Convolutional Neural Networks for Sentence Classification

Yoon Kim

EMNLP 2014

김웅희

• Index

Introduction

• Abstract

Abstract

We report on a series of experiments with convolutional neural networks (CNN) trained on top of pre-trained word vectors for sentence-level classification tasks. We show that a simple CNN with little hyperparameter tuning and static vectors achieves excellent results on multiple benchmarks. Learning task-specific vectors through fine-tuning offers further gains in performance. We additionally propose a simple modification to the architecture to allow for the use of both task-specific and static vectors. The CNN models discussed herein improve upon the state of the art on 4 out of 7 tasks, which include sentiment analysis and question classification.

이미 pretrained된 word vector와 CNN을 활용해
sentence classification task를 수행.

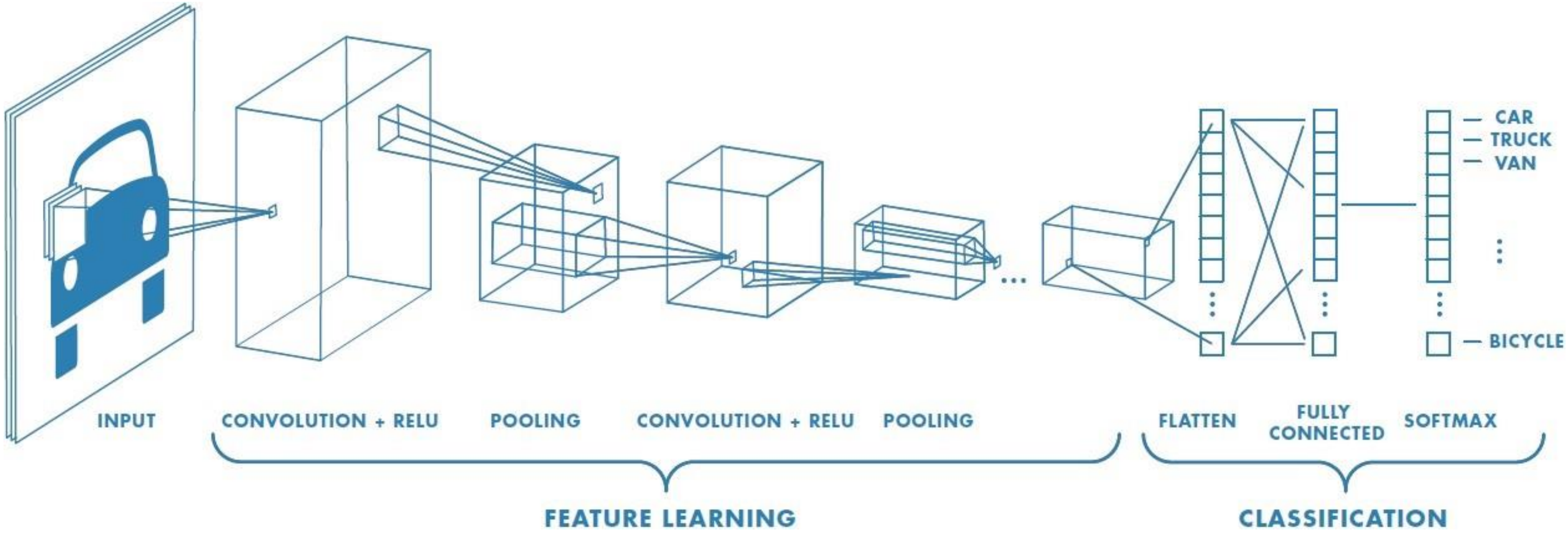
Word Vector를 task-specific하게 finetuning 시켜 성능
을 향상시킬 수 있음.

