CS769 Advanced NLP

Sequence Labeling II

Junjie Hu

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Slides adapted from Yulia

https://junjiehu.github.io/cs769-spring23/

Goals for Today

- Comparison of Generative vs. Discriminative Modeling
  - Text classification
  - Sequence labeling
- Conditional Random Field (CRF)
- Neural CRF
- Sequence labeling tasks

Recap: HMM

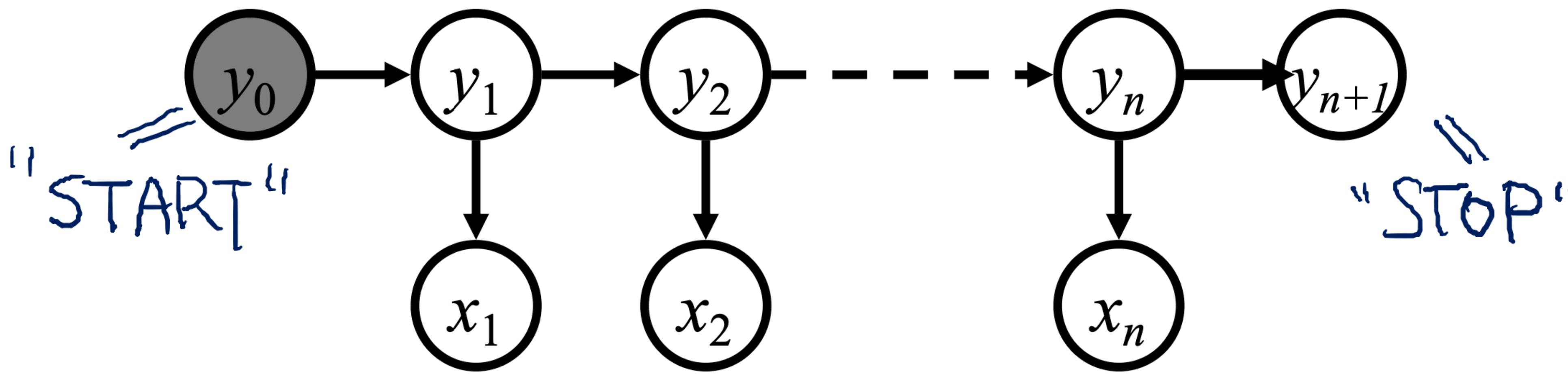
- **Generative model:** Learn a joint probability of $p(x_1 \cdots x_n, y_1 \cdots y_{n+1})$

- Use the 1st order Markov assumption

$$p(x_1 ... x_n, y_1 ... y_{n+1}) = q(\text{STOP}|y_n) \prod_{i=1}^{n} q(y_i|y_{i-1}) e(x_i|y_i)$$

- $e(x_i|y_i)$: Probability of state y_i generating x_i
- $q(y_{i+1}|y_i)$: Probability of state y_i transitioning to y_{i+1}
- $q(\text{STOP}|y_n)$: Probability of y_n being the last state

Graphical Model Representation of HMM



$$p(x_1 \ldots x_n, y_1 \ldots y_{n+1}) = q(\mathrm{STOP}|y_n) \prod_{i=1}^{n} q(y_i|y_{i-1}) e(x_i|y_i)$$

where $y_0 = \mathrm{START}$ and we call $q(y'|y)$ the transition distribution and $e(x|y)$ the emission (or observation) distribution.

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):

$$P(X,Y) = q(\mathrm{STOP}|y_n) \prod_{i=1}^{n} q(y_i|y_{i-1}) e(x_i|y_i)$$

$$= \left(q(\mathrm{STOP}|y_n) \prod_{i=1}^{n} q(y_i|y_{i-1} \right) \left(\prod_{i=1}^{n} e(x_i|y_i) \right)$$

