Text Classification and Convolutional Neural Networks

COSC 7336: Advanced Natural Language Processing
Fall 2017

Some content on these slides was borrowed from J&M
Today’s lecture

- Text Classification: task definition
- Classical approaches to Text Classification
- Convolutional Neural Networks (CNN)
- Recent work using CNNs for Text Classification problems
- Demo: CNN for text
- Practical
Hello Dear
My name is: Engineer Mrs. Eva Rose, I am a citizen of Austria, a business woman specialized in mining of raw Gold in Africa; but now I am critically sick with esophageal cancer which has damaged almost all the cells in my body system and I will soon die according to my doctors.

My late husband died in an accident with our two daughters few years ago leaving me with our only son whom is just 10 years old and he is my most concern now as he is still a child and does not know anything about live and has nobody to take care of him after I am dead; because I and my late husband does not have any relatives, we both grew up in the orphanage home and got married under orphanage as orphans. So if I die now my innocent child would be left alone in this wicked world and I do not wish to send him to any orphanage home, I want him to grow up in the hands of an individual, not orphanage.

Please, i am begging you in the name of God to sincerely accept my proposal; let me instruct my bank to wire transfer my fund worth the sum of US$ 15,000,000.00 (FIFTEEN MILLION DOLLARS) to your account in your country immediately, then you take my son to your home and raise him as your own son. As you receive the fund into your account, you are entitled to take 30 percent and invest 70 percent for my son so that he will not suffer in his entire life.
What do these books have in common?
Other tasks that can be solved as TC

★ Sentiment classification
★ Native language identification
★ Profiling
Formal definition of the TC task

★ Input:
  ○ a document \( d \)
  ○ a fixed set of classes \( C = \{c_1, c_2, \ldots, c_J\} \)

★ Output: a predicted class \( c \in C \)
Methods for TC tasks

★ Rule based approaches
★ Machine Learning algorithms
  ○ Naive Bayes
  ○ Support Vector Machines
  ○ Logistic Regression
  ○ And now deep learning approaches
Naive Bayes for Text Classification

★ Simple approach
★ Based on the bag-of-words representation
Bag of words

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

The first reference to Bag of Words is attributed to a 1954 paper by Zellig Harris.
Naive Bayes

Probabilistic classifier

\[ \hat{c} = \arg\max_{c \in C} P(c|d) \quad (\text{eq. 1}) \]

According to Bayes rule:

\[ P(x|y) = \frac{P(y|x)P(x)}{P(y)} \quad (\text{eq. 2}) \]

Replacing eq. 2 into eq. 1:

\[ \hat{c} = \arg\max_{c \in C} P(c|d) = \arg\max_{c \in C} \frac{P(d|c)P(c)}{P(d)} \]

Dropping the denominator:

\[ \arg\max_{c \in C} P(d|c)P(c) \]
Naive Bayes

A document $d$ is represented as a set of features $f_1, f_2, \ldots, f_n$

$$\hat{c} = \arg\max_{c \in C} \frac{P(f_1, f_2, \ldots, f_n | c) \cdot P(c)}{P(c)}$$

How many parameters do we need to learn in this model?
Naive Bayes Assumptions

1. Position doesn’t matter
2. Naive Bayes assumption: probabilities $P(f_i | c)$ are independent given the class $c$ and thus we can multiply them:

$$P(f_1, f_2, \ldots, f_n | c) = P(f_1 | c) \cdot P(f_2 | c) \cdot \ldots \cdot P(f_n | c)$$

This leads us to:

$$c_{NB} = \arg\max_{c \in C} P(c) \prod_{f \in F} P(f | c)$$
Naive Bayes in Practice

We consider word positions:

\[ \text{positions} \leftarrow \text{all word positions in test document} \]

\[ c_{NB} = \arg\max_{c \in C} P(c) \prod_{i \in \text{positions}} P(w_i | c) \]

We also do everything in log space:

\[ c_{NB} = \arg\max_{c \in C} \log P(c) + \sum_{i \in \text{positions}} \log P(w_i | c) \]
Naive Bayes: Training

How do we compute $P(c)$ and $P(f_i|c)$?
Is Naive Bayes a good option for TC?

Scaling to Very Very Large Corpora for Natural Language Disambiguation

Michele Banko and Eric Brill
Microsoft Research
1 Microsoft Way
Redmond, WA 98052 USA
{mbanko,brill}@microsoft.com

Abstract

The amount of readily available on-line text has reached hundreds of billions of words and continues to grow. Yet for most core natural language tasks, algorithms continue to be optimized, tested and compared after training on corpora consisting of only one million words or less. In this paper, we potentially large cost of annotating data for those learning methods that rely on labeled text.

