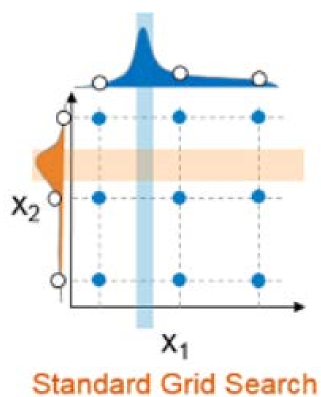
Convolutional Neural Networks for Sentence Classification

고려대 산업경영공학과
박사과정 김동화

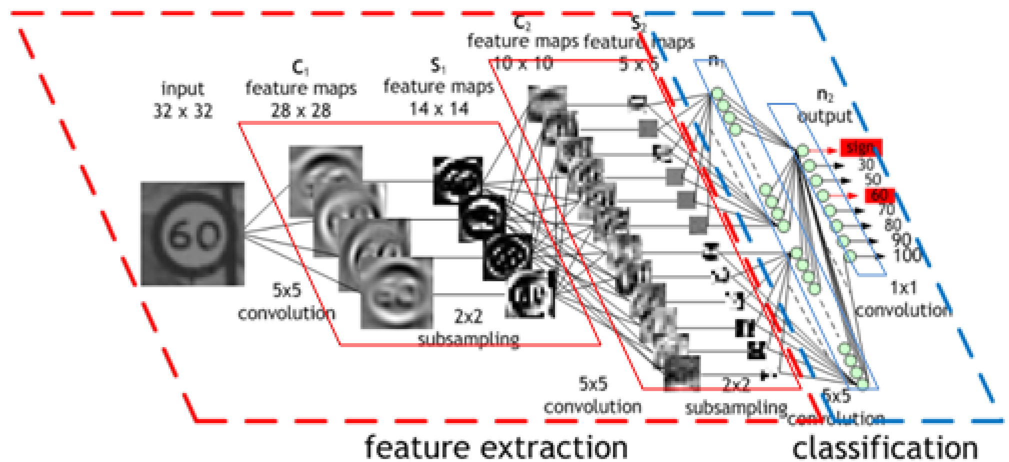
Introduction

- CNN은 Hyperparameter tuning이 거의 없음
- Static 벡터에 대해 뛰어난 성능을 가짐 (Pre-training)
- Fine-tuning을 통한 Task-specific 벡터는 더 좋은 성능을 보임
- 본 논문은 Model(Pre-training + Task-specific vectors) 제안

