Natural Language Processing (almost) from Scratch

Ronan Collobert
Jason Weston
Léon Bottou

Michael Karlen
Koray Kavukcuoglu
Pavel Kuksa

Speaker: Jason Chen
Motivation & Benchmark Test

Part-of-speech tagging (POS)
Chunking (CHUNK)
Named entity recognition (NER)
Semantic Role Labeling (SRL)
Part-of-speech tagging
Chunking
Named entity recognition
Semantic Role Labeling

Motivation
& Benchmark Test

• Labeling each word with an unique tag that indicates its syntactic role.

• E.g. Noun Phrases (NP), Verb Phrases (VP), Determiner (Det), etc.
Part-of-speech tagging

**Chunking**

Named entity recognition

Semantic Role Labeling

- Labeling segments of a sentence with syntactic constituents such as noun or verb phrases (NP or VP).
- Each word is assigned only one unique tag

Motivation & Benchmark Test
Part-of-speech tagging

Chunking

**Named entity recognition**

- Labeling atomic elements in the sentence into categories.
- E.g. PERSON, LOCATION, DATE, NUMERIC

Motivation & Benchmark Test
Semantics Role Labeling

- Giving a semantic role to a syntactic constituent of a sentence.
- Feature categories include:
  - The POS and syntactic labels of words;
  - The Node’s position in relation to the verb;
  - The syntactic path to the verb in the parse tree;
  - The verb sub-categorization;
  - The voice of the sentence: active or passive.
- E.g. John ate the apple

ARG0 REL ARG1

Motivation & Benchmark Test
Neural Network Structure

Traditional Approach

1. Extract a rich set of hand designed feature from sentence.
2. Feed the feature set into classification algorithm, such as Support Vector Machine (SVM) with a linear kernel.

Neural Network Approach

1. Takes the feature vectors of complete sentence/segment of text (window).
2. Passes through the lookup table layer, transform words into feature vectors.
3. Produces local features around each word of the sentence using linear/convolutional layers.
4. Combine local features into a global feature vector.
5. Feed into standard affine layers.
Transform Words into Feature Vectors

- Maps each word indices into feature vectors by a look up table operation.
- Consider of efficiency, words are feeding as indices.
- Formally noted as

\[
L_{T_w}(w) = \langle W \rangle^1_w, \ w \in D
\]

- The above equation can be extend as

\[
L_{T_w}(\lbrack w \rbrack^T) = \langle W \rangle^1_w
\]
Transform Words into Feature Vectors - Extend

- Extend to any discrete features, provide features other than words if one suspects that these features.
- E.g. pre-processing keeps case information.

Formally noted as

\[
LT_{w_1, \ldots, w_k}(\text{w}) = \begin{pmatrix}
\langle W \rangle_{w_K}^1 \\
\vdots \\
\langle W \rangle_{w_K}^k
\end{pmatrix}
\]

The above equation can be extend as

\[
LT_{w_1, \ldots, w_k}([w]_T^T) = \begin{pmatrix}
\langle W \rangle_{[w_1]}^1 & \cdots & \langle W \rangle_{[w_1]}^T \\
\vdots & \ddots & \vdots \\
\langle W \rangle_{[w_K]}^1 & \cdots & \langle W \rangle_{[w_K]}^T
\end{pmatrix}
\]
Extract High Level Features from Word Feature Vector
Extract High Level Features from Word Feature Vector – Window Approach

- Variable length equals to width of the window $k_{sz}$.
- Given a word to tag, a fixed size window of words around this word.
- Each window passed through the lookup table layer, producing a word features matrix with size $d_{wrd} \times k_{sz}$

$$f_{\theta}^{1} = \langle LT_{W}(\left[w_{r}\right]_{1}^{T})\rangle_{t}^{d_{win}} = \begin{pmatrix} \langle W \rangle_{1}^{1}[w_{r}]_{t-d_{win}/2} \\ \vdots \\ \langle W \rangle_{1}^{1}[w_{r}]_{t} \\ \vdots \\ \langle W \rangle_{1}^{1}[w_{r}]_{t+d_{win}/2} \end{pmatrix}$$
Extract High Level Features from Word Feature Vector – Window Approach

