CS639 Deep Learning for NLP

Sequence Labeling II

Junjie Hu

Slides adapted from Luke, Yulia, Bob

https://junjiehu.github.io/cs639-spring26/

Outline

- Sequence Labeling
- (**Generative**) **Hidden Markov Model**
- (**Discriminative**) **Conditional Random Field**
- Comparison

Recap: Sequence labeling problems

- Map **a sequence of words** to **a sequence of labels**
  - Part-of-speech tagging (Church, 1988; Brants, 2000)
  - Named entity recognition (Bikel et al., 1990)
  - Text chunking and shallow parsing (Ramshaw and Marcus, 1995)
  - Word alignment of parallel text (Vogel et al., 1996)
  - Compression (Conroy and O'Leary, 2001)
  - Acoustic models, discourse segmentation, etc.

Conditional Random Fields
(Sequential Version of Logistic Regression)

Recap: Logistic Regression (Log Linear Models)

- **Text classification:** $X = \{x_1 \cdots, x_n\}, y \in \{1 \cdots C\}$

$$P(y = c|X) = \frac{\exp(w_c^T f(X) + b_c)}{\sum_k \exp(w_k^T f(X) + b_k)}, \qquad w_c, f(X) \in \mathbb{R}^d$$

$F(X, y = c)$ Scoring function

$Z(X)$ Normalization constant or partition function

- **"Log-linear" assumption:**
  - The features of the input is "log-linear" to the output

$$\log P(y = c|X) = F(y = c, X) - \log Z(X)$$

  - Very flexible to include hand-crafted features (or learned features by neural networks)

Linear chain Conditional Random Fields
("Log-Linear" 1st order Sequential Model)

- **Sequence labeling** $X = \{x_1 \cdots x_n\}, Y = \{y_1 \cdots y_n, \text{STOP}\}$:

$$P(Y|X) = \frac{1}{Z(X)} \exp\left(\sum_{j=1}^{d_1}\sum_{i=2}^{n+1} \lambda_j q_j(y_{i-1}, y_i, X) + \sum_{j=1}^{d_2}\sum_{i=1}^{n} \mu_j g_j(y_i, X)\right)$$

$$Z(X) = \sum_{Y} \exp(F(Y, X))$$

d features scoring transitions

d features scoring each state w/ input sequence

$$F(Y, X) = w^T f(Y, X) = \sum_{j=1}^{d_1+d_2} w_j f_j(Y, X), \quad w, f(Y, X) \in \mathbb{R}^{d_1+d_2}$$

$$f_j(Y, X) = \begin{cases} \sum_{i=2}^{n} q_j(y_{i-1}, y_i, X), & j \in [1, d_1] \\ \sum_{i=1}^{n} g_{j'}(y_i, X), & j = j' + d_1, j' \in [1, d_2], \end{cases}$$

$$w = [\lambda_1, \cdots, \lambda_{d_1}, \mu_1, \cdots, \mu_{d_2}]$$

Linear chain Conditional Random Fields
("Log-Linear" 1st order Sequential Model)

- **Sequence labeling** $X = \{x_1 \cdots x_n\}, Y = \{y_1 \cdots y_n, \text{STOP}\}$:

$$P(Y|X) = \frac{1}{Z(X)} \exp\left(\sum_{i=2}^{n+1} \lambda \cdot q(y_{i-1}, y_i, X) + \sum_{i=1}^{n} \mu \cdot g(y_i, X)\right)$$

$$Z(X) = \sum_{Y} \exp(F(Y, X))$$

d_1 features scoring transitions

d_2 features scoring each state w/ input sequence

$$F(Y, X) = w \cdot f(Y, X) = \sum_{i=1}^{n} w \cdot f(y_i, y_{i+1}, X), \quad w, f(Y, X) \in \mathbb{R}^d$$

$$f(y_i, y_{i+1}, X) = [q(y_i, y_{i+1}, X);\ g(y_i, X)]$$

$$w = [\lambda;\ \mu], \lambda \in \mathbb{R}^{d_1}, \mu \in \mathbb{R}^{d_2}$$

CRF: Learning

- **Learning**: maximize the log-likelihood over the training data

$$\mathcal{L}(w) = \sum_{(X,Y)\sim\mathcal{D}_{\text{train}}} \log P(Y|X)$$

$$= \sum_{(X,Y)\sim\mathcal{D}_{\text{train}}} w^\top f(Y,X) - \log Z(X)$$

$$w^* = \arg\max_w \mathcal{L}(w)$$

Sum over all possible outputs *Y* for an input *X* — Brute force solution: score n^C outputs
Can we do faster?

