Sequence labeling problems

Sequence labeling problems

- Many problems in NLP can be formulated as sequence labeling problems
  - POS tagging:
    - The_DT man_NN who_WP whistles_VBZ tunes_VBZ pianos_NNS
  - Named Entity Recognition (NER)
    - The_O company_O is_O backed_O by_O Microsoft_B-ORG cofounder_O Bill_B-PER Gates_I-PER and_O venture_O capitalist_O Andressen_B-PER Horowitz_I-PER
  - Time expression detection
    - Bedford_O police_O said_O they_O received_O a_O call_O about_O 3:45_B-TIMEX p.m._I-TIMEX Monday_B-TIMEX
  - Spoken language understanding
    - Which_O flights_FLIGHT arrive_ARRIVE in_O Burbank_CITY from_O Denver_CITY on_ON Saturday_Day
  - . . . . . .

Search and Learning

Recall most natural language problems can be formulated mathematically as optimization:

$$\hat{\boldsymbol{y}} = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} \Psi(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})$$

There are two modules to this:

- Search, the module that is responsible for finding the argmax of the score function Ψ
- Learning, the module that is responsible for finding the optimal parameters $\boldsymbol{\theta}$

For simple text classification problems, the search module is fairly straightforward, and most of the work goes to learning. For sequence labeling and more complicated NLP problems, the search module is getting more complicated.

Sequence labeling: first idea

- Classify the sequence one element at a time

Sequence labeling example: POS tagging

- Let's use POS tagging as an example
- The most common used data set for training POS taggers is the Penn TreeBank

DT	NN	WP	VBZ	VBZ	NNS
The	man	who	whistles	tunes	pianos

- DT: Determiner
- NN: uncountable noun or noun in singular form
- WP: Wh-pronoun
- VBZ: 3rd person singular verb
- NNS: plural noun

How do we extract features from sequences in a linear model?

- We take as input a sequence of word tokens $\boldsymbol{x}$ and their corresponding POS tags $\boldsymbol{y}$, as well as a position m, and return a set of features associated with that position.
- Typically we make the assumption that the context that matters for classifying the word at position m are its surrounding words. We define a window that is centered on position m of size k, and only extract contextual information from this window.

Extracting features from a window size of 1

- Assuming a window of 1, the features we will be extracting from the example sentence will be:

$$\boldsymbol{f}((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 1), DT)$$
$$= (w_0 = the, DT)$$
$$\boldsymbol{f}((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 2), NN)$$
$$= (w_0 = man, NN)$$
$$\ldots\ldots$$

How many features will we extract if we use a window of size 1?

Weights

We can then train a classifier using these features and get a weight for each feature:

	DT	NN	VBP	VBZ	NNS
w_0=the	-0.05	-3.9	-4.6	-4.6	-4.6
w_0=man	-4.6	-0.35	-4.6	-1.4	-3.5
w_0=who	-4.6	-4.6	-0.05	-4.6	-4.6
w_0=whistles	-4.6	-4.6	-4.6	-0.8	-0.63
w_0=tunes	-4.6	-4.6	-4.6	-0.8	-0.6
w_0=pianos	-4.6	-4.6	-4.6	-3.0	-0.08

For example, the weight $\theta_1 = -0.05$ for the feature $f_1(w_0 = the, DT)$

Using these weights we can classify each word in the sequence

$$\psi((\boldsymbol{w} = \text{the man who whistles tunes pianes}, m = 1), DT)$$
$$= \sum_i f_i \theta_i = -0.05$$

$$\psi((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 1), NN)$$
$$= \sum_i f_i \theta_i = -3.9$$

$$\psi((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 1), WP)$$
$$= \sum_i f_i \theta_i = -4.6$$

$$\psi((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 1), VBZ)$$
$$= \sum_i f_i \theta_i = -4.6$$

$$\psi((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 1), NNS)$$
$$= \sum_i f_i \theta_i = -4.6$$

Predicting the tag for each word in the sequence

After finding the argmax at all positions of the sentence, we get:

DT	NN	WP	NNS	NNS	NNS
The	man	who	whistles	tunes	pianos

	DT	NN	WP	VBZ	NNS
w_0=the	-0.05	-3.9	-4.6	-4.6	-4.6
w_0=man	-4.6	-0.35	-4.6	-1.4	-3.5
w_0=who	-4.6	-4.6	-0.05	-4.6	-4.6
w_0=whistles	-4.6	-4.6	-4.6	-0.8	-0.63
w_0=tunes	-4.6	-4.6	-4.6	-0.8	-0.6
w_0=pianos	-4.6	-4.6	-4.6	-3.0	-0.08

Extracting features from a larger window

- If we increase the window size to 2 and also include the previous word in the context

$\boldsymbol{f}((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 1), DT)$
$= \{(w_0 = the, DT), (w_{-1} = START, DT)\}$

$\boldsymbol{f}((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 2), NN)$
$= \{(w_0 = man, NN), (w_{-1} = the, NN)\}$

.....