- $f^l_\theta$ can feed to one or several standard linear neural network layers.
  $$f^l_\theta = W^l f^{l-1}_\theta + b^l$$

- Use HardTanh layer as the activation function.
  $$\text{HardTanh}(x) = \begin{cases} 
-1 & \text{if } x < -1 \\
    x & \text{if } -1 \leq x \leq 1 \\
    1 & \text{if } x > 1 
\end{cases}$$

- Padding special "PADDING" word $d_{\text{win}}/2$ times at the beginning and the end.
Extract High Level Features from Word Feature Vector – Sentence Approach

- Window approach fails with SRL, where the tag of a word depends on a verb chosen beforehand in the sentence and the verb falls outside the window.
- In this case, tagging a word requires the consideration of the whole sentence.
- Implementing convolutional layer for sentence approach. A convolutional layer can be seen as a generalization of a window approach.
Formally, using previous notation, the convolutional layer can be noted as
\[
\langle f_{\theta}^{l} \rangle_t^1 = W^l \langle f_{\theta}^{l-1} \rangle_t^{d_{\text{win}}} + b^l
\]
- \(W^l\) is shared across all windows \(t\) in the sequence.
- Convolutional layers are often stacked to extract higher level features, so, it must be followed a non-linearity layer.
- We use Max Layer here.
\[
[f_{\theta}^{l}] = \max_t[f_{\theta}^{l-1}]_{i,t} \quad 1 \leq i \leq n_{hu}^{l-1}
\]
Everything is for Tagging

• The network output layers compute scores for all the possible tags for the task of interest.
• In the window approach, theses tags apply to the word located in the center of the window.
• In the sentence approach, these tags apply to the word designated by additional markers in the network input.
• Use IOBES tagging scheme for all tasks, in order to eliminate additional source of variations that different task using different tagging schemes.
\[ \theta \rightarrow \sum_{(x,y) \in \tau} \log p(y|x, \theta) \]
• The log-likelihood can be expressed as
\[ \log p(y|x, \theta) = [f_\theta]_y - \log \sum_i e^{[f_\theta]_i} \]

• The score can be interpreted as a conditional tag probability by applying a softmax operation
\[ p(i|x, \theta) = \frac{e^{[f_\theta]_i}}{\sum_j e^{[f_\theta]_j}} \]
• Finding the best path of tags during training.

• Introducing a transition score $[A]_{i,j}$ for jumping from $i$ to $j$ tags in successive words.

• The new training parameter is

$$\tilde{\theta} = \theta \cup \{ [A]_{i,j} \ orall i, j \}$$

• The score of a sentence along a path of tags is given as

$$s([x]_1^T, [i]_1^T, \tilde{\theta}) = \sum_{t=1}^T ([A][i]_{t-1}, [i]_t + [f_\theta][i]_{t,t})$$

• The probability of true path

$$\log p([y]_1^T | [x]_1^T, \tilde{\theta}) = s([x]_1^T, [y]_1^T, \tilde{\theta}) - \log \text{add} s([x]_1^T, [j]_1^T, \tilde{\theta}) \ orall [j]_1^T$$

• After mathematical simplification, we find

$$\arg\max s([x]_1^T, [j]_1^T, \tilde{\theta})$$

Training
**Stochastic Gradient**

Training

\[ \theta \leftarrow \theta + \lambda \frac{\partial \log p(y|x, \theta)}{\partial \theta} \]
Benchmark Results

• **Tasks:**
  1. POS
  2. CHUNK
  3. NER
  4. SRL

• **Methods:**
  1. Benchmark Systems
  2. Neural Network + Word-Level Log-Likelihood
  3. Neural Network + Sentence-Level Log-Likelihood