Hyperparameter tuning

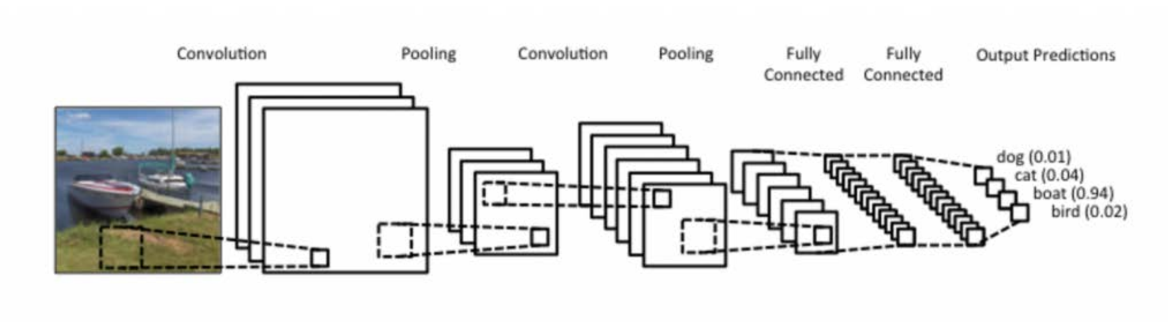


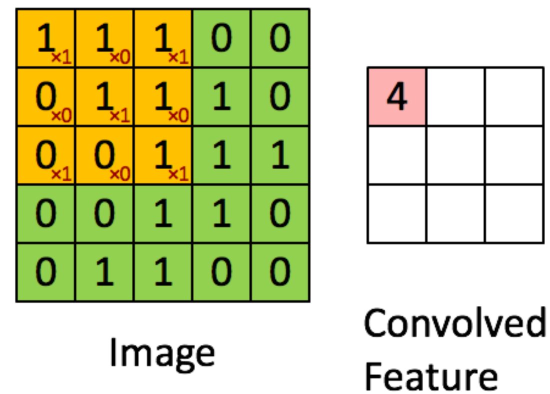
Pre-training & fine-tuning



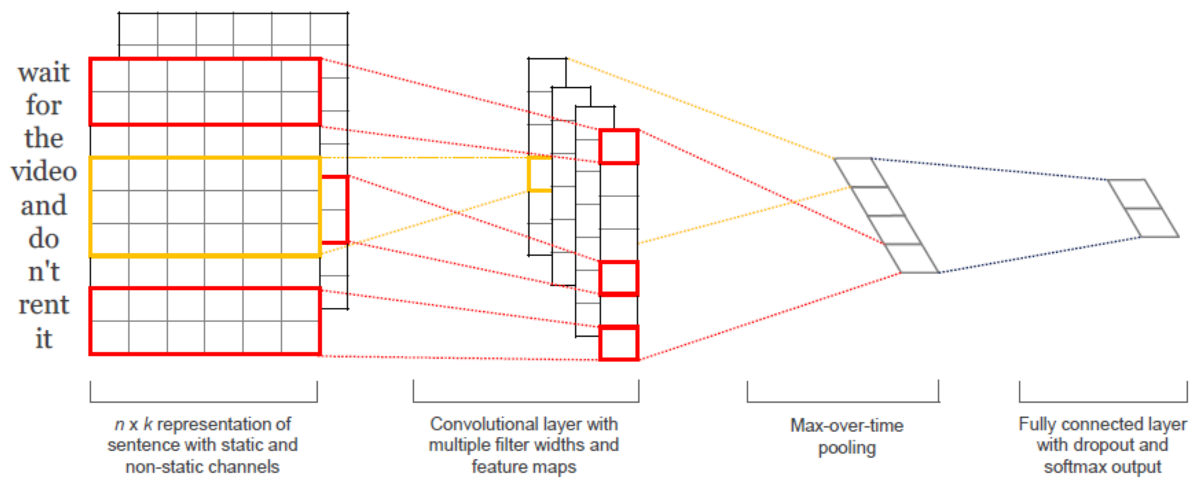
Images vs Texts

CNN for images





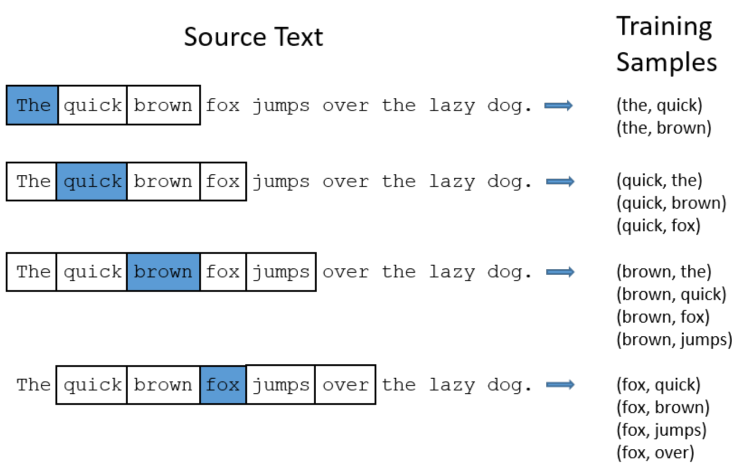
CNN for texts



Keywords

- Word2Vec: 유사한 것은 낮은 차원에서도 가까운 거리에 존재함
- Convolution: 구문분석, 데이터 검색에서 많이 사용됨
- Pre-training: 이미 학습된 파라미터들을 가져와서 없는 부분들만 Training
  - 100 billion words of Google News

Word2Vec (CBOW)



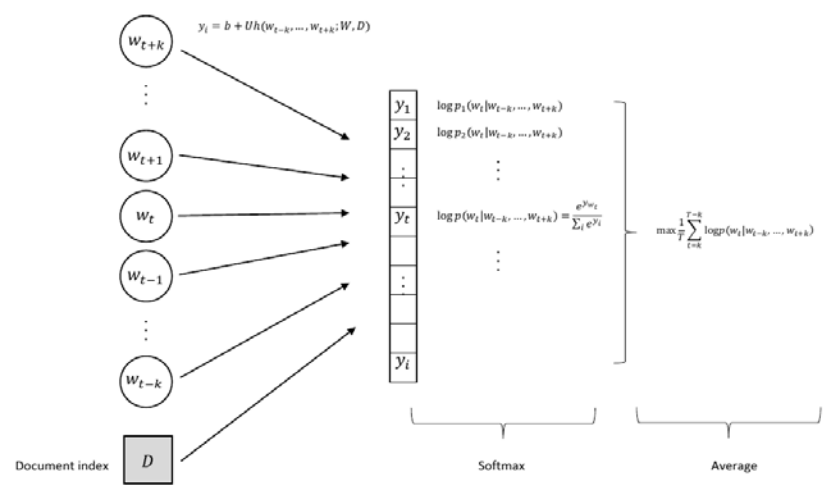
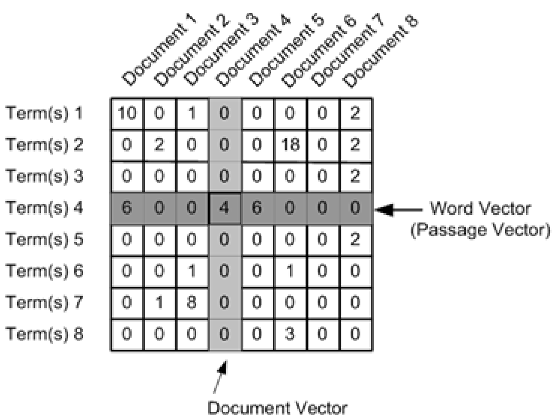


