Convolutional Neural Networks with Recurrent Neural Filters

Yi Yang
ASAPP

Chunyang Xiao
Bloomberg
CNNs for NLP problems

[Yoon Kim (2014)]
CNNs for NLP problems

Convolutional Neural Networks for Sentence Classification

Yoon Kim
New York University
yik255@nyu.edu

n x k representation of sentence with static and non-static channels

Convolutional layer with multiple filter widths and feature maps

Max-over-time pooling

Fully connected layer with dropout and softmax output

[Yoon Kim (2014)]
CNNs for NLP problems

Convolutionsal Neural Networks for Sentence Classification

Natural Language Processing (Almost) from Scratch

Ronan Collobert*
Jason Weston†
Léon Bottou‡
Michael Karlen
Koray Kavukcuoglu§
Pavel Kuksa∣

NEC Laboratories America
4 Independence Way
Princeton, NJ 08540

RONAN@COLLOBERT.COM
JWESTON@GOOGLE.COM
LEON@BOTTOU.ORG
MICHAEL.KARLEN@GMAIL.COM
KORAY@CS.NYU.EDU
PKUKSA@CS.RUTGERS.EDU

wait for the video and don't rent it

non-static channels  feature maps  softmax output
CNNs for NLP problems

Convolutional Neural Networks for Sentence Classification

Natural Language Processing (Almost) from Scratch

Question Answering over Freebase with Multi-Column Convolutional Neural Networks

Ronan Collobert
Jason Wee Meng Lee
Léon Bottou
Michael I. Jordan
Koray Kavukcuoglu
Pavel Kušáček
NEC Labs
4 Indepen
Princeton, nj

Li Dong†* Furu Wei†† Ming Zhou‡ Ke Xu†
†SKLSDE Lab, Beihang University, Beijing, China
‡Microsoft Research, Beijing, China

donglixp@gmail.com {fuwei, mingzhou}@microsoft.com
kexu@nlsde.buaa.edu.cn

BERT.COM
OGLE.COM
TTOU.ORG
MAIL.COM
S.NYU.EDU
TUGERS.EDU

non-static channels
feature maps
softmax output
CNNs for NLP problems

Convolutional Neural Networks for Sentence Classification

Natural Language Processing (Almost) from Scratch

Question Answering over Freebase with Multi-Column Convolutional Neural Networks

Convolutional Sequence to Sequence Learning

Yoon Kim (2014)
Linear convolution filters

wait for
wait for $x_{i:i+1}$

\[
\begin{array}{c}
\times \\
\end{array}
\]

$w_j$

$= c_{i,j} = f(w_j x_{i:i+1} + b_j)$
Linear convolution filters

\[ c_{i,j} = f(w_j x_{i:i+1} + b_j) \]

- Limit high-order filters
- Compositionality
- Long-term deps.
Linear convolution filters

\[
\begin{align*}
\text{wait for} & \quad \begin{array}{ccccccc}
\text{wait} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\end{array}
\end{align*}
\]

\[x_{i:i+1} \times w_j = c_{i,j} = f(w_j x_{i:i+1} + b_j)\]

- Limit high-order filters
  - compositionality
  - long-term deps.
- Filters are independent
  - duplication
Local label consistency ratio

Ratio of m-grams that share the same labels as the original sentences.

Socher et al. (2013)
Recurrent neural filters

don’t give up until it’s too late
Recurrent neural filters

$\mathbf{x}_{i:i+6}$

don’t give up until it’s too late
Recurrent neural filters

\[ h_{i:i+6} = \text{RNN}(h_{t-1}, x_t) \]

\[ x_{i:i+6} = \text{don’t give up until it’s too late} \]
Recurrent neural filters

\[ c_i = h_{i+6} \]

\[ h_t = \text{RNN}(h_{t-1}, x_t) \]

Text:

don't    give      up      until     it's      too      late
Recurrent neural filters