기존 Static Word Vector와 Task Specific Word Vector
둘 다 사용하는 새롭고 간단한 아키텍처 제안

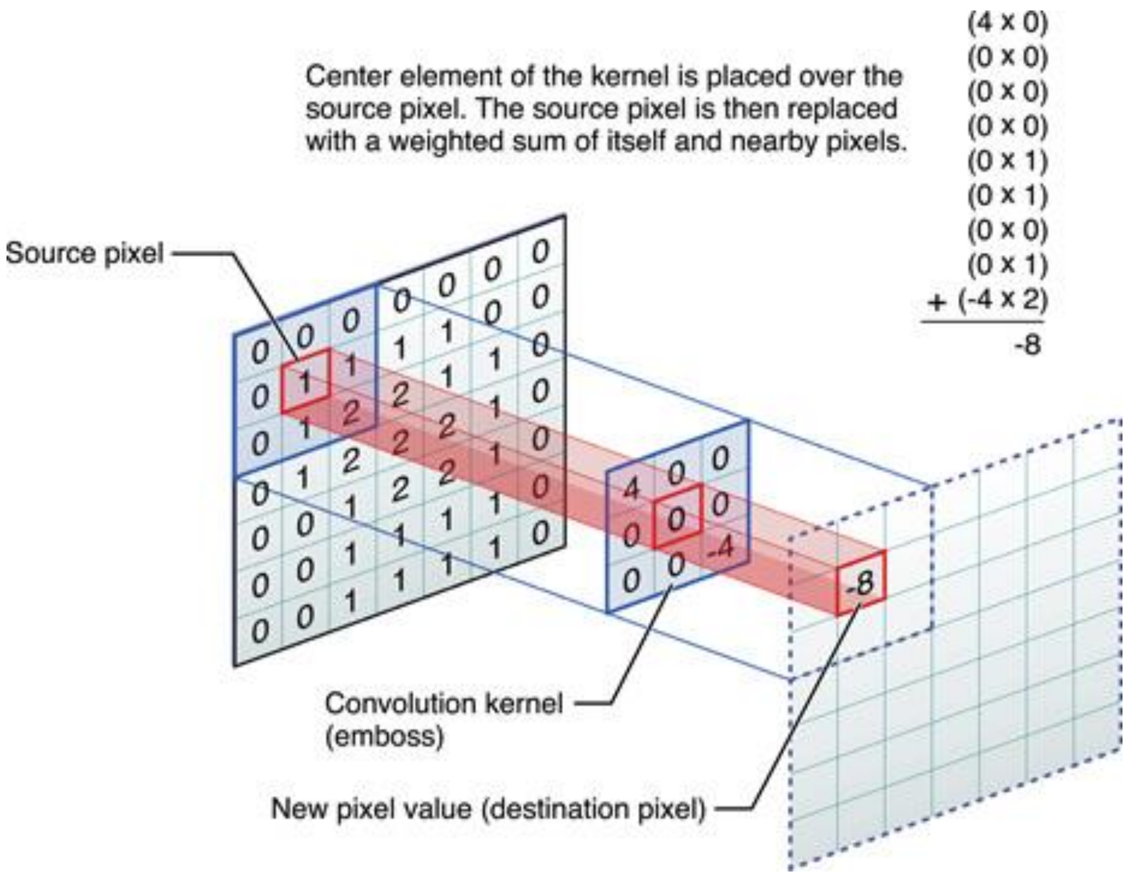
7개의 benchmark dataset에 대해 높은 정확도를 가지며
그 중 4개의 dataset에서는 SOTA를 가짐

Terminology

- **Terminology : Convolution Neural Networks (CNNs)**



• Terminology : Convolution Neural Networks (CNNs)



• Terminology : Convolution Neural Networks (CNNs)