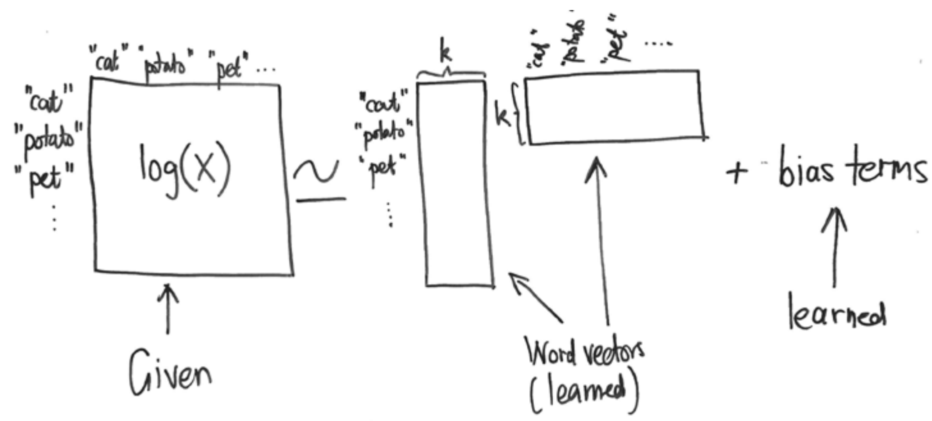
Figure 3: Distributed Memory Model of Paragraph Vectors [14]

CNN for Sentence Classification

- **Step1): Convert DTM or STM to Word2Vec**
  (*DTM: Document Term Matrix, *STM: Sentence Term Matrix)



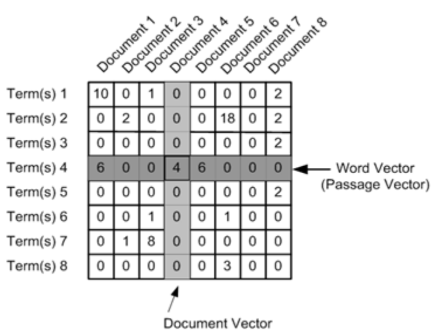
	Document 1	Document 2	Document 3	Document 4	Document 5	Document 6	Document 7	Document 8
Term(s) 1	10	0	1	0	0	0	0	2
Term(s) 2	0	2	0	0	0	18	0	2
Term(s) 3	0	0	0	0	0	0	0	2
Term(s) 4	6	0	0	4	6	0	0	0
Term(s) 5	0	0	0	0	0	0	0	2
Term(s) 6	0	0	1	0	0	1	0	0
Term(s) 7	0	1	8	0	0	0	0	0
Term(s) 8	0	0	0	0	0	3	0	0



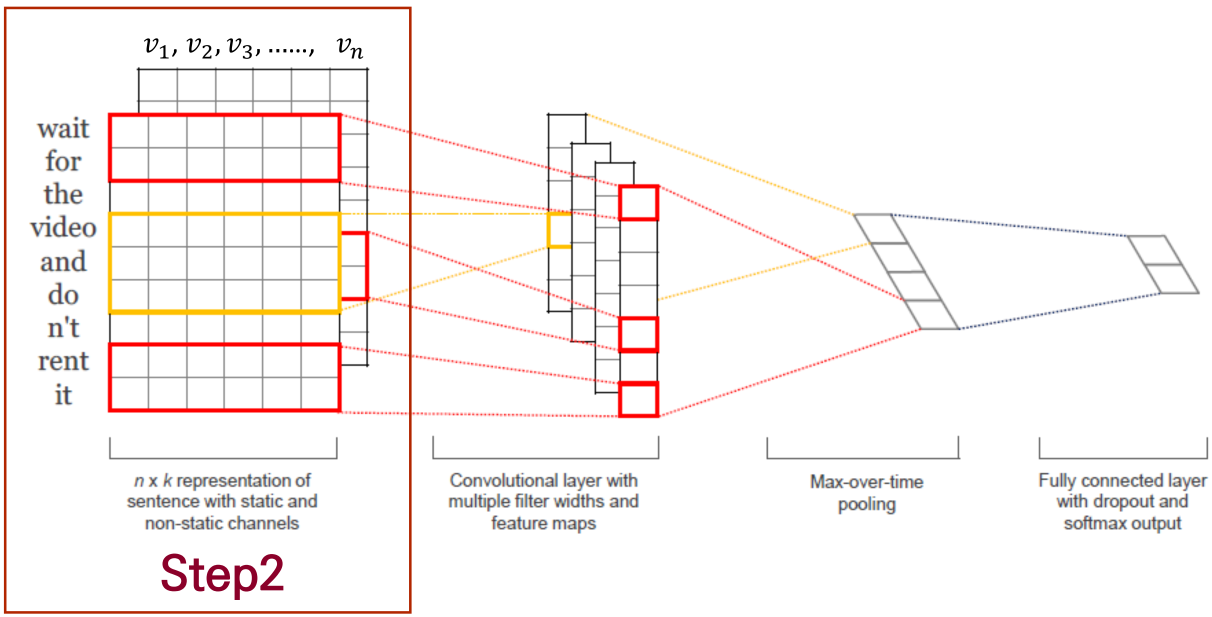
CNN for Sentence Classification

- **Step2): Decision of Window size (h)**
  - Window size = Filter size
- **Step3): Activation for window of words**

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



	Document 1	Document 2	Document 3	Document 4	Document 5	Document 6	Document 7	Document 8
Term(s) 1	10	0	1	0	0	0	0	2
Term(s) 2	0	2	0	0	0	18	0	2
Term(s) 3	0	0	0	0	0	0	0	2
Term(s) 4	6	0	0	4	6	0	0	0
Term(s) 5	0	0	0	0	0	0	0	2
Term(s) 6	0	0	1	0	0	1	0	0
Term(s) 7	0	1	8	0	0	0	0	0
Term(s) 8	0	0	0	0	0	3	0	0



