

Recurrent Convolutional Neural Networks for Text Classification

Siwei Lai, Liheng Xu, Kang Liu, Jun Zhao

AAAI 2015

読み手: 周 双双

Background

- Text classification

- Feature representation in previous studies: bag-of-words (BoW) model, where unigrams, bigrams , n-grams or some exquisitely designed patterns.
- Disadvantages: “ignore the contextual information or word order in texts and remain unsatisfactory for capturing the semantics of the words.”



Why Recurrent Convolutional Neural Network?

- **Recursive Neural Network (RecursiveNN)**

- **Disadvantage:** "Its performance heavily depends on the performance of the textual tree exhibits a time complexity of at least $O(n^2)$, where n is the length of the text."

- **Recurrent Neural Network(RecurrentNN)**

- **Disadvantage:** "a biased model, where later words are more dominant than earlier words"

- **Convolutional Neural Network(CNN)**

- **Advantages:**

- capture the semantic of texts better than recursive or recurrent NN.
 - Time complexity of CNN is $O(n)$.

- **Disadvantages:**

- It is difficult to determine the window size



Recurrent Convolutional Neural Network (RCNN): Learn more contextual information than conventional window-based neural networks

Recurrent Convolutional Neural Network

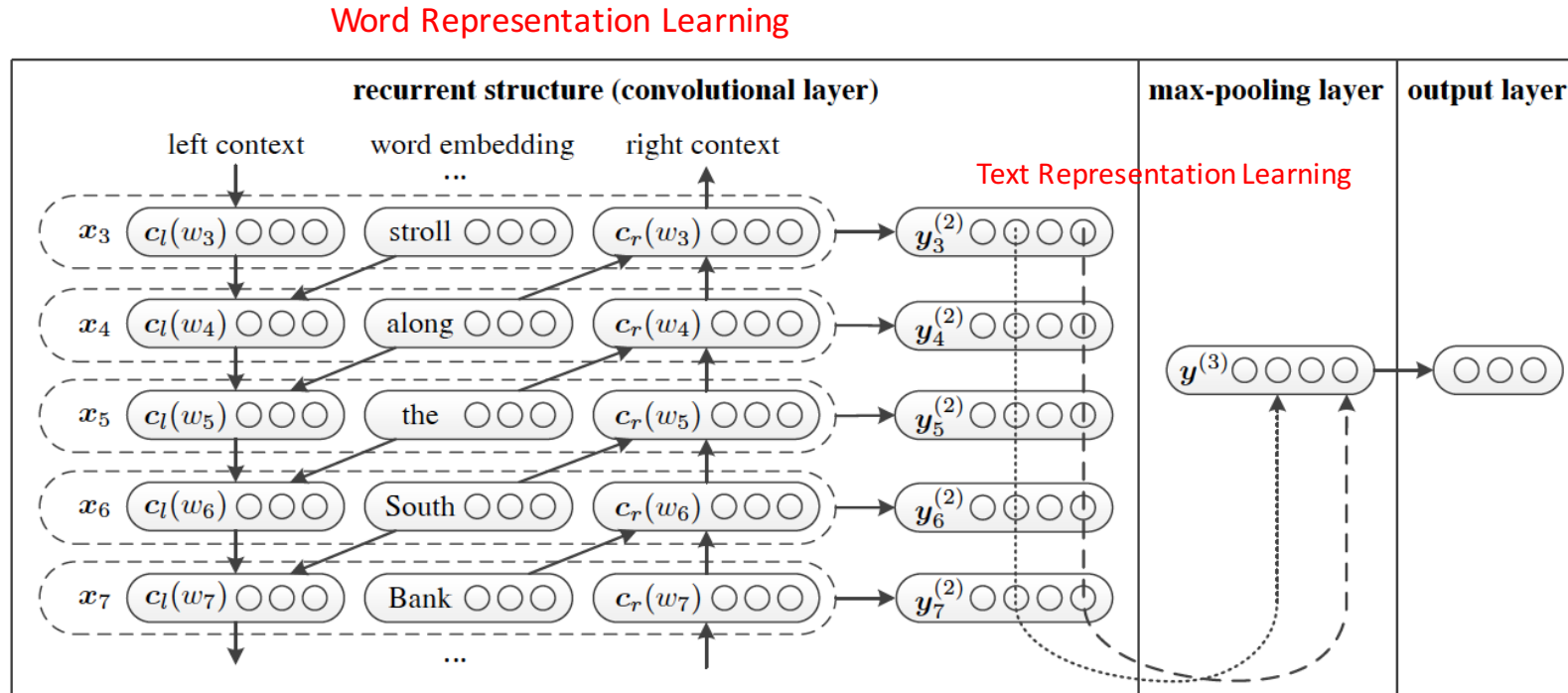



Figure 1: The structure of the recurrent convolutional neural network. This figure is a partial example of the sentence “A sunset stroll along the South Bank affords an array of stunning vantage points”, and the subscript denotes the position of the corresponding word in the original sentence.

Word Representation Learning

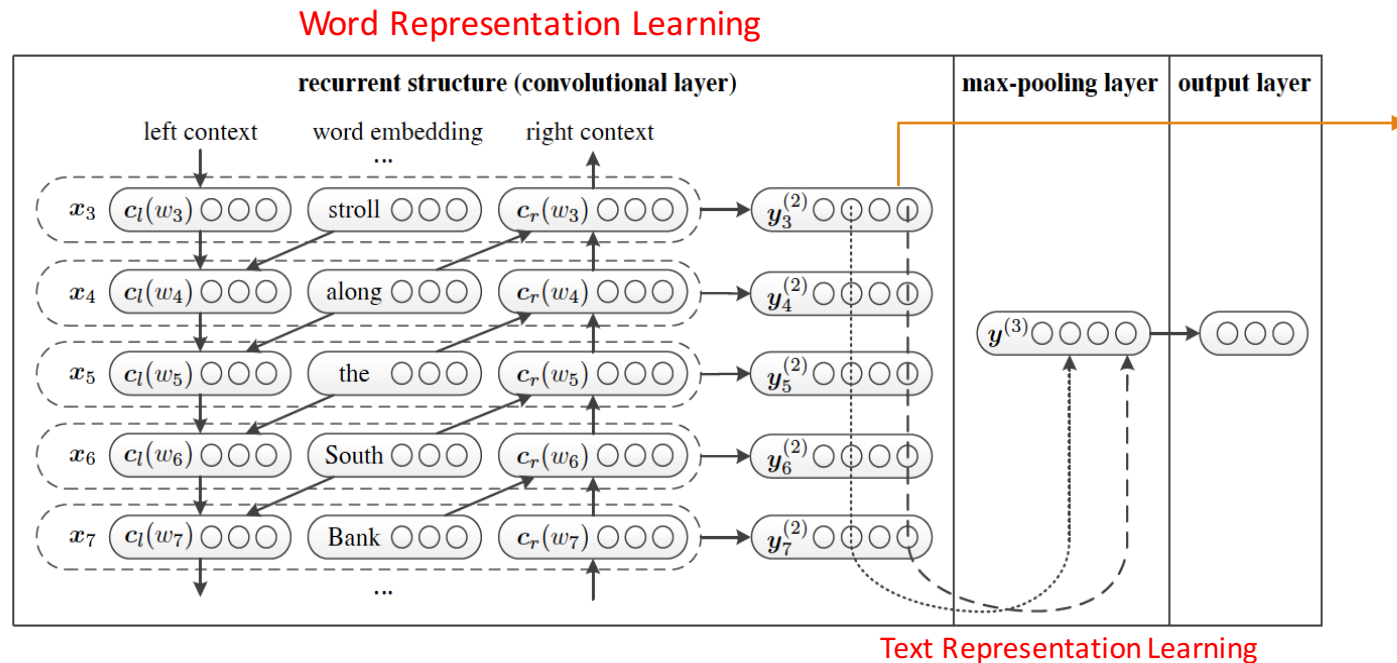
The representation of word w_i is the concatenation of the left-side context vector $c_l(w_i)$, the word embedding $e(w_i)$ and the right-side context vector $c_r(w_i)$

$$\mathbf{x}_i = [\mathbf{c}_l(w_i); \mathbf{e}(w_i); \mathbf{c}_r(w_i)]$$


$$\mathbf{c}_l(w_i) = f(W^{(l)}\mathbf{c}_l(w_{i-1}) + W^{(sl)}\mathbf{e}(w_{i-1})) \quad \mathbf{c}_r(w_i) = f(W^{(r)}\mathbf{c}_r(w_{i+1}) + W^{(sr)}\mathbf{e}(w_{i+1}))$$

