Word-level CNN & Char-level CNN

서울대학교병원 정보화실

고태훈 (taehoonko@snuh.org)

최세원 (swc@snuh.org)

How to design input

- Bag-of-words
- n-gram features
- Distributed representation for words

Bag-of-words

- **문서들을 등장 단어에 대한 가중치로 표현하는 방법**
  - 가중치
    - 등장 유무 (0 or 1)
    - 등장 빈도 수 (term frequency)
    - TF-IDF
- **장점**
  - 문서의 벡터 표현이 직관적임
- **단점**
  - 문서 벡터의 차원이 지나치게 큼
    - 전체 문서에 등장하는 vocabulary size만큼의 차원
    - Curse of dimensionality (차원의 저주)
  - 생성되는 문서-단어 행렬 (document-term matrix) 가 너무 희소함 (too sparse).
    - 희소행렬 (sparse matrix) 은 모델링하기 까다로움

Bag-of-words: Example

- **예제 문장**
  - Korean: ‘머신러닝과 딥러닝을 이용하여 뉴스를 분류하다.’
  - English: ‘Classify news using machine learning and deep learning.’

‘*term-frequency*’ BOW

머신	러닝	과	딥	을	이용	하여	뉴스	분류	하다
1	2	1	1	2	1	1	1	1	1

classify	news	using	machine	learning	and	deep
1	1	1	1	2	1	1

Bag of words with n-gram features

- **n-gram**
  - n개의 연속된 단어 열
  - Unigram (n=1) 보다 그 이상의 단어 열이 정보를 더 잘 표현할 수 있음
    - [machine, deep, learning] ***vs.*** [machine learning, deep learning]
- **BOW + n-gram features: 예제 문장**
  - English: 'Classify news using machine learning and deep learning.'

'*term-frequency*' BOW with unigrams and bigrams

classify	news	using	machine	learning	and	deep
1	1	1	1	2	1	1

classify news	news using	using machine	machine learning	learning and	and deep	deep learning
1	1	1	1	1	1	1

Bag of words with n-gram features

- **장점**
  - 풍부한 정보를 생성할 수 있음
- **단점**
  - Bag-of-words 표현에서 n-gram features를 같이 포함하는 경우, 벡터의 차원이 너무 커짐
    - Curse of dimensionality (차원의 저주)

Distributed representation for words

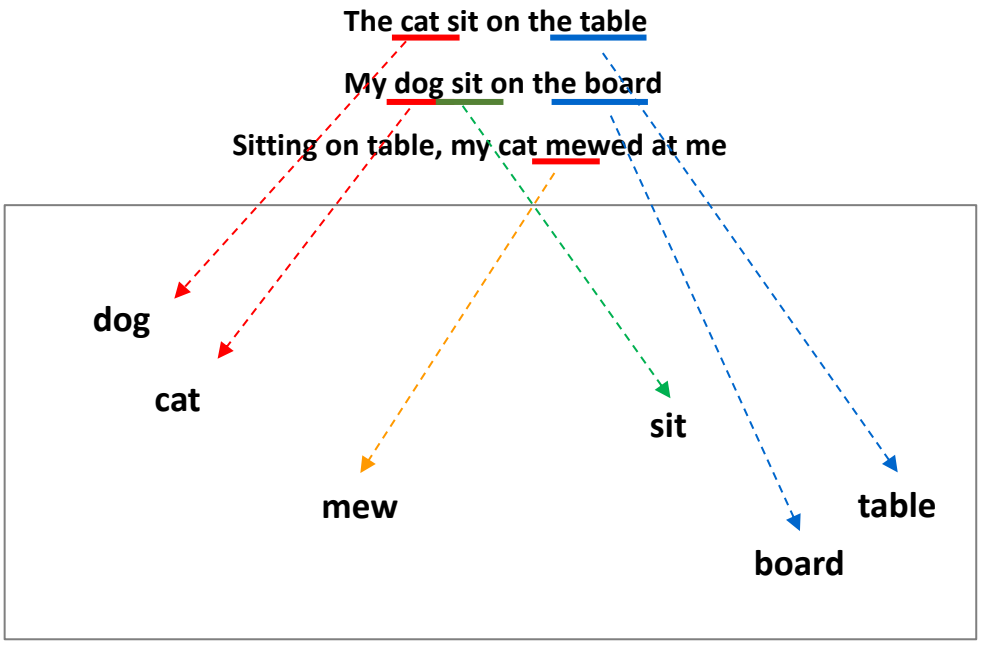
- **Bengio et al. proposed to fight the curse of dimensionality by *learning a distributed representation for words*.**
  - learning simultaneously (1) a distributed representation for each word along with (2) the probability function for word sequences, expressed in terms of these representations
- **Mikolov et al. proposed word2vec algorithm to quickly calculate word vectors.**

'dog'=	[**0.31**, **0.21**, 0.01, 0.01, …]
'cat'=	[**0.45**, **0.17**, 0.01, 0.01, …]
'topic modeling'=	[0.01, 0.01, **0.22**, **0.54**, …]
'dim. reduction'=	[0.01, 0.01, **0.19**, **0.45**, …]