The empirical NLP community has put substantial effort into evaluating performance of a large number of machine learning methods over fixed, and relatively small, data sets. Yet since we now have access to significantly more data, one has to wonder what conclusions that have been drawn on small data sets may carry over when these learning methods are trained using much larger corpora.
Evaluation in TC

Confusion table

<table>
<thead>
<tr>
<th>Gold Standard</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>TP = true positives</td>
</tr>
<tr>
<td>False</td>
<td>FN = false negatives</td>
</tr>
</tbody>
</table>

Accuracy = \( \frac{TP + TN}{TP + TN + FN + FP} \)
Suppose we want to learn to classify each message in a web forum as “extremely negative”. We have a collected gold standard data:

- 990 instances are labeled as negative
- 10 instances are labeled as positive
- Test data has 100 instances (99- and 1+)
- A dumb classifier can get 99% accuracy by always predicting “negative”!
More Sensible Metrics: Precision, Recall and F-measure

\[ P = \frac{TP}{TP + FP} \]
\[ R = \frac{TP}{TP + FN} \]

<table>
<thead>
<tr>
<th>Gold Standard</th>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>TP = true positives</td>
<td>FP = False positives</td>
</tr>
<tr>
<td>False</td>
<td>FN = false negatives</td>
<td>TN = True negatives</td>
</tr>
</tbody>
</table>

F-measure = \[ \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \]
What about Multi-class problems?

- Multi-class: $c > 2$
- P, R, and F-measure are defined for a single class
- We assume classes are mutually exclusive
- We use per class evaluation metrics

\[
P = \frac{\sum_{j} C_{ii}}{\sum_{j} C_{ji}} \quad R = \frac{\sum_{j} C_{ii}}{\sum_{j} C_{ij}}
\]
Micro vs Macro Average

- **Macro average**: measure performance per class and then average
- **Micro average**: collect predictions for all classes then compute TP, FP, FN, and TN
- **Weighted average**: compute performance per label and then average where each label score is weighted by its support
### Example

<table>
<thead>
<tr>
<th>Class 1: Urgent</th>
<th>Class 2: Normal</th>
<th>Class 3: Spam</th>
<th>Pooled</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>true</strong>&lt;br&gt;system urgent</td>
<td><strong>true</strong>&lt;br&gt;system normal</td>
<td><strong>true</strong>&lt;br&gt;system spam</td>
<td><strong>true</strong>&lt;br&gt;system yes</td>
</tr>
<tr>
<td>8</td>
<td>60</td>
<td>200</td>
<td>268</td>
</tr>
<tr>
<td>11</td>
<td>55</td>
<td>33</td>
<td>99</td>
</tr>
<tr>
<td><strong>false</strong>&lt;br&gt;system not</td>
<td><strong>false</strong>&lt;br&gt;system not</td>
<td><strong>false</strong>&lt;br&gt;system not</td>
<td><strong>false</strong>&lt;br&gt;system no</td>
</tr>
<tr>
<td>8</td>
<td>40</td>
<td>51</td>
<td>99</td>
</tr>
<tr>
<td>340</td>
<td>212</td>
<td>83</td>
<td>635</td>
</tr>
</tbody>
</table>

**Precision**

- Class 1: \(\text{precision} = \frac{8}{8+11} = .42\)
- Class 2: \(\text{precision} = \frac{60}{60+55} = .52\)
- Class 3: \(\text{precision} = \frac{200}{200+33} = .86\)
- Pooled: \(\text{microaverage precision} = \frac{268}{268+99} = .73\)

**Macroaverage Precision**

\(\text{macroaverage precision} = \frac{.42 + .52 + .86}{3} = .60\)
Train/Test Data Separation
Convolutional Neural Networks
Neocognitron (Fukushima, 1980)
LeNet (LeCun, 1998)
Convolution

(source: ICML2013 Deep Learning Tutorial, Yan LeCun et al.)
Convolution

(Source: Feature extraction using convolution, Stanford Deep Learning Wiki)
Convolution

(Source: Feature extraction using convolution, Stanford Deep Learning Wiki)
Pooling or Subsampling
Pooling

(source: Karpathy, CS231n Convolutional Neural Networks for Visual Recognition)
Pooling

(source: Karpathy, CS231n Convolutional Neural Networks for Visual Recognition)
Properties

★ Local invariance
★ Compositionality

Adapted from: http://cs.nyu.edu/~fergus/tutorials/deep_learning_cvpr12/
CNNs for NLP

★ Same as images, text exhibits some local invariance properties that can be modeled by CNNs
★ CNNs are not as popular as recurrent neural networks (to be discussed next class) for text analysis, but there are many cases where they work pretty well.
★ Big advantage: CNNs can be trained efficiently since they take full advantage of parallelism.
A character-level CNN

Example from Sebastián Sierra
A character-level CNN
A character-level CNN
A character-level CNN
A character-level CNN
A character-level CNN
A character-level CNN
A character-level CNN
A character-level CNN
A character-level CNN
A character-level CNN

They are fed to the FC layer.

