Combining Deep Learning with Conditional Random Fields

“Segmental Recurrent Neural Networks,” Kong, Dyer, and Smith, ICLR 2016

Kristen Masada
Example: Named Entity Recognition (NER)

“John Lennon met Paul McCartney in 1957 and invited McCartney to join his music group.”
Example: Named Entity Recognition (NER)

“John Lennon met Paul McCartney in 1957 and invited McCartney to join his music group.”
Example: Named Entity Recognition (NER)

“John Lennon met Paul McCartney in 1957 and invited McCartney to join his music group.”
Example: Chord Recognition
Example: Chord Recognition
Example: Chord Recognition

Eb:maj

D:maj

mf
Two Approaches to Classification

1. Token-level classification.
   a. **Token** - A single unit (e.g. individual word, music event).

2. Segment-level classification.
   a. **Segment** - Consecutive group of tokens with same label (e.g. 'John Lennon').
   b. *Joint segmentation and labeling.
Two Approaches to Token-Level Classification

1. Classify tokens **individually** using traditional ML algorithms.
   a. e.g. Logistic regression - classify events based on whether or not they belong to a single class.

2. Classify tokens **jointly**.
   a. *Not as simple as using softmax.*
      i. Labels for each event are dependent on one another.
Two Probabilistic Models to Classify Tokens Jointly

   a. Directly model $P(X, Y)$.
   b. $X$ is the list of tokens \{\(x_1, x_2, \ldots, x_N\}\).
   c. $Y$ is the list of labels \{\(y_1, y_2, \ldots, y_N\)\} corresponding to each token.
   d. E.g. **Hidden Markov Models (HMMs)**.
2. Discriminative Probabilistic Models.
   a. Directly model the posterior label probability $P(Y|X)$.
   b. E.g. **Linear Conditional Random Fields (CRFs).**
Hidden Markov Models (HMMs)

Assumptions:
1. An event $x_i$ depends only on its label $y_i$.
   a. $p(x_1, ..., x_N | y_1, ..., y_N) = \prod_{i=1}^{N} p(x_i | y_i)$
2. A label $y_i$ depends only on the previous label $y_{i-1}$.
   a. $p(y_1, ..., y_N) = \prod_{i=1}^{N} p(y_i | y_{i-1})$

Then uses Bayes’s Theorem to compute $P(Y|X)$:

$$\hat{y}_1 \ldots \hat{y}_N = \arg\max_{y_1 \ldots y_N} \prod_{i=1}^{N} p(x_i | y_i)p(y_i | y_{i-1})$$
Linear Conditional Random Fields (CRFs)

Computes global feature vector:

- \( F(Y, X) = \sum_{i=1}^{N} f(y_i, y_{i-1}, X) \)

Also computes a normalization constant \( Z(X) \):

- \( Z(X) = \sum_{Y} e^{w^T F(Y, X)} \)

Then computes \( P(Y|X) \):

\[
P(Y|X) = \frac{e^{w^T F(Y, X)}}{Z(X)}
\]
Semi-Markov Conditional Random Fields (CRFs)

• *semi-Markov CRFs = semi-CRFs = segmental CRFs.
• \( S \) is the list of segmentations for input \( X, \{s_1, s_2, ..., s_N\} \).
• \( s_i.f \) is the index of the first token of segment \( s_i \); \( s_i.l \) is index of last token of \( s_i \).
• \( Y \) becomes list of labels for each corresponding segment in \( S \).
• Again computes global feature vector, but now with segment \( s_i \): 
  \[
  F(S,Y,X) = \sum_{i=1}^{N} f(s_i,y_i,y_{i-1},X).
  \]


Computes normalization constant $Z(X)$ again:

- $Z(X) = \sum_{S,Y} e^{w^T F(S,Y,X)}$

Then computes $P(Y|X)$:

$$P(Y|X) = \frac{e^{w^T F(S,Y,X)}}{Z(X)}$$
Segmental RNNs (SRNNs)

