

5. Part-of-speech Tagging with CRFs

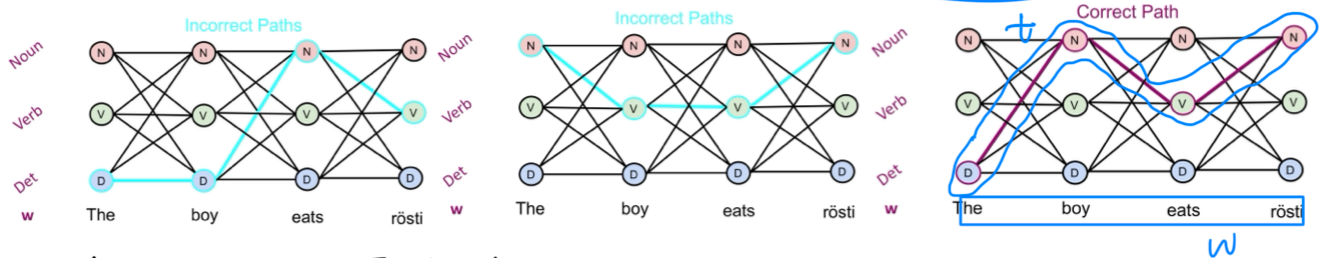
What is Part-of-speech Tagging

Assign each word in a sentence to one of these grammatical categories

- Noun, Verb, Adjective, Adverb, Determiner, etc.

* The simplest structured prediction in NLP (arguably)

Example: The boy eats rösti $\Rightarrow 3 \text{ tags } (D, N, V) \Rightarrow 3^4 = 81 \text{ assignments}$
 Determiner Noun Verb Noun
 Score of a path: $\text{score}(\langle D, N, V, N \rangle, \mathbf{w})$ full tag sequence $\mathbf{t}$ (len=n) word sequence $\mathbf{w}$ (sentence) of len=n



Conditional Random Fields (Log-linear Models on Structures)

The conditional random field (CRF) is a conditional probabilistic model for sequence labeling, built on the logistic regression classifier.

$$P(\mathbf{t} | \mathbf{w}) = \frac{\exp\{\text{score}(\mathbf{t}, \mathbf{w})\}}{\sum_{\mathbf{t}' \in T^N} \exp\{\text{score}(\mathbf{t}', \mathbf{w})\}}, \quad \text{score}(\mathbf{t}, \mathbf{w}) = \text{"goodness"}(\mathbf{t}, \mathbf{w}),$$

where $\mathbf{t}$ is a part-of-speech tagging, $\mathbf{w}$ is an input sentence, $N = |\mathbf{w}|$ sentence length

* Higher score means $\mathbf{t}$ is a better tagging for $\mathbf{w}$.

* Score(.) can be any function

- Linear: $\text{score}(\mathbf{t}, \mathbf{w}) = \theta \cdot f(\mathbf{t}, \mathbf{w})$

- Non-linear: $\text{score}(\mathbf{t}, \mathbf{w}) = NN_{\theta}(\mathbf{t}, \mathbf{w})$

Normalizer, $O(1T1N)$
 Computational complexity!!!
 Need assumptions & algorithms.

Encode structure assumption - combinatorial

Additively decomposable score function

$$\text{score}(\mathbf{t}, \mathbf{w}) = \sum_{n=1}^N \text{score}(\langle t_{n-1}, t_n \rangle, \mathbf{w})$$

↑
score full tag $\mathbf{t}$ for word sequence $\mathbf{w}$

↑
tag bigram

* Use special start symbol when $n=0$

Example (continued)

$$\begin{aligned} \text{score}(\langle D, N, V, N \rangle, \mathbf{w}) &= \text{score}(\langle \text{start}, D \rangle, \mathbf{w}) + \text{score}(\langle D, N \rangle, \mathbf{w}) + \text{score}(\langle N, V \rangle, \mathbf{w}) \\ &\quad + \text{score}(\langle V, N \rangle, \mathbf{w}) \\ &= \dots + \text{transition}(\langle D, N \rangle) + \text{emission}(\text{boy}, N) + \dots \end{aligned}$$

How likely N follows D How likely 'boy' is a Noun

New equation with structured score function

$$p(t, w) = \frac{\exp\{\sum_{n=1}^N \text{score}(\langle t_{n-1}, t_n \rangle, w)\}}{\sum_{t' \in T^N} \exp\{\sum_{n=1}^N \text{score}(\langle t'_{n-1}, t'_n \rangle, w)\}} \implies \prod_{n=1}^N \exp\{\text{score}(\langle t_{n-1}, t_n \rangle, w)\}$$

