A Convolutional Neural Network for Modelling Sentences

Nal Kalchbrenner    Edward Grefenstette
Phil Blunsom

Department of Computer Science, Oxford University
Overview of Model

Represent sentences by extracting more abstract features

Input: sequence of word embeddings

Output: classification probabilities

Each layer involves

1. Convolution
2. Dynamic $k$-Max Pooling
3. Apply a non-linearity (tanh)
One-Dimensional Convolution

1. The filter \( m \in \mathbb{R}^m \)
2. The sequence \( s \in \mathbb{R}^s \)

Returns sequence \( c \in \mathbb{R}^{s-m+1} \)

\[
c_j = m^T s_{j-m+1:j}, j = 1, ..., s - m + 1
\]

Takes a dot product between length \( m \) subsequences of \( s \) and the filter \( m \)

Wide convolution pads \( s \) with \( m - 1 \) zeros on the left.
Convolution with Word Embeddings

Assume word embeddings of dimension $d$
Filter $\mathbf{m}$ will be in $\mathbb{R}^{d \times m}$
Sequence $\mathbf{s}$ will be in $\mathbb{R}^{d \times s}$
Each row of $\mathbf{m}$ will be convolved with the corresponding row of $\mathbf{s}$
**k-Max Pooling (LeCun et al.)**

Given \( k \) and sequence \( p \in \mathbb{R}^p, p \geq k \)

1. Return \( k \) largest elements of \( p \)
2. Keep elements in their original order

Denoted \( p_{\max}^k \in \mathbb{R}^k \)
Dynamic $k$-Max Pooling

“Smooth extraction of higher-order features”

$$k_L = \max \left( k_{top}, \left\lfloor \frac{L - l}{L} \right\rfloor s \right)$$

- $k_{top}$ is fixed parameter
- $l$ is current layer
- $L$ is total number of layers
- $s$ is sentence length
Folding

Elementwise sum of pairs rows of a matrix

\[ f : \mathbb{R}^{d \times n} \rightarrow \mathbb{R}^{d/2 \times n} \]

\[ f(M) = N \text{ where} \]

\[ N[i, j] = M[2i, j] + M[2i + 1, j], \]

\[ i = 0, \ldots, d/2 - 1, \quad j = 0, \ldots, n - 1 \]

- Introduces dependencies between different feature rows
- No added parameters
The cat sat on the red mat
### Size of Network

<table>
<thead>
<tr>
<th>Model</th>
<th>First Layer</th>
<th></th>
<th>Second Layer</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Width</td>
<td>Filters</td>
<td>Width</td>
<td>Filters</td>
<td>$k$-top</td>
</tr>
<tr>
<td>Binary</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>14</td>
<td>4</td>
</tr>
<tr>
<td>Multi-class</td>
<td>10</td>
<td>6</td>
<td>7</td>
<td>12</td>
<td>5</td>
</tr>
</tbody>
</table>
Training

Top layer is soft-max nonlinearity to predict probability distribution

$L_2$ regularization of parameters in objective function

Parameters are word embeddings, filter weights, & fully connected layers

Trained using Adagrad with mini-batches

“Processes multiple millions of sentences per hour on one GPU”
Experiments

1. Predicting sentiment of movie reviews - binary (Socher et al. 2013)
2. Predicting sentiment of movie reviews - multi-class (Socher et al. 2013)
3. Categorization of questions (Li and Roth 2002)
4. Sentiment of Tweets, labels based on emoticons (Go et al. 2009)

Feature embedding dimensionality chosen based on size of dataset
### Movies accuracy

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Fine-grained (%)</th>
<th>Binary (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>41.0</td>
<td>81.8</td>
</tr>
<tr>
<td>BiNB</td>
<td>41.9</td>
<td>83.1</td>
</tr>
<tr>
<td>SVM</td>
<td>40.7</td>
<td>79.4</td>
</tr>
<tr>
<td>RecNTN</td>
<td>45.7</td>
<td>85.4</td>
</tr>
<tr>
<td>Max-TDNN</td>
<td>37.4</td>
<td>77.1</td>
</tr>
<tr>
<td>NBoW</td>
<td>42.4</td>
<td>80.5</td>
</tr>
<tr>
<td>DCNN</td>
<td><strong>48.5</strong></td>
<td><strong>86.8</strong></td>
</tr>
</tbody>
</table>
First layer feature-detectors

**POSITIVE**
lovely comedic moments and several fine performances
good script, good dialogue, funny
sustains throughout is daring, inventive and
well written, nicely acted and beautifully
remarkably solid and subtly satirical tour de

**NEGATIVE**
nonexistent plot and pretentious visual style
it fails the most basic test as
so stupid, so ill conceived,
too dull and pretentious to be
hood rats butt their ugly heads in

**'NOT'**
n't have any huge laughs in its
no movement,
not stop me from enjoying much of
not that kung pow is n't funny
not a moment that is not false

**'TOO'**
too dull and pretentious to be
either too serious or too lighthearted,
too slow, too long and too
feels too formulaic and too familiar to
is too predictable and too self conscious
TREC 6-way classification accuracy

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Features</th>
<th>Acc. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HIER</td>
<td>unigram, POS, head chunks NE, semantic relations</td>
<td>91.0</td>
</tr>
<tr>
<td>MAXENT</td>
<td>unigram, bigram, trigram POS, chunks, NE, supertags CCG parser, WordNet</td>
<td>92.6</td>
</tr>
<tr>
<td>MAXENT</td>
<td>unigram, bigram, trigram POS, wh-word, head word shape, parser hypernyms, WordNet</td>
<td>93.6</td>
</tr>
<tr>
<td>SVM</td>
<td>unigram, POS, wh-word head word, parser hypernyms, WordNet 60 hand-coded rules</td>
<td>95.0</td>
</tr>
<tr>
<td>MAX-TDNN</td>
<td>unsupervised vectors</td>
<td>84.4</td>
</tr>
<tr>
<td>NBoW</td>
<td>unsupervised vectors</td>
<td>88.2</td>
</tr>
<tr>
<td>DCNN</td>
<td>unsupervised vectors</td>
<td>93.0</td>
</tr>
</tbody>
</table>
## Twitter sentiment

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>81.6</td>
</tr>
<tr>
<td>BiNB</td>
<td>82.7</td>
</tr>
<tr>
<td>MaxEnt</td>
<td>83.0</td>
</tr>
<tr>
<td>Max-TDNN</td>
<td>78.8</td>
</tr>
<tr>
<td>NBoW</td>
<td>80.9</td>
</tr>
<tr>
<td>DCNN</td>
<td>87.4</td>
</tr>
</tbody>
</table>
Dynamic Convolutional Neural Networks

- Convolutions apply function to n-grams
- Dynamic $k$-max pooling extracts most active feature, and chooses $k$ based on layer and sentence length
- Composing these two operations can be seen as feature detection
- Outperformed/stayed competitive with other neural approaches, baseline models, and state-of-the-art approaches without needing handcrafted features