- The left-side context vector $c_l(w_i)$ and the right-side context vector $c_r(w_i)$ are calculated by the similar way.
- $e(w_{i-1})$ is the word embedding of word w_{i-1} , $c_l(w_{i-1})$ is the left-side context vector of the previous word w_{i-1}
- $e(w_{i+1})$ is the word embedding of word w_{i+1} , $c_r(w_{i+1})$ is the right-side context vector of the next word w_{i+1}

Recurrent Convolutional Neural Network



$$y_i^{(2)} = \tanh \left(W^{(2)} x_i + b^{(2)} \right)$$

- Apply a linear transformation together with the tanh activation function to x_i and send the result to the next layer.
- $y_i^{(2)}$ is a latent semantic vector, will be used to determine the most useful factor for representing the text.

Figure 1: The structure of the recurrent convolutional neural network. This figure is a partial example of the sentence “A sunset stroll along the South Bank affords an array of stunning vantage points”, and the subscript denotes the position of the corresponding word in the original sentence.

Text Representation Learning

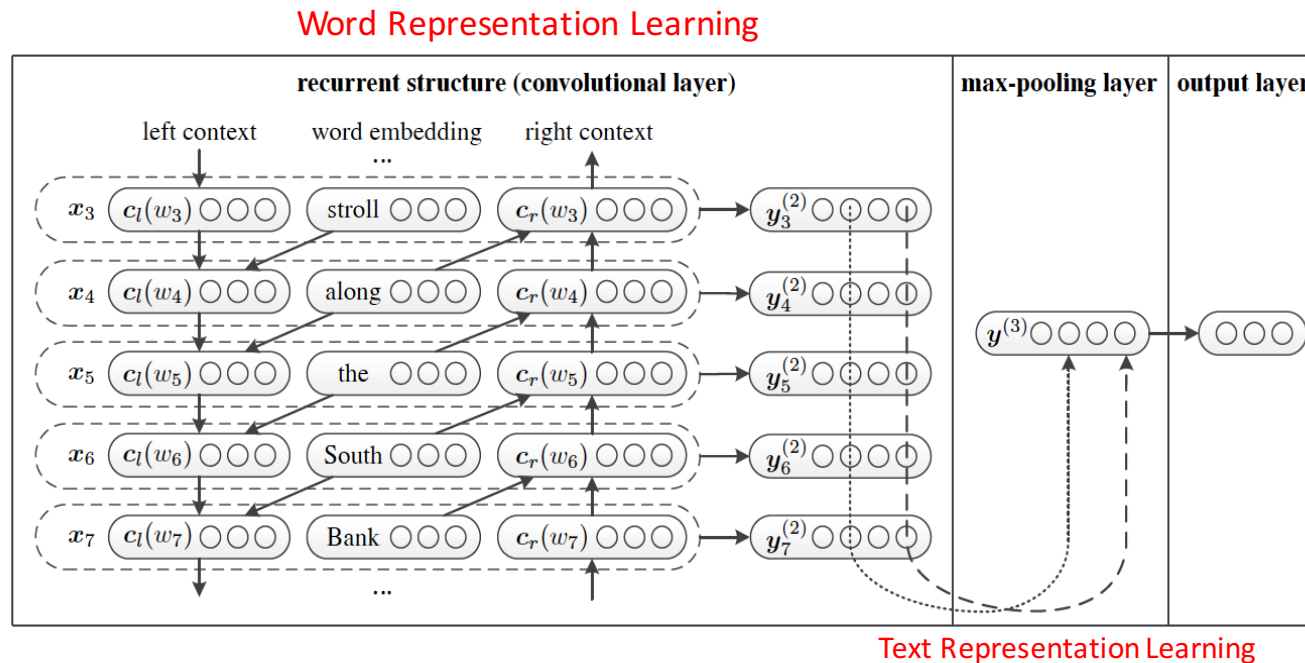


Figure 1: The structure of the recurrent convolutional neural network. This figure is a partial example of the sentence “A sunset stroll along the South Bank affords an array of stunning vantage points”, and the subscript denotes the position of the corresponding word in the original sentence.