13	20	30	0
8	12	3	0
34	70	33	5
111	80	10	23

Activation Map

20	30
111	33

Max Pooling

13	8
66	18

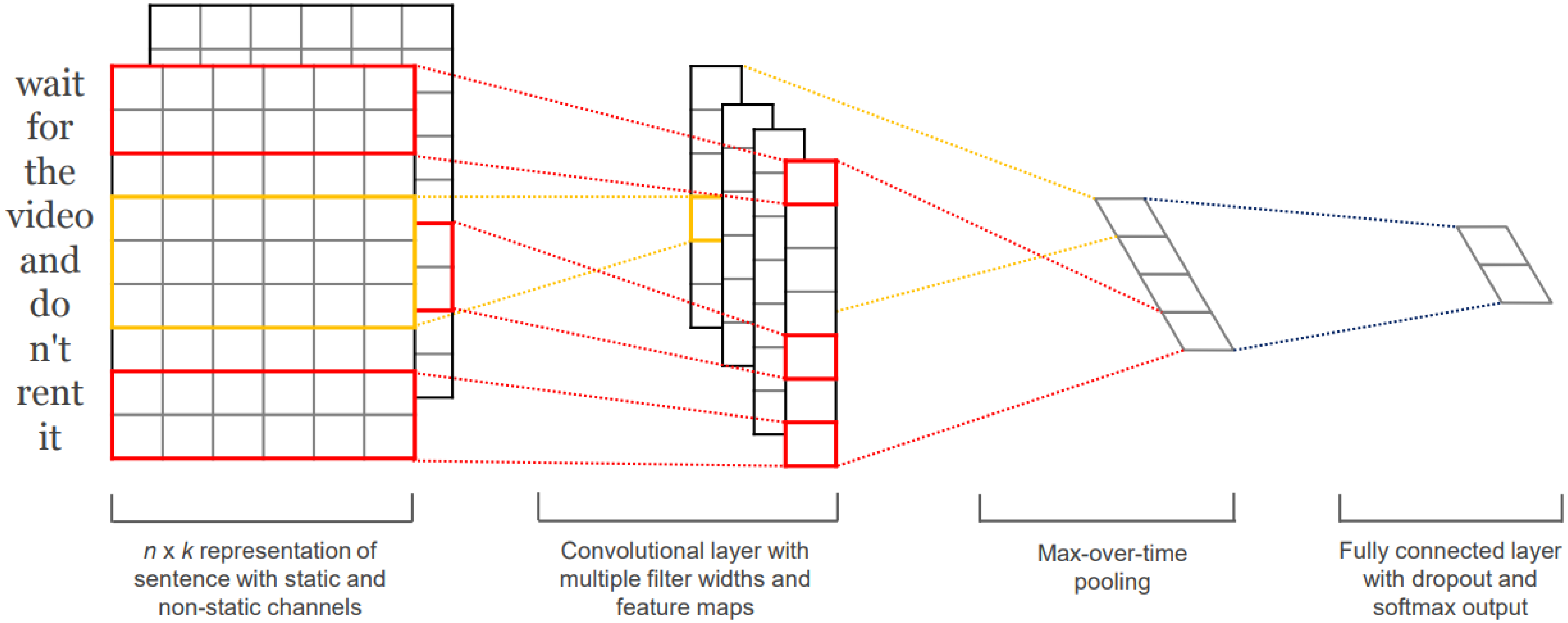
Average Pooling

8	0
34	5

Min Pooling

Model Framework

• Model Framework



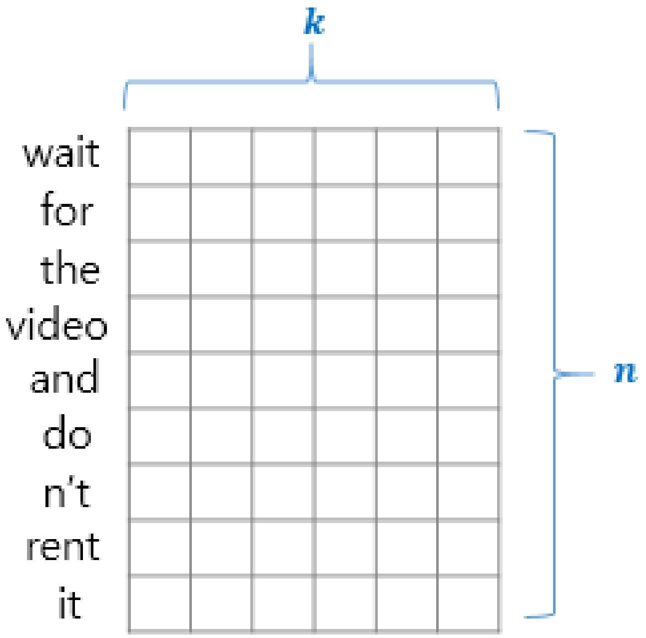
① Input ② Convolution ③ Max-Pooling ④ Fully Connected Layer and Regularization