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, *3*(Feb), 1137-1155.
Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Word embedding and example (Revisited)

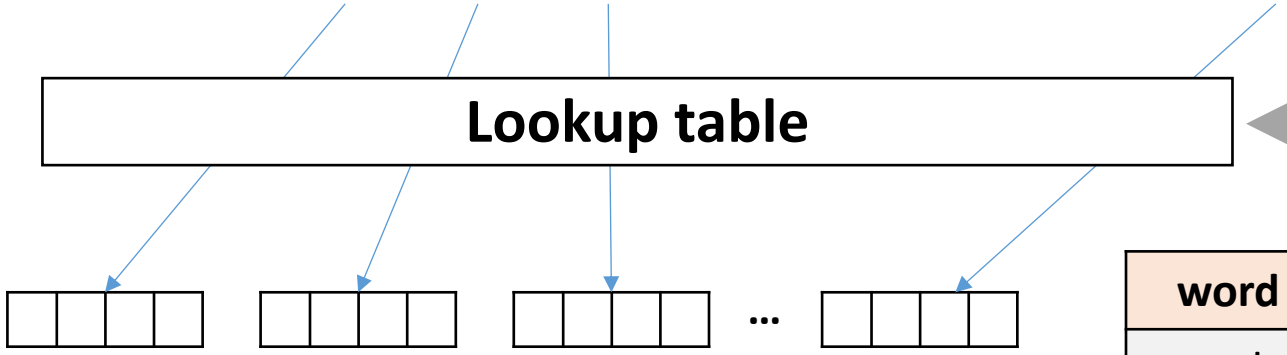
- **Word2Vec, Doc2Vec 은 행렬 형태로 데이터를 가공하지 않은 채 단어나 문서를 벡터로 표현하는 임베딩 방법**
  - 행렬 형태로 가공하지 않는다는 것은 알고리즘을 돌릴 때의 입장이며,
  - 수학적으로 해석하면 매우 큰 sparse vector를 저차원의 dense vector로 표현하는 방법
  - Word2Vec, Doc2Vec을 이해하는 키는 그들이 **"어떤 정보를 보존"**하며 저차원 dense vector를 학습하는지를 파악하는 것



< 단어의 의미 공간 >

Distributed representation for words

- 예제 문장
  - English: ‘Classify news using machine learning and deep learning.’



word	word vector			
dog	0.22	0.42	…	-0.54
cat	-0.30	0.38	…	-0.60
news	…			
…	…			
learning	…			
…	…			
deep	…			
…	…			

[Lookup table]

Convolutional Neural Networks for sentence classification

- Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences.
- Kim, Y. (2014). Convolutional neural networks for sentence classification.
- Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification.

A convolutional neural network for modelling sentences

A Convolutional Neural Network for Modelling Sentences

Nal Kalchbrenner, Edward Grefenstette, Phil Blunsom

(Submitted on 8 Apr 2014)

The ability to accurately represent sentences is central to language understanding. We describe a convolutional architecture dubbed the Dynamic Convolutional Neural Network (DCNN) that we adopt for the semantic modelling of sentences. The network uses Dynamic k-Max Pooling, a global pooling operation over linear sequences. The network handles input sentences of varying length and induces a feature graph over the sentence that is capable of explicitly capturing short and long-range relations. The network does not rely on a parse tree and is easily applicable to any language. We test the DCNN in four experiments: small scale binary and multi-class sentiment prediction, six-way question classification and Twitter sentiment prediction by distant supervision. The network achieves excellent performance in the first three tasks and a greater than 25% error reduction in the last task with respect to the strongest baseline.

Subjects: **Computation and Language (cs.CL)**
Cite as: **arXiv:1404.2188 [cs.CL]**
(or **arXiv:1404.2188v1 [cs.CL]** for this version)

https://arxiv.org/abs/1404.2188

Convolutional Neural Networks for Sentence Classification

Convolutional Neural Networks for Sentence Classification

Yoon Kim

(Submitted on 25 Aug 2014 (v1), last revised 3 Sep 2014 (this version, v2))

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.

Comments: To appear in EMNLP 2014
Subjects: **Computation and Language (cs.CL)**; Neural and Evolutionary Computing (cs.NE)
Cite as: arXiv:1408.5882 [cs.CL]
(or arXiv:1408.5882v2 [cs.CL] for this version)

https://arxiv.org/abs/1408.5882

A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification

A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification

Ye Zhang, Byron Wallace

(Submitted on 13 Oct 2015 (v1), last revised 6 Apr 2016 (this version, v4))

Convolutional Neural Networks (CNNs) have recently achieved remarkably strong performance on the practically important task of sentence classification (kim 2014, kalchbrenner 2014, johnson 2014). However, these models require practitioners to specify an exact model architecture and set accompanying hyperparameters, including the filter region size, regularization parameters, and so on. It is currently unknown how sensitive model performance is to changes in these configurations for the task of sentence classification. We thus conduct a sensitivity analysis of one-layer CNNs to explore the effect of architecture components on model performance; our aim is to distinguish between important and comparatively inconsequential design decisions for sentence classification. We focus on one-layer CNNs (to the exclusion of more complex models) due to their comparative simplicity and strong empirical performance, which makes it a modern standard baseline method akin to Support Vector Machine (SVMs) and logistic regression. We derive practical advice from our extensive empirical results for those interested in getting the most out of CNNs for sentence classification in real world settings.

