Introduction to Structured Prediction

He He
Slides based on Lecture 09 from David Rosenberg's course materials
(https://github.com/davidrosenberg/mlcourse)

CDS, NYU

March 30, 2021

Example: Part-of-speech (POS) Tagging

- Given a sentence, give a part of speech tag for each word:

x	$\underbrace{\text{[START]}}_{x_0}$	$\underbrace{\text{He}}_{x_1}$	$\underbrace{\text{eats}}_{x_2}$	$\underbrace{\text{apples}}_{x_3}$
y	$\underbrace{\text{[START]}}_{y_0}$	$\underbrace{\text{Pronoun}}_{y_1}$	$\underbrace{\text{Verb}}_{y_2}$	$\underbrace{\text{Noun}}_{y_3}$

- $\mathcal{V} = \{\text{all English words}\} \cup \{\text{[START]}, \text{"."}\}$
- $\mathcal{X} = \mathcal{V}^n$, $n = 1, 2, 3, \ldots$ [Word sequences of any length]
- $\mathcal{P} = \{\text{START}, \text{Pronoun}, \text{Verb}, \text{Noun}, \text{Adjective}\}$
- $\mathcal{Y} = \mathcal{P}^n$, $n = 1, 2, 3, \ldots$[Part of speech sequence of any length]

Multiclass Hypothesis Space

- Discrete output space: $\mathcal{Y}(x)$ $\quad \mathcal{Y} = \{1, \ldots k\}$
    - Very large but has structure, e.g., linear chain (sequence labeling), tree (parsing)
    - Size depends on input x
- Base Hypothesis Space: $\mathcal{H} = \{h : \mathcal{X} \times \mathcal{Y} \to \mathrm{R}\}$
    - $h(x, y)$ gives compatibility score between input x and output y
- Multiclass hypothesis space

$$\mathcal{F} = \left\{ x \mapsto \underset{y \in \mathcal{Y}}{\arg\max}\, h(x, y) \mid h \in \mathcal{H} \right\}$$

- Final prediction function is an $f \in \mathcal{F}$.
- For each $f \in \mathcal{F}$ there is an underlying compatibility score function $h \in \mathcal{H}$.

Structured Prediction

- Part-of-speech tagging

x:	he	eats	apples
y:	pronoun	verb	noun

- Multiclass hypothesis space:

$$h(x,y) = w^T \Psi(x,y) \tag{1}$$

$$\mathcal{F} = \left\{ x \mapsto \arg\max_{y \in \mathcal{Y}(x)} h(x,y) \mid h \in \mathcal{H} \right\} \tag{2}$$

- A special case of multiclass classification
- How to design the feature map Ψ? What are the considerations?

Unary features

- A **unary feature** only depends on
  - the label at a single position, y_i, and x

$f(y_i, x)$

- Example:

$$
\begin{aligned}
\phi_1(x, y_i) &= 1(x_i = \text{runs})1(y_i = \text{Verb}) \\
\phi_2(x, y_i) &= 1(x_i = \text{runs})1(y_i = \text{Noun}) \\
\phi_3(x, y_i) &= 1(x_{i-1} = \text{He})1(x_i = \text{runs})1(\underline{y_i} = \text{Verb})
\end{aligned}
$$

$1[x_{i-2} = \langle s \rangle]$

$1(x_{i-1}$ has suffix ly$)$

Markov features

- A **markov feature** only depends on
  - two adjacent labels, y_{i-1} and y_i, and x
- Example:

$$
\begin{aligned}
\theta_1(x, y_{i-1}, y_i) &= 1(y_{i-1} = \text{Pronoun})1(y_i = \text{Verb}) \\
\theta_2(x, y_{i-1}, y_i) &= 1(y_{i-1} = \text{Pronoun})1(y_i = \text{Noun})
\end{aligned}
$$

- Reminiscent of Markov models in the output space
- Possible to have higher-order features