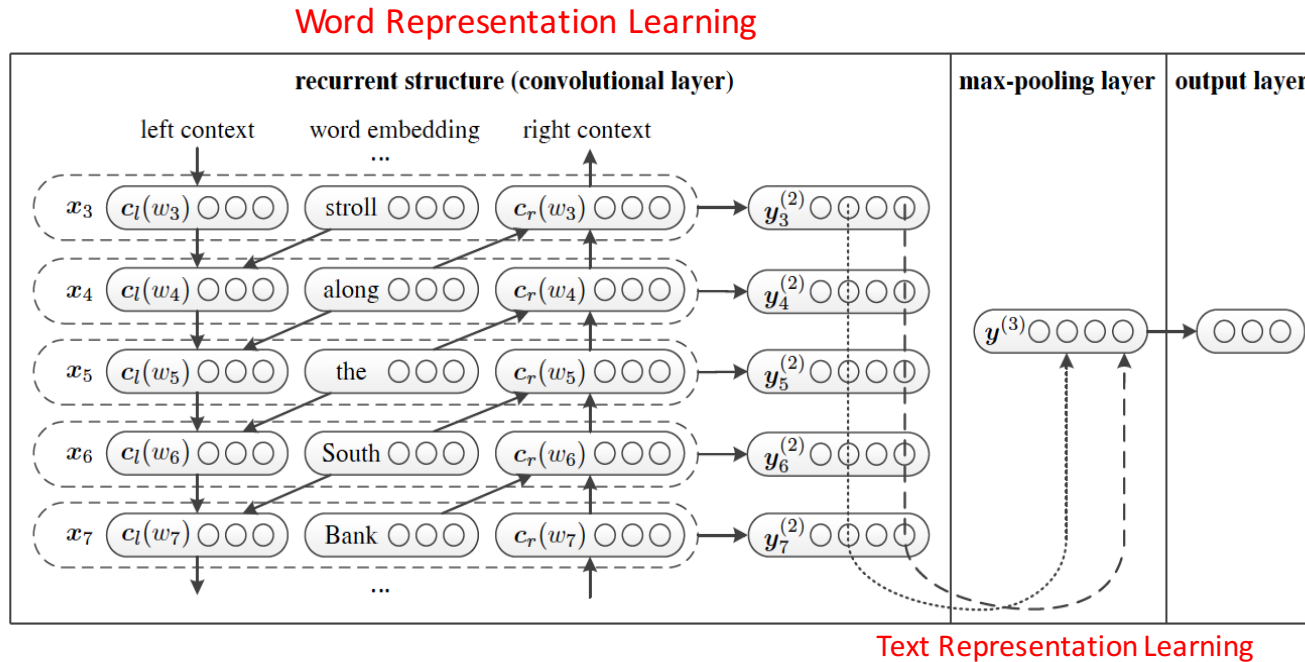
$$y^{(3)} = \max_{i=1}^n y_i^{(2)}$$

- A max-pooling layer is applied on all the representations of words.
- A various lengths text is converted into a fixed-length vector.
- The k -th element of $y^{(3)}$ is the maximum in the k -th elements of $y_i^{(2)}$

Why max pooling, not average pooling?

- Aim to find the most important latent semantic factors in the texts.

Output layer



$$y^{(4)} = W^{(4)}y^{(3)} + b^{(4)}$$

Softmax function is applied to $y^{(4)}$

$$p_i = \frac{\exp(y_i^{(4)})}{\sum_{k=1}^n \exp(y_k^{(4)})}$$

Figure 1: The structure of the recurrent convolutional neural network. This figure is a partial example of the sentence “A sunset stroll along the South Bank affords an array of stunning vantage points”, and the subscript denotes the position of the corresponding word in the original sentence.