Subjects: **Computation and Language (cs.CL)**; Learning (cs.LG); Neural and Evolutionary Computing (cs.NE)
Cite as: arXiv:1510.03820 **[cs.CL]**
(or **arXiv:1510.03820v4 [cs.CL]** for this version)

https://arxiv.org/abs/1510.03820

CNN for sentence classification

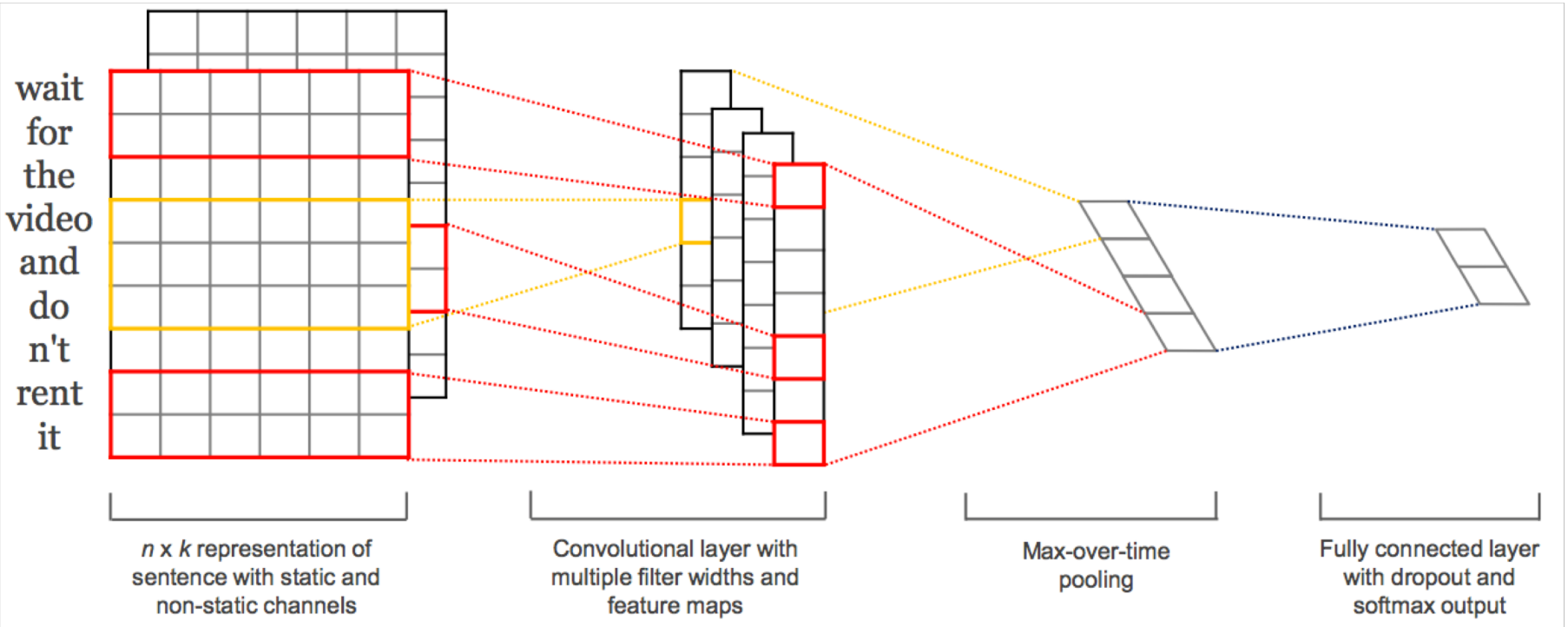
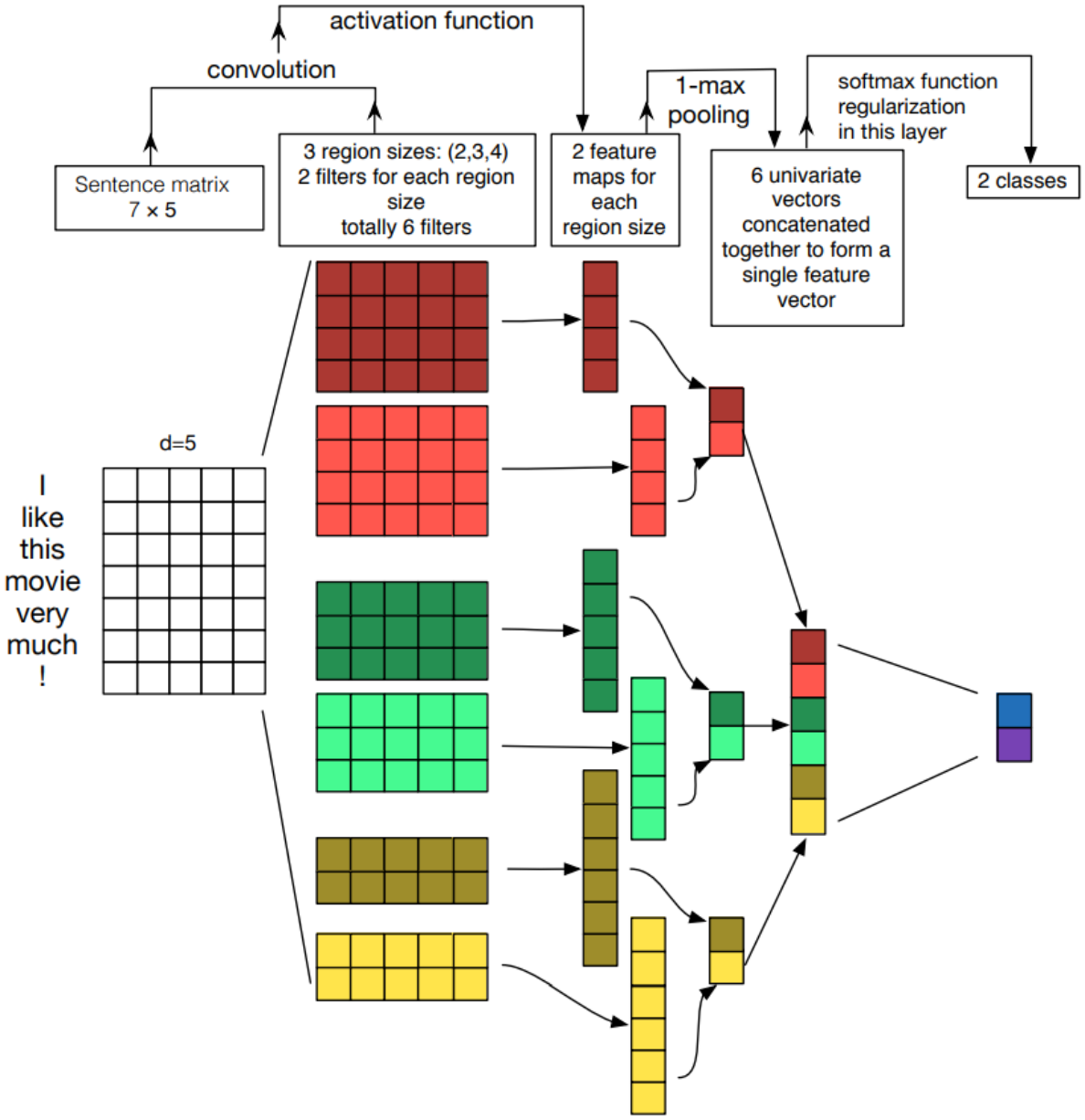