Efficiently compute the normalizer

$$\sum_{t \in T^N} \exp\left\{\sum_{n=1}^N \text{score}(\langle t_{n-1}, t_n \rangle, w)\right\} \quad * \text{ exponential number of terms !!!}$$

$$= \sum_{t_{1:N} \in T^N} \prod_{n=1}^N \exp\{\text{score}(\langle t_{n-1}, t_n \rangle, w)\}$$

$$= \sum_{t_{1:N-1} \in T^{N-1}} \sum_{t_N \in T} \prod_{n=1}^N \exp\{\text{score}(\langle t_{n-1}, t_n \rangle, w)\} \quad * \text{ Product distributes over sum}$$

* Linear number of terms now!

$$= \sum_{t_{1:N-1} \in T^{N-1}} \prod_{n=1}^{N-1} \exp\{\text{score}(\langle t_{n-1}, t_n \rangle, w)\} \times \sum_{t_N \in T} \exp\{\text{score}(\langle t_{N-1}, t_N \rangle, w)\}$$

$$= \sum_{t_1 \in T} \left(\exp\{\text{score}(\langle t_0, t_1 \rangle, w)\} \times \dots \times \sum_{t_{N-1} \in T} \left(\exp\{\text{score}(\langle t_{N-2}, t_{N-1} \rangle, w)\} \times \exp\{\text{score}(\langle t_{N-1}, t_N \rangle, w)\} \right) \dots \right)$$

Algorithm With Dynamic Programming

$$\beta(w, t_N) \leftarrow 1$$

for $n \leftarrow N-1, \dots, 0$:

$$\beta(w, t_n) \leftarrow \sum_{t_{n+1} \in T} \exp\{\text{score}(\langle t_n, t_{n+1} \rangle, w)\} \times \beta(w, t_{n+1})$$

* Assume $\forall t_n \in T$

* $z = \beta(w, t_0)$

The Backward Algorithm

Algorithm 5.1

def BACKWARD ALGORITHM(w, T, N):

for $t_{N-1} \in T$:

$\beta(w, t_{N-1}, N-1) \leftarrow \exp(\text{score}(\langle t_{N-1}, \text{EOS} \rangle, w))$ $\triangleright$ Handles the scores of the incoming edges of node EOS

end for

for $n \in N-2, \dots, 1$:

$\triangleright$ Handles the scores over normal edges

for $t_n \in T$:

$$\beta(w, t_n, n) \leftarrow \sum_{t_{n+1} \in T} \exp(\text{score}(\langle t_n, t_{n+1} \rangle, w)) \times \beta(w, t_{n+1}, n+1)$$

end for

end for

$$\beta(w, \text{BOS}, 0) \leftarrow 0$$

for $t_1 \in T$:

$\triangleright$ Handles the scores over the outgoing edges of the BOS node

$$\beta(w, \text{BOS}, 0) \leftarrow \beta(w, \text{BOS}, 0) + \exp(\text{score}(\langle \text{BOS}, t_1 \rangle, w)) \times \beta(w, t_1, 1)$$

end for

return $\beta(w, \text{BOS}, 0)$

Initialization

Example (Continued):

$$T = \{\text{BOT}, N, V, D\}$$

w	$\langle \text{BOS} \rangle$	The	boy	eats	rösti
BOT				0.03	0.01
N			80241.5	0.4	3.5

		Transition			
From	To	BOT	N	V	D
N		-100	2	4	1
:		:	:	:	:
		Emission			

V				0.9	0.4
D				0.8	0.2

	BoT	N	V	D
boy	-100	2	5	-10
:	:	:	:	:

assume we have already computed these

$$\begin{aligned}
 & \text{Score}(\langle N, \text{BoT} \rangle, w) + \text{Score}(\langle N, N \rangle, w) + \text{Score}(\langle N, V \rangle, w) + \text{Score}(\langle N, D \rangle, w) \\
 & \downarrow \qquad \qquad \qquad \downarrow \qquad \qquad \qquad \downarrow \qquad \qquad \qquad \downarrow \\
 & \text{Trans}(N, \text{BoT}) + \text{Emis}(N, \text{BoT}) \quad \text{Trans}(N, N) + \text{Emis}(N, N) \quad \text{Trans}(N, V) + \text{Emis}(N, V) \quad \text{Trans}(N, D) + \text{Emis}(N, D) \\
 & = -100 + -100 = -200 \quad = 2 + 2 = 4 \quad = 4 + 5 = 9 \quad = 1 - 10 = -9 \\
 & \exp(-200) = 0 \quad \exp(4) = 54.6 \quad \exp(9) = 8103.1 \quad \exp(-9) = 0
 \end{aligned}$$

