CS114B Lab 8

Kenneth Lai

March 24, 2022

► Perceptrons for sequence labeling

- Perceptrons for sequence labeling
 - Given a sequence $X = [x_1, \dots, x_T]$, predict labels $Y = [y_1, \dots, y_T]$

- Perceptrons for sequence labeling
 - Given a sequence $X = [x_1, \dots, x_T]$, predict labels $Y = [y_1, \dots, y_T]$
 - We do not care about the probability P(Y|X), just which Y has the highest score Z

- Perceptrons for sequence labeling
 - Given a sequence $X = [x_1, \dots, x_T]$, predict labels $Y = [y_1, \dots, y_T]$
 - We do not care about the probability P(Y|X), just which Y has the highest score Z
- $\hat{Y} = \operatorname*{argmax}_{k \in K^T} Z_k$

► Score Z decomposes into a sum of local parts

- ► Score Z decomposes into a sum of local parts
 - At each time step i, for each possible combination of current tag y_i and previous tag y_{i-1} , compute a local score $z(y_i, y_{i-1})$

- Score Z decomposes into a sum of local parts
 - At each time step i, for each possible combination of current tag y_i and previous tag y_{i-1} , compute a local score $z(y_i, y_{i-1})$
 - Use the Viterbi algorithm to combine the local scores across the sequence, and find the argmax

Suppose that at each time step i, we want to predict the current tag y_i using the following features:

- ► Suppose that at each time step *i*, we want to predict the current tag *y_i* using the following features:
 - ▶ Previous tag y_{i-1}

- ► Suppose that at each time step *i*, we want to predict the current tag *y_i* using the following features:
 - \triangleright Previous tag y_{i-1}
 - ► At the beginning of the sentence, let the previous tag *y*_{*i*-1} be the start symbol <S>

- ► Suppose that at each time step *i*, we want to predict the current tag *y_i* using the following features:
 - \triangleright Previous tag y_{i-1}
 - At the beginning of the sentence, let the previous tag y_{i-1} be the start symbol <S>
 - Current word x_i

- ► Suppose that at each time step *i*, we want to predict the current tag *y_i* using the following features:
 - \triangleright Previous tag y_{i-1}
 - ▶ At the beginning of the sentence, let the previous tag y_{i-1} be the start symbol <S>
 - Current word x_i
- ► For simplicity, we will assume that these are the only features, and we will ignore the bias term

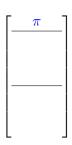
- ► Suppose that at each time step *i*, we want to predict the current tag *y_i* using the following features:
 - \triangleright Previous tag y_{i-1}
 - ▶ At the beginning of the sentence, let the previous tag y_{i-1} be the start symbol <S>
 - Current word x_i
- ► For simplicity, we will assume that these are the only features, and we will ignore the bias term
- Let $f(X, y_i, y_{i-1}, i)$ be the feature vector at time step i

- ► Suppose that at each time step *i*, we want to predict the current tag *y_i* using the following features:
 - \triangleright Previous tag y_{i-1}
 - ▶ At the beginning of the sentence, let the previous tag y_{i-1} be the start symbol <S>
 - ightharpoonup Current word x_i
- ► For simplicity, we will assume that these are the only features, and we will ignore the bias term
- ▶ Let $\mathbf{f}(X, y_i, y_{i-1}, i)$ be the feature vector at time step i
 - Using f instead of x, because features can include more than just the input

ightharpoonup We can arrange our weight matrix Θ as follows:

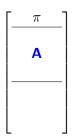


ightharpoonup We can arrange our weight matrix Θ as follows:



- ► Initial features
 - ▶ $y_{i-1} = \langle S \rangle, y_i = ...$

 \blacktriangleright We can arrange our weight matrix Θ as follows:



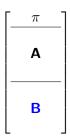
Initial features

$$y_{i-1} = \langle S \rangle, y_i = \dots$$

► Transition features

$$ightharpoonup y_{i-1} = \dots, y_i = \dots$$

 \blacktriangleright We can arrange our weight matrix Θ as follows:



Initial features

$$y_{i-1} = \langle S \rangle, y_i = \dots$$

Transition features

$$y_{i-1} = \dots, y_i = \dots$$

► Emission features

$$\triangleright$$
 $x_i = \ldots, y_i = \ldots$

• We want to compute local scores $z(y_1, \le)$ for each possible y_1

- We want to compute local scores $z(y_1, \le)$ for each possible y_1
 - ▶ These are the elements of $\mathbf{z}_1 = \mathbf{f}(X, y_1, \langle S \rangle, 1) \cdot \Theta$

$$\mathbf{z}_1 = \mathbf{f}(X, y_1, \langle \mathbb{S} \rangle, 1) \cdot \Theta$$

$$= \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \cdot \begin{bmatrix} & \pi & & \\ & & \mathbf{A} & & \\ & & & \\ & & & \mathbf{B} & \end{bmatrix}$$

$$\mathbf{z}_1 = \mathbf{f}(X, y_1, \langle \mathbb{S} \rangle, 1) \cdot \Theta$$
 $= \begin{bmatrix} 1 & & & \\ & \mathbf{A} & & \\ & & \mathbf{B} & \end{bmatrix}$