$$= P(Y) \left(\prod_{i=1}^{n} P(x_i|y_i) \right)$$

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

Generative v.s. Discriminative

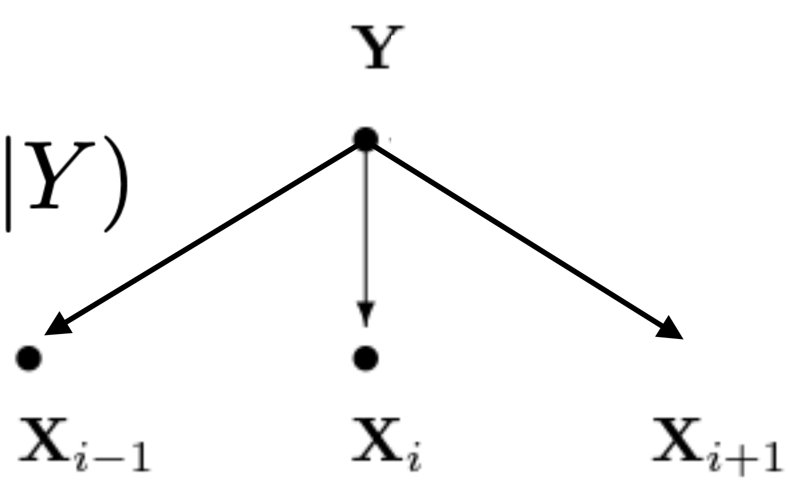
- **Generative Models**:
  - Joint probability: $P(X, Y)$
  - Make prediction by $\arg\max_Y P(X, Y)$
  - Can generate new samples (X, Y)
  - Examples: **HMMs, Naive Bayes**
- **Discriminative Models**:
  - Conditional probability: $P(Y|X)$
  - Can directly predict $\arg\max_Y P(Y|X)$
  - Examples: **Conditional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

Compare Naive Bayes and Logistic Regression

- Directed graphical model vs undirected graphical model

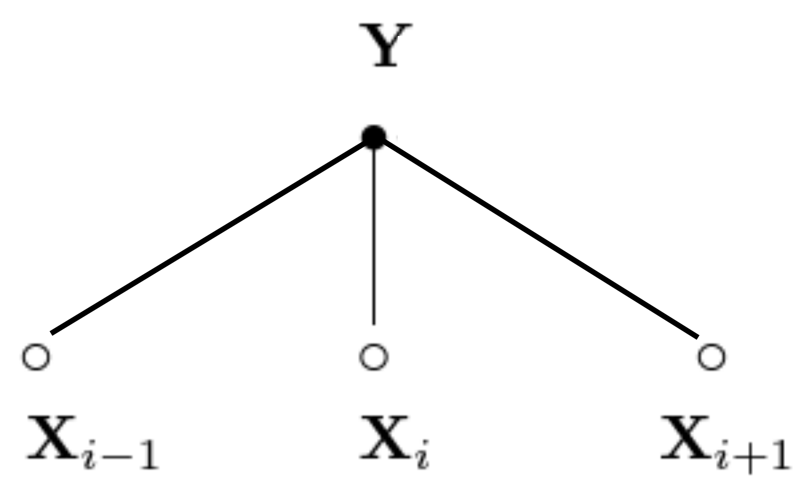
$$Y \sim P(Y)$$

$$P(X|Y)$$



Naive Bayes
(Generative)

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



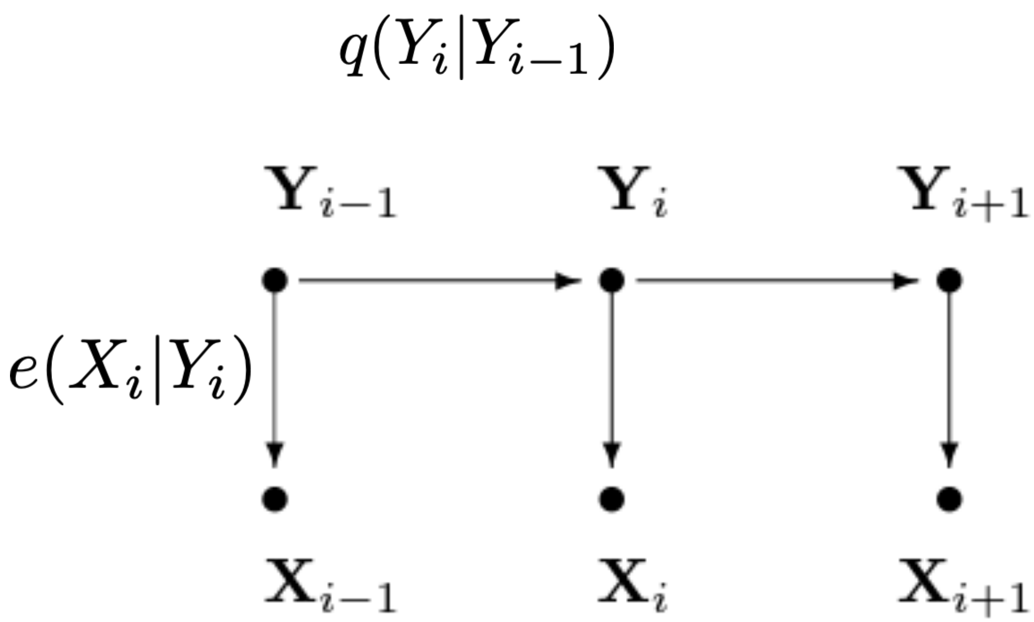
Logistic Regression
(Discriminative)