• **Tricks:**
  1. All networks were fed with two raw text features: low case words and capital letter feature.
  2. Number was replaced as NUMBER, words outside the dictionary is replaced as RARE.
## Stochastic Gradient Benchmark Results

<table>
<thead>
<tr>
<th>Approach</th>
<th>POS (PWA)</th>
<th>Chunking (F1)</th>
<th>NER (F1)</th>
<th>SRL (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Systems</td>
<td>97.24</td>
<td>94.29</td>
<td>89.31</td>
<td>77.92</td>
</tr>
<tr>
<td>NN+WLL</td>
<td>96.31</td>
<td>89.13</td>
<td>79.53</td>
<td>55.40</td>
</tr>
<tr>
<td>NN+SLL</td>
<td>96.37</td>
<td>90.33</td>
<td>81.47</td>
<td>70.99</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task</th>
<th>Window/Conv. size</th>
<th>Word dim.</th>
<th>Caps dim.</th>
<th>Hidden units</th>
<th>Learning rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>POS</td>
<td>$d_{win} = 5$</td>
<td>$d^0 = 50$</td>
<td>$d^1 = 5$</td>
<td>$n_{hu}^1 = 300$</td>
<td>$\lambda = 0.01$</td>
</tr>
<tr>
<td>CHUNK</td>
<td>&quot;</td>
<td>&quot;</td>
<td>&quot;</td>
<td>&quot;</td>
<td>&quot;</td>
</tr>
<tr>
<td>NER</td>
<td>&quot;</td>
<td>&quot;</td>
<td>&quot;</td>
<td>&quot;</td>
<td>&quot;</td>
</tr>
<tr>
<td>SRL</td>
<td>&quot;</td>
<td>&quot;</td>
<td>&quot;</td>
<td>$n_{hu}^1 = 300$</td>
<td>&quot;</td>
</tr>
</tbody>
</table>

Training
Benchmark Results

(a) POS

(b) CHUNK

(c) NER

(d) SRL

Training
Not yet Improvement

Using unlabeled data
Improvements

Pairwise Criterion

$$\theta \rightarrow \sum_{x \in X} \sum_{w \in D} \max\{0, 1 - f_\theta(x) + f_\theta(x^{(w)})\}$$
1. Initially, choose $k$ parameters choices from the set of all possible parameters.
2. Select the best ones using the validation set error rate.
3. In next iteration, we choose another set of $k$ parameters from the possible grid of values that permute slightly the most successful candidates from previous round.
4. Repeat 2 and 3.

**Benefits:** Many of parameter choice can share weights.
Improvements

**Language Model LM1**

Language model LM1 has a window size $d_{win} = 11$ and a hidden layer with $n^1_{hu} = 100$ units. The embedding layers were dimensioned as 50. Model LM1 was trained on our first English corpus (Wikipedia) using successive dictionaries composed of the 5000, 10,000, 30,000, 50,000 and finally 100,000 most common WSJ words.

**Language Model LM2**

Based on the word embeddings obtained by LM1, trained on the Wikipedia+Reuters corpus for addition.
**Improvements**

Old

Neither syntactic nor semantic relationship

New

The syntactic and semantic are clearly related
## Improvements

<table>
<thead>
<tr>
<th>Approach</th>
<th>POS (PWA)</th>
<th>CHUNK (F1)</th>
<th>NER (F1)</th>
<th>SRL (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Systems</td>
<td>97.24</td>
<td>94.29</td>
<td>89.31</td>
<td>77.92</td>
</tr>
<tr>
<td>NN+WLL</td>
<td>96.31</td>
<td>89.13</td>
<td>79.53</td>
<td>55.40</td>
</tr>
<tr>
<td>NN+SLL</td>
<td>96.37</td>
<td>90.33</td>
<td>81.47</td>
<td>70.99</td>
</tr>
<tr>
<td>NN+WLL+LM1</td>
<td>97.05</td>
<td>91.91</td>
<td>85.68</td>
<td>58.18</td>
</tr>
<tr>
<td>NN+SLL+LM1</td>
<td>97.10</td>
<td>93.65</td>
<td>87.58</td>
<td>73.84</td>
</tr>
<tr>
<td>NN+WLL+LM2</td>
<td>97.14</td>
<td>92.04</td>
<td>86.96</td>
<td>58.34</td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.20</td>
<td>93.63</td>
<td>88.67</td>
<td>74.15</td>
</tr>
</tbody>
</table>
Not yet

