Semi-Supervised Structured Prediction with Neural CRF Autoencoder

Xiao Zhang¹, Yong Jiang², Hao Peng¹, Kewei Tu² and Dan Goldwaswer¹
Purdue University¹, ShanghaiTech University²

ABSTRACT

We propose an end-to-end neural CRF autoencoder (NCRF-AE) model for semi-supervised learning of sequential structured prediction problems. Our NCRF-AE consists of two parts: an encoder which is a CRF model enhanced by deep neural networks, and a decoder which is a generative model trying to reconstruct the input. Our model has a unified structure with different loss functions for labeled and unlabeled data with shared parameters. We developed a variation of the EM algorithm for optimizing both the encoder and the decoder simultaneously by decoupling their parameters. Our experimental results over the Part-of-Speech (POS) tagging task on eight different languages show that the NCRF-AE model can outperform competitive systems in both supervised and semi-supervised scenarios.

LEARNING

Unified learning framework Loss functions for labeled and unlabeled data with shared parameters:

**Labeled data:** \( \text{loss}_l = - \log \text{P}_{\text{AE}}(\hat{x}, y|x); \) **Unlabeled data:** \( \text{loss}_u = - \log \text{P}_{\text{AE}}(\hat{x}|x). \)

Parameter learning using EM

1. **E-step:**
   \[
   E: \sum_t \log \text{P}(\hat{x}|x) \geq \sum_t \sum_y \text{Q}(y_t|x) \log \text{P}(\hat{x}|y_t|x),
   \]
   Decoding using Viterbi
   \[
   \hat{y}_t = \arg \max_y \text{P}_{\text{AE}}(\hat{x}, y|x).
   \]

Algorithm 1 Obtain Expected Count (\( T_t \))

1. **Require:** the expected count table \( T_t \)
2. **for** an unlabeled data example \( x, y \) **do**
3. **Compute** the forward messages:
   \[
   \alpha(y_t, t) \equiv \text{y}_t, t \uparrow \text{x} \text{is the position in a sequence}.
   \]
4. **Compute** the backward messages:
   \[
   \beta(y_t, t) \equiv \text{y}_t, t \downarrow \text{x}.
   \]
5. **Calculate** the expected count for each \( x \):
   \[
   T_t(x, y_t) 
   \]
6. **end for**

Algorithm 2 Mixed Expectation-Maximization

1. **Initialize** expected count table \( T_t \) using labeled data \( x, y \), and use it as \( \Theta^{(0)} \) in the decoder.
2. **Initialize** \( \Lambda^{(0)} \) in the encoder randomly.
3. **for** \( t \) **in** epochs **do**
4. **Train** the encoder on labeled data \( x, y \) and unlabeled data \( \{ \text{x} \} \) to update \( \Lambda^{(t-1)} \) to \( \Lambda^{(t)} \).
5. **Re-initialize** expected count table \( T_t \), with \( x \).
6. **Use** labeled data \( x, y \) to calculate real counts and update \( T_t \).
7. **Use** unlabeled data \( x \) to compute the expected counts with parameters \( \Lambda^{(t)} \) and \( \Theta^{(t-1)} \) and update \( T_t \).
8. **Obtain** \( \Theta^{(t)} \) globally and analytically based on \( T_t \).
9. **end for**

RESULTS

<table>
<thead>
<tr>
<th>Models (supervised)</th>
<th>English</th>
<th>French</th>
<th>German</th>
<th>Italian</th>
<th>Russian</th>
<th>Spanish</th>
<th>Indonesian</th>
<th>Croatian</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM</td>
<td>86.28%</td>
<td>91.23%</td>
<td>85.59%</td>
<td>92.03%</td>
<td>79.82%</td>
<td>91.31%</td>
<td>89.40%</td>
<td>96.89%</td>
</tr>
<tr>
<td>CRF</td>
<td>89.96%</td>
<td>93.40%</td>
<td>86.83%</td>
<td>94.07%</td>
<td>83.38%</td>
<td>91.47%</td>
<td>88.63%</td>
<td>86.90%</td>
</tr>
<tr>
<td>LSTM</td>
<td>90.50%</td>
<td>94.16%</td>
<td>88.40%</td>
<td>94.96%</td>
<td>84.87%</td>
<td>93.17%</td>
<td>89.42%</td>
<td>88.95%</td>
</tr>
<tr>
<td>NCRF</td>
<td>91.52%</td>
<td>95.07%</td>
<td>90.27%</td>
<td>96.20%</td>
<td>93.39%</td>
<td>93.34%</td>
<td>92.32%</td>
<td>93.85%</td>
</tr>
<tr>
<td>NCRF-AE</td>
<td>92.50%</td>
<td>95.28%</td>
<td>90.50%</td>
<td>96.64%</td>
<td>93.66%</td>
<td>93.86%</td>
<td>93.96%</td>
<td>94.32%</td>
</tr>
<tr>
<td>Models (semi-supervised)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NCRF-AE (Only Labeled)</td>
<td>88.41%</td>
<td>93.69%</td>
<td>90.75%</td>
<td>92.17%</td>
<td>87.82%</td>
<td>91.70%</td>
<td>89.06%</td>
<td>87.92%</td>
</tr>
<tr>
<td>NCRF-AE (All Labeled)</td>
<td>79.92%</td>
<td>88.15%</td>
<td>77.01%</td>
<td>84.57%</td>
<td>72.96%</td>
<td>86.77%</td>
<td>83.61%</td>
<td>77.20%</td>
</tr>
<tr>
<td>NCRF-AE (Scaled Labeled)</td>
<td>86.79%</td>
<td>92.85%</td>
<td>88.78%</td>
<td>90.68%</td>
<td>86.39%</td>
<td>91.30%</td>
<td>88.86%</td>
<td>86.55%</td>
</tr>
<tr>
<td>NCRF-AE</td>
<td>89.43%</td>
<td>93.89%</td>
<td>90.99%</td>
<td>92.85%</td>
<td>88.93%</td>
<td>92.17%</td>
<td>89.41%</td>
<td>89.14%</td>
</tr>
</tbody>
</table>

EXPERIMENTS

We evaluated our model on the POS tagging task, in both the supervised and semi-supervised learning settings, over 8 different languages from the UD (Universal Dependencies) 1.4 dataset.

Error analysis: an example form the test set

Semi-supervised learning effect

UD POS tagging accuracy versus increasing proportion of unlabeled sequences using 20% labeled data. The green straight line is the performance of the neural CRF trained over the labeled data.

Varying sizes of labeled data on English

We gradually increased the proportion of labeled data, and in accordance decreased the proportion of unlabeled.

FUTURE WORK

- Use embeddings for POS tags to compute both the transition score and the generative decoder score.
- Add a prior for predicted labels and cast it into the variational inference framework.