Figure 1: Model architecture with two channels for an example sentence.

CNN for sentence classification



Input: **Sentence**

↓

Embedded lookup table

↓

(wide) **convolution layer** with multiple filters

↓

1-max **pooling**

↓

concatenate

↓

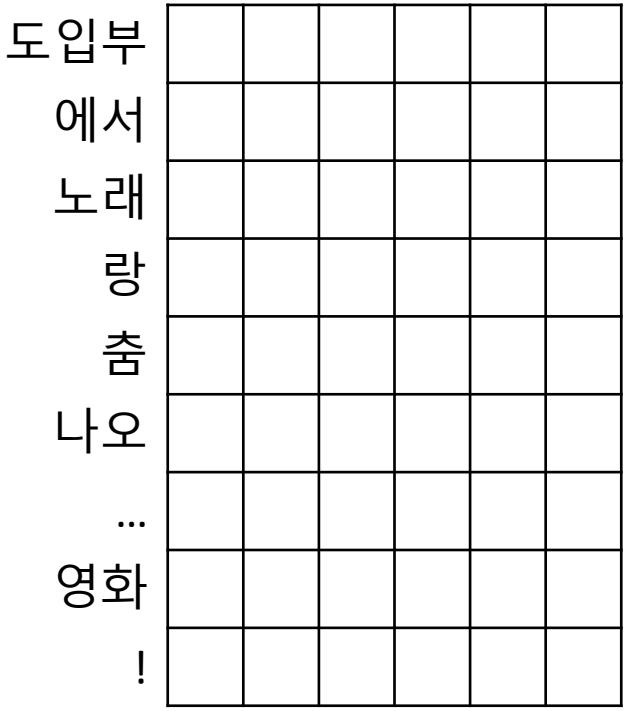
softmax

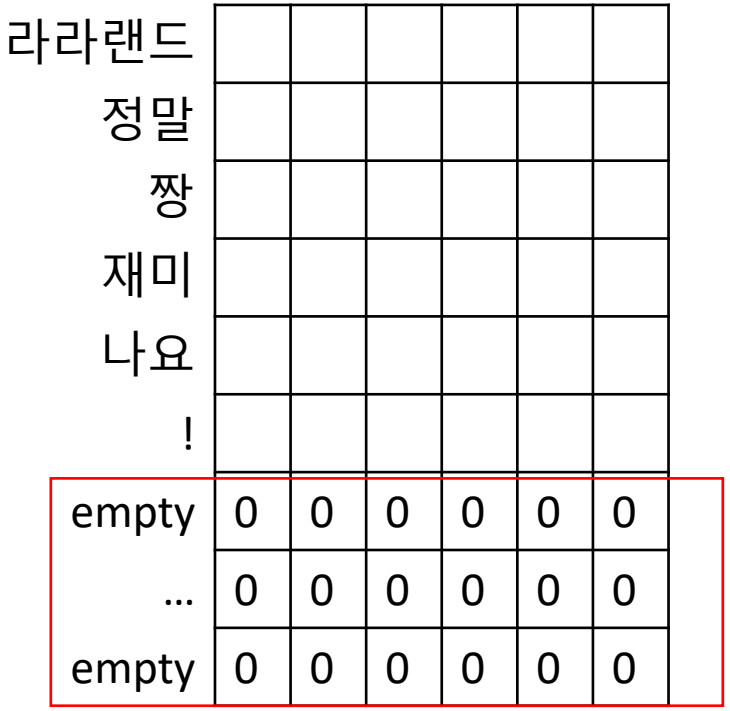
CNN for sentence classification

- **Example sentence**
  - 도입부에서 노래랑 춤 나오길래 꽤 큰 뮤지컬 영화구나..하고 보기 시작했는데, 노래가 끝나는 지점에서 영화 연출로 한 테이크에 이어지는 거보고 숨을 못 쉬었다.그 뒤부터 계속 놀라움에 연속!헐리우드의 과거와 현재를 모두 담은 영화!
  - 라라랜드 정말 짱 재미나요!

CNN for sentence classification

- **n x k representation of sentence**