Improvement

Using Multi-Task Learning
Improvements

Joint Decoding

1. Considering additional probabilistic dependency paths between the models.
2. Therefore, it defines an implicit supermodel that describes all the tasks in the same probabilistic framework.
3. However, separately training cannot make dependency paths directly involve unobserved variables.

Joint Training

1. Good find relation for the case that training sets for the individual tasks contain the same patterns with different labels.
2. Sufficient to train a model that computes multiple outputs form each pattern.
Improvements
## Improvements

<table>
<thead>
<tr>
<th>Approach</th>
<th>POS (PWA)</th>
<th>CHUNK (F1)</th>
<th>NER (F1)</th>
<th>SRL (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Systems</td>
<td>97.24</td>
<td>94.29</td>
<td>89.31</td>
<td>–</td>
</tr>
<tr>
<td>Window Approach</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.20</td>
<td>93.63</td>
<td>88.67</td>
<td>–</td>
</tr>
<tr>
<td>NN+SLL+LM2+MTL</td>
<td>97.22</td>
<td>94.10</td>
<td>88.62</td>
<td>–</td>
</tr>
<tr>
<td>Sentence Approach</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.12</td>
<td>93.37</td>
<td>88.78</td>
<td>74.15</td>
</tr>
<tr>
<td>NN+SLL+LM2+MTL</td>
<td>97.22</td>
<td>93.75</td>
<td>88.27</td>
<td>74.29</td>
</tr>
</tbody>
</table>
Not yet

Improvement

Using Tricks
Improvements

1. **Suffix Features:** Strong predictors of the syntactic function in western languages.
2. **Gazetteers:** Large (8,000) category dictionary of name entity.
3. **Cascading:** Tags obtained for one task might useful for taking decisions in others.
4. **Ensembles:** Use multiple learning algorithms to obtain better performance.
5. **Parsing:** Provide parse tree information as additional input features to the system.
6. **Word Representations:** Induced word embedding on large amount of unlabeled text data.
## Improvements

### Final Results

<table>
<thead>
<tr>
<th>Task</th>
<th>Benchmark</th>
<th>SENNA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part of Speech (POS) (Accuracy)</td>
<td>97.24 %</td>
<td>97.29 %</td>
</tr>
<tr>
<td>Chunking (CHUNK) (F1)</td>
<td>94.29 %</td>
<td>94.32 %</td>
</tr>
<tr>
<td>Named Entity Recognition (NER) (F1)</td>
<td>89.31 %</td>
<td>89.59 %</td>
</tr>
<tr>
<td>Parse Tree level 0 (PT0) (F1)</td>
<td>91.94 %</td>
<td>92.25 %</td>
</tr>
<tr>
<td>Semantic Role Labeling (SRL) (F1)</td>
<td>77.92 %</td>
<td>75.49 %</td>
</tr>
</tbody>
</table>
## Improvements

### Resources Usage

<table>
<thead>
<tr>
<th></th>
<th>RAM (MB)</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>POS System</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Toutanova et al. (2003)</td>
<td>800</td>
<td>64</td>
</tr>
<tr>
<td>Shen et al. (2007)</td>
<td>2200</td>
<td>833</td>
</tr>
<tr>
<td>Senna</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td><strong>SRL System</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Koomen et al. (2005)</td>
<td>3400</td>
<td>6253</td>
</tr>
<tr>
<td>Senna</td>
<td>124</td>
<td>51</td>
</tr>
</tbody>
</table>
Conclusion
Q & A

😊 Thanks for your attention 😊