CNN for Sentence Classification

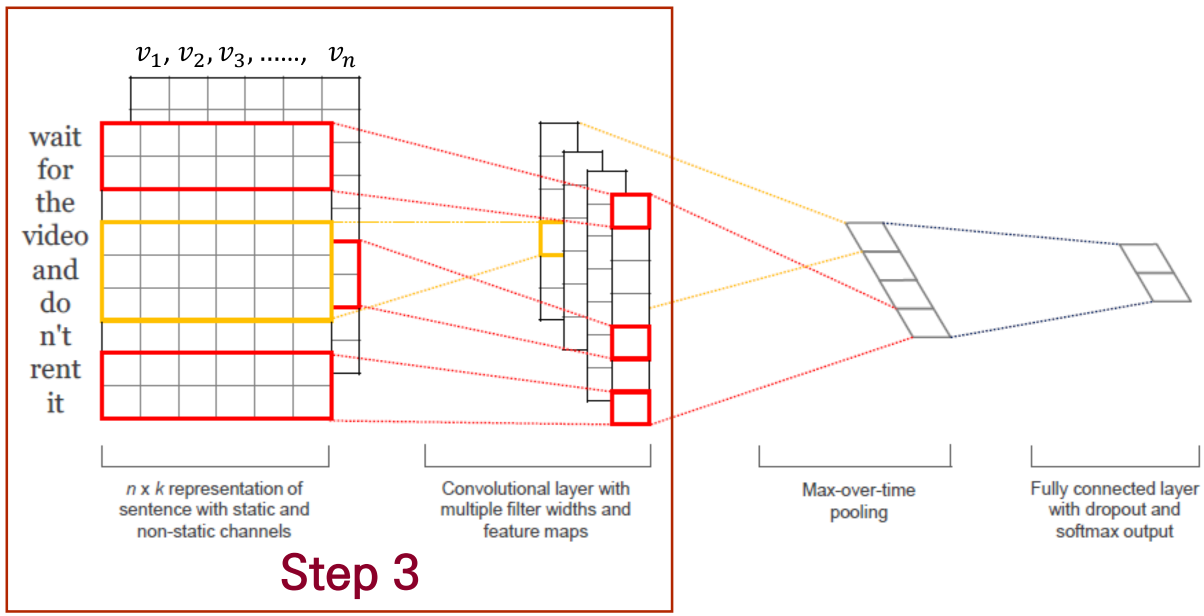
- **Step2): Decision of Window size (h)**
  - Window size = Filter size
- **Step3): Activation for window of words**

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

	Document 1	Document 2	Document 3	Document 4	Document 5	Document 6	Document 7	Document 8
Term(s) 1	10	0	1	0	0	0	0	2
Term(s) 2	0	2	0	0	0	18	0	2
Term(s) 3	0	0	0	0	0	0	0	2
Term(s) 4	6	0	0	4	6	0	0	0
Term(s) 5	0	0	0	0	0	0	0	2
Term(s) 6	0	0	1	0	0	1	0	0
Term(s) 7	0	1	8	0	0	0	0	0
Term(s) 8	0	0	0	0	0	3	0	0

Word Vector (Passage Vector)

Document Vector



CNN for Sentence Classification

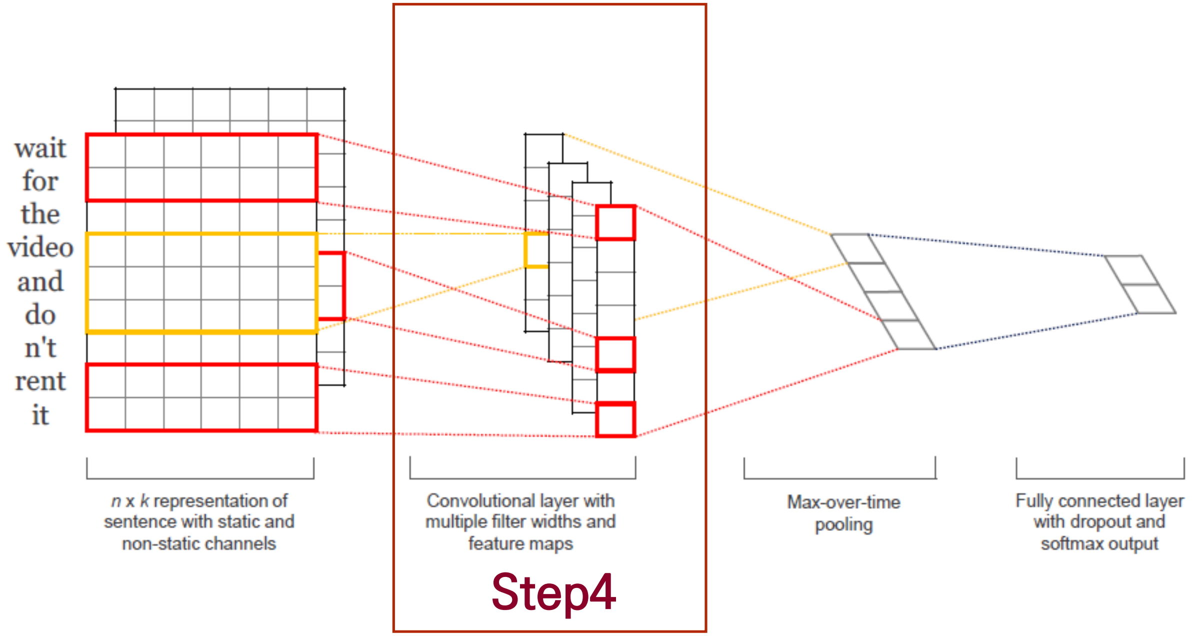
- **Step4): Concatenate**

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

- **Step5): Max Pooling**

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

- **Step6): A fully connected softmax layer**
  - Probability distribution over labels