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

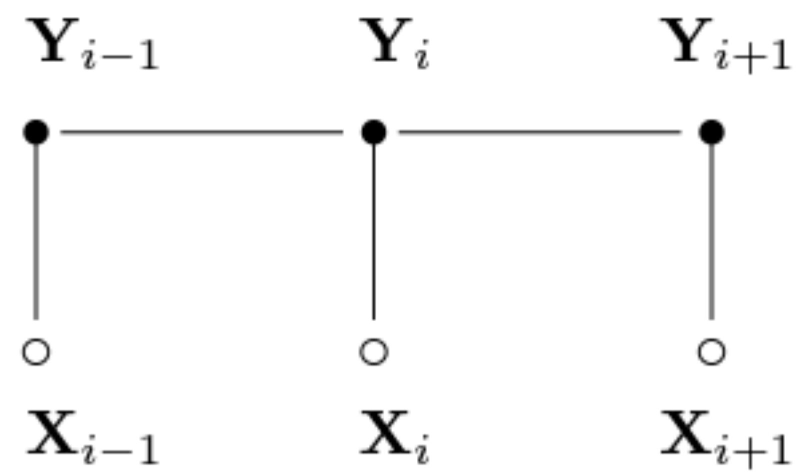
Compare HMM and linear chain CRF

- Directed graphical model vs undirected graphical model

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



HMM
(Generative)



Chain-structure CRF
(Discriminative)

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

Conditional Random Fields

(Sequential Version of Logistic Regression)

Recap: Logistic Regression (Log Linear Models)

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

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

$F(X, y = c)$ 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** $X = \{x_1 \cdots x_n\}, Y = \{y_1 \cdots y_n, \text{STOP}\}$:

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

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

d1 features scoring transitions

d2 features scoring each state w/ input sequence

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

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

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

ICML "test-of-time" paper: Lafferty et al. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data

CRF: Learning

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

$$
\begin{aligned}
\mathcal{L}(w) &= \sum_{(X,Y)\sim\mathcal{D}_{\text{train}}} \log P(Y|X) \\
&= \sum_{(X,Y)\sim\mathcal{D}_{\text{train}}} w^\top f(Y,X) - \log \boxed{Z(X)}
\end{aligned}
$$

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

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

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

CRF: Learning

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

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

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

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

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

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

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

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

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

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

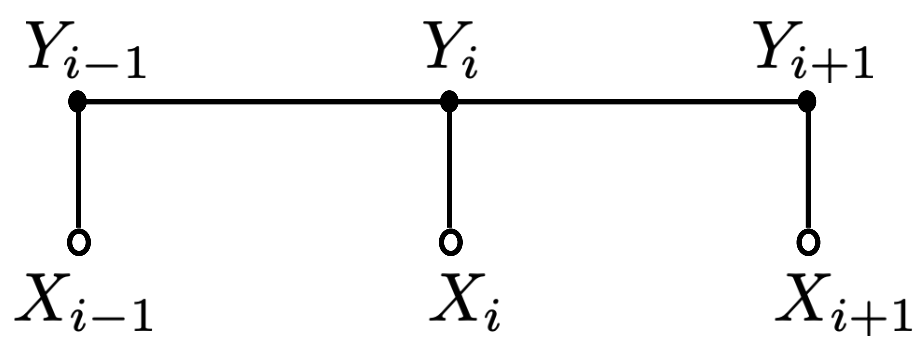
Feature values are not limited to just binary values, can be real-values too.
Number of features can be tens of thousands or more.

Feature Selection

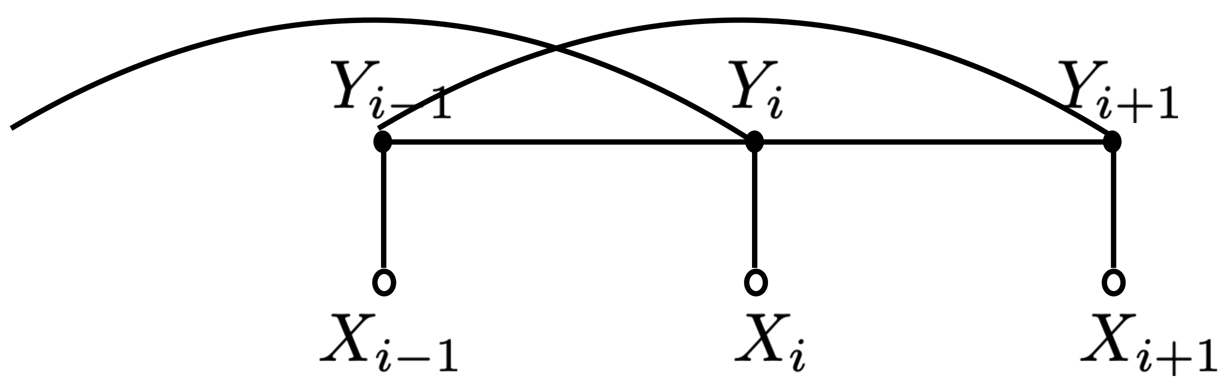
1. Initially CRF model has no features (uniform prediction)
2. Create some candidate feature sets, e.g., (combination of any word-tag pairs, x=John, y_i= PERSON). There are VK possible pairs
3. Build a new CRF w/ a subset of features
4. Include the selected features that improves over the previous CRF
5. Go to step 3 until enough features have been added to CRF