Training network parameters

They define all of the parameters to be trained as θ

$$\theta = \{E, \mathbf{b}^{(2)}, \mathbf{b}^{(4)}, \mathbf{c}_l(w_1), \mathbf{c}_r(w_n), W^{(2)}, W^{(4)}, W^{(l)}, W^{(r)}, W^{(sl)}, W^{(sr)}\}$$

The training target of the network is used to maximize the log-likelihood with respect to θ , where D is the training document set and $class_D$ is the correct class of document D .

$$\theta \mapsto \sum_{D \in \mathbb{D}} \log p(class_D | D, \theta)$$

Stochastic gradient descent is used to optimize the training target.

$$\theta \leftarrow \theta + \alpha \frac{\partial \log p(class_D | D, \theta)}{\partial \theta}$$

Dataset and Experiments settings

Dataset	C	Train/Dev/Test	Len	Lang	Measure
20News	4	7520/836/5563	429	EN	Marco-F1
Fudan	20	8823/981/9832	2981	CH	Accuracy
ACL	5	146257/28565/28157	25	EN	Accuracy
SST	5	8544/1101/2210	19	EN	Accuracy

- Chose a common used hyper-parameters following previous studies (Collobert et al. 2011; Turian, Ratinov, and Bengio 2010)
- Set learning rate α as 0.01. Hidden layer size H as 100, word embedding dimension as 50, context vector dimension as 50.
- Word embedding are respectively pre-trained on English and Chinese Wikipedia dumps by the default parameter in word2vec with the skip-gram algorithm.

Experiments Results – Comparison of Methods

Model	20News	Fudan	ACL	SST
BoW + LR	92.81	92.08	46.67	40.86
Bigram + LR	93.12	92.97	47.00	36.24
BoW + SVM	92.43	93.02	45.24	40.70
Bigram + SVM	92.32	93.03	46.14	36.61
Average Embedding	89.39	86.89	41.32	32.70
ClassifyLDA-EM (Hingmire et al. 2013)	93.60	-	-	-
Labeled-LDA (Li, Sun, and Zhang 2008)	-	90.80	-	-
CFG (Post and Bergsma 2013)	-	-	39.20	-
C&J (Post and Bergsma 2013)	-	-	49.20	-
RecursiveNN (Socher et al. 2011b)	-	-	-	43.20
RNTN (Socher et al. 2013)	-	-	-	45.70
Paragraph-Vector (Le and Mikolov 2014)	-	-	-	48.70
CNN	94.79	94.04	47.47	46.35
RCNN	96.49	95.20	49.19	47.21

- **Bag-of-words(BoW)/Bigram** + logistic regression(LR)/SVM (Wang and Manning 2012)
- **Average Embedding + LR** : a tf-idf weighted average of the word embeddings and subsequently applies a softmax layer
- **LDA**: ClassifyLDA-EM and Labeled-LDA
- **Tree Kernels**: Post and Bergsma (2013) used various tree kernels as features. Here, they compared context-free grammar(**CFG**) produced by Berkeley parser and the ranking feature set of **C&J**.
- **RecursiveNN**: two recursive-based methods. RecursiveNN (Socher et al. 2011a) and Recursive Neural Tensor Networks (RNTNs) (Socher et al. 2013)
- **CNN**: a convolution kernel concatenated the word embeddings in a pre-defined window in Collobert et al. 2011.
- **Paragraph-Vector**: Le and Mikolov 2014.

Experiments Results and Discussion

Model	20News	Fudan	ACL	SST
BoW + LR	92.81	92.08	46.67	40.86
Bigram + LR	93.12	92.97	47.00	36.24
BoW + SVM	92.43	93.02	45.24	40.70
Bigram + SVM	92.32	93.03	46.14	36.61
Average Embedding	89.39	86.89	41.32	32.70
ClassifyLDA-EM (Hingmire et al. 2013)	93.60	-	-	-
Labeled-LDA (Li, Sun, and Zhang 2008)	-	90.80	-	-
CFG (Post and Bergsma 2013)	-	-	39.20	-
C&J (Post and Bergsma 2013)	-	-	49.20	-
RecursiveNN (Socher et al. 2011b)	-	-	-	43.20
RNTN (Socher et al. 2013)	-	-	-	45.70
Paragraph-Vector (Le and Mikolov 2014)	-	-	-	48.70
CNN	94.79	94.04	47.47	46.35
RCNN	96.49	95.20	49.19	47.21

- Neural network approaches (**RecursiveNN**, **CNN**, **RCNN**) outperform the traditional methods (e.g.BoW+LR) for all four datasets. Neural networks can capture more contextual information of features.
- RecursiveNN outperformed CNN, RCNN on SST dataset. The importance of max-pooling layer and convolutional layer. RNTN (3-5 hours) vs. RCNN (several minutes)
- Comparing RCNN with tree kernels methods, RCNN might be useful in low-resource languages.
- RCNN vs. CNN. Recurrent structure is better than window-based structure.