CNN for Sentence Classification

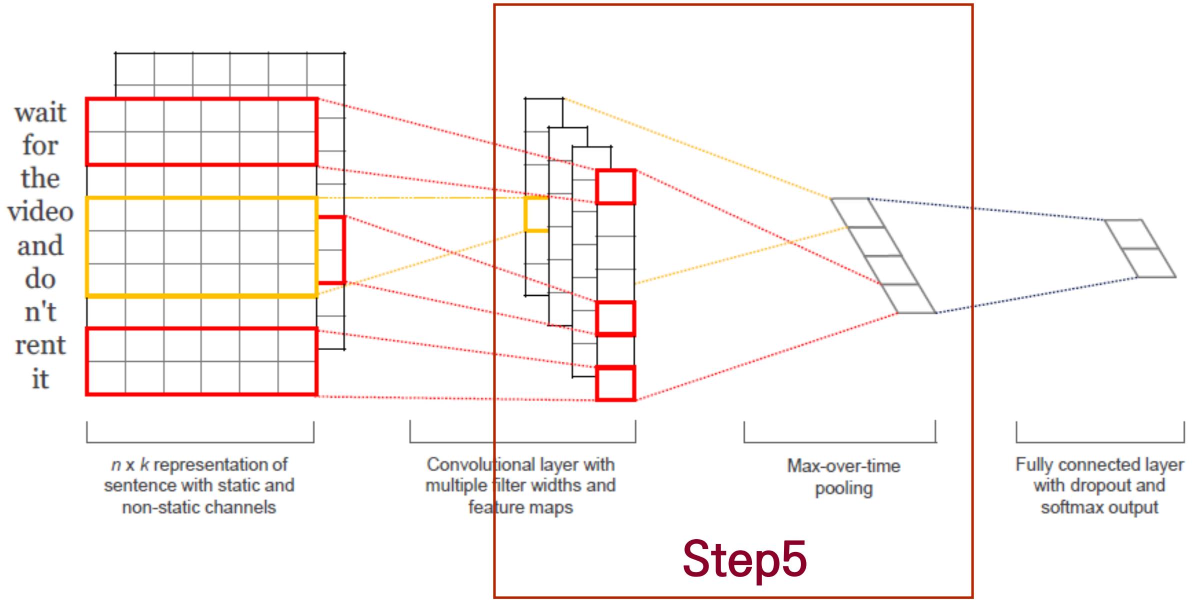
- **Step4): Concatenate**

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

- **Step5): Max Pooling**

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

- **Step6): A fully connected softmax layer**
  - Probability distribution over labels