Neural Conditional Random Fields

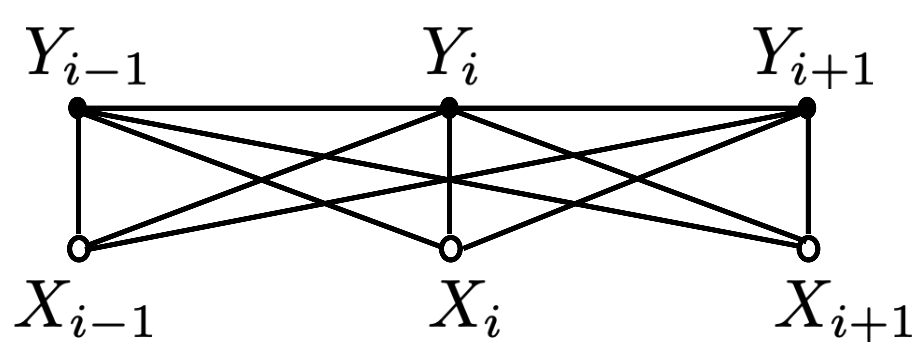
Variants of CRF Layers



- 1th order linear chain



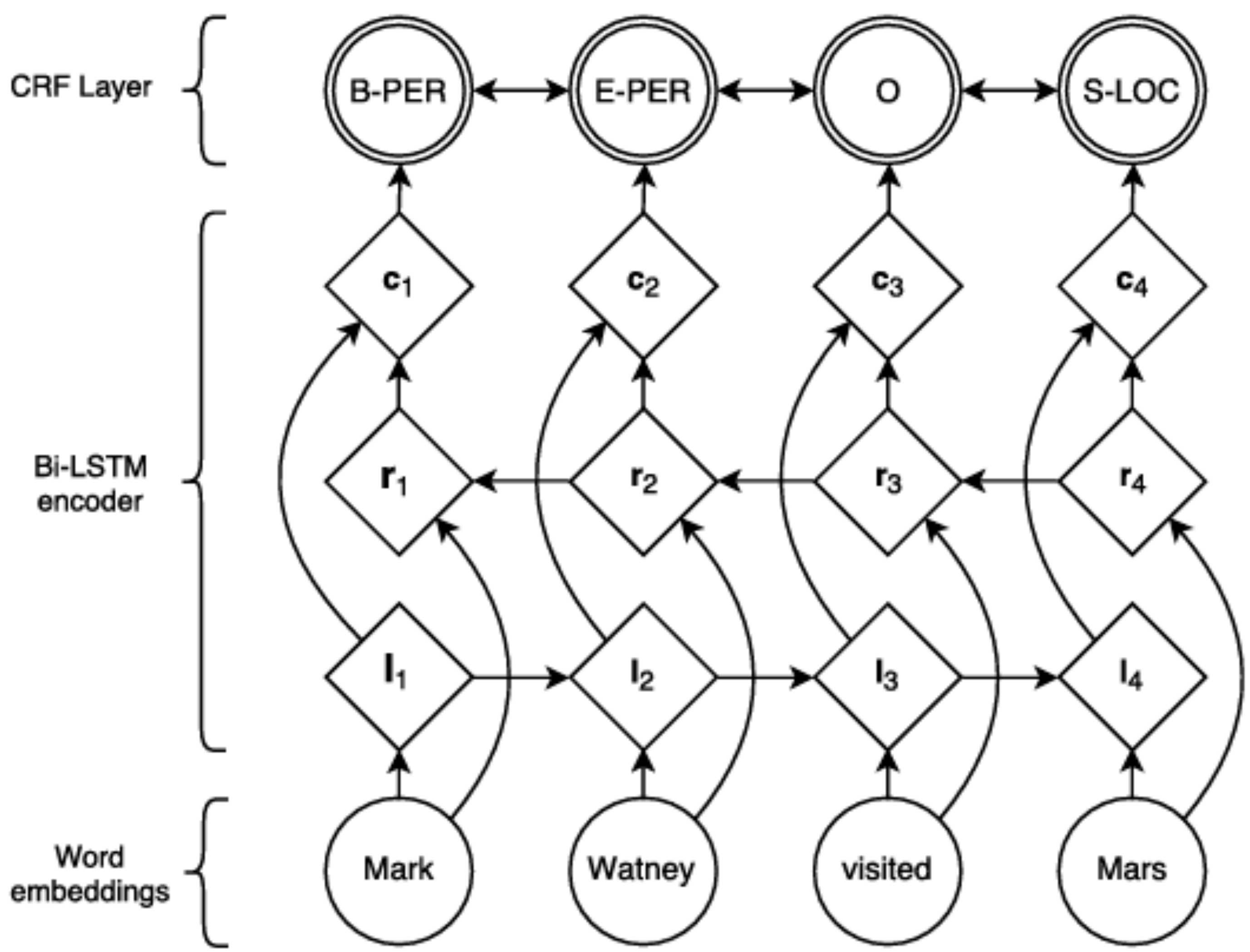
- 2nd order linear chain



- Local vs. Global context

Neural CRF

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



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