- **Update:** stochastic gradient descent to move in a direction that decreases the loss

$$w \leftarrow w - \alpha \frac{\partial \mathcal{L}(w)}{\partial w}$$

Reflection of Gradient

- The gradient w.r.t. each feature weight

$$\frac{\partial \mathcal{L}(w)}{\partial w_j} = \sum_{(X,Y)\in\mathcal{D}_{\text{train}}} \sum_{i=1}^{n} f_j(y_i, y_{i+1}, X) - \sum_{(X)\in\mathcal{D}_{\text{train}}} \mathbb{E}_{y'_i, y'_{i+1}} f_j(y'_i, y'_{i+1}, X)$$

observed feature counts — expected feature counts

Dynamic Programing

- **Learning**: maximize the log-likelihood over the training data

$$\frac{\partial \log Z(X)}{\partial w_j} = \mathbb{E}_Y \left[\sum_{i=1}^{n} f_j(y'_i, y'_{i+1}, X) \right]$$

$$= \sum_{i=1}^{n} \mathbb{E}_{y'_i, y'_{i+1}} \left[P(y'_i, y'_{i+1}|X) f_j(y'_i, y'_{i+1}, X) \right]$$

$$= \sum_{i=1}^{n} \sum_{y'_i, y'_{i+1}} P(y'_i, y'_{i+1}|X) f_j(y'_i, y'_{i+1}, X)$$

$P(y'_i, y'_{i+1}|X)$ can be computed by dynamic programing (forward-backward algorithm) — sum production algorithm, basically replace the max operation in Viterbi algorithm by sum operation

Dynamic Programing

- **Learning**: maximize the log-likelihood over the training data

$$\frac{\partial \log Z(X)}{\partial w_j} = \mathbb{E}_Y \left[\sum_{i=1}^{n} f_j(y_{i-1}, y_i, X, i) \right]$$

$$= \sum_{i=1}^{n} \mathbb{E}_{y'_{i-1}, y'_i} [P(y'_{i-1}, y'_i | X) f_j(y'_{i-1}, y'_i, X, i)]$$

$$= \sum_{i=1}^{n} \sum_{y'_{i-1}, y'_i} P(y'_{i-1}, y'_i | X) f_j(y'_{i-1}, y'_i, X, i)$$

$P(y'_{i-1}, y' | X)$ can be computed by dynamic programing (forward-backward algorithm) — sum production algorithm, basically replace the max operation in Viterbi algorithm by sum operation

CRF Decoding: Viterbi

- Same as HMM decoding
- Viterbi (max-production algorithm): define the recursive function to compute the max value of the past partial sequence

$$
\begin{aligned}
Y^* &= \arg\max_Y \log P(Y|X) \\
&= \arg\max_Y w \cdot f(Y, X) - \log Z(X) \\
&= \arg\max_Y \sum_{i=1}^{n} w \cdot f(y_i, y_{i+1}, X)
\end{aligned}
$$

Decoding output doesn't depend on the second term

Feature functions

- Feature functions based on **possible combination of words and tags**, or other information such as POS tag (if given), whether the word is capitalized or not

$$q_1(y_{i-1}, y_i, X) = \begin{cases} 1 & \text{if } y_{i-1} = \text{OTHER and } y_i = \text{PERSON} \\ 0 & \text{otherwise} \end{cases}$$

$$g_2(y_i, X) = \begin{cases} 1 & \text{if } y_i = \text{PERSON and } x_i = \text{John} \\ 0 & \text{otherwise} \end{cases}$$

Feature values are not limited to just binary values, can be real-values too.
Number of features can be tens of thousands or more.

Feature Selection

1. Initially CRF model has no features (uniform prediction)

2. Create some candidate feature sets, e.g., (combination of any word-tag pairs, x=John, y_i= PERSON). There are VK possible pairs