$\boldsymbol{f}((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 4), VBZ)$
$= \{(w_0 = whistles, VBZ), (w_{-1} = who, VBZ)\}$

Include weights for the new features

	DT	NN	WP	VBZ	NNS
w_0=the	-0.05	-3.9	-4.6	-4.6	-4.6
w_0=man	-4.6	-0.35	-4.6	-1.4	-3.5
w_0=who	-4.6	-4.6	-0.03	-4.6	-4.6
w_0=whistles	-4.6	-4.6	-4.6	-0.8	-0.63
w_0=tunes	-4.6	-4.6	-4.6	-0.8	-0.6
w_0=pianos	-4.6	-4.6	-4.6	-3.0	-0.08
w_{-1}=START	-0.92	-3.9	-1.9	-3.5	-0.92
w_{-1}=the	-4.6	-1.7	-4.6	-4.6	-0.75
w_{-1}=man	-1.6	-2.3	-0.9	-1.6	-2.3
w_{-1}=who	-1.8	-4.6	-4.6	-0.2	-4.6
w_{-1}=whistles	-2.3	-4.6	-4.6	-1.6	-0.4
w_{-1}=tunes	-1.6	-4.6	-4.6	-4.6	-0.26

Classification with the new weights

$$\psi((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 4), DT)$$
$$= \sum_i f_i\theta_i = -4.6 - 1.8 = -6.4$$

$$\psi((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 4), NN)$$
$$= \sum_i f_i\theta_i = -4.6 - 4.6 = -9.2$$

$$\psi((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 4), WP)$$
$$= \sum_i f_i\theta_i = -4.6 - 4.6 = -9.2$$

$$\psi((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 4), VBZ)$$
$$= \sum_i f_i\theta_i = -0.8 - 0.2 = -1$$

$$\psi((\boldsymbol{w} = \text{the man who whistles tunes pianos}, m = 4), NNS)$$
$$= \sum_i f_i\theta_i = -0.63 + -4.6 = -5.23$$

So VBZ receives the highest score when classifying position 4.

Updated classification results

DT	NN	WP	VBZ	NNS	NNS
The	man	who	whistles	tunes	pianes

	DT	NN	WP	VBZ	NNS
w_0=the	-0.05	-3.9	-4.6	-4.6	-4.6
w_0=man	-4.6	-0.35	-4.6	-1.4	-3.5
w_0=who	-4.6	-4.6	-0.05	-4.6	-4.6
w_0=whistles	-4.6	-4.6	-4.6	-0.8	-0.63
w_0=tunes	-4.6	-4.6	-4.6	-0.8	-0.6
w_0=pianes	-4.6	-4.6	-4.6	-3.0	-0.08
w_{-1}=START	-0.92	-3.9	-1.9	-3.5	-0.92
w_{-1}=the	-4.6	-0.7	-4.6	-4.6	-0.75
w_{-1}=man	-1.6	-2.3	-0.9	-1.6	-2.3
w_{-1}=who	-1.8	-4.6	-4.6	-0.2	-4.6
w_{-1}=whistles	-2.3	-4.6	-4.6	-1.6	-0.4
w_{-1}=tunes	-1.6	-4.6	-4.6	-4.6	-0.26

Sequence labeling as structured prediction

- Enlarging the window to include more context helps, to a degree

- To further improve the classifier requires also evaluating sequences of tags. For example, the tag sequence "NNS NNS" should receives a very low score as it rarely appears. Incorporating such information in the model would help improve tagging accuracy

- The tags are of course not observable in the data, and they need to be predicted together.