  - n : 문장을 구성하는 단어 수
  - k : 단어에 대한 임베디드 벡터의 차원





문장의 단어 수를 맞춰주기 위한 zero-padding

CNN for sentence classification

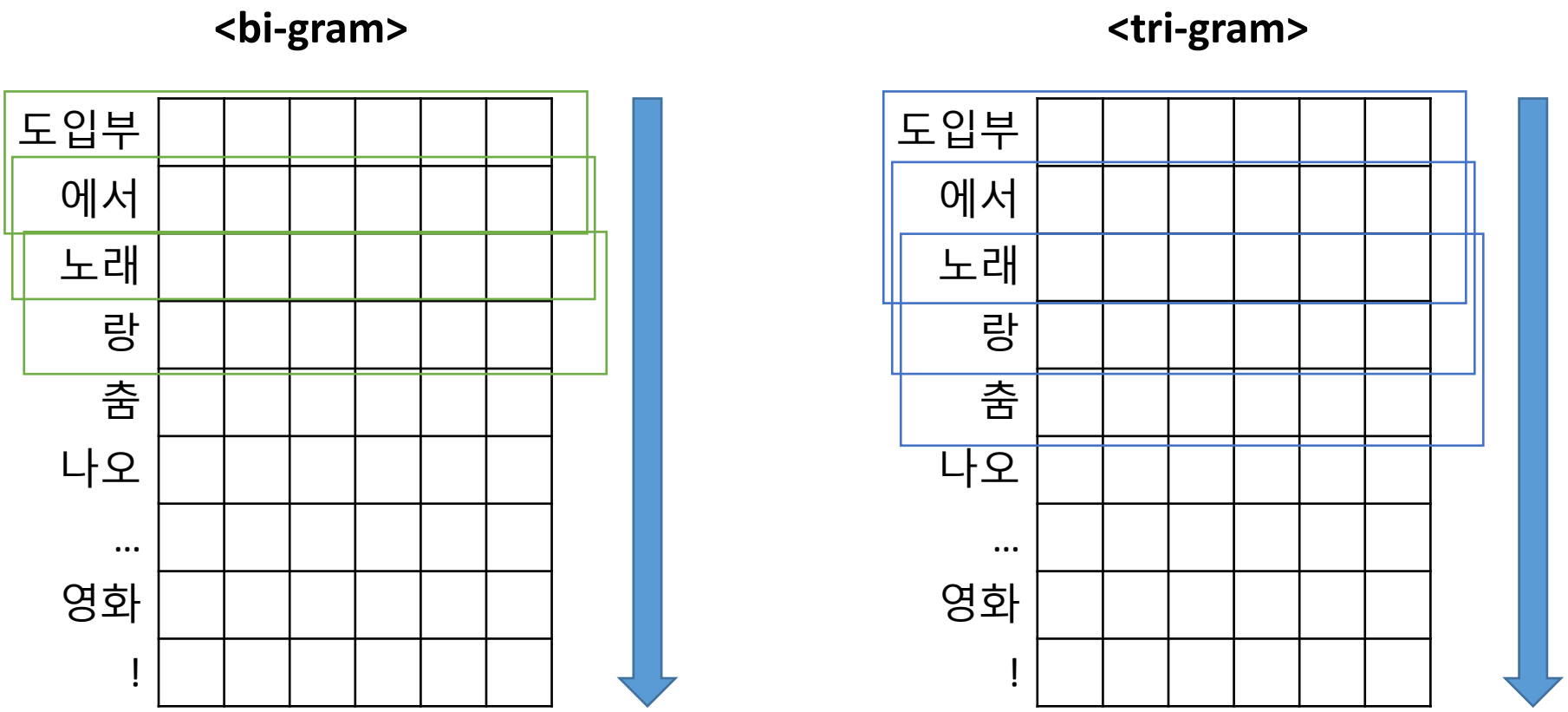
- **How to construct embedded vectors**
  - **CNN-static**
    - 먼저 word2vec과 같은 임베딩 기법으로 각 단어를 임베딩한 후에 해당 벡터를 사용
    - 단, Kim은 같은 도메인의 텍스트 데이터로 임베딩한 것이 아니라 구글 뉴스에서 등장하는 100 billion 개의 단어를 미리 word2vec으로 학습한 후의 산출물을 이용하였다.
  - **CNN-nonstatic**
    - 앞과 동일한 임베디드 벡터들을 사용하는데, CNN을 학습하는 과정에서 임베디드 벡터를 조금씩 fine-tuning
    - 일종의 전이학습 (Transfer learning)?
  - **CNN-rand**
    - 각 단어의 임베디드 벡터의 성분을 random하게 초기화한 후에, 모델을 학습하는 과정에서 임베디드 벡터를 같이 생성