CNN for Sentence Classification

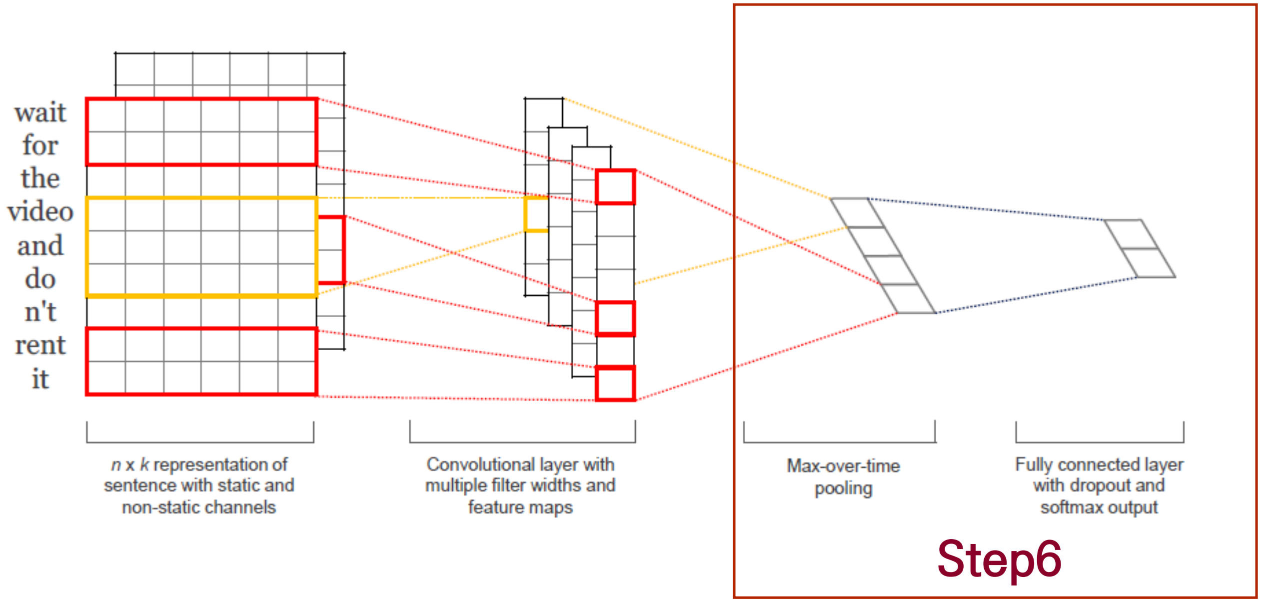
- **Step4): Concatenate**

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

- **Step5): Max Pooling**

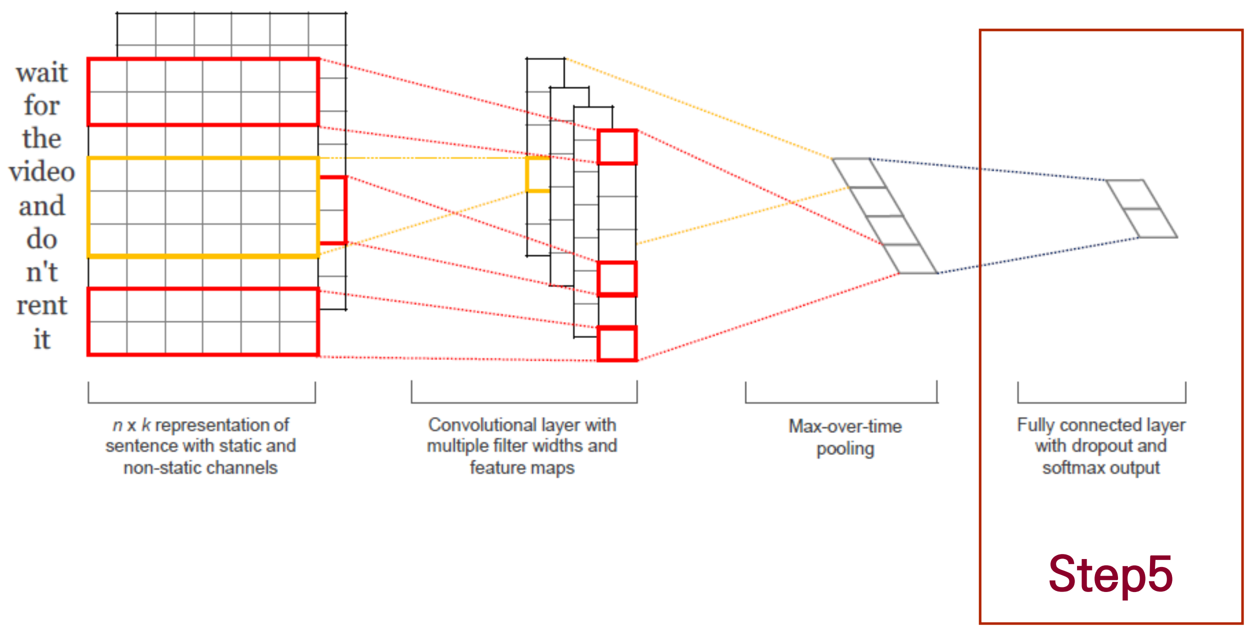
$$\hat{c} = \max\{\mathbf{c}\}$$

- **Step6): A fully connected softmax layer**
  - Probability distribution over labels



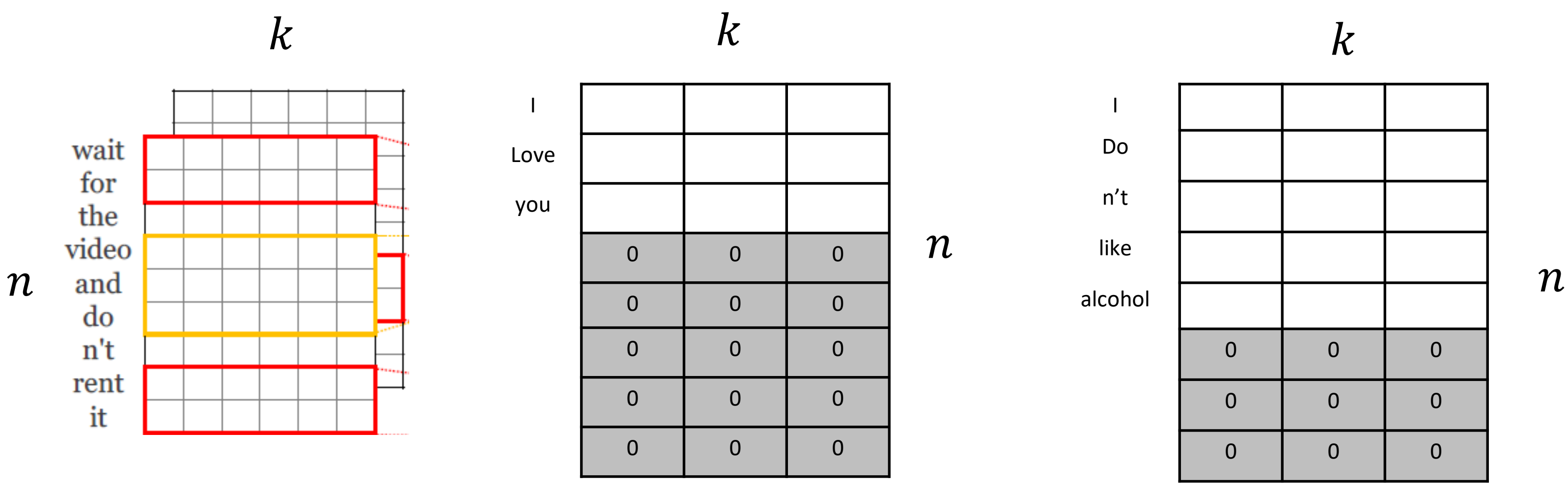
Characteristics

- Two channel: 1) Static Word2vec, 2) fine-tuned Word2Vec via backpropa
- Filter size = 단어문맥 순서의 길이
- Max Pooling = 가장 중요한 단어문맥 순서 추출 & 일정한 feature maps
- a filter = a feature