Sequence labeling: Computing a global score for the entire sequence

- Consider all possible label sequences for the input sequence, and choose the one that has the highest score

$$\Psi(\boldsymbol{w}, (DT, NN, WP, VBZ, NNS, NNS)) =$$

$$\Psi(\boldsymbol{w}, (DT, NN, WP, VBZ, VBZ, NNS)) =$$

- For a sequence of M elements with a tagset of size N, there are N^M possible sequences, a very large number!
- To find the sequence with the highest score, we need to do this efficiently
- The common solution is the Viterbi Algorithm

Sequence labeling as structured prediction

- The goal of the model is to find the tag sequence that yields the highest score for the input sequence:

$$\hat{\boldsymbol{y}} = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{w})} \Psi(\boldsymbol{w}, \boldsymbol{y})$$

- To make the computation tractable we factor the score for the entire sequence into the sum of local scores at each position m

$$\Psi(\boldsymbol{w}, \boldsymbol{y}) = \sum_{m=1}^{M+1} \psi(\boldsymbol{w}, y_m, y_{m-1}, m)$$

- The local score is a weighted sum of the local features at position m.

$$\psi(\boldsymbol{w}_{1:M}, y_m, y_{m-1}, m) = \boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{w}, y_m, y_{m-1}, m)$$

Feature representation for sequences

$$\boldsymbol{f}(\boldsymbol{w} = \text{the man who whistles tunes pianos}, \boldsymbol{y} = \text{DT NN WP VBZ VBZ NNS})$$

$$= \sum_{m=1}^{M+1} \boldsymbol{f}(\boldsymbol{w}, y_m, y_{m-1}, m)$$

$$= \boldsymbol{f}(\boldsymbol{w}, DT, \Diamond, 1) + \boldsymbol{f}(\boldsymbol{w}, NN, DT, 2) + \boldsymbol{f}(\boldsymbol{w}, WP, NN, 3)$$

$$+ \boldsymbol{f}(\boldsymbol{w}, VBZ, WP, 4) + \boldsymbol{f}(\boldsymbol{w}, VBZ, VBZ, 5) + \boldsymbol{f}(\boldsymbol{w}, NNS, VBZ, 6)$$

$$+ \boldsymbol{f}(\boldsymbol{w}, \blacklozenge, NNS, 7)$$

$$= f(w_0 = the, y_0 = DT) + f(y_0 = DT, y_{-1} = \Diamond)$$

$$+ f(w_0 = man, y_0 = NN) + f(y_0 = NN, y_{-1} = DT)$$

$$+ f(w_0 = who, y_0 = WP) + f(y_0 = WP, y_{-1} = NN)$$

$$+ f(w_0 = whistles, y_0 = VBZ) + f(y_0 = VBZ, y_{-1} = WP)$$

$$+ f(w_0 = tunes, y_m = VBZ) + f(y_0 = VBZ, y_{-1} = VBZ)$$

$$+ f(w_0 = pianos, y_0 = NNS) + f(y_0 = NNS, y_{-1} = VBZ)$$

$$+ f(y_0 = \blacklozenge, y_{-1} = NNS)$$

Decoding for sequences: The Viterbi algorithm

- The goal is to find the sequence of tags with the highest score:

$$\hat{\boldsymbol{y}} = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{w})} \Psi(\boldsymbol{w}, \boldsymbol{y})$$

$$= \operatorname*{argmax}_{\boldsymbol{y}_{1:M}} \sum_{m=1}^{M+1} \psi(\boldsymbol{w}, y_m, y_{m-1}, m)$$

$$= \operatorname*{argmax}_{\boldsymbol{y}_{1:M}} \sum_{m=1}^{M+1} s_m(y_m, y_{m-1})$$

- Instead of finding the argmax for the entire sequence directly, we start by finding the max up to position m and keep a sequence of back pointers

Finding the max score for the sequence

$$\max_{\boldsymbol{y}_{1:M}} \Psi(\boldsymbol{x}, \boldsymbol{y}_{1:M}) = \max_{\boldsymbol{y}_{1:M}} \sum_{m=1}^{M+1} s_m(y_m, y_{m-1})$$

$$= \left(\max_{y_M} s_{M+1}(\blacklozenge, y_M) \right) + \left(\max_{\boldsymbol{y}_{1:M-1}} \sum_{m=1}^{M} s_m(y_m, y_{m-1}) \right)$$

