					
like					
this					
movie					
very					
much					
.					
[EMPTY]	0	0	0	0	0

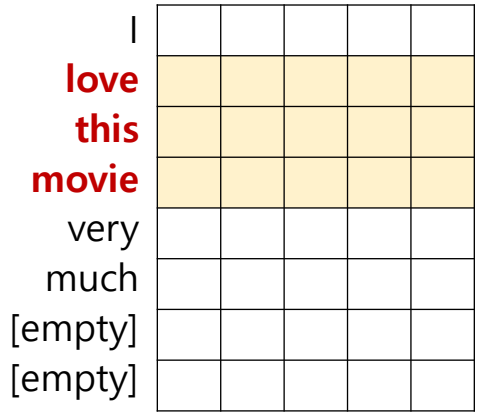
Word – level convolutional neural network

- (Yoon Kim, 2014) 는 아주 간단한 구조의 CNN 구조를 제안했습니다.
  - 길이가 각각 3, 4, 5 인 필터 100 개 (총 300 개)로 이뤄진 하나의 convolution layer 만을 이용합니다.
  - 각 필터에 1 – max pooling 이 적용되어 300 차원의 fully connected layer 의 input 이 만들어집니다.

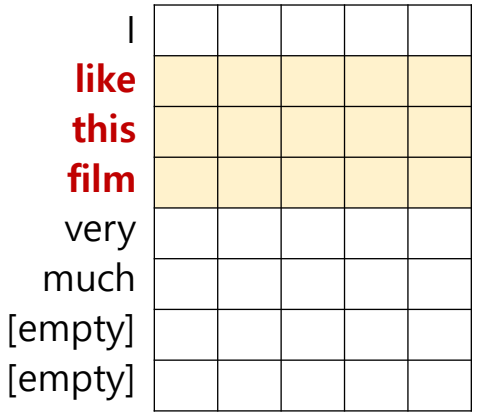
Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Word – level convolutional neural network

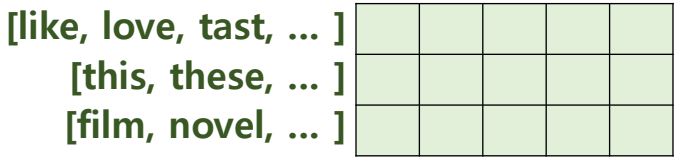
- 하나의 필터는 semantically similar 한 단어로 구성된 n-grams 들을 한번에 인식할 수 있습니다.
  - 문장 이미지와 필터와의 내적이 비슷합니다.



sentence 1



sentence 2



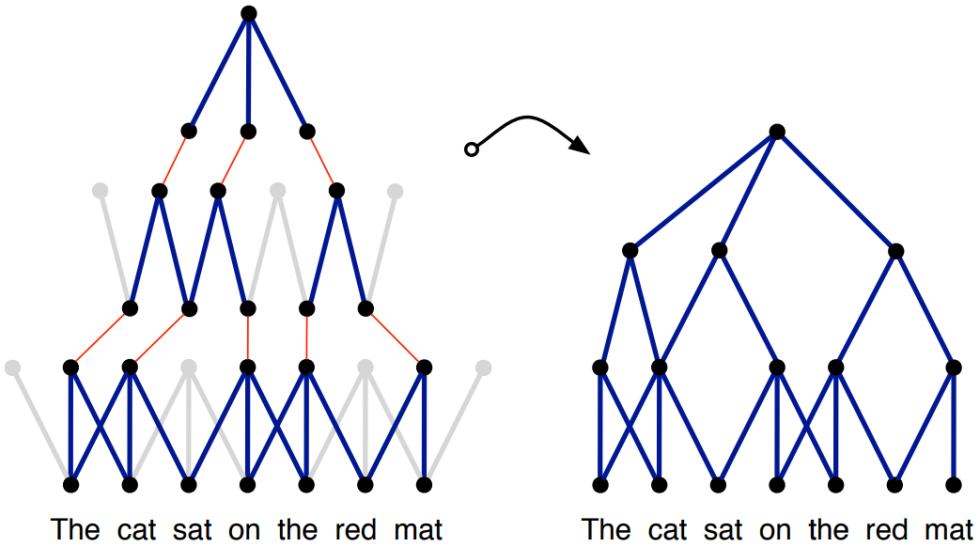
filter 1

Word – level convolutional neural network

- Word – level CNN 은 문장에 특정한 n-grams 이 존재하는지를 확인하는 것입니다.
  - max pooling 은 문장에 각각의 필터들이 표현하는 n-gram, (word vectors) 이 존재하는지를 확인합니다.
  - pooling 에 의하여 word order 는 사라집니다.