3. Build a new CRF w/ a subset of features

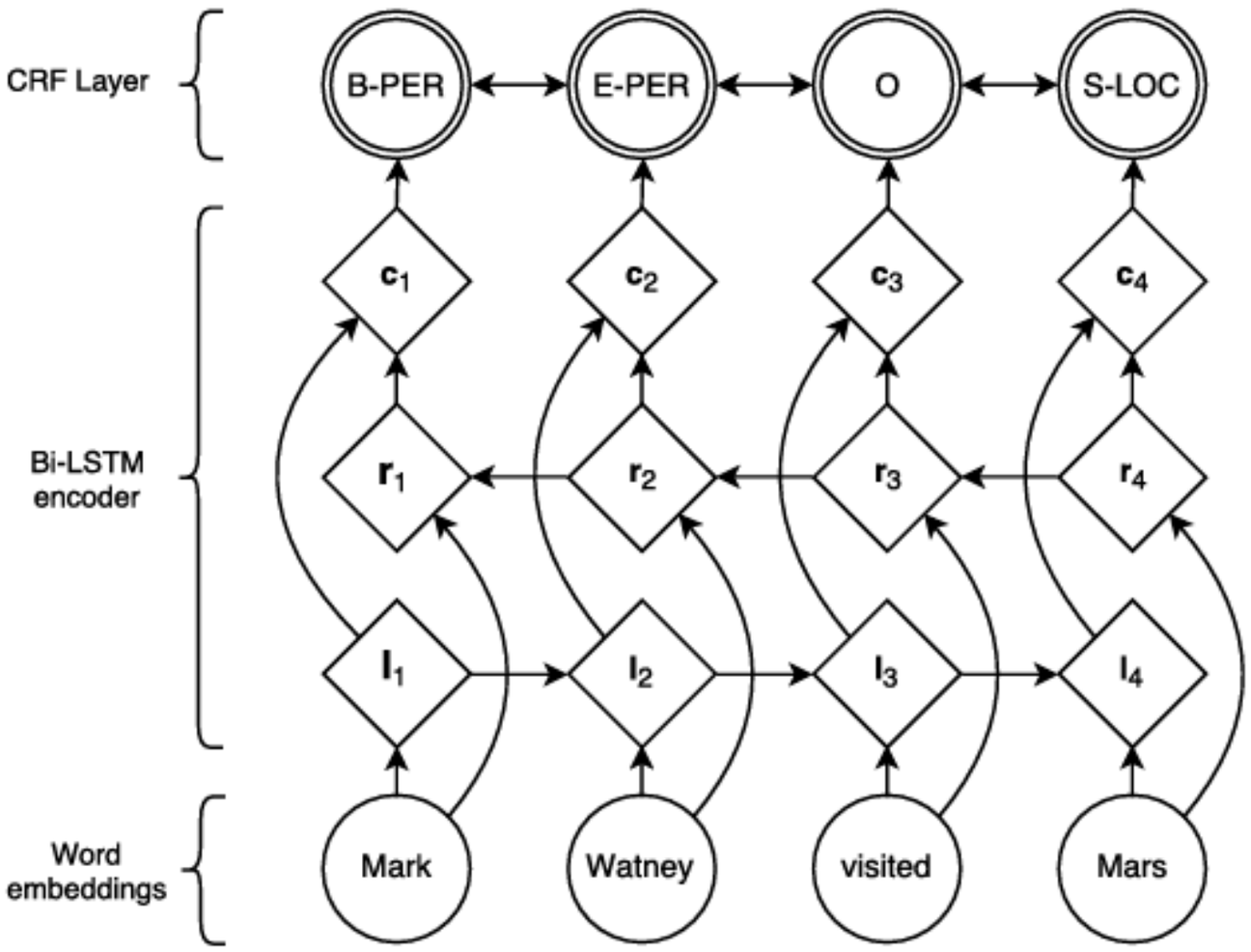
4. Include the selected features that improves over the previous CRF

5. Go to step 3 until enough features have been added to CRF

Neural Conditional Random Fields

Neural CRF

- Rather than hand-crafted features, let's use NN to learn features.



$$p(\mathbf{y}|\mathbf{X}) = \frac{e^{s(\mathbf{X},\mathbf{y})}}{\sum_{\tilde{\mathbf{y}}\in\mathbf{Y_X}} e^{s(\mathbf{X},\tilde{\mathbf{y}})}}$$

$$s(\mathbf{X},\mathbf{y}) = \sum_{i=0}^{n} A_{y_i,y_{i+1}} + \sum_{i=1}^{n} P_{i,y_i}$$

Lample et. al 2016 Neural Architectures for Named Entity Recognition

Learned Feature

- P_{i,y_i} : the output of the bi-LSTM model followed by a linear projection layer. $P \in \mathbb{R}^{n \times C}$

- $A \in \mathbb{R}^{C+2 \times C+2}$: is the transition matrix from one state (tag) to the other state, including the start/end states (so *C+2*).

$$s(\mathbf{X}, \mathbf{y}) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} P_{i, y_i}$$

Scoring the transition

Scoring the association
Of tag yi w/ the input X

Training: forward pass

- During training, we need to compute the log of the condition probability:

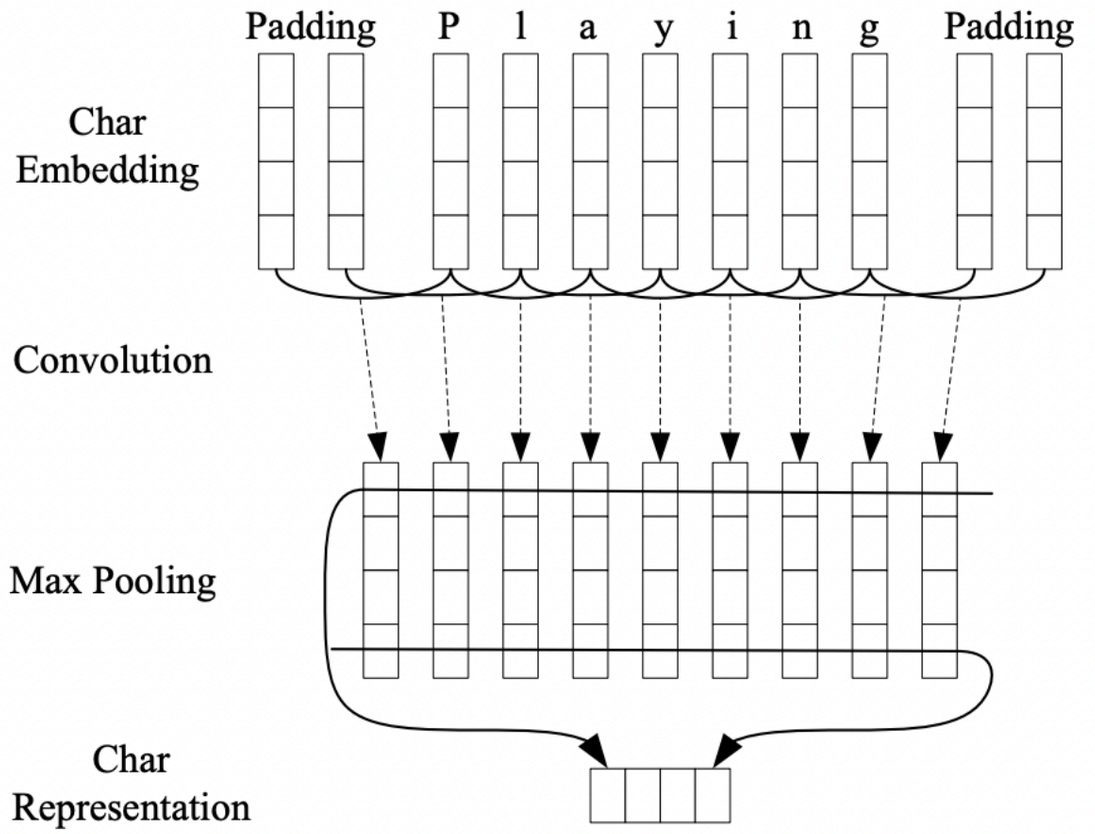
$$\log(p(\mathbf{y}|\mathbf{X})) = s(\mathbf{X},\mathbf{y}) - \log\left(\sum_{\widetilde{\mathbf{y}}\in\mathbf{Y_X}} e^{s(\mathbf{X},\widetilde{\mathbf{y}})}\right)$$
$$= s(\mathbf{X},\mathbf{y}) - \operatorname{logadd}_{\widetilde{\mathbf{y}}\in\mathbf{Y_X}} s(\mathbf{X},\widetilde{\mathbf{y}}), \quad (1)$$

- Why?
  - Avoid floating-point issues, more stable.
  - The second term can be solved by dynamic programming (sum-product)

- Use MLE as objective function, and NN-based back-propogation to update the gradient of each learning parameters (including Bi-LSTM, CRF layer)