• Model Framework : Input

wait for the video and don’t rent it

$$\mathbf{x}_{1:n} = \mathbf{x}_1 \oplus \mathbf{x}_2 \oplus \ldots \oplus \mathbf{x}_n$$

※ 단어벡터 : x^i
※ 단어 벡터 길이 : k
※ 문장 길이 : n

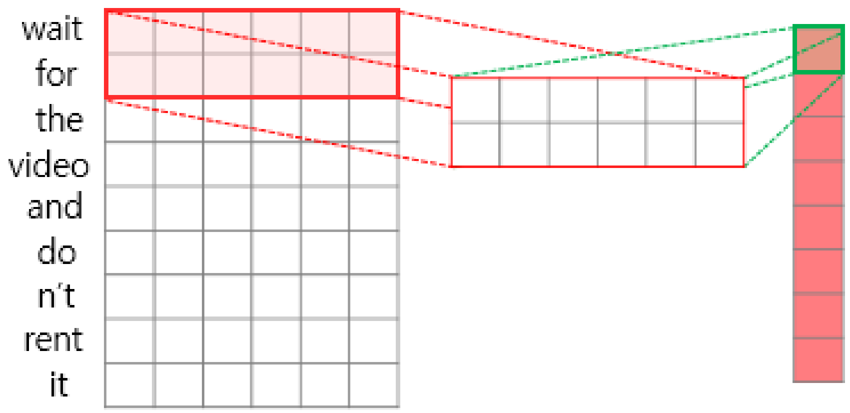


• Model Framework : Convolution

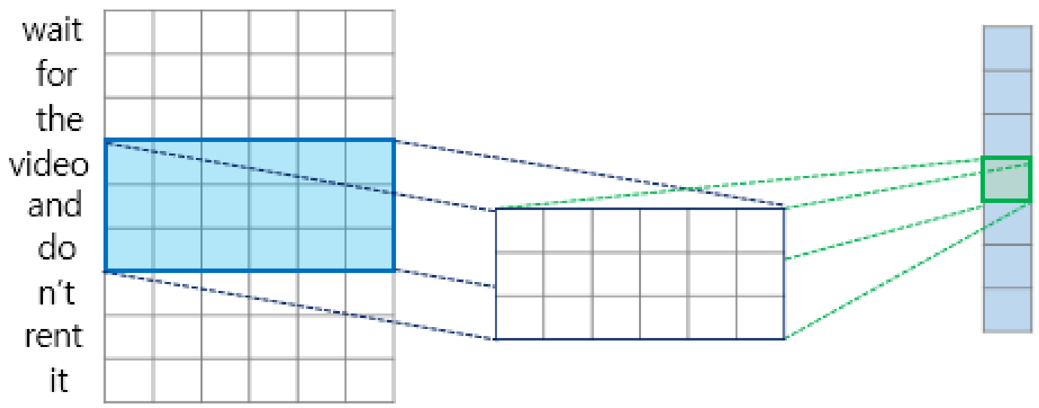
※ Window(Filter) Size : h

$$c_i = f(\mathbf{w} \cdot \mathbf{x}_{i:i+h-1} + b)$$

$$\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}]$$



h = 2