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

Lampple 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:

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

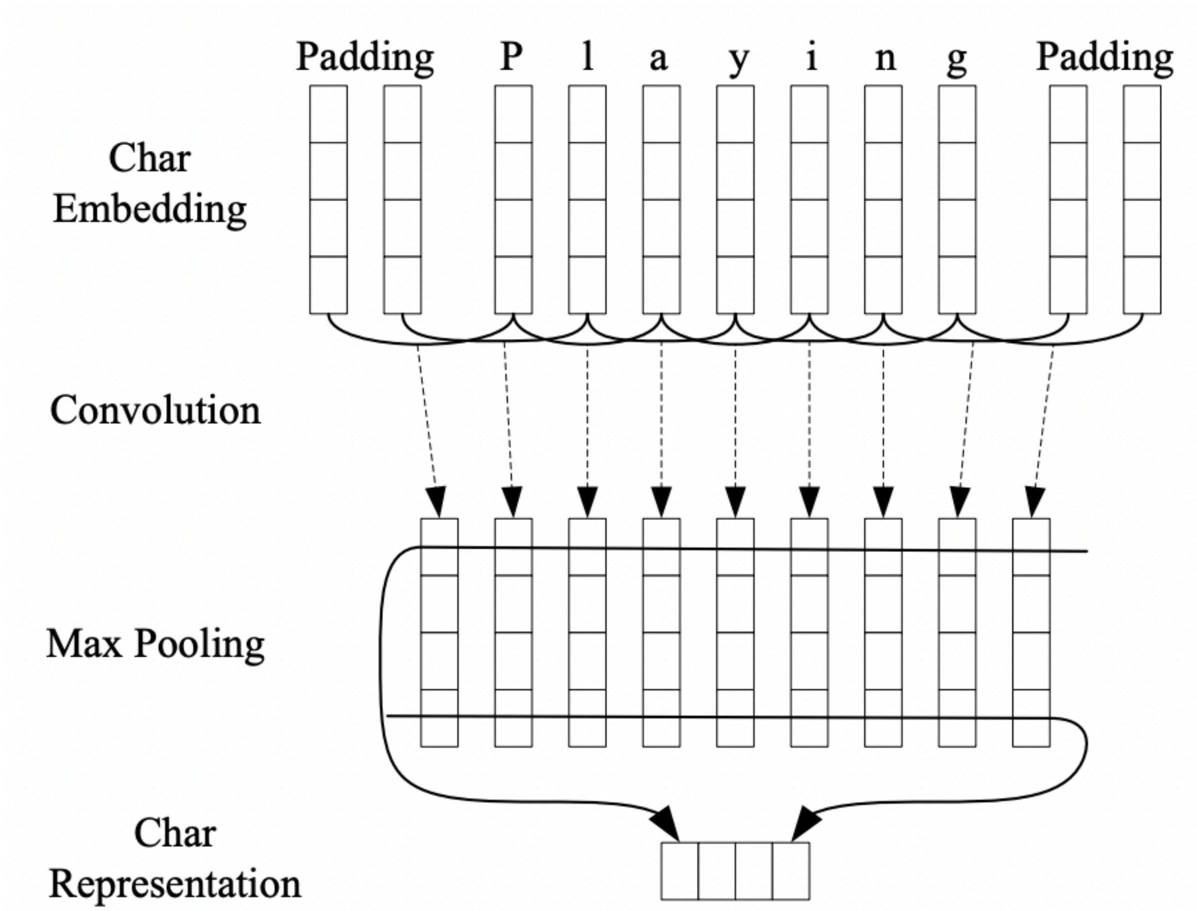
$$= s(\mathbf{X}, \mathbf{y}) - \underset{\widetilde{\mathbf{y}} \in \mathbf{Y_X}}{\operatorname{logadd}}\ s(\mathbf{X}, \widetilde{\mathbf{y}}), \quad (1)$$