▶ We know that $y_{i-1} = \langle S \rangle$

$$\mathbf{z}_1 = \mathbf{f}(X, y_1, \langle \mathbb{S} \rangle, 1) \cdot \Theta$$

$$= \left[\begin{array}{c|c} 1 & \mathbf{0} \end{array}\right] \cdot \begin{bmatrix} \frac{\pi}{\mathbf{A}} \\ \mathbf{A} \\ \mathbf{B} \end{bmatrix}$$

▶ We know that y_{i-1} cannot be any other tag

$$\mathbf{z}_1 = \mathbf{f}(X, y_1, \langle \mathrm{S} \rangle, 1) \cdot \Theta$$

$$= \left[\begin{array}{c|c} 1 & \mathbf{0} & \mathbf{1}\{x_1 = o_1\} \end{array}\right] \cdot \begin{bmatrix} \frac{\pi}{\mathbf{A}} & \mathbf{A} \\ \mathbf{B} & \mathbf{B} \end{bmatrix}$$

One-hot vector of the first word

$$\mathbf{z}_1 = \mathbf{f}(X, y_1, \langle \mathbf{S} \rangle, 1) \cdot \Theta$$

$$= \begin{bmatrix} 1 & \mathbf{0} & \mathbf{1}\{x_1 = o_1\} \end{bmatrix} \cdot \begin{bmatrix} \frac{\pi}{\mathbf{A}} \\ \mathbf{B} \end{bmatrix}$$

$$= 1 \cdot \pi + \mathbf{0} \cdot \mathbf{A} + \mathbf{1}\{x_1 = o_1\} \cdot \mathbf{B}$$

$$\mathbf{z}_1 = \mathbf{f}(X, y_1, \langle \mathbf{S} \rangle, 1) \cdot \Theta$$

$$= \begin{bmatrix} 1 & \mathbf{0} & \mathbf{1}\{x_1 = o_1\} \end{bmatrix} \cdot \begin{bmatrix} \frac{\pi}{\mathbf{A}} \\ \mathbf{A} \end{bmatrix}$$

$$= 1 \cdot \pi + \mathbf{0} \cdot \mathbf{A} + \mathbf{1}\{x_1 = o_1\} \cdot \mathbf{B}$$

$$= \pi + \mathbf{B}[o_1]$$

$$\mathbf{z}_{1} = \mathbf{f}(X, y_{1}, \langle \mathbf{S} \rangle, 1) \cdot \Theta$$

$$= \begin{bmatrix} 1 & \mathbf{0} & \mathbf{1}\{x_{1} = o_{1}\} \end{bmatrix} \cdot \begin{bmatrix} \frac{\pi}{\mathbf{A}} & \mathbf{A} \\ \mathbf{B} & \mathbf{B} \end{bmatrix}$$

$$= 1 \cdot \pi + \mathbf{0} \cdot \mathbf{A} + \mathbf{1}\{x_{1} = o_{1}\} \cdot \mathbf{B}$$

$$= \pi + \mathbf{B}[o_{1}]$$

These local scores go into the first column of the Viterbi trellis

▶ We want to compute local scores $z(y_i, y_{i-1})$ for each possible combination of y_i and y_{i-1}

- We want to compute local scores $z(y_i, y_{i-1})$ for each possible combination of y_i and y_{i-1}
 - We can stack the feature vectors for each possible y_{i-1} , $f(X, y_i, y_{i-1}, i)$, on top of each other, in order

- We want to compute local scores $z(y_i, y_{i-1})$ for each possible combination of y_i and y_{i-1}
 - We can stack the feature vectors for each possible y_{i-1} , $\mathbf{f}(X, y_i, y_{i-1}, i)$, on top of each other, in order
 - \triangleright Form a feature matrix \mathbf{F}_i

- We want to compute local scores $z(y_i, y_{i-1})$ for each possible combination of y_i and y_{i-1}
 - We can stack the feature vectors for each possible y_{i-1} , $f(X, y_i, y_{i-1}, i)$, on top of each other, in order
 - \triangleright Form a feature matrix \mathbf{F}_i
 - ► Compute $\mathbf{Z}_i = \mathbf{F}_i \cdot \Theta$

$$\mathbf{Z}_i = \mathbf{F}_i \cdot \Theta$$

$$= \left[\begin{array}{c|c} & & & \\ &$$

$$\mathbf{Z}_i = \mathbf{F}_i \cdot \Theta$$

$$= \begin{bmatrix} \mathbf{0} & & & \\ & \mathbf{A} & & \\ & & \mathbf{B} \end{bmatrix}$$

▶ We know that $y_{i-1} \neq \langle S \rangle$

$$\mathbf{Z}_i = \mathbf{F}_i \cdot \Theta$$

$$= \left[egin{array}{c|c} \mathbf{0} & \mathbf{I} & & & \\ & &$$

► Identity matrix!