BiLSTM-CNN CRF

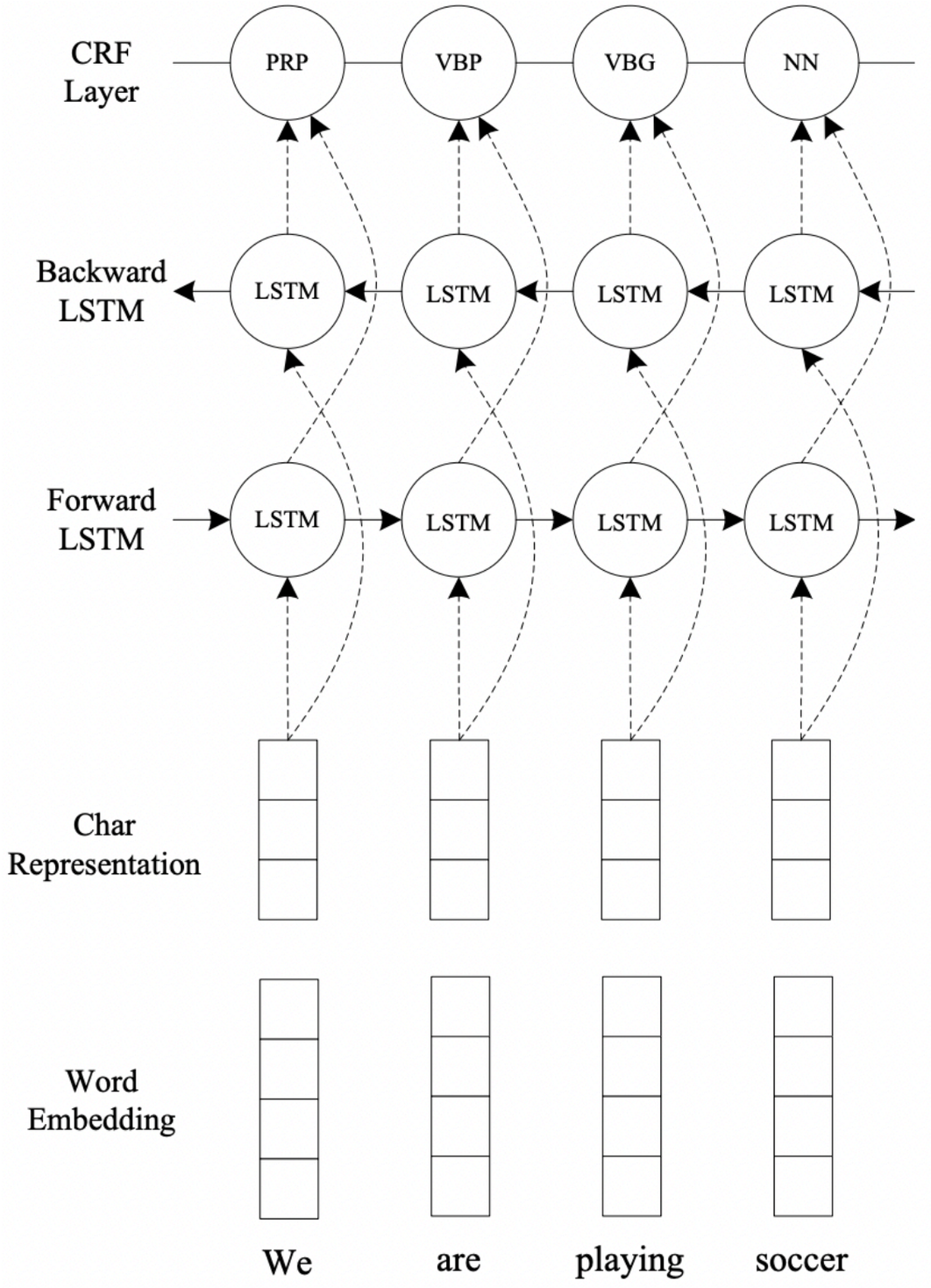
- Use CNN to encode character embeddings