Word – level convolutional neural network

- 두 개 이상의 convolution – pooling layers 를 이용하는 것은, pooling 에 의하여 선택된 단어들의 조합이 존재하는지를 확인하는 것과 같습니다.



Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*.

Word – level convolutional neural network

- Word – level CNN 이 이용하는 word embedding vector 는 미리 학습된 벡터를 이용할 수도 있고, 각 classification 마다 학습할 수도 있습니다.
- (Yoon Kim, 2014) 는 네 가지 word embedding vector 를 비교하였습니다.
  - rand : randomly initialized. trained by CNN
  - static : use pre-trained vector, no more training
  - non-static : initialized with pre-trained vector, but fine-tune with CNN
  - multichannel : both static + non-static with 2 channels

Word – level convolutional neural network

- Word – level CNN 이 이용하는 word embedding vector 는 미리 학습된 벡터를 이용할 수도 있고, 각 classification 마다 학습할 수도 있습니다.

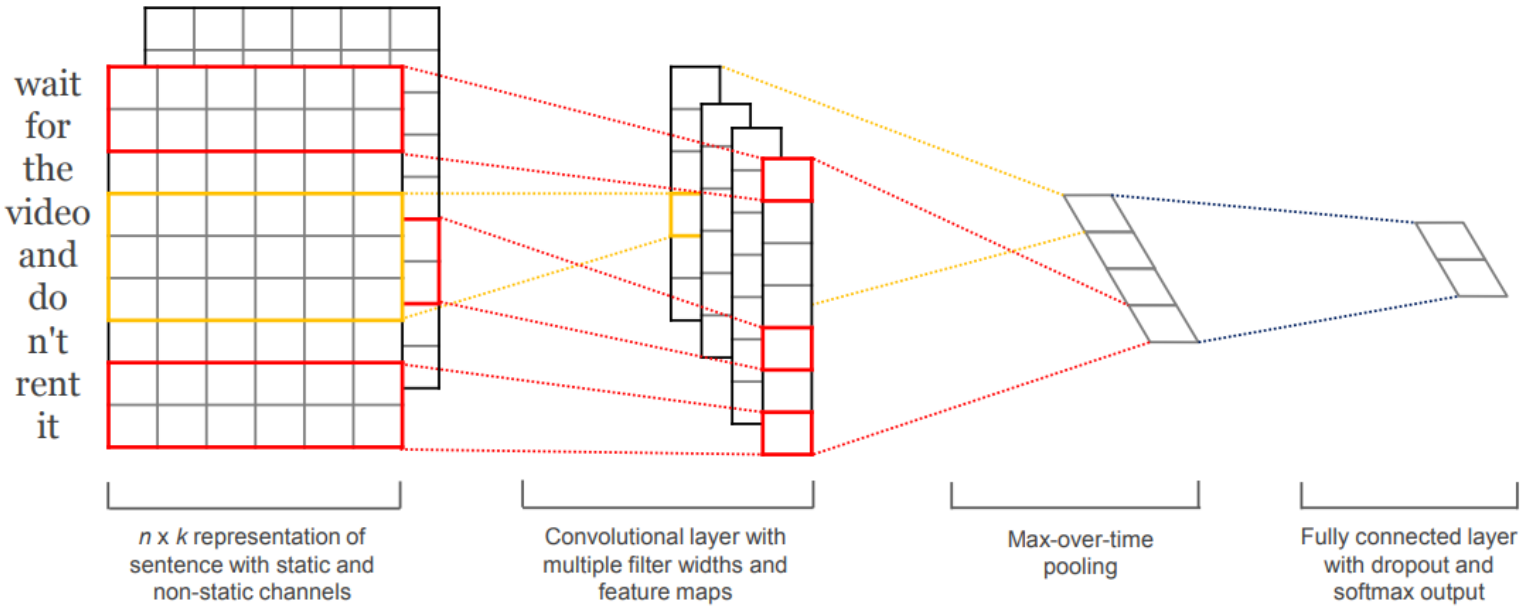


Figure 1: Model architecture with two channels for an example sentence.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Word – level convolutional neural network

- Pre-trained vector 는 도움이 됩니다.
- 간단한 구조의 CNN 만으로도 복잡한 모델만큼의 성능을 보입니다.
- rand 만 아니라면 static / non-static / multichannel 의 성능은 비슷합니다.

Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-rand	76.1	45.0	82.7	89.6	91.2	79.8	83.4
CNN-static	81.0	45.5	86.8	93.0	92.8	84.7	**89.6**
CNN-non-static	**81.5**	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	**88.1**	93.2	92.2	**85.0**	89.4
RAE (Socher et al., 2011)	77.7	43.2	82.4	–	–	–	86.4
MV-RNN (Socher et al., 2012)	79.0	44.4	82.9	–	–	–	–
RNTN (Socher et al., 2013)	–	45.7	85.4	–	–	–	–
DCNN (Kalchbrenner et al., 2014)	–	48.5	86.8	–	93.0	–	–
Paragraph-Vec (Le and Mikolov, 2014)	–	**48.7**	87.8	–	–	–	–
CCAE (Hermann and Blunsom, 2013)	77.8	–	–	–	–	–	87.2
Sent-Parser (Dong et al., 2014)	79.5	–	–	–	–	–	86.3
NBSVM (Wang and Manning, 2012)	79.4	–	–	93.2	–	81.8	86.3
MNB (Wang and Manning, 2012)	79.0	–	–	**93.6**	–	80.0	86.3
G-Dropout (Wang and Manning, 2013)	79.0	–	–	93.4	–	82.1	86.1
F-Dropout (Wang and Manning, 2013)	79.1	–	–	**93.6**	–	81.9	86.3
Tree-CRF (Nakagawa et al., 2010)	77.3	–	–	–	–	81.4	86.1
CRF-PR (Yang and Cardie, 2014)	–	–	–	–	–	82.7	–
SVM_S (Silva et al., 2011)	–	–	–	–	**95.0**	–	–

Table 2: Results of our CNN models against other methods. **RAE**: Recursive Autoencoders with pre-trained word vectors from Wikipedia (Socher et al., 2011). **MV-RNN**: Matrix-Vector Recursive Neural Network with parse trees (Socher et al., 2012). **RNTN**: Recursive Neural Tensor Network with tensor-based feature function and parse trees (Socher et al., 2013). **DCNN**: Dynamic Convolutional Neural Network with k-max pooling (Kalchbrenner et al., 2014). **Paragraph-Vec**: Logistic regression on top of paragraph vectors (Le and Mikolov, 2014). **CCAE**: Combinatorial Category Autoencoders with combinatorial category grammar operators (Hermann and Blunsom, 2013). **Sent-Parser**: Sentiment analysis-specific parser (Dong et al., 2014). **NBSVM, MNB**: Naive Bayes SVM and Multinomial Naive Bayes with uni-bigrams from Wang and Manning (2012). **G-Dropout, F-Dropout**: Gaussian Dropout and Fast Dropout from Wang and Manning (2013). **Tree-CRF**: Dependency tree with Conditional Random Fields (Nakagawa et al., 2010). **CRF-PR**: Conditional Random Fields with Posterior Regularization (Yang and Cardie, 2014). **SVM_S**: SVM with uni-bi-trigrams, wh word, head word, POS, parser, hypernyms, and 60 hand-coded rules as features from Silva et al. (2011).

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Word – level convolutional neural network

- Non-static channel 로부터 task – specific word representation 을 얻을 수 있습니다.