- 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

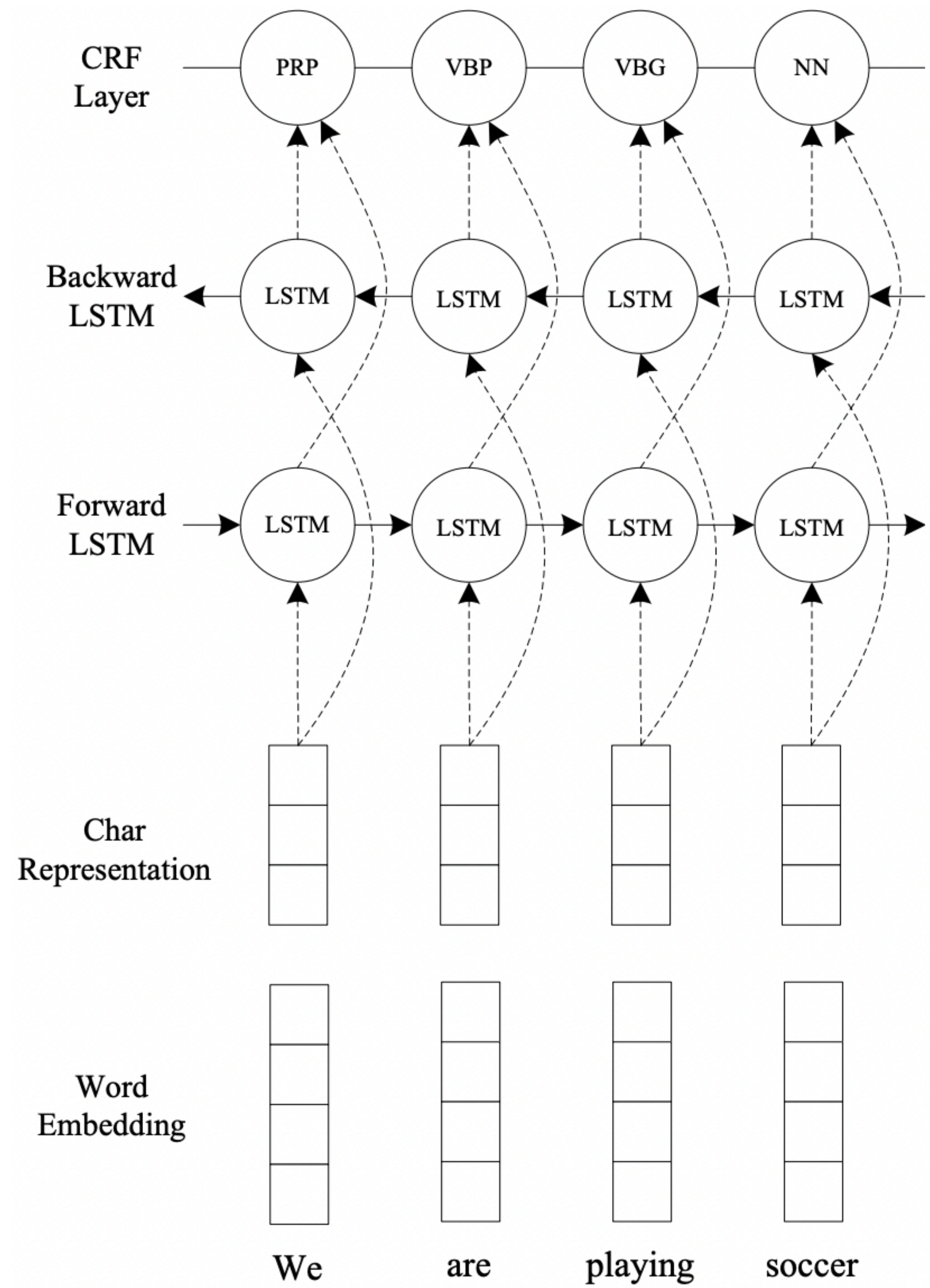
- Use CNN to encode character embeddings