Ma et al. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF

BiLSTM-CNN CRF

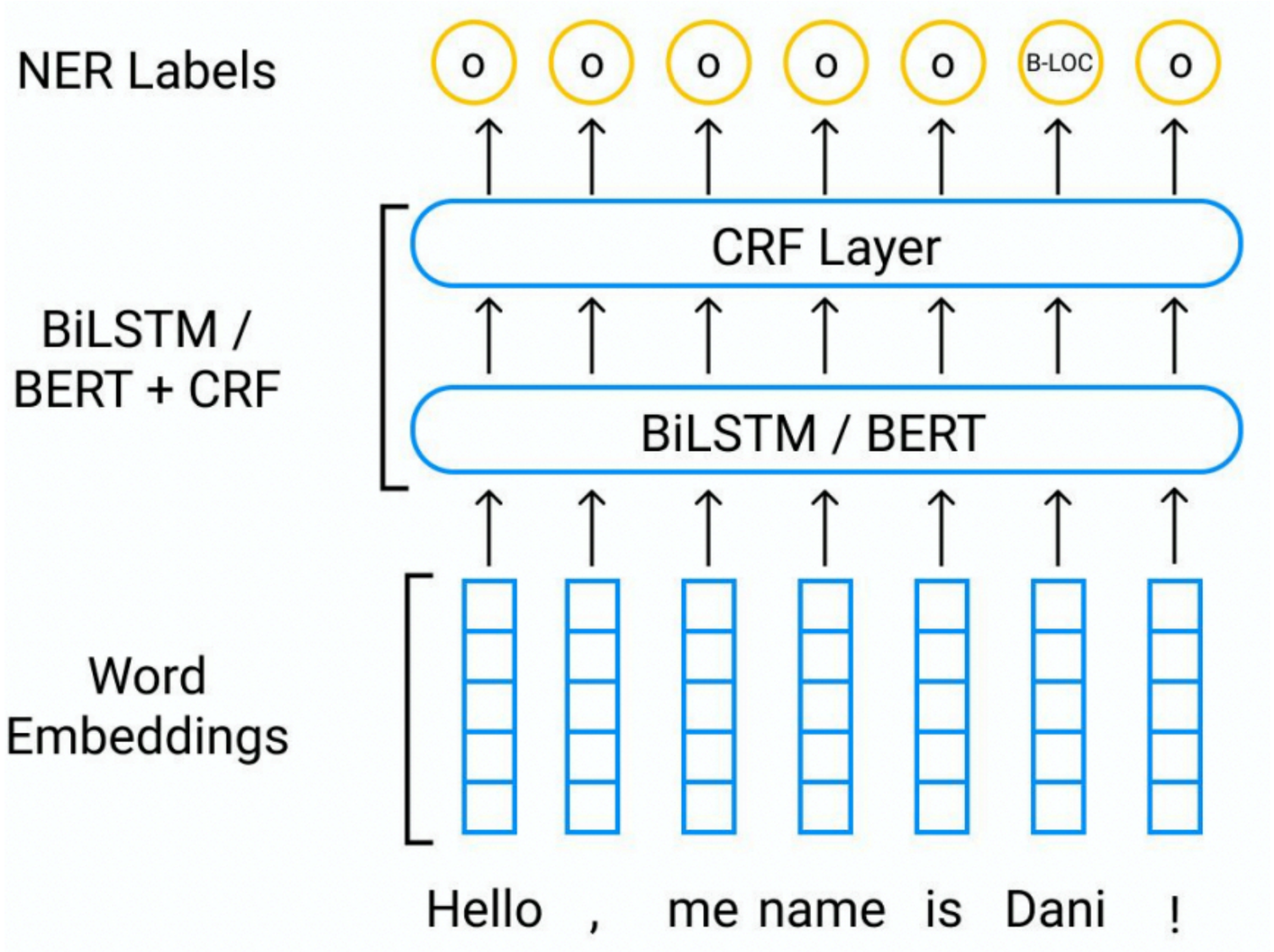
- Use CNN to encode character embeddings
- Combine char and word embeddings together
- Further encode by BiLSTM model to learn the sequence representations
- Add a CRF layer



Ma et al. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF

BERT-CRF

- Replace BiLSTM with a BERT encoder



Comparison: Naive Bayes -> HMM

Logistic Regression -> CRF

Recap: Naive Bayes & HMMs

- Naive Bayes (for text classification):

$$P(X, y) = P(X|y)P(y) = \left(\prod_{x_i} P(x_i|y) \right) P(y)$$

- Hidden Markov Models (for sequence labeling):

$$\begin{aligned} P(X, Y) &= q(\text{STOP}|y_n) \prod_{i=1}^{n} q(y_i|y_{i-1}) e(x_i|y_i) \\ &= \left(q(\text{STOP}|y_n) \prod_{i=1}^{n} q(y_i|y_{i-1} \right) \left(\prod_{i=1}^{n} e(x_i|y_i) \right) \\ &= P(Y) \left(\prod_{i=1}^{n} P(x_i|y_i) \right) \end{aligned}$$

HMMs ≈ sequence version of Naive Bayes!
Both are generative models.

Logistic Regression & CRF

- Logistic Regression (for text classification):

$$P(Y = c|X) \propto w_c \cdot F(X, Y = c)$$

- Conditional Random Field (for sequence labeling):

$$P(Y|X) = \prod_{i=1}^{T} P(Y_i|X_i, Y_{i-1}), \quad Y_0 = [\text{START}]$$

$$\begin{aligned} P(Y_i = c|X_i, Y_{i-1}) &\propto w_c \cdot f(Y_i = c, Y_{i-1}, X_i) \\ &= \lambda \cdot q(Y_i = c, Y_{i-1}, X_i) + \mu \cdot g(Y_i = c, X_i) \end{aligned}$$

CRF ≈ sequence version of Logistic Regression!
Both are discriminative models.