	Most Similar Words for	
	Static Channel	Non-static Channel
bad	*good* *terrible* *horrible* *lousy*	*terrible* *horrible* *lousy* *stupid*
good	*great* *bad* *terrific* *decent*	*nice* *decent* *solid* *terrific*
n't	*os* *ca* *ireland* *wo*	*not* *never* *nothing* *neither*
!	*2,500* *entire* *jez* *changer*	*2,500* *lush* *beautiful* *terrific*
,	*decasia* *abysmally* *demise* *valiant*	*but* *dragon* *a* *and*

Table 3: Top 4 neighboring words—based on cosine similarity—for vectors in the static channel (left) and fine-tuned vectors in the non-static channel (right) from the multichannel model on the SST-2 dataset after training.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Word – level convolutional neural network

- k-max pooling
  - 1 – max pooling 은 region 에서 가장 큰 하나의 값을 선택합니다. 여러 개의 n-grams 중 하나만을 이용합니다.
  - k-max pooling 은 activated 된 n-grams 중 k 개를 이용합니다.

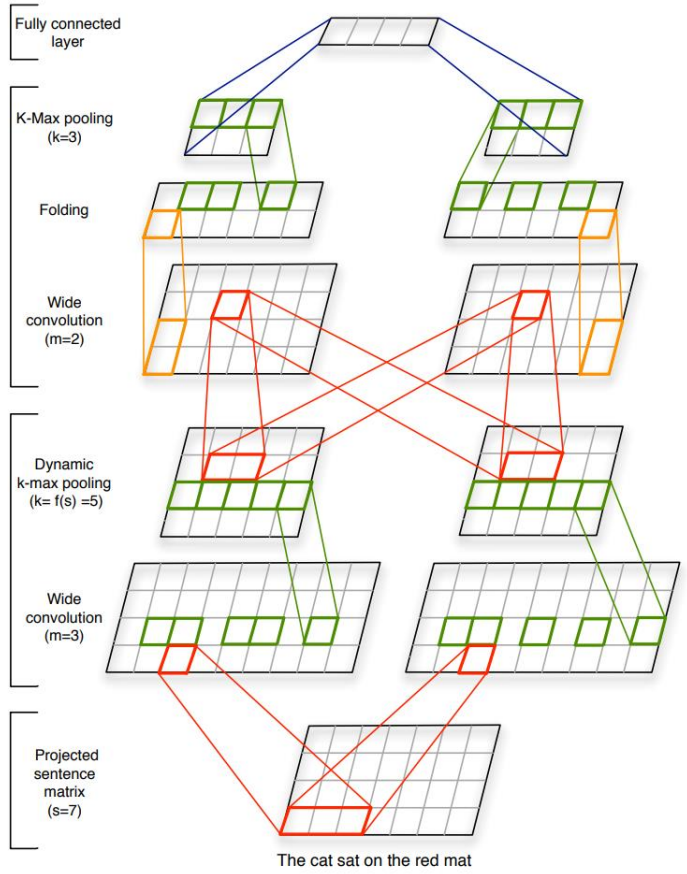


Figure 3: A DCNN for the seven word input sentence. Word embeddings have size $d = 4$. The network has two convolutional layers with two feature maps each. The widths of the filters at the two layers are respectively 3 and 2. The (dynamic) k-max pooling layers have values k of 5 and 3.

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.

Word – level convolutional neural network

- (Kalchbrenner et al., 2014) 에서는 convolution layers 의 깊이에 따라 서로 다른 k 를 이용하는 dynamic k – max pooling 을 제안하였습니다.
  - (Kalchbrenner et al., 2014) 에서는 1 – max pooling 보다 좋은 성능을 보인다고 주장하지만,
  - (Zhang and Wallace, 2015) 에서는 오히려 성능이 좋지 않다고 검증하였습니다.

[1] Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.
[2] Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820.

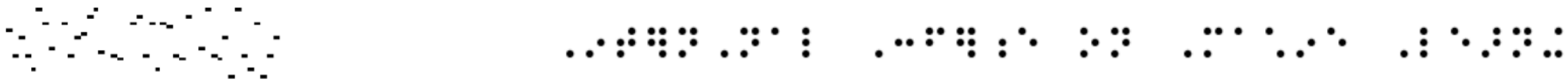
Word – level convolutional neural network

- (Zhang & Wallace, 2015) 는 word – level CNN 구조를 설정하는데 필요한 패러메터들에 대한 전반적인 실험을 하였습니다.
- 위 논문에서 주목해야 하는 점들입니다.
  - Filter region 의 크기는 어느 정도만 크면 큰 차이는 없습니다.
  - 여러 종류의 필터를 굳이 섞어써야 하는 것도 아닙니다.
  - 한 개의 convolution layer 만을 이용할 경우, activation function 은 큰 영향이 없습니다.
  - 그 외의 details 도 많습니다. 직접 모델링을 하시면서 함께 읽어보세요.

Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820.

Character – level convolutional neural network

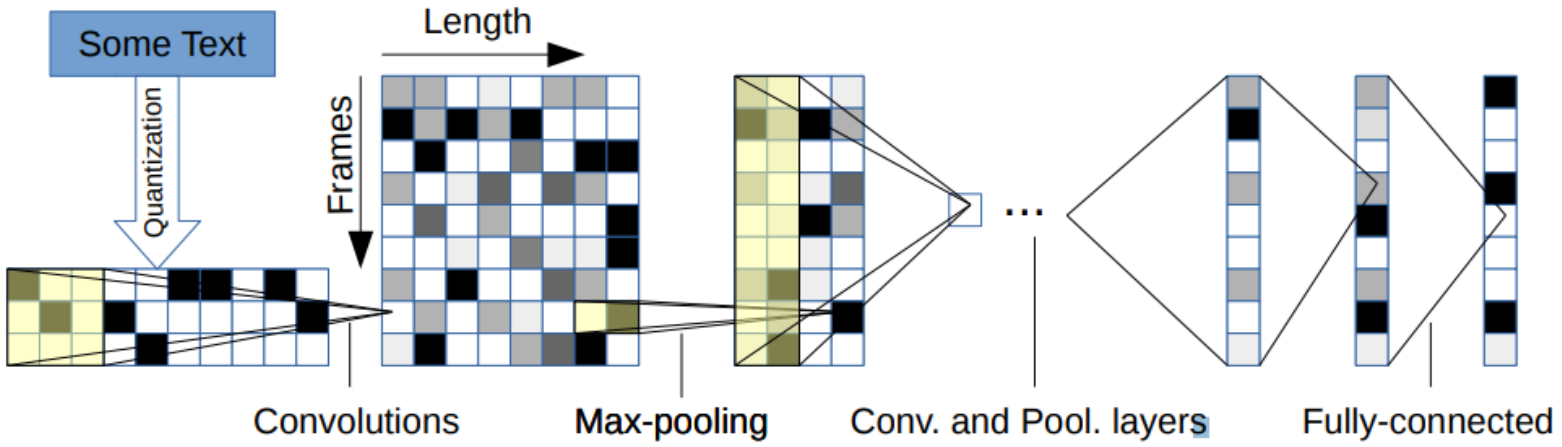
- 문장의 글자를 binary 로 표현하는 것과 점자로 표현하는 것의 모양이 유사합니다.
  - 가로 축이 문장의 방향이며, 새로축은 dimension 입니다.
  - 점자의 한 글자는 binary 가 아닌 integer vector 입니다.



(a) Binary (b) Braille

Character – level convolutional neural network

- 문장의 한 글자를 binary vector 로 표현하여 이를 쌓아서 문장을 binary image 로 표현합니다. 여기에 CNN 을 적용합니다.
- 언어적 지식을 필요로 하는 토크나이징의 과정을 생략할 수 있습니다.



Zhang, X., & LeCun, Y. (2015). Text understanding from scratch. *arXiv preprint arXiv:1502.01710*.

Character – level convolutional neural network

- CNN structure (in_features, out_features)
  - 입력되는 문장의 최대 길이는 1014 입니다.
  - 데이터의 크기에 따라 large, small 중에서 선택합니다.

Table 1. Convolutional layers used in our experiments. The convolutional layers do not use stride and pooling layers are all non-overlapping ones, so we omit the description of their strides.

Layer	Large Frame	Small Frame	Kernel	Pool
1	1024	256	7	3
2	1024	256	7	3
3	1024	256	3	N/A
4	1024	256	3	N/A
5	1024	256	3	N/A
6	1024	256	3	3

Table 2. Fully-connected layers used in our experiments. The number of output units for the last layer is determined by the problem. For example, for a 10-class classification problem it will be 10.