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

BiLSTM-CNN CRF

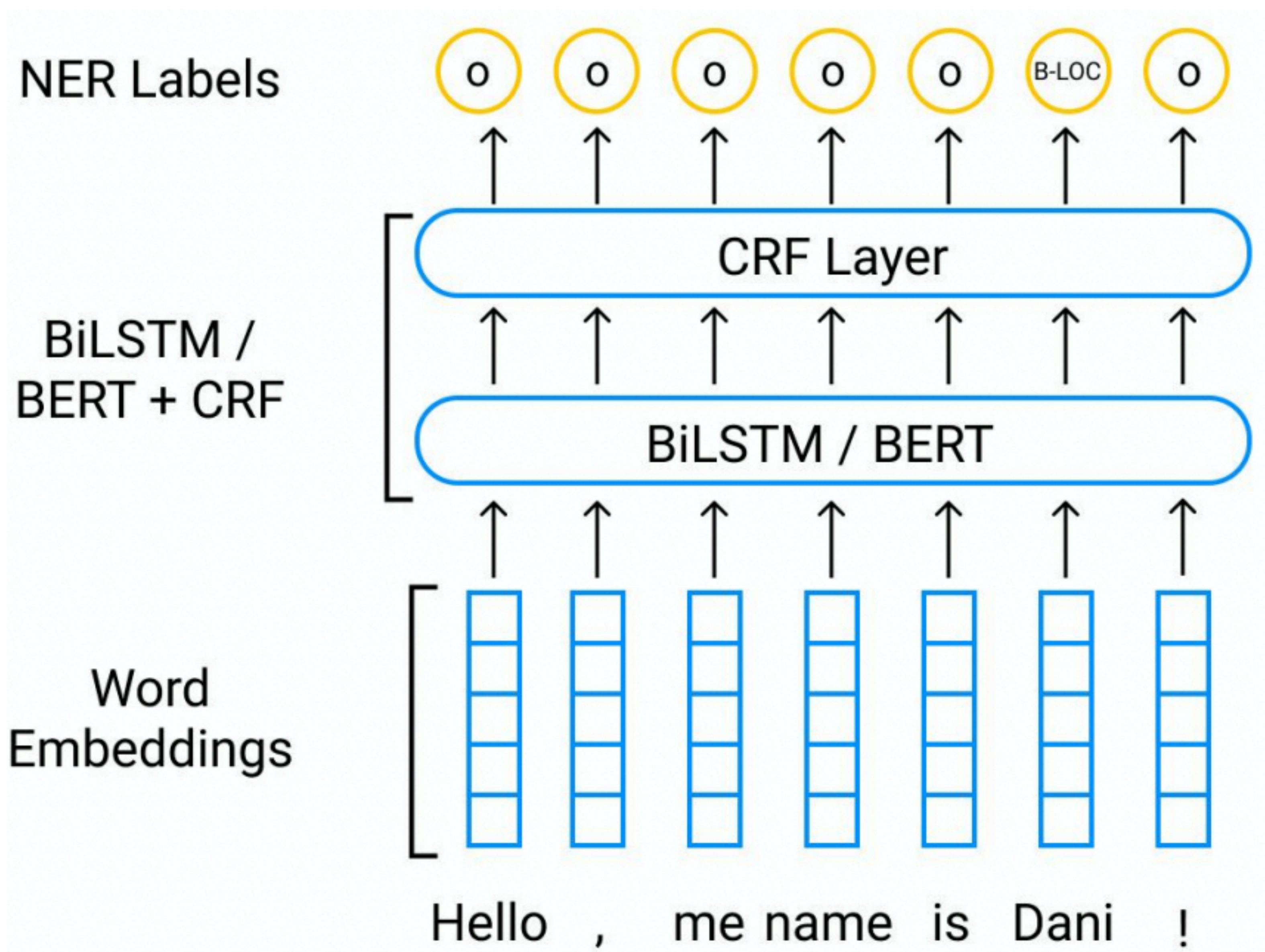
- Use CNN to encode character embeddings
- Combine char and word embeddings together
- Further encode by BiLSTM model to learn the sequence representations
- Add a CRF layer



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

BERT-CRF

- Replace BiLSTM with a BERT encoder



Code Walk: Neural CRF Implementation

https://pytorch-crf.readthedocs.io/en/stable/_modules/torchcrf.html#CRF.forward

More Sequence Labeling Examples

Named Entity Recognition

- Goal: Segment text into spans with certain properties
- e.g., entities: PER, ORG, and LOC

Germany 's representative to the European Union 's veterinary committee Werner Zwingman said on Wednesday consumers should…

↓

[Germany]$_{LOC}$'s representative to the [European Union]$_{ORG}$'s veterinary committee [Werner Zwingman]$_{PER}$ said on Wednesday consumers should…

Is this a sequence labeling task?

NER (IOB format)

- BL, BO, BP: beginning of LOC, ORG, PER respectively
- CL, CO, CP: continuation of chunks for LOC, ORG, PER
- NA: other words

> Germany ’s representative to the European Union ’s veterinary committee Werner Zwingman said on Wednesday consumers should…

↓

Germany/BL ’s/NA representative/NA to/NA the/NA European/BO Union/CO ’s/NA veterinary/NA committee/NA Werner/BP Zwingman/CP said/NA on/NA Wednesday/NA consumers/NA should/NA…

NER as Sequence Labeling

- IOB tagging scheme

[$_{ORG}$ **American Airlines**], a unit of [$_{ORG}$ **AMR Corp.**], immediately matched the move, spokesman [$_{PER}$ **Tim Wagner**] said.