Viterbi variable

Caching Viterbi variables as intermediate results:

$$\upsilon_m(y_m) \triangleq \max_{\boldsymbol{y}_{1:m-1}} \sum_{n=1}^{m} s_n(y_n, y_{n-1})$$

$$= \max_{y_{m-1}} s_m(y_m, y_{m-1}) + \max_{\boldsymbol{y}_{1:m-2}} \sum_{n=1}^{m-1} s_n(y_n, y_{n-1})$$

$$= \max_{y_{m-1}} s_m(y_m, y_{m-1}) + \upsilon_{m-1}(y_{m-1})$$

Note that $\upsilon_1(y_1) \triangleq s_1(y_1, \lozenge)$ and the maximum overall score for the sequence is the final Viterbi variable
$\max_{\boldsymbol{y}_{1:M}} \Psi(\boldsymbol{w}_{1:M}, \boldsymbol{y}_{1:M}) = \upsilon_{M+1}(\blacklozenge)$

The Viterbi Algorithm

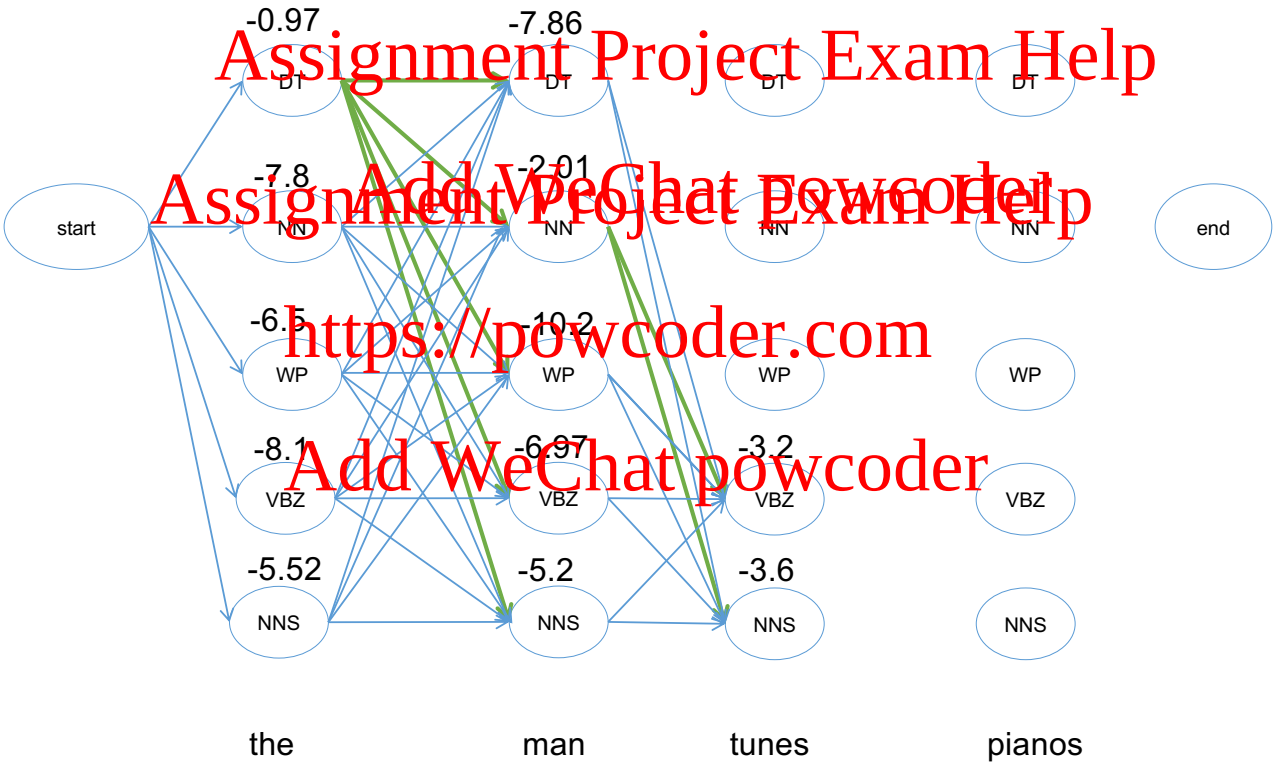
Viterbi Algorithm: Each $s_m(k, k')$ is a local score for tag $y_m = k$ and $y_{m-1} = k'$

for $k \in \{0, \ldots, K\}$ **do**
    $v_1(k) \leftarrow s_1(k, \Diamond)$
for $m \in \{2 \ldots M\}$ **do**
    for $k \in \{0 \ldots K\}$ **do**
        $v_m(k) \leftarrow \max_{k'} s_m(k, k') + v_{m-1}(k')$
        $b_m(k) \leftarrow \text{argmax}_{k'} s_m(k, k') + v_{m-1}(k')$
$y_M \leftarrow \text{argmax}_k s_{M+1}(\blacklozenge, k) + v_M(k)$
for $m \in \{M-1, \cdots, 1\}$ **do**
    $y_m \leftarrow b_m(y_{m+1})$
return $y_{1:M}$