Layer	Output Units Large	Output Units Small
7	2048	1024
8	2048	1024
9	Depends on the problem	

Character – level convolutional neural network

- CNN structure (in_features, out_features)

Layer 1 : (1014)

Layer 2 : (336)

Layer 3 : (110)

Layer 4 : (108)

Layer 5 : (106)

Layer 6 : (104)

$$w` = \frac{w - f + 2 \cdot p}{s} + 1 = \frac{1014 - 7}{1} + 1 = 1008 \rightarrow \frac{1008}{3} = 336$$

$$w` = \frac{336 - 7}{1} + 1 = 330 \rightarrow \frac{330}{3} = 110$$

$$w` = \frac{110 - 3}{1} + 1 = 108$$

$$w` = \frac{108 - 3}{1} + 1 = 106$$

$$w` = \frac{106 - 3}{1} + 1 = 104$$

$$w` = \frac{104 - 3}{1} + 1 = 102 \rightarrow \frac{102}{3} = 34$$

convolution

Layer 7 : (34 x 1024, 2048)

Layer 8 : (2048, 2048)

Layer 9 : (2048, # class)

fully connected

Character – level convolutional neural network

- 첫번째 convolution layer 의 8 개의 필터 예시입니다.
  - $(7, 69)$ 의 이미지가 학습되었습니다만, 해석은 어렵습니다.

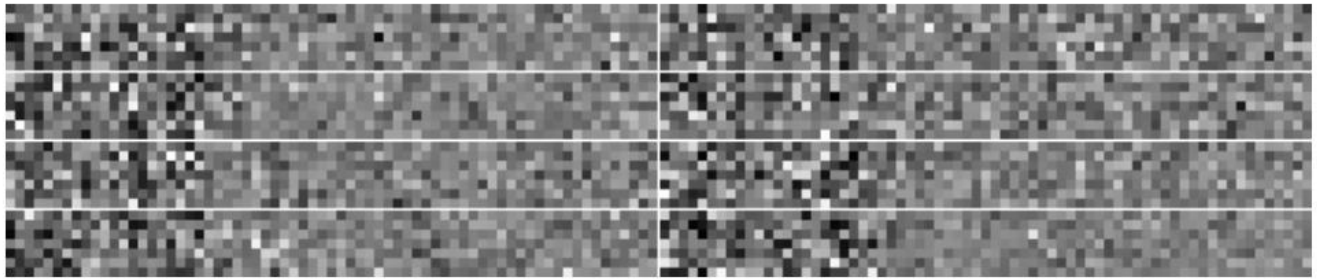


Figure 3. Visualization of first layer weights

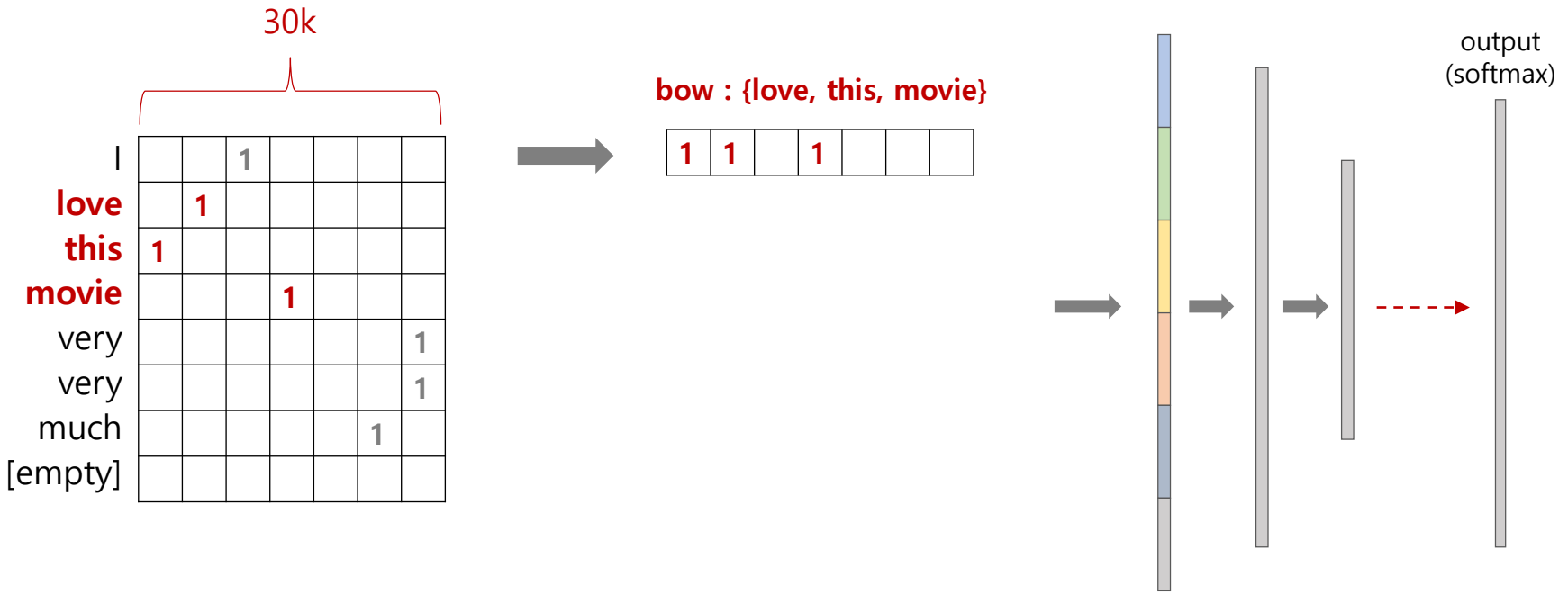
Zhang, X., & LeCun, Y. (2015). Text understanding from scratch. *arXiv preprint arXiv:1502.01710*.