$$(0 \times 0.03 = 0) + (54.6 \times 0.4 = 21.84) + (8103.1 \times 0.9 = 80220.69) + (0 \times 0.8 = 0) \Rightarrow 80241.5$$

The decoding problem

We want $\mathbf{t}^* \in \underset{\mathbf{t}' \in T^N}{\text{argmax}} \text{Score}(\mathbf{t}', w) \xrightarrow{\text{backpointers}} \underset{\mathbf{t}' \in T^N}{\text{Max}} \text{Score}(\mathbf{t}', w).$

Viterbi Algorithm

$$\gamma(w, t_N) \leftarrow 1$$

for $n \leftarrow N-1, \dots, 0$:

$$\gamma(w, t_n) \leftarrow \underset{t_{n+1} \in T}{\text{Max}} \exp\{\text{score}(\langle t_n, t_{n+1} \rangle, w)\} \times \gamma(w, t_{n+1})$$

* Very similar to the backward algorithm.

* We can generalize it with semiring.

Algorithm 5.2

```

def VITERBI_ALGORITHM(w, T, N):
    for tN-1 ∈ T:
        v(w, tN-1, N-1) ← exp(score(⟨tN-1, eos⟩, w))
    end for
    for n ∈ N-2, ..., 1:
        for tn ∈ T:
            v(w, tn, n) ← maxtn+1 ∈ T exp(score(⟨tn, tn+1⟩, w)) × v(w, tn+1, n+1)
            b(tn, n) ← argmaxtn+1 ∈ T exp(score(⟨tn, tn+1⟩, w)) × v(w, tn+1, n+1) ▷ Keeps track of
the best tags
        end for
    end for
    v(w, bos, 0) ← maxt1 ∈ T (v(w, bos, 0), exp(score(⟨bos, t1⟩, w)) × v(w, t1, 1))
    b(bos, 0) ← argmaxt1 ∈ T (v(w, bos, 0), exp(score(⟨bos, t1⟩, w)) × v(w, t1, 1))
    for n ∈ 1, ..., N:
        tn ← b(tn-1, n-1) ▷ Recovers the best tagging sequence using backpointers
    end for
    return t1:N, v(w, bos, 0)

```

Semirings

A semiring is a 5-tuple $R = (A, \oplus, \otimes, \bar{0}, \bar{1})$ such that

- $(A, \oplus, \bar{0})$ is a commutative monoid. * both associative and commutative
- $(A, \otimes, \bar{1})$ is a monoid. * associative
- $\otimes$ distributes over $\oplus$: for all a, b, c in A ,
 - $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$

- $c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)$
- $\bar{0}$ is an annihilator for $\otimes$, $\forall a \in A, \bar{0} \otimes a = a \otimes \bar{0} = \bar{0}$.

Example:

$$\langle \mathbb{R}_{\geq 0}, +, \times, 0, 1 \rangle$$

For $\forall a, b, c \in \mathbb{R}_{\geq 0}$:

$$\begin{aligned} - (a+b)+c &= a+(b+c) & - (a \times b) \times c &= a \times (b \times c) & - a \times (b+c) &= (a \times b) + (a \times c) \\ 0+a &= a+0 = a & 1 \times a &= a \times 1 & (a+b) \times c &= (a \times c) + (b \times c) \\ a+b &= b+a \\ - 0 \times a &= a \times 0 = 0 \end{aligned}$$

So it is a valid semiring.