Characteristics

- Zero padding: 각 문장의 길이는 서로 다르기 때문에 padding 적용



Improvement methods

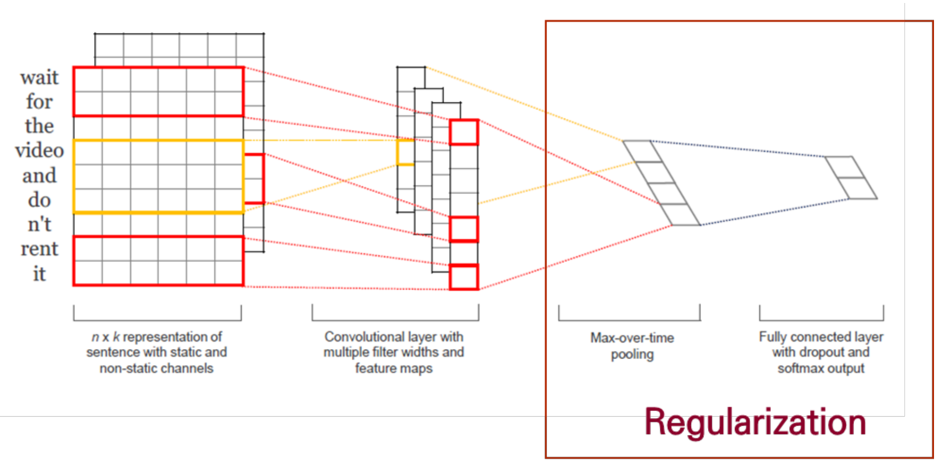
- **Regularization**
  - 마지막 뒤에서 두번째 레이어의 features

    $\mathbf{z} = [\hat{c}_1, \ldots, \hat{c}_m]$ (= the number of filters)
  - Fully connected layer

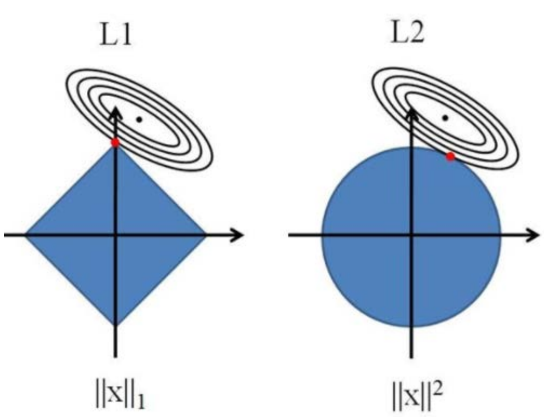
    $y = \mathbf{w} \cdot \mathbf{z} + b$
  - Regularized Fully connected layer

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

    Hadamard product
  - Masking (0 or 1) depended on a probability p



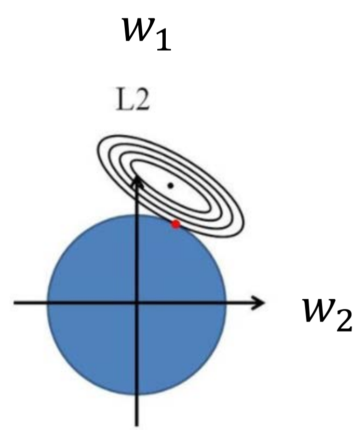
Ln norm (n=1, n=2)



Improvement methods

Regularization

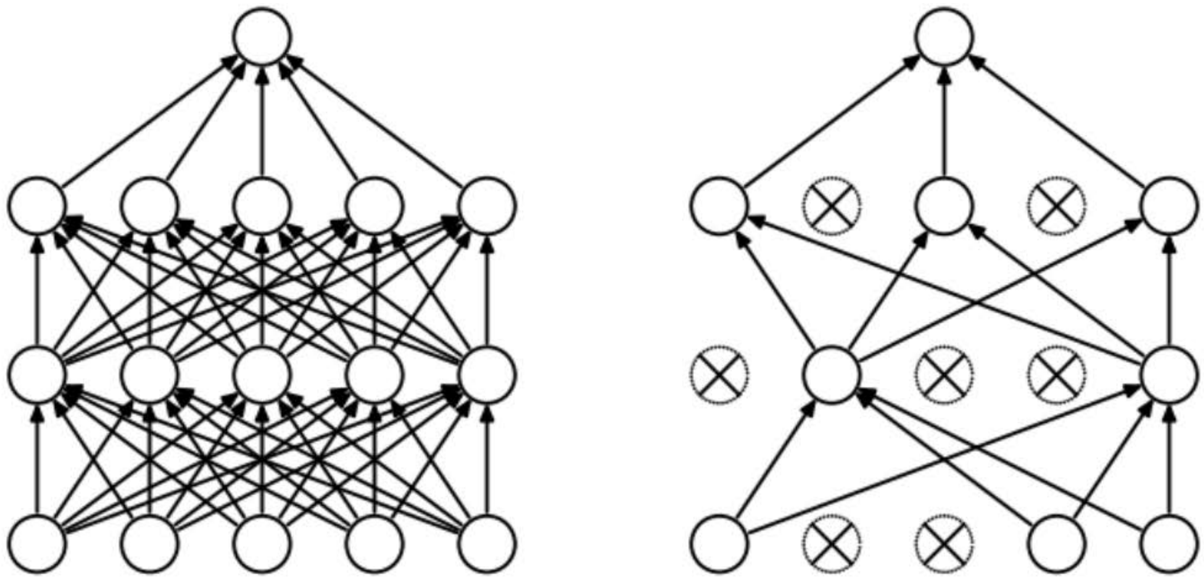
- 켜져있는 히든 유닛만 backprop 적용
- 테스트할때 확률 p 사용하여 모든 히든유닛에 적용 ($\hat{w} = pw$)
- $||w||_2 > s \rightarrow ||w||_2 = s$