$$\mathbf{Z}_i = \mathbf{F}_i \cdot \Theta$$

$$= \left[\begin{array}{c|c} \mathbf{0} & \mathbf{I} & \mathbf{1} \{x_i = o_i\} \end{array} \right] \cdot \left[\begin{array}{c} \pi & \\ \mathbf{A} & \\ \mathbf{B} \end{array} \right]$$

Stack of one-hot vectors

$$\mathbf{Z}_i = \mathbf{F}_i \cdot \Theta$$

$$= \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{1} \{x_i = o_i\} \end{bmatrix} \cdot \begin{bmatrix} \frac{\pi}{\mathbf{A}} \\ \mathbf{B} \end{bmatrix}$$

$$= 0 \cdot \pi + \mathbf{I} \cdot \mathbf{A} + \mathbf{1} \{x_i = o_i\} \cdot \mathbf{B}$$

$$\mathbf{Z}_{i} = \mathbf{F}_{i} \cdot \Theta$$

$$= \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{1} \{x_{i} = o_{i}\} \end{bmatrix} \cdot \begin{bmatrix} \frac{\pi}{\mathbf{A}} \\ \mathbf{A} \end{bmatrix}$$

$$= 0 \cdot \pi + \mathbf{I} \cdot \mathbf{A} + \mathbf{1} \{x_{i} = o_{i}\} \cdot \mathbf{B}$$

$$= \mathbf{A} + \mathbf{B}[o_{i}]$$

$$\mathbf{Z}_{i} = \mathbf{F}_{i} \cdot \Theta$$

$$= \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{1} \{x_{i} = o_{i}\} \end{bmatrix} \cdot \begin{bmatrix} \frac{\pi}{\mathbf{A}} \\ \mathbf{A} \end{bmatrix}$$

$$= 0 \cdot \pi + \mathbf{I} \cdot \mathbf{A} + \mathbf{1} \{x_{i} = o_{i}\} \cdot \mathbf{B}$$

$$= \mathbf{A} + \mathbf{B}[o_{i}]$$

Use the Viterbi algorithm to combine these local scores with scores from the rest of the sequence



▶ Use the Viterbi algorithm to combine these local scores with scores from the rest of the sequence

▶ Use the Viterbi algorithm to combine these local scores with scores from the rest of the sequence

$$\mathbf{Z}_i = \mathbf{F}_i \cdot \Theta = \mathbf{A} + \mathbf{B}[o_i]$$

▶ Use the Viterbi algorithm to combine these local scores with scores from the rest of the sequence

$$\mathbf{Z}_i = \mathbf{F}_i \cdot \Theta = \mathbf{A} + \mathbf{B}[o_i]$$

$$Viterbi[:, i] = max(Viterbi[:, i - 1:i] + \mathbf{A} + \mathbf{B}[o_i], axis=0)$$

► Best last tag is the argmax of the last column of the Viterbi trellis

- Best last tag is the argmax of the last column of the Viterbi trellis
- ► Follow backpointers in the backpointer trellis

- Best last tag is the argmax of the last column of the Viterbi trellis
- ► Follow backpointers in the backpointer trellis
 - Nothing fancy; we'll let you figure it out on your own

- Best last tag is the argmax of the last column of the Viterbi trellis
- ► Follow backpointers in the backpointer trellis
 - Nothing fancy; we'll let you figure it out on your own
 - ► For HW4, you do not have to return the path (log-)probability/score, just the backtrace path

▶ Use the Viterbi algorithm to compute the best tag sequence

- ▶ Use the Viterbi algorithm to compute the best tag sequence
- ▶ If $\hat{Y} = Y$, then do nothing

- Use the Viterbi algorithm to compute the best tag sequence
- ▶ If $\hat{Y} = Y$, then do nothing
- ► Else:
 - Increment weights for features in \hat{Y} , decrement weights for features in \hat{Y}

- Use the Viterbi algorithm to compute the best tag sequence
- ▶ If $\hat{Y} = Y$, then do nothing
- ► Else:
 - Increment weights for features in \hat{Y} , decrement weights for features in \hat{Y}
 - ▶ In other words, for each time step *i*:

- Use the Viterbi algorithm to compute the best tag sequence
- If $\hat{Y} = Y$, then do nothing
- ► Else:
 - Increment weights for features in \hat{Y} , decrement weights for features in \hat{Y}
 - In other words, for each time step *i*:
 - Increment weights for features in y_i , decrement weights for features in \hat{y}_i

- Use the Viterbi algorithm to compute the best tag sequence
- If $\hat{Y} = Y$, then do nothing
- ► Else:
 - Increment weights for features in \hat{Y} , decrement weights for features in \hat{Y}
 - In other words, for each time step *i*:
 - Increment weights for features in y_i, decrement weights for features in ŷ_i
 - Nothing fancy; no Numpy tricks needed