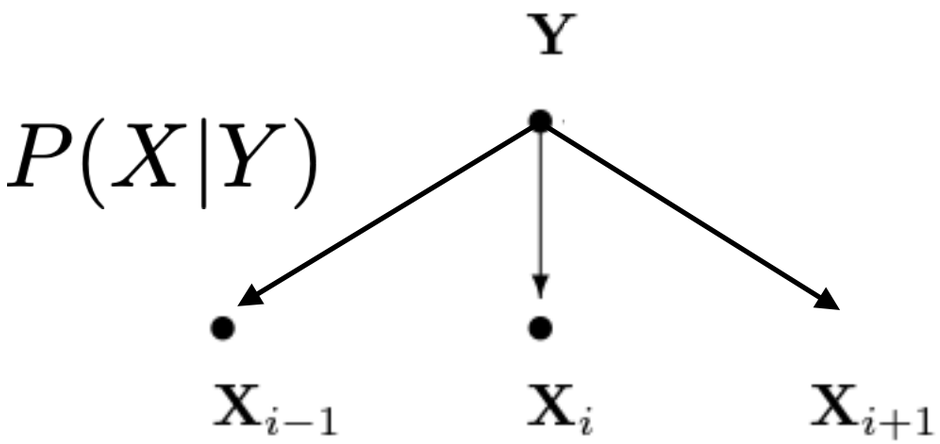
Generative v.s. Discriminative

- **Generative Models**:
  - Joint probability: $P(X, Y)$
  - Make prediction by $\arg\max_Y P(X, Y)$
  - Can generate new samples (X, Y)
    - Examples: **HMMs, Naive Bayes**
- **Discriminative Models**:
  - Conditional probability: $P(Y|X)$
  - Can directly predict $\arg\max_Y P(Y|X)$
  - Examples: **Conditional Random Fields, Logistic Regression**
- Both trained via Maximum Likelihood Estimation

Compare Naive Bayes and Logistic Regression

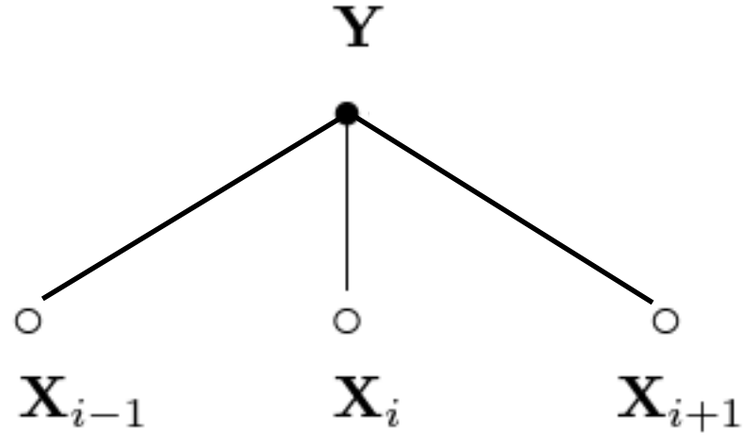
- Directed graphical model vs undirected graphical model

$$Y \sim P(Y)$$



Naive Bayes
(Generative)

$$P(Y = c|X) \propto w_c \cdot F(X, Y = c)$$



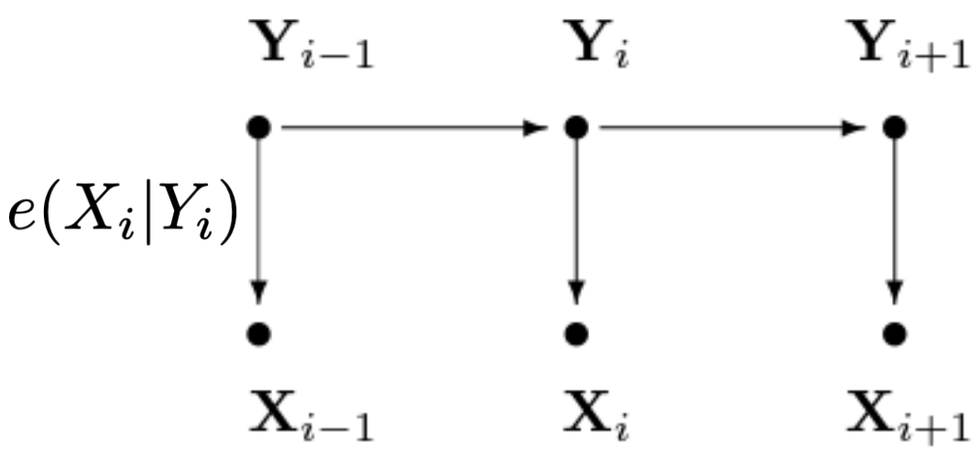
Logistic Regression
(Discriminative)

An open circle indicates that the variable is not generated by the model.