Words	IOB Label	IO Label
American	B-ORG	I-ORG
Airlines	I-ORG	I-ORG
,	O	O
a	O	O
unit	O	O
of	O	O
AMR	B-ORG	I-ORG
Corp.	I-ORG	I-ORG
,	O	O
immediately	O	O

Named Entity tags

Type	Tag	Sample Categories	Example sentences
People	PER	people, characters	**Turing** is a giant of computer science.
Organization	ORG	companies, sports teams	The **IPCC** warned about the cyclone.
Location	LOC	regions, mountains, seas	The **Mt. Sanitas** loop is in **Sunshine Canyon**.
Geo-Political Entity	GPE	countries, states, provinces	**Palo Alto** is raising the fees for parking.
Facility	FAC	bridges, buildings, airports	Consider the **Golden Gate Bridge**.
Vehicles	VEH	planes, trains, automobiles	It was a classic **Ford Falcon**.

Ambiguity in NER

- Washington can be PER or ORG or LOC, VEH

[$_{\text{PER}}$ Washington] was born into slavery on the farm of James Burroughs.
[$_{\text{ORG}}$ Washington] went up 2 games to 1 in the four-game series.
Blair arrived in [$_{\text{LOC}}$ Washington] for what may well be his last state visit.
In June, [$_{\text{GPE}}$ Washington] passed a primary seatbelt law.
The [$_{\text{VEH}}$ Washington] had proved to be a leaky ship, every passage I made...

Common hand-crafted Features

identity of w_i, identity of neighboring words
embeddings for w_i, embeddings for neighboring words
part of speech of w_i, part of speech of neighboring words
base-phrase syntactic chunk label of w_i and neighboring words
presence of w_i in a **gazetteer**
w_i contains a particular prefix (from all prefixes of length ≤ 4)
w_i contains a particular suffix (from all suffixes of length ≤ 4)
w_i is all upper case
word shape of w_i, word shape of neighboring words
short word shape of w_i, short word shape of neighboring words
presence of hyphen

Common hand-crafted Features

- gazetteers
  - A list of place names providing millions of entries for locations with
    detailed geographical and political information
  - binary indicator features: define the condition for some prefix, suffix, etc.

$$
\begin{array}{ll}
\text{prefix}(w_i) = \texttt{L} & \text{suffix}(w_i) = \texttt{tane} \\
\text{prefix}(w_i) = \texttt{L'} & \text{suffix}(w_i) = \texttt{ane} \\
\text{prefix}(w_i) = \texttt{L'O} & \text{suffix}(w_i) = \texttt{ne} \\
\text{prefix}(w_i) = \texttt{L'Oc} & \text{suffix}(w_i) = \texttt{e} \\
\text{word-shape}(w_i) = \texttt{X'Xxxxxxxx} & \text{short-word-shape}(w_i) = \texttt{X'Xx}
\end{array}
$$

Semantic Role Labeling

- A *semantic role* in language is the relationship that a syntactic constituent has with a predicate.
- Typical semantic arguments include **Agent, Patient, Instrument**, etc. and also adjunctive arguments indicating **Locative, Temporal, Manner, Cause**, etc. aspects.
- Recognizing and labeling semantic arguments is a key task for answering **"Who", "When", "What", "Where", "Why"**, etc. questions in Information Extraction, Question Answering, Summarization.

[$_{\text{A0}}$ He] [$_{\text{AM-MOD}}$ would] [$_{\text{AM-NEG}}$ n't] [$_{\text{V}}$ **accept**] [$_{\text{A1}}$ anything of value] from [$_{\text{A2}}$ those he was writing about] .

V: verb
A0: acceptor
A1: thing accepted
A2: accepted-from
A3: attribute
AM-MOD: modal
AM-NEG: negation

CoNLL-2004

Multilingual POS tagging

- In morphologically-rich languages like Czech, Hungarian, Turkish
  - a 250,000 word token corpus of Hungarian has more than twice as many word types as a similarly sized corpus of English
  - a 10 million word token corpus of Turkish contains four times as many word types as a similarly sized English corpus
- => Many UNKs
- More information is coded in morphology

Yerdeki **izin** temizlenmesi gerek. **The trace** on the floor should be cleaned.	iz + Noun+A3sg+Pnon+Gen
Üzerinde parmak **izin** kalmiş **Your** finger **print** is left on (it).	iz + Noun+A3sg+P2sg+Nom

Multilingual POS tagging

- AsdfasIn non-word-space languages like Chinese word segmentation is either applied before tagging or done jointly
    - UNKs are difficult: the majority of unknown words are common nouns and verbs because of extensive compounding

- Universal POS tagset accounts for cross-linguistic differences

Questions?