$$r = \sqrt{w_1 + w_2}$$

Improvement methods

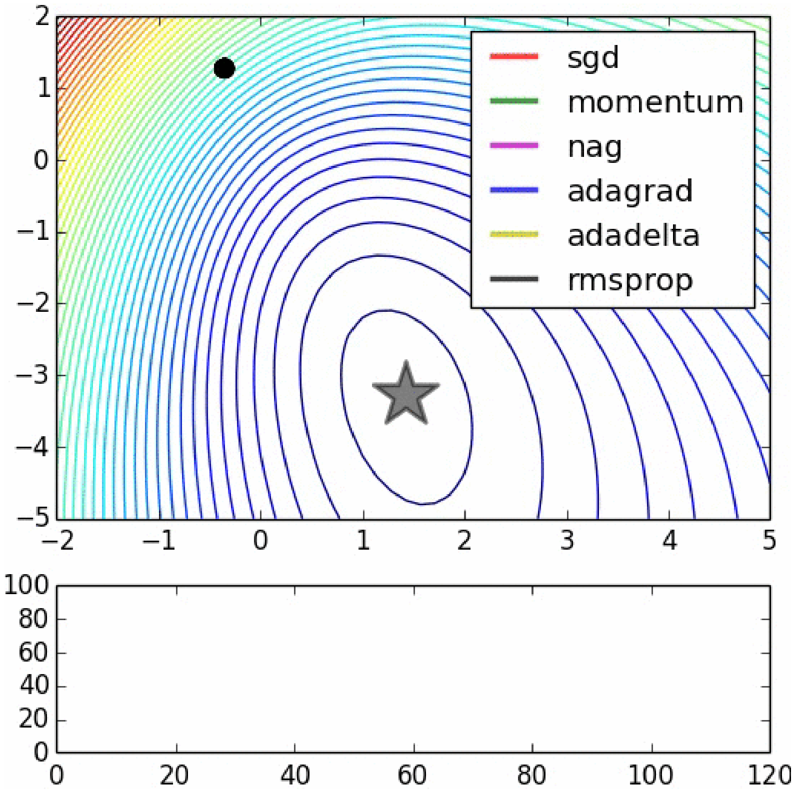
- **Dropout**
  - 2%~4%정도



(a) Standard Neural Net (b) After applying dropout.

Optimization

- **Adadelta**
  - Adagrad의 epoch가 많은 버전



Experiments

- MR: 영화 리뷰 Positive/Negative
- SST-1: fine-grained labels (very positive, positive, neutral, negative, very negative)
- SST-2: positive/negative from SST-1
- Subj: subjective or objective
- TREC: 6 question types
- CR: products (positive/negative)

- CNN-rand: randomly initialized
- CNN-static: pre-trained vectors from word2vec
- CNN-non-static: Same as above but the pretrained vectors are fine-tuned for each task.
- CNN-multichannel: (CNN-static + CNN-non-static)

Result

- Pre-training을 활용한 CNN의 성능은 복잡한 알고리즘에 비해 탁월하다
  - 본 연구자는 다양한 필터 사이즈와 feature maps 때문에 성능이 좋은 것이라고 생각
- CNN-multichannel은 적은 데이터셋에 대한 과적합을 방지한다
- Fine-grained labels에 대한 문제는 제안된 방법론의 성능이 다소 낮음

Data	c	l	N	$\|V\|$	$\|V_{pre}\|$	*Test*
MR	2	20	10662	18765	16448	CV
SST-1	5	18	11855	17836	16262	2210
SST-2	2	19	9613	16185	14838	1821
Subj	2	23	10000	21323	17913	CV
TREC	6	10	5952	9592	9125	500
CR	2	19	3775	5340	5046	CV
MPQA	2	3	10606	6246	6083	CV

Table 1: Summary statistics for the datasets after tokenization. c: Number of target classes. l: Average sentence length. N: Dataset size. $|V|$: Vocabulary size. $|V_{pre}|$: Number of words present in the set of pre-trained word vectors. *Test*: Test set size (CV means there was no standard train/test split and thus 10-fold CV was used).

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**	–	–

Result

- **구문론적인 결과해석**
  - Static channel: Best case(good -> great)// Worst case(good->bad)
  - Non-static channel: Best case(good-> nice)// Worst case(good-> solid)

	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*