Some Semirings

Semiring	Set	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$	intuition/application
Boolean	$\{0, 1\}$	$\vee$	$\wedge$	0	1	logical deduction, recognition
Viterbi	$[0, 1]$	max	$\times$	0	1	prob. of the best derivation
Inside	$\mathbb{R}^+ \cup \{+\infty\}$	+	$\times$	0	1	prob. of a string
Real	$\mathbb{R} \cup \{+\infty\}$	min	+	$+\infty$	0	shortest-distance
Tropical	$\mathbb{R}^+ \cup \{+\infty\}$	min	+	$+\infty$	0	with non-negative weights
Counting	$\mathbb{N}$	+	$\times$	0	1	number of paths

Generalizing the algorithm with semirings

$$\begin{aligned} & \bigoplus_{t_1:N \in \mathcal{T}^N} \bigotimes_{n=1}^N \exp \{ \text{score}(\langle t_{n-1}, t_n \rangle, \mathbf{w}) \} \\ &= \bigoplus_{t_1:N-1 \in \mathcal{T}^{N-1}} \bigoplus_{t_N \in \mathcal{T}} \bigotimes_{n=1}^N \exp \{ \text{score}(\langle t_{n-1}, t_n \rangle, \mathbf{w}) \} \\ &= \bigoplus_{t_1:N-1 \in \mathcal{T}^{N-1}} \bigotimes_{n=1}^{N-1} \exp \{ \text{score}(\langle t_{n-1}, t_n \rangle, \mathbf{w}) \} \otimes \bigoplus_{t_N \in \mathcal{T}} \exp \{ \text{score}(\langle t_{N-1}, t_N \rangle, \mathbf{w}) \} \\ &= \bigoplus_{t_1 \in \mathcal{T}} \exp \{ \text{score}(\langle t_0, t_1 \rangle, \mathbf{w}) \} \otimes \bigoplus_{t_2 \in \mathcal{T}} \exp \{ \text{score}(\langle t_1, t_2 \rangle, \mathbf{w}) \} \otimes \\ & \quad \dots \otimes \bigoplus_{t_N \in \mathcal{T}^N} \exp \{ \text{score}(\langle t_{N-1}, t_N \rangle, \mathbf{w}) \} \end{aligned}$$

Correctness is
guaranteed because
of semiring usage!

There are some other algorithms for shortest-path computation: Dijkstra, Bellman-Ford, Floyd-Warshall, etc.

* Dijkstra might be slower than Viterbi due to the queues. See HW2.

CRFs as a softmax & structured perception

Parameter estimation

Assume we have a data D consisting of K i.i.d. data points $(\mathbf{t}^{(k)}, \mathbf{w}^{(k)})$.

$$\begin{aligned} \log \prod_{k=1}^K p(\mathbf{t}^{(k)} | \mathbf{w}^{(k)}) &= \sum_{k=1}^K \log p(\mathbf{t}^{(k)} | \mathbf{w}^{(k)}) \\ &= \sum_{k=1}^K \log \left(\frac{\exp \{ \text{score}(\mathbf{t}^{(k)}, \mathbf{w}^{(k)}) \}}{\sum_{\mathbf{t} \in \mathcal{T}^N} \exp \{ \text{score}(\mathbf{t}, \mathbf{w}^{(k)}) \}} \right) \end{aligned}$$

$$= \sum_{k=1}^K \left(\text{score}(\mathbf{t}^{(k)}, \mathbf{w}^{(k)}) - \log \sum_{\mathbf{t}' \in T^N} \exp \{ \text{score}(\mathbf{t}', \mathbf{w}^{(k)}) \} \right)$$

* Use backpropagation to compute the gradient. Linear time due to generalized Viterbi, forward pass can be computed linearly in N .

CRF and softmax

$$\begin{aligned} & \sum_{k=1}^K \left(\text{score}(\mathbf{t}^{(k)}, \mathbf{w}^{(k)}) - \log \sum_{\mathbf{t}' \in T^N} \exp \{ \text{score}(\mathbf{t}', \mathbf{w}^{(k)}) \} \right) \\ &= \sum_{k=1}^K \left(\text{score}(\mathbf{t}^{(k)}, \mathbf{w}^{(k)}) - \underbrace{\left(\frac{1}{T} \log \sum_{\mathbf{t}' \in T^N} \frac{\exp \{ \text{score}(\mathbf{t}', \mathbf{w}^{(k)}) \}}{T} \right)}_{\text{temperature in softmax}} \right) \end{aligned}$$

$$\xrightarrow{T \rightarrow 0} \sum_{k=1}^K \left(\text{score}(\mathbf{t}^{(k)}, \mathbf{w}^{(k)}) - \max_{\mathbf{t}' \in T^N} \text{score}(\mathbf{t}', \mathbf{w}^{(k)}) \right)$$

* If the minibatch size and the learning rate are both equal to 1, the above is known as the structured perceptron update rule.