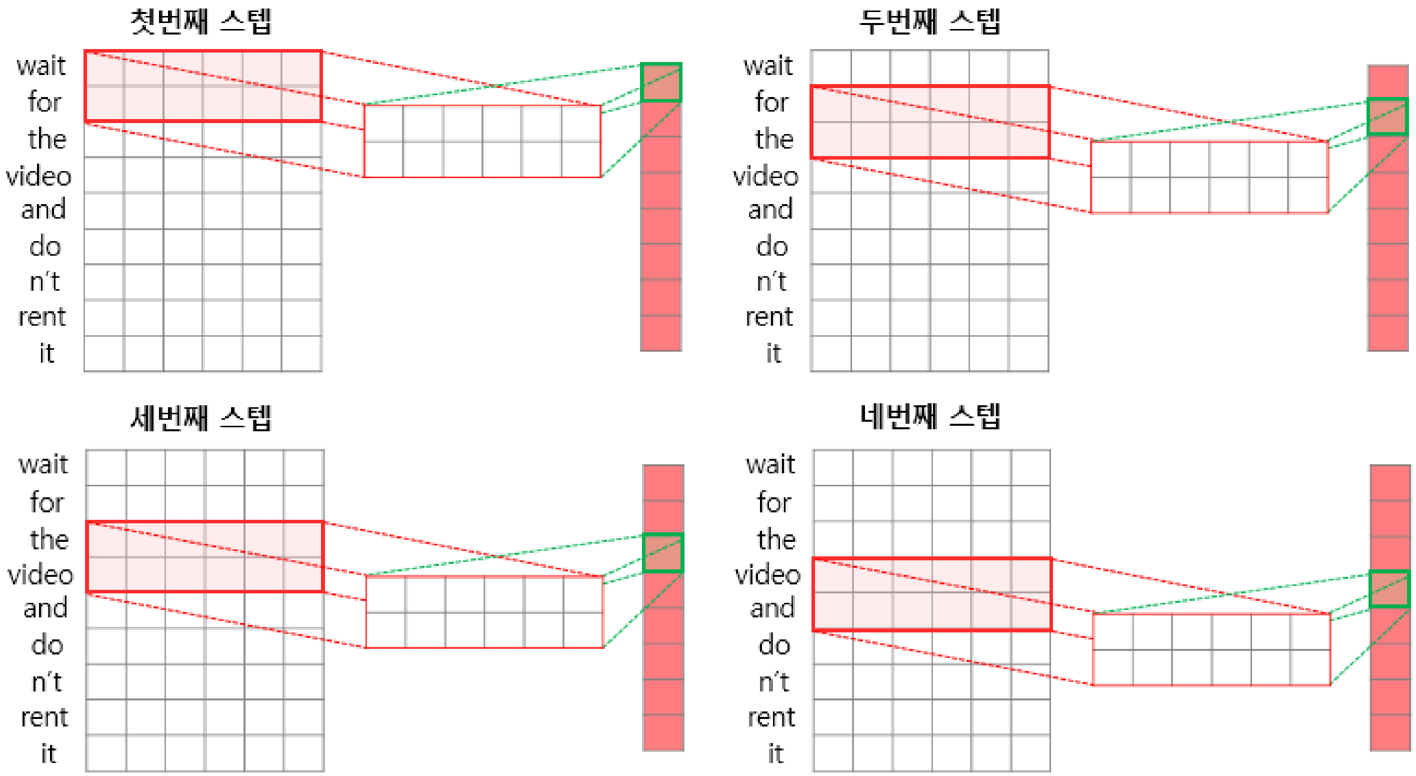


h = 3

• Model Framework : Convolution

※ Window(Filter) Size : h

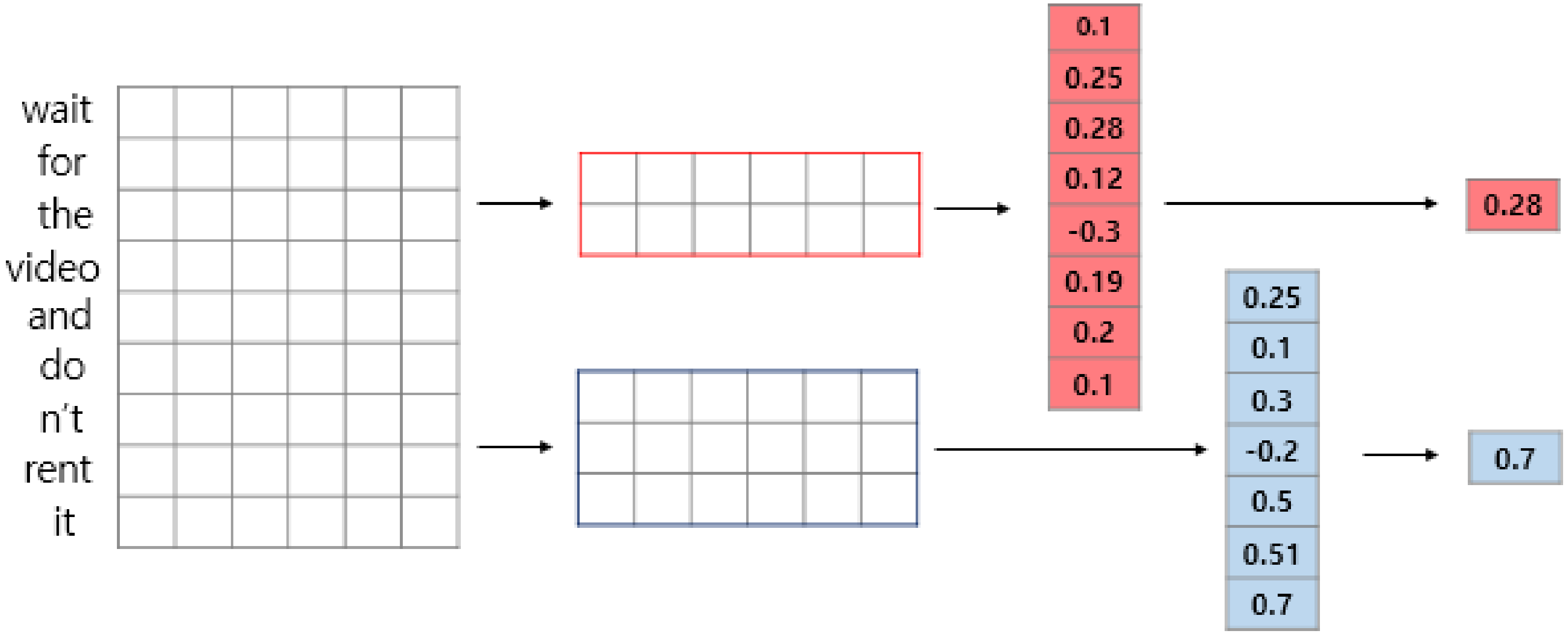
h = 2
Stride = 1



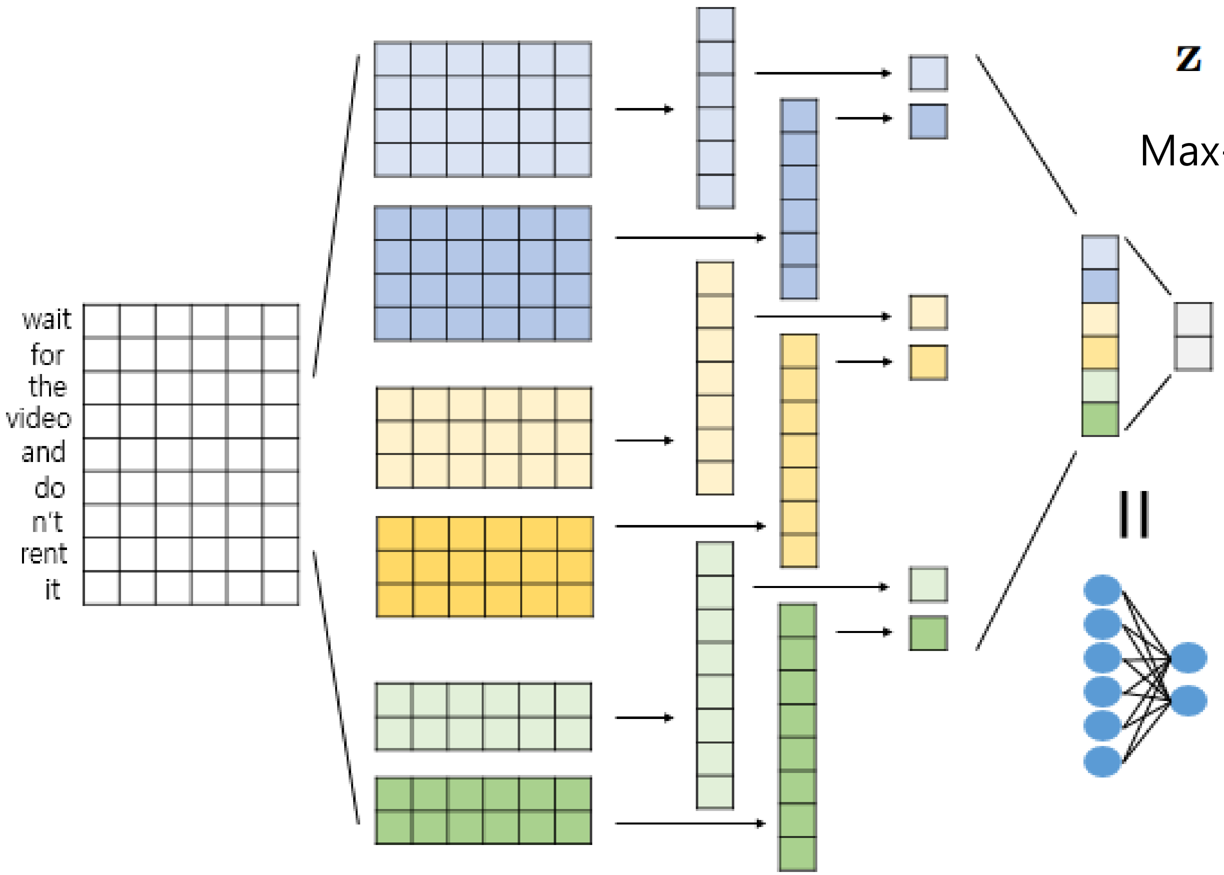
• Model Framework : Convolution

$$\hat{c} = \max\{\mathbf{c}\}$$

$$\mathbf{z} = [\hat{c}_1, \ldots, \hat{c}_m]$$



• Model Framework : Fully Connected Layer and Regularization



$$\mathbf{z} = [\hat{c}_1, \ldots, \hat{c}_m]$$

Max-pooling output : m

Number of filters used : m

$$y = w \cdot z + b$$

L2 norms of the weight vectors by rescaling w -> $||w|| = s$

$$y = w \cdot (z \circ r) + b$$

Masking vector of Bernoulli Random Variable

Experiments

• Experiments : Datasets

- MR : Movie reviews with one sentence per review (Label : positive / negative)
- SST-1 : Stanford Sentiment Treebank (Label : very positive, positive, neutral, negative, very negative)
- SST-2 : Stanford Sentiment Treebank (Label : positive, negative)
- Subj : Subjectivity dataset (Label : subjective / objective)
- TREC : TREC question dataset (Label : abbreviation, entity, description, human, location, numeric)
- CR : Customer review of various products (Label : positive, negative)
- MPQA : Opinion polarity detection subtask of the MPQA dataset (Label : positive, negative, neutral, both)

• Experiments : Hyperparameters and Training

- Activation function : ReLU

- Filter window (h) :
  - 3 with 100 feature maps
  - 4 with 100 feature maps
  - 5 with 100 feature maps