CNN for sentence classification

- **Multi-channel input data?**
  - **CNN-multichannel**
  - 문장을 두 개의 channel로 보고 학습
    - 처음 두 channel은 pre-trained embedded vector로 생성
    - fine-tuning하는 과정에서 하나의 channel은 임베디드 벡터를 계속 업데이트하는데, 다른 channel은 기존의 벡터를 그대로 유지
    - 즉, **CNN-static**과 **CNN-nonstatic**을 동시에 고려하는 효과

CNN for sentence classification

- 하나의 Filter size: [number of grams, length of embedded vectors]
- stride = 1



CNN for sentence classification

- 제안한 방법이 다른 모델들보다 좋은 성능을 보임
- CNN-rand보다는 미리 학습된 임베디드 벡터를 사용할 때 성능이 더 좋음
- CNN-multichannel의 효과가 생각보다 미미함

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

CNN for sentence classification

- Static channel과 non-static channel에 대해 특정 단어와 가장 유사한 공간에 임베딩된 단어 4개를 추출

	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.

Pooling strategy

- Kalchbrenner et al. proposed ***k*-max pooling** and **dynamic *k*-max pooling**.
  - ***k*-max pooling**: 1-max pooling과는 다르게, features map에서 상위 k개의 값을 샘플링하여 다음 레이어로 전달
  - **Dynamic *k*-max pooling**: (convolution layer를 여러 개 쌓은 경우) 현재의 convolution layer가 전체 네트워크에서 상대적으로 어느 위치에 있는지를 반영하여 k값을 다이나믹하게 변동
    - input layer에서 멀어질수록 k값이 작아져서 지속적으로 텐서의 차원을 줄임

- Zhang and Wallace는 **1-max pooling**이 다른 pooling 전략보다 outperformed 한 결과가 나왔다고 설명함
  - Why?

참고

- 또 다른 읽을거리
  - 김건영, 이창기 (2016). Convolutional neural network를 이용한 한국어 영화평 감성 분석
  - 조휘열, 김진화, 윤상웅, 김경민, 장병탁 (2015). 컨볼루션 신경망 기반 대용량 텍스트 데이터 분류 기술
- References
  - http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/
  - http://docs.likejazz.com/cnn-text-classification-tf/

Character-level CNN

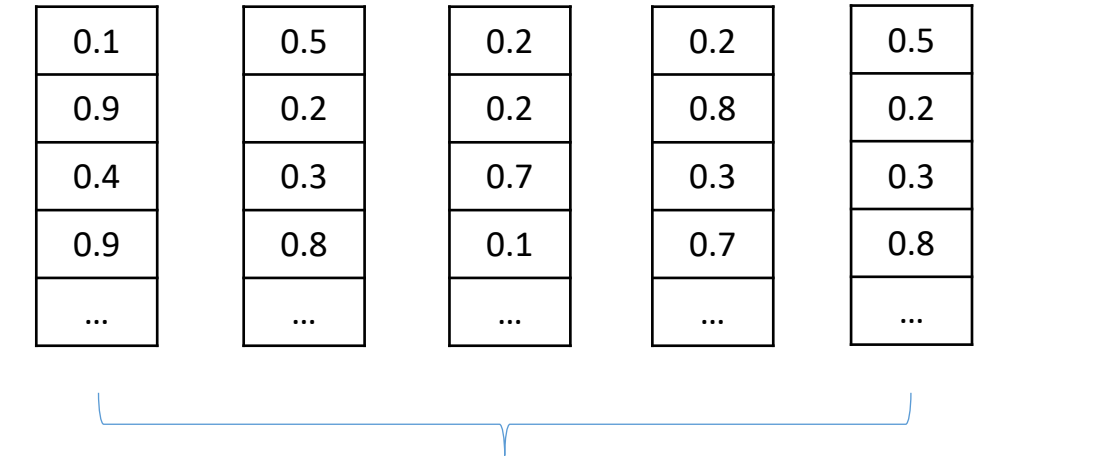
서울대학교병원 정보화실

고태훈 (taehoonko@snuh.org)