Character – level convolutional neural network

- Character – level CNN 의 첫번째 layer 는 토크나이징을,
  두번째 layer 는 n-gram 의 역할을 하는 것으로 해석할 수 있습니다.
- 어떠한 언어적 지식 없이도 모델을 학습할 수 있다는 점은 장점이지만,
  (Yoon Kim, 2014) 와 비교하여 매우 복잡한 구조의 모델입니다.

bow-CNN

- (Johnson & Zhang, 2014) 는 문장 이미지를 one hot encoding 으로 표현합니다.
  - 30k 이하의 vocabulary 만을 이용합니다.
  - 필터가 적용되는 region 은 row sum 을 한 뒤, 필터를 적용합니다.



Johnson, R., & Zhang, T. (2014). Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058*.

bow-CNN

- (Johnson & Zhang, 2014) 는 문장 이미지를 one hot encoding 으로 표현합니다.
  - 특정한 맥락의 단어들을 하나의 필터에 학습시킬 수 있습니다.
  - Region 의 크기도 30 처럼 매우 크게 설정합니다.

N1	completely useless ., return policy .
N2	it won't even, but doesn't work
N3	product is defective, very disappointing !
N4	is totally unacceptable, is so bad
N5	was very poor, it has failed
P1	works perfectly !, love this product
P2	very pleased !, super easy to, i am pleased
P3	'm so happy, it works perfect, is awesome !
P4	highly recommend it, highly recommended !
P5	am extremely satisfied, is super fast

Examples of predictive text regions in the training set.

were unacceptably bad, is abysmally bad, were universally poor, was hugely disappointed, was enormously disappointed, is monumentally frustrating, are endlessly frustrating
best concept ever, best ideas ever, best hub ever, am wholly satisfied, am entirely satisfied, am incredicbly satisfied, 'm overall impressed, am awfully pleased, am exceptionally pleased, 'm entirely happy, are acoustically good, is blindingly fast,

Examples of text regions that contribute to prediction. They are from the test set, and they did not appear in the training set, either entirely or partially as bi-grams.

Johnson, R., & Zhang, T. (2014). Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058*.

References

- Papers

  - Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.
  - Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.
  - Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820.
  - Zhang, X., & LeCun, Y. (2015). Text understanding from scratch. *arXiv preprint arXiv:1502.01710*.
  - Johnson, R., & Zhang, T. (2014). Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058*.

References

- Stanford CS lecture, CS231
  - Visual Recognition 강좌
  - http://cs231n.stanford.edu/
- Stanford CS lecture, CS224
  - Deep learning for Natural Language Processing
  - http://cs224d.stanford.edu/