- Dropout rate (p) : 0.5

- L2 Constraint (s) : 3

- Mini batch size : 50

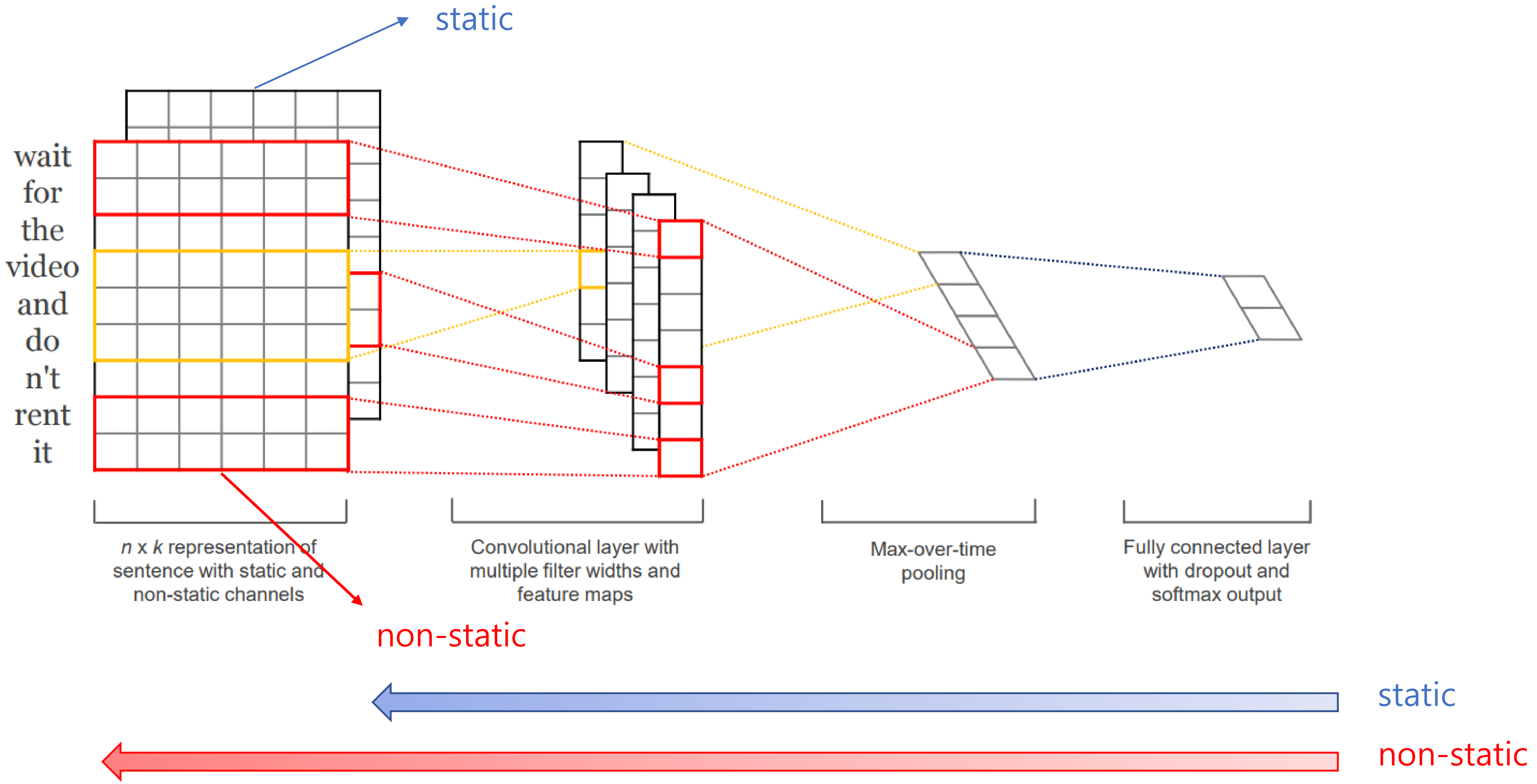
• Experiments : Hyperparameters and Training

모델	Input vector 초기화	학습 중 변경되는지 여부
CNN-rand	랜덤하게	O
CNN-static	word2vec	X
CNN-non-static	word2vec	O
CNN-multichannel	두 개의 채널 (CNN-static + CNN-non-static)	하나는 X, 하나는 O

• Experiments : Hyperparameters and Training

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	$\boxed{\mathbf{89.6}}$
CNN-non-static	$\boxed{\mathbf{81.5}}$	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	$\boxed{\mathbf{88.1}}$	93.2	92.2	$\boxed{\mathbf{85.0}}$	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**	–	–

• Experiments : Static vs Non-Static Representations



• Experiments : Static vs Non-Static Representations

	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*

Conclusion

• Conclusion

- 다양한 필터 사이즈와 여러 개의 Feature map을 사용할수록 모델의 성능이 좋아진다.
- 다른 단어 벡터들의 variance와 유사하게 word2vec에 없는 단어 벡터들을 초기화하면 성능을 향상시킬 수 있다.
- Word2Vec을 사용했을 때, 다른 pretrained word vector를 사용한 것보다 성능이 좋은 것을 통해 universal한 feature extractor임을 확인할 수 있다.