RCNN vs. CNN

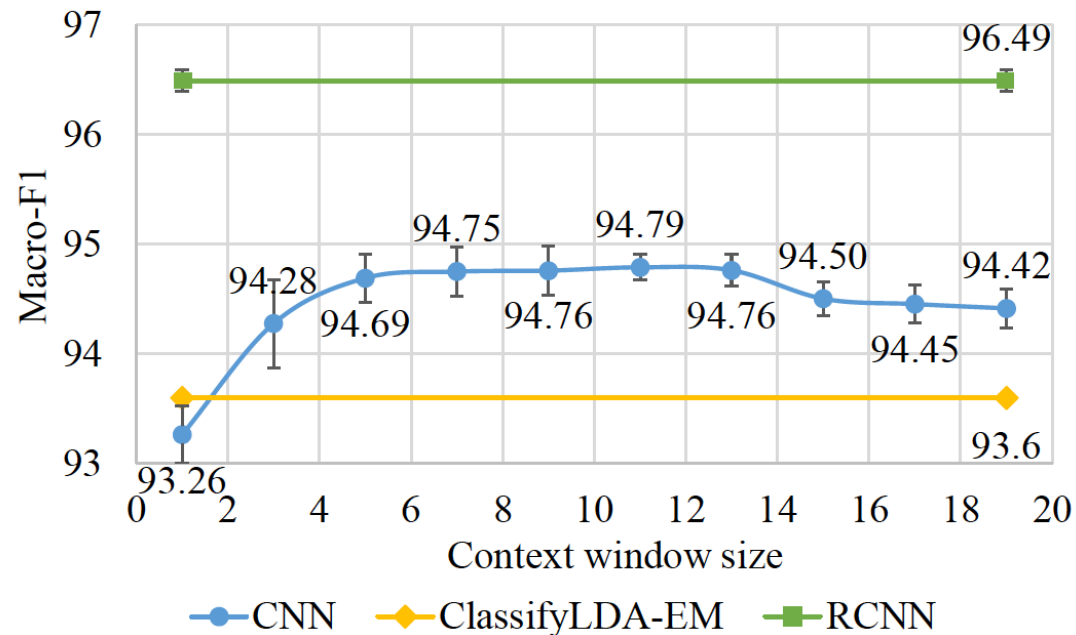


Figure 2: Macro-F1 curve for how context window size influences the performance of the 20Newsgroups classification

- They consider all odd window size from 1 to 19 to train and test the CNN model.
- RCNN outperforms the CNN for all window sizes and it does not rely on the window size.
- The recurrent structure can preserve longer contextual information and introduces less noise.

Learned Keywords

RCNN	
P	well <i>worth</i> the; a <i>wonderful</i> movie; even <i>stinging</i> at; and <i>invigorating</i> film; and <i>ingenious</i> entertainment; and <i>enjoy</i> .; 's <i>sweetest</i> movie
N	A <i>dreadful</i> live-action; Extremely <i>boring</i> .; is <i>n't</i> a; 's <i>painful</i> .; Extremely <i>dumb</i> .; an <i>awfully</i> derivative; 's <i>weaker</i> than; incredibly <i>dull</i> .; very <i>bad</i> sign;
RNTN	
P	an amazing performance; most visually stunning; wonderful all-ages triumph; a wonderful movie
N	for worst movie; A lousy movie; a complete failure; most painfully marginal; very bad sign

Table 3: Comparison of positive and negative features extracted by the RCNN and the RNTN

- The max-pooling layer selects the most important words in texts.
- Present the center word and its neighboring trigram.
- List the comparison results with RNTN (Socher et al.2013) on sentimental classification.

感想

- RCNN Does rely on a syntactic parser.
- RCNN might be useful in low-resource languages.
- RCNN has a lower time complexity of $O(n)$.
- Limitation of RCNN?
- Their model can capture the key components in texts for text classification



Capture the key components for disambiguation?
Plan to apply RCNN on other tasks

Source codes:

<https://github.com/knok/rcnn-text-classification>