최세원 (swc@snuh.org)

Embedded dense vector (revisited)

- **How to represent *"Machine learning and deep learning are interesting."***
  - Word-based (embedded dense vector)

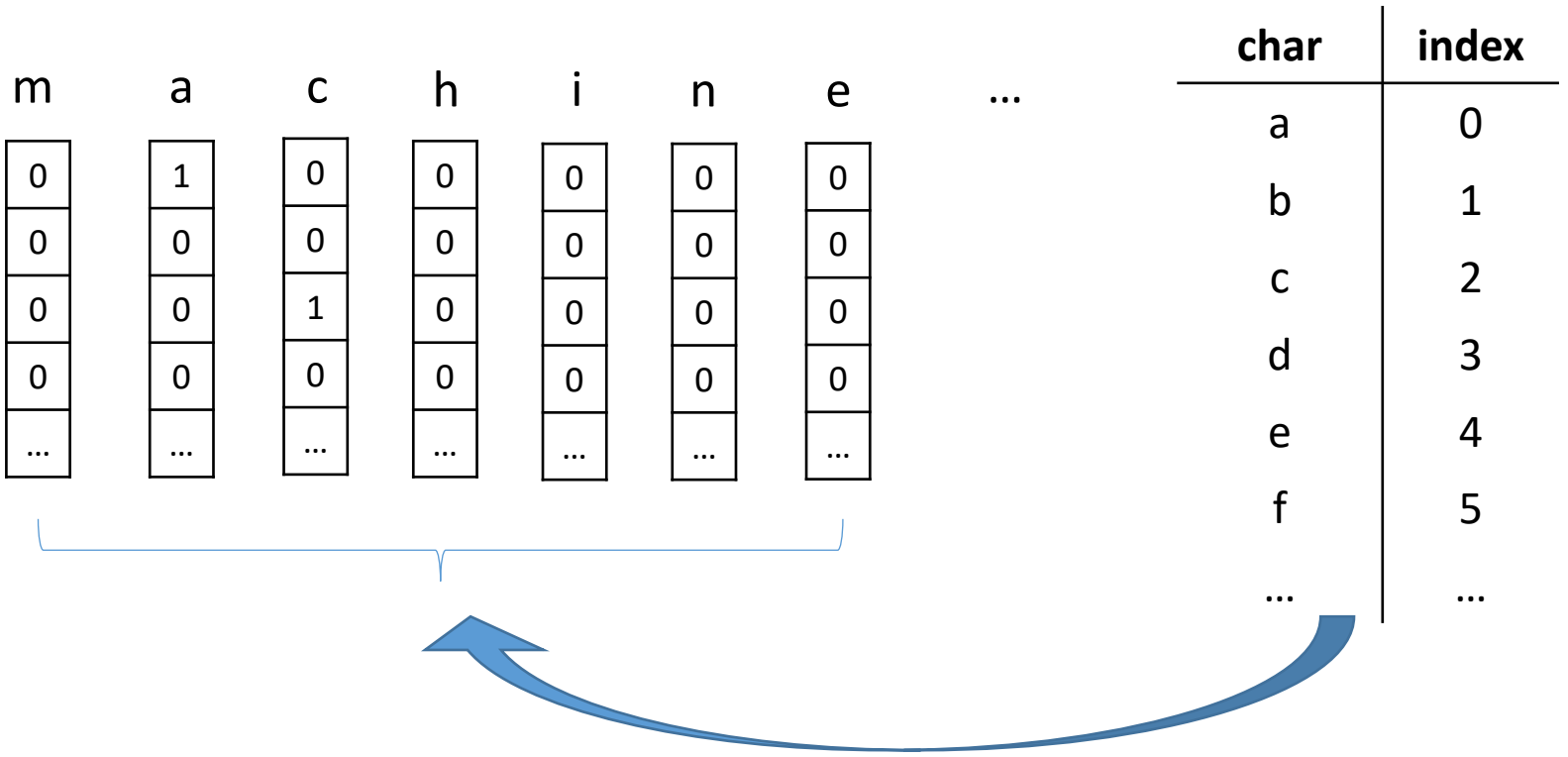


word	v1	v2	…
machine	0.1	0.9	…
learning	0.5	0.2	…
and	0.2	0.2	…
deep	0.2	0.8	…
are	0.9	0.7	…
interesting	0.5	0.4	…
…	…	…	…

lookup an embedded vector table

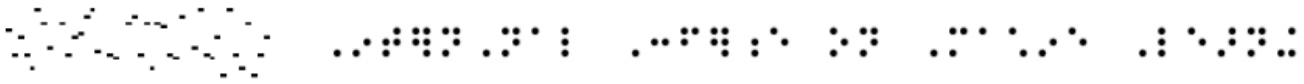
Character-level representation (revisited)

- **How to represent *"Machine learning and deep learning are interesting."***
  - Character-based (one-hot vector)