$c_i = h_{i+6}$

$h_{i:i+6}$

$x_{i:i+6}$

don’t give up until it’s too late

$h_t = RNN(h_{t-1}, x_t)$

- RNN is implemented as
  - gated recurrent unit
  - LSTM unit

Cho et al. (2014); Hochreiter and Schmidhuber (1997)
CNN architectures

- CNN sentence encoder
  \[ v = \max \{ C \}, \text{ where } C = [c_1, c_2, \cdots, c_{n-m+1}] \]

Yoon Kim (2014); Yang et al. (2015)
CNN architectures

- CNN sentence encoder
  \[ v = \max \{ C \}, \text{ where } C = [c_1, c_2, \cdots, c_{n-m+1}] \]

- Sentence classification (e.g., sentiment classification)
  \[ p(y|v) = \text{Softmax}(W_v v) \]

Yoon Kim (2014); Yang et al. (2015)
CNN architectures

- **CNN sentence encoder**
  \[ v = \max \{ C \}, \text{ where } C = [c_1, c_2, \cdots, c_{n-m+1}] \]

- **Sentence classification (e.g., sentiment classification)**
  \[ p(y|v) = \text{Softmax}(W_v v) \]

- **Sentence matching (e.g., answer sentence selection)**
  \[ p(y|v_1, v_2) = \text{Sigmoid}(v_1^\top W_v v_2) \]
Data

- Sentence classification
  - Stanford Sentiment Treebank (SST)
    - Binary classification / fine-grained classification

Socher et al. (2013); Wang et al. (2007); Yang et al. (2015)
Data

- Sentence classification
  - Stanford Sentiment Treebank (SST)
    - Binary classification / fine-grained classification
- Sentence matching
  - QASent
  - WikiQA

Socher et al. (2013); Wang et al. (2007); Yang et al. (2015)
Results: sentence classification

- Accuracy results for fine-grained sentiment classification

- CNN: linear-filter
- CNN: RNF-LSTM
- LSTM
- LSTM-maxpool

48.0
Results: sentence classification

- Accuracy results for fine-grained sentiment classification

- CNN: linear-filter (48.0)
- CNN: RNF-LSTM (53.4)
- LSTM
- LSTM-maxpool
Results: sentence classification

- Accuracy results for fine-grained sentiment classification
Results: sentence classification

- Accuracy results for binary sentiment classification

- CNN: linear-filter
  - Accuracy: 86.1

- CNN: RNF-LSTM
  - Accuracy: 90.0

- LSTM
  - Accuracy: 89.3

- LSTM-maxpool
  - Accuracy: 89.8
Results: sentence matching

- MAP results on the WikiQA dataset

<table>
<thead>
<tr>
<th>Method</th>
<th>MAP Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN: linear-filter</td>
<td>0.668</td>
</tr>
<tr>
<td>CNN: RNF-LSTM</td>
<td></td>
</tr>
<tr>
<td>LSTM</td>
<td></td>
</tr>
<tr>
<td>LSTM-maxpool</td>
<td></td>
</tr>
</tbody>
</table>
Results: sentence matching

- MAP results on the WikiQA dataset

- CNN: linear-filter
- CNN: RNF-LSTM
- LSTM
- LSTM-maxpool

Results:
- 0.668
- 0.729
Results: sentence matching

- MAP results on the WikiQA dataset

CNN: linear-filter
CNN: RNF-LSTM
LSTM
LSTM-maxpool

0.668
0.729
0.651
0.701
Results: sentence matching

- MAP results on the QASent dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>MAP Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN: linear-filter</td>
<td>0.750</td>
</tr>
<tr>
<td>CNN: RNF-LSTM</td>
<td>0.780</td>
</tr>
<tr>
<td>LSTM</td>
<td>0.745</td>
</tr>
<tr>
<td>LSTM-maxpool</td>
<td>0.762</td>
</tr>
</tbody>
</table>
Key phrases: the phrase label is the same as the sentence label (SST)
Conclusions

- Conventional CNNs adopt linear convolution filters that fail to account for language compositionality.

- Recurrent neural filters (RNFs) yield much better results than linear filters on many NLP tasks.

- Code: https://github.com/bloomberg/cnn-rnf