Assuming these parameters

	DT	NN	WP	VBZ	NNS	♦
w_0=the	-0.05	-3.9	-4.6	-4.6	-4.6	$-\infty$
w_0=man	-4.6	-0.35	-4.6	-1.4	-3.5	$-\infty$
w_0=who	-4.6	-4.6	-0.25	-4.6	-4.6	$-\infty$
w_0=whistles	-4.6	-4.6	-4.6	-0.8	-0.63	$-\infty$
w_0=tunes	-4.6	-4.6	-4.6	-0.8	-0.6	$-\infty$
w_0=pianos	-4.6	-4.6	-4.6	-3.0	-0.08	$-\infty$
$t_{-1} = \diamondsuit$	-0.92	-3.9	-1.9	-3.5	-0.92	$-\infty$
t_{-1}=DT	-2.3	-0.69	-4.6	-4.6	-0.76	-4.6
t_{-1}=NN	-4.6	-1.6	-0.3	-0.36	-1.0	-0.7
t_{-1}=WP	-3.8	-4.6	-4.6	-0.2	-4.6	-4.6
t_{-1}=VBZ	-0.2	-1.3	-1.6	-4.6	-0.92	-2.3
w_{-1}=NNS	-4.6	-4.6	-0.1	-4.6	-3.9	-1.2

Example Viterbi computation



Additional features (and their weights) can be added

	DT	NN	WP	VBZ	NNS	♦
w_0=the	-0.05	-3.9	-4.6	-4.6	-4.6	$-\infty$
w_0=man	-4.6	-0.35	-4.6	-1.4	-3.5	$-\infty$
w_0=who	-4.6	-4.6	-0.05	-4.6	-4.6	$-\infty$
w_0=whistles	-4.6	-4.6	-4.6	-0.8	-0.63	$-\infty$
w_0=tunes	-4.6	-4.6	-4.6	-0.8	-0.6	$-\infty$
w_0=pianos	-4.6	-4.6	-4.6	-3.0	-0.08	$-\infty$
$t_{-1} = \Diamond$	-0.92	-3.9	-1.9	-3.5	-0.92	$-\infty$
t_{-1}=DT	-2.3	-0.69	-4.6	-4.6	-0.76	-4.6
t_{-1}=NN	-4.6	-1.6	-0.3	-0.36	-1.0	-0.7
t_{-1}=WP	-3.8	-4.6	-4.6	-0.2	-4.6	-4.6
t_{-1}=VBZ	-0.2	-1.3	-1.6	-4.6	-0.92	-2.3
w_{-1}=NNS	-4.6	-4.6	-0.1	-4.6	-3.9	-1.2
w_{-1}=START	-0.92	-3.9	-1.9	-3.5	-0.92	$-\infty$
w_{-1}=the	-4.6	-0.7	-4.6	-4.6	-0.75	-10
w_{-1}=man	-1.6	-2.3	-0.9	-1.6	-2.3	-1
w_{-1}=who	-1.8	-4.6	-4.6	-0.2	-4.6	-9
w_{-1}=whistles	-2.3	-4.6	-4.6	-1.6	-0.4	-0.5
w_{-1}=tunes	-1.6	-4.6	-4.6	-4.6	-0.26	-0.3

Feature templates used in SoA models

State-of-the-art models tend to use richer set of features and high-order transitions

- current word, w_0

- previous words, w_{-1}, w_{-2}
- next words, w_1, w_2

- previous two tags, y_{-1}, y_{-2}
- for rare words:

  - first k characters, up to $K = 4$
  - last k characters, up to $k = 4$
  - whether w_m contains a number, uppercase character, or hyphen

Parameter estimation for sequence labeling

We can extend the text classification models to sequence labeling:

Text classification	**Sequence Labeling**
Naïve Bayes	Hidden Markov Models (HMM)
Logistic Regression	Conditional Random Fields (CRF)
Perceptron	Perceptron
Support Vector Machines (SVM)	Support Vector Machines (SVM)