Compare HMM and linear chain CRF

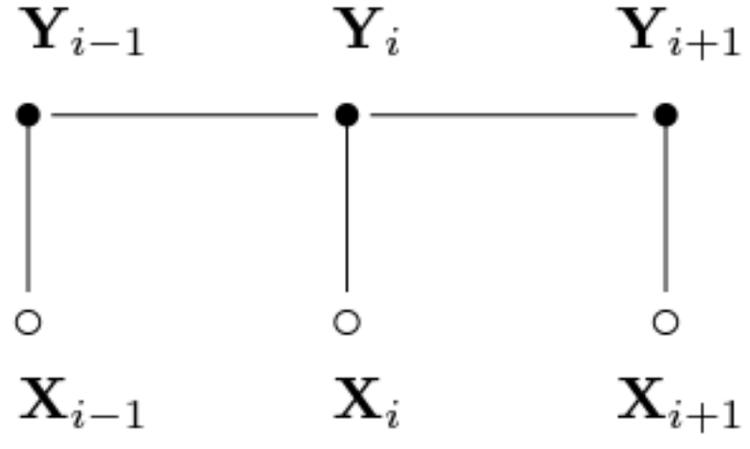
- Directed graphical model vs undirected graphical model

$$q(Y_i|Y_{i-1})$$



HMM
(Generative)

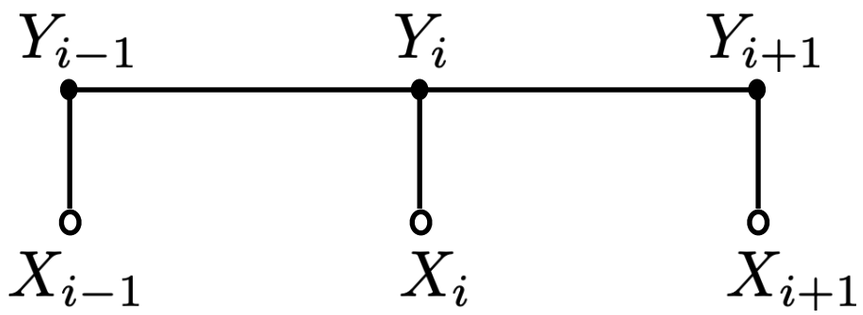
$$P(Y_i = c|X_i, Y_{i-1}) \propto w_c \cdot f(Y_i = c, Y_{i-1}, X_i)$$
$$= \lambda \cdot q(Y_i = c, Y_{i-1}, X_i) + \mu \cdot g(Y_i = c, X_i)$$



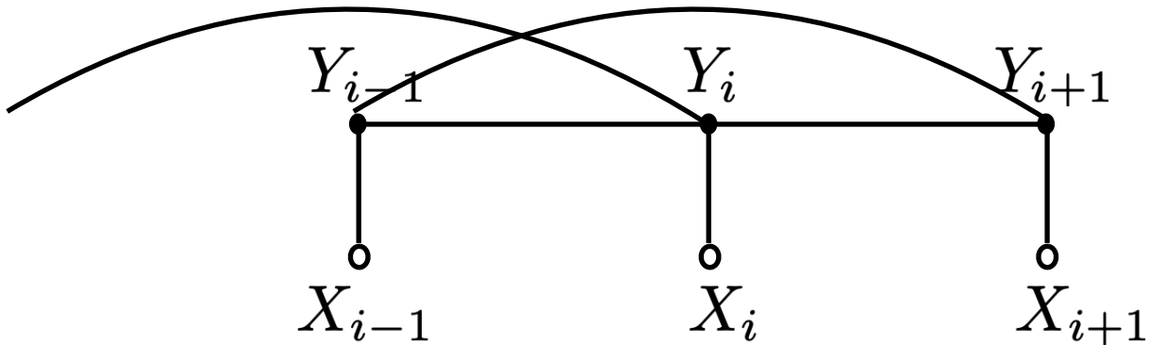
Chain-structure CRF
(Discriminative)

An open circle indicates that the variable is not generated by the model.

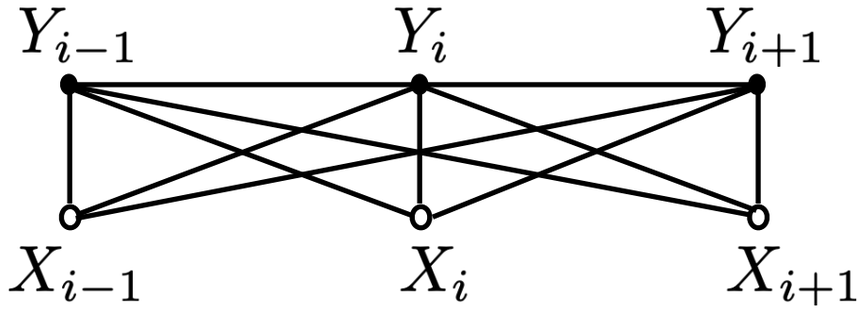
Variants of CRF Layers



- 1th order linear chain



- 2nd order linear chain



- Local vs. Global context

Questions?