We expect to capture high-level features at this layer.

hidden_dims: 150

<table>
<thead>
<tr>
<th>G</th>
<th>T</th>
<th>C</th>
<th>A</th>
<th>A</th>
<th>C</th>
<th>A</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Convolutional neural networks for sentence classification

Convolutional neural networks for sentence classification

<table>
<thead>
<tr>
<th>Model</th>
<th>MR</th>
<th>SST-1</th>
<th>SST-2</th>
<th>Subj</th>
<th>TREC</th>
<th>CR</th>
<th>MPQA</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN-rand</td>
<td>76.1</td>
<td>45.0</td>
<td>82.7</td>
<td>89.6</td>
<td>91.2</td>
<td>79.8</td>
<td>83.4</td>
</tr>
<tr>
<td>CNN-static</td>
<td>81.0</td>
<td>45.5</td>
<td>86.8</td>
<td>93.0</td>
<td>92.8</td>
<td>84.7</td>
<td>89.6</td>
</tr>
<tr>
<td>CNN-non-static</td>
<td>81.5</td>
<td>48.0</td>
<td>87.2</td>
<td>93.4</td>
<td>93.6</td>
<td>84.3</td>
<td>89.5</td>
</tr>
<tr>
<td>CNN-multichannel</td>
<td>81.1</td>
<td>47.4</td>
<td><strong>88.1</strong></td>
<td>93.2</td>
<td>92.2</td>
<td><strong>85.0</strong></td>
<td>89.4</td>
</tr>
<tr>
<td>RAE (Socher et al., 2011)</td>
<td>77.7</td>
<td>43.2</td>
<td>82.4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>86.4</td>
</tr>
<tr>
<td>MV-RNN (Socher et al., 2012)</td>
<td>79.0</td>
<td>44.4</td>
<td>82.9</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>RNTN (Socher et al., 2013)</td>
<td>-</td>
<td>45.7</td>
<td>85.4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DCNN (Kalchbrenner et al., 2014)</td>
<td>-</td>
<td>48.5</td>
<td>86.8</td>
<td>-</td>
<td>93.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Paragraph-Vec (Le and Mikolov, 2014)</td>
<td>-</td>
<td>48.7</td>
<td>87.8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CCAE (Hermann and Blunsom, 2013)</td>
<td>77.8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>87.2</td>
</tr>
<tr>
<td>Sent-Parser (Dong et al., 2014)</td>
<td>79.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>86.3</td>
</tr>
<tr>
<td>NBSVM (Wang and Manning, 2012)</td>
<td>79.4</td>
<td>-</td>
<td>-</td>
<td>93.2</td>
<td>-</td>
<td>81.8</td>
<td>86.3</td>
</tr>
<tr>
<td>MNB (Wang and Manning, 2012)</td>
<td>79.0</td>
<td>-</td>
<td>-</td>
<td><strong>93.6</strong></td>
<td>-</td>
<td>80.0</td>
<td>86.3</td>
</tr>
<tr>
<td>G-Dropout (Wang and Manning, 2013)</td>
<td>79.0</td>
<td>-</td>
<td>-</td>
<td>93.4</td>
<td>-</td>
<td>82.1</td>
<td>86.1</td>
</tr>
<tr>
<td>F-Dropout (Wang and Manning, 2013)</td>
<td>79.1</td>
<td>-</td>
<td>-</td>
<td><strong>93.6</strong></td>
<td>-</td>
<td>81.9</td>
<td>86.3</td>
</tr>
<tr>
<td>Tree-CRF (Nakagawa et al., 2010)</td>
<td>77.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>81.4</td>
<td>86.1</td>
</tr>
<tr>
<td>CRF-PR (Yang and Cardie, 2014)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>82.7</td>
<td>-</td>
</tr>
<tr>
<td>SVM$_S$ (Silva et al., 2011)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td><strong>95.0</strong></td>
<td>-</td>
</tr>
</tbody>
</table>
Recent work using CNNs: Text Classification

- Text Classification
- Architecture with up to 29 convolutional layers
- Idea is to learn a hierarchical representation of text
- Achieve state of the art on most datasets and outperform recent work using shallow CNNs
- They reach state of the art on large data sets > 630k
- No statistical tests for significance
- They couldn’t outperform a hierarchical method adapted for multiple sentences.
Recent work using CNNs: Authorship Attribution

<table>
<thead>
<tr>
<th># of tweets</th>
<th>CNN-2</th>
<th>CNN-1</th>
<th>SCH</th>
<th>CHAR</th>
<th>LSTM-2</th>
<th>CNN-W</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>0.724</td>
<td>0.717</td>
<td>0.672</td>
<td>0.655</td>
<td>0.597</td>
<td>0.509</td>
</tr>
<tr>
<td>200</td>
<td>0.665</td>
<td>0.665</td>
<td>0.614</td>
<td>0.585</td>
<td>0.528</td>
<td>0.460</td>
</tr>
<tr>
<td>100</td>
<td>0.613</td>
<td>0.617</td>
<td>0.565</td>
<td>0.517</td>
<td>0.438</td>
<td>0.417</td>
</tr>
<tr>
<td>50</td>
<td>0.542</td>
<td>0.562</td>
<td>0.507</td>
<td>0.466</td>
<td>0.364</td>
<td>0.366</td>
</tr>
</tbody>
</table>

A statistical t-test on the results over the 10 disjoint datasets shows that the difference between CNN-2 and CHAR, LSTM-2, and CNN-W are statistically significant at $p < 0.001$. We could
CNNs for Sentence Classification Demo