Character-level CNN

- **Character quantization**
  - First, Quantize each character using 1-of-***m*** encoding (a.k.a. one-hot encoding).
  - And then, the sequence of characters (such as a word, a sentence) is transformed to a sequence of such ***m*** sized vectors with fixed length ***l***.
    - ***m***: Number of characters used in the input language
    - ***l***: Length of the sequence of characters



(a) Binary (b) Braille

Figure 1. Comparison of our binary encoding and Braille on the text "International Conference on Machine Learning"

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

Character-level CNN

- **Illustration of Zhang's model**

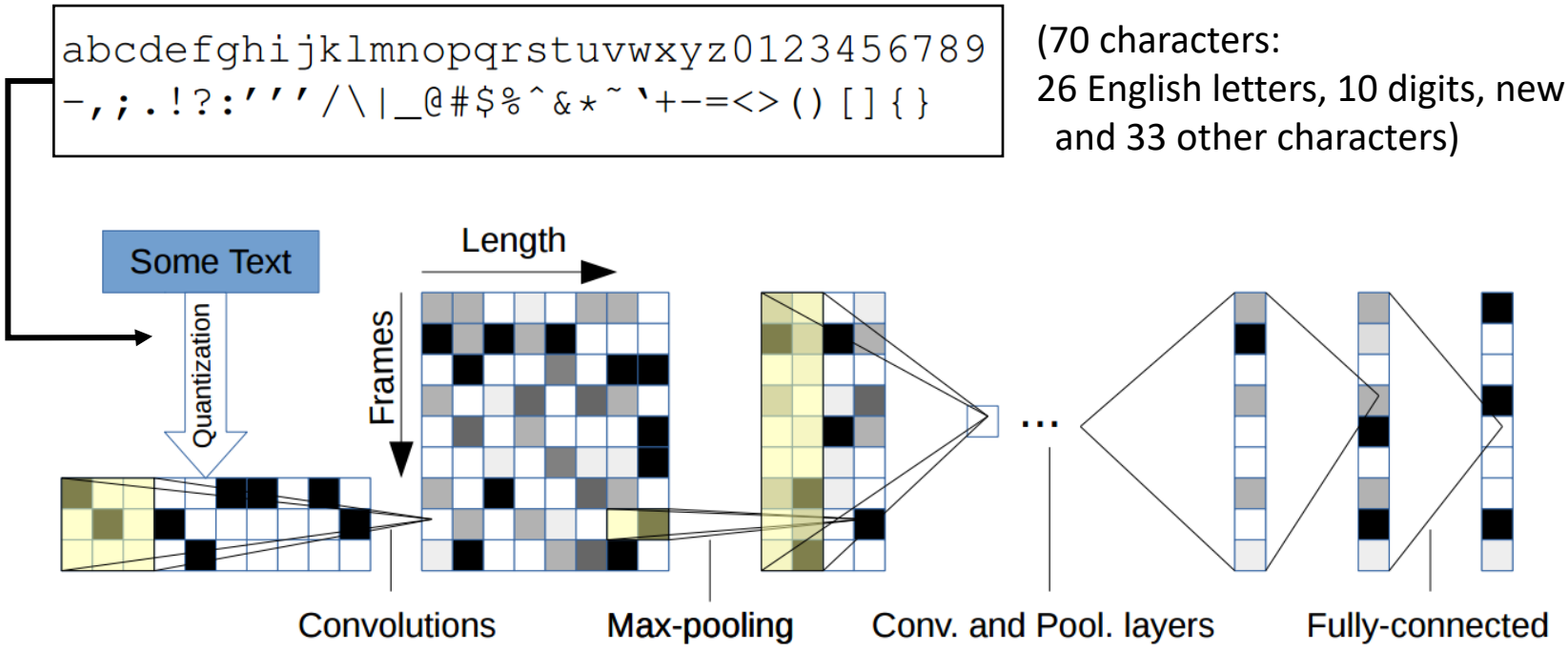


Figure 2. Illustration of our model

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

Character-level CNN

- **Results**
  - Please see the paper: https://arxiv.org/pdf/1502.01710.pdf

Table 3. DBpedia ontology classes. The numbers contain only samples with both a title and a short abstract.

Class	Total	Train	Test
Company	63,058	40,000	5,000
Educational Institution	50,450	40,000	5,000
Artist	95,505	40,000	5,000
Athlete	268,104	40,000	5,000
Office Holder	47,417	40,000	5,000
Mean Of Transportation	47,473	40,000	5,000
Building	67,788	40,000	5,000
Natural Place	60,091	40,000	5,000
Village	159,977	40,000	5,000
Animal	187,587	40,000	5,000
Plant	50,585	40,000	5,000
Album	117,683	40,000	5,000
Film	86,486	40,000	5,000
Written Work	55,174	40,000	5,000

Table 4. DBpedia results. The numbers are accuracy.

Model	Thesaurus	Train	Test
Large ConvNet	No	**99.96%**	98.27%
Large ConvNet	Yes	99.89%	**98.40%**
Small ConvNet	No	99.37%	98.02%
Small ConvNet	Yes	99.62%	98.15%
Bag of Words	No	96.29%	96.19%
word2vec	No	89.32%	89.09%