- Combine semi-CRFs with RNNs.
- Computes local feature vector as follows:
  \[ f(y_i, s_i, X) = \phi(V[g_y(y_i); g_s(s_i); \overrightarrow{\text{RNN}}(c_{s_i,f:s_i,e}); \overrightarrow{\text{RNN}}(c_{s_i,e:s_i,f})] + a) \]
- Semicolons (e.g. \([a; b; c]\)) indicate vector concatenation.
- \(g_y\) and \(g_s\) are functions that map \(y_i\) and \(s_i\) to a vector representation (e.g. one-hot encoding).
- \(\overrightarrow{\text{RNN}}(c_{s_i,f:s_i,e})\) is an RNN that computes the forward segment embedding starting at \(s_i.f\); \(\overleftarrow{\text{RNN}}(c_{s_i,e:s_i,f})\) computes the backward segment embedding.
- \(V\) is a matrix that transforms the input to a vector of the size of the number of rows in \(V\).
- \(a\) is a bias term.
- \(\phi\) is a nonlinear activation function (e.g. tanh).
SRNNs (Cont.)

- Computes global feature vector like semi-CRF:
  \[ F(S, Y, X) = \sum_{i=1}^{N} f(s_i, y_i, X) \]
- Also computes normalization constant \( Z(X) \):
  \[ Z(X) = \sum_{S,Y} e^{w^T F(S,Y,X)} \]
- Then computes \( P(Y|X) \):
  \[ P(Y|X) = \frac{e^{w^T F(S,Y,X)}}{Z(X)} \]

*Only difference between regular semi-CRF and SRNNs is that \( F(S, Y, X) \) is manually specified for semi-CRF, but learned by SRNNs.
SRNNs (Cont.): Architecture

Bi-directional LSTM

Encoder

*Outputted segments have durations 3, 2, and 1.
SRNNs (Cont.): Architecture

\[
\begin{align*}
\mathbf{s}_i, \mathbf{f} & \rightarrow \mathbf{h}_{rl} \\
\mathbf{h}_{rl} & \rightarrow \mathbf{g}_y(\mathbf{y}_i), \mathbf{g}_s(\mathbf{s}_i), \mathbf{h}_{lr} \\
\mathbf{h}_{lr} & \rightarrow \mathbf{s}_i, \mathbf{l}
\end{align*}
\]

(Bi-directional LSTM)
SRNNs (Cont.): Computing Embeddings

- Uses dynamic programming algorithm Viterbi to compute segment embeddings in $O(|x|^2)$.
- 4 recursive update rules:
  \[
  \overrightarrow{h}_{i,i} = \overrightarrow{\text{RNN}}(\overrightarrow{h}_0, c_i)
  \\
  \overrightarrow{h}_{i,j} = \overrightarrow{\text{RNN}}(\overrightarrow{h}_{i,j-1}, c_j)
  \\
  \overleftarrow{h}_{i,i} = \overleftarrow{\text{RNN}}(\overleftarrow{h}_0, c_i)
  \\
  \overleftarrow{h}_{i,j} = \overleftarrow{\text{RNN}}(\overleftarrow{h}_{i+1,j}, c_i)
  \]
SRNNs (Cont.): Learning

- Uses the negative log likelihood of $P(Y,Z|X)$ in training.

$$\mathcal{L} = \sum_{(X,Y,Z) \in \mathcal{D}} -\log P(Y,Z|X)$$
SRNNs (Cont.): Evaluation

- Evaluated on handwriting recognition, as well as Chinese word segmentation and POS tagging tasks.
- Compared against implementations that do not incorporate segmentation.

<table>
<thead>
<tr>
<th></th>
<th>Dev</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$P_{seg}$</td>
<td>$R_{seg}$</td>
</tr>
<tr>
<td>SRNNs (Partial)</td>
<td>98.7%</td>
<td>98.4%</td>
</tr>
<tr>
<td>SRNNs (Full)</td>
<td>98.9%</td>
<td>98.6%</td>
</tr>
<tr>
<td>CTC</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: Hand-writing Recognition Task

<table>
<thead>
<tr>
<th></th>
<th>Dev</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$P_{tag}$</td>
<td>$R_{tag}$</td>
</tr>
<tr>
<td>BiRNNs</td>
<td>93.2%</td>
<td>92.9%</td>
</tr>
<tr>
<td>SRNNs</td>
<td>93.8%</td>
<td>93.8%</td>
</tr>
</tbody>
</table>

Table 3: Joint Chinese Word Segmentation and POS Tagging
Questions