Local Feature Vector and Compatibility Score

- At each position i in sequence, define the **local feature vector** (unary and markov):

$$\begin{aligned}\Psi_i(x, y_{i-1}, y_i) &= (\phi_1(x, y_i), \phi_2(x, y_i), \ldots, \\ &\quad \theta_1(x, y_{i-1}, y_i), \theta_2(x, y_{i-1}, y_i), \ldots)\end{aligned}$$

- And **local compatibility score** at position i: $\langle w, \Psi_i(x, y_{i-1}, y_i)\rangle$.
- The compatibility score for (x, y) is the sum of local compatibility scores:

$$\sum_i \langle w, \Psi_i(x, y_{i-1}, y_i)\rangle = \left\langle w, \sum_i \Psi_i(x, y_{i-1}, y_i)\right\rangle = \langle w, \Psi(x, y)\rangle, \tag{3}$$

where we define the **sequence feature vector** by

$$\Psi(x, y) = \sum_i \Psi_i(x, y_{i-1}, y_i). \qquad \text{decomposable}$$

Structured perceptron

Given a dataset $\mathcal{D} = \{(x, y)\}$;
Initialize $w \leftarrow 0$;
for $iter = 1, 2, \ldots, T$ **do**
    for $(x, y) \in \mathcal{D}$ **do**
        $\hat{y} = \arg\max_{y' \in \mathcal{Y}(x)} w^T \psi(x, y')$;
        if $\hat{y} \neq y$ **then** `// We've made a mistake`
            $w \leftarrow w + \Psi(x, y)$; `// Move the scorer towards` $\psi(x, y)$
            $w \leftarrow w - \Psi(x, \hat{y})$; `// Move the scorer away from` $\psi(x, \hat{y})$
        end
    end
end

Identical to the multiclass perceptron algorithm except the $\arg\max$ is now over the structured output space $\mathcal{Y}(x)$.

Structured hinge loss

- Recall the generalized hinge loss

$$\ell_{\text{hinge}}(y, \hat{y}) \stackrel{\text{def}}{=} \max_{y' \in \mathcal{Y}(x)} \left(\Delta(y, y') + \left\langle w, \left(\Psi(x, y') - \Psi(x, y) \right) \right\rangle \right) \tag{4}$$

- What is $\Delta(y, y')$ for two sequences? $= \begin{cases} 1 & y \neq y' \\ 0 & y = y' \end{cases}$

- **Hamming loss** is common:

$$\Delta(y, y') = \frac{1}{L} \sum_{i=1}^{L} 1(y_i \neq y_i')$$

  where L is the sequence length.

$$\begin{matrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \end{matrix} \to \Delta(y, y') = 1$$

- Can generalize to the cost-sensitive version using $\delta(y_i, y_i')$

Structured SVM

Exercise:

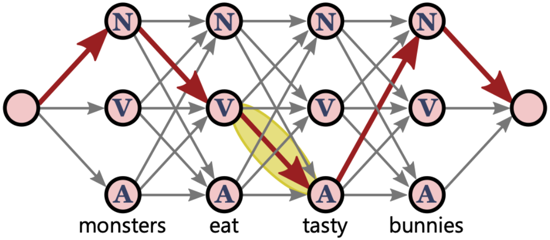
- Write down the objective of structured SVM using the structured hinge loss.
- Stochastic sub-gradient descent for structured SVM (similar to HW3 P3)
- Compare with the structured perceptron algorithm

The argmax problem for sequences [BONUS]

Problem To compute predictions, we need to find $\arg\max_{y \in \mathcal{Y}(x)} \langle w, \Psi(x, y) \rangle$, and $|\mathcal{Y}(x)|$ is exponentially large.

Observation $\Psi(x, y)$ decomposes to $\sum_i \Psi_i(x, y)$. $\rightarrow y_i, y_{i-1}$

Solution Dynamic programming (similar to the Viterbi algorithm)



What's the running time?

Figure by Daumé III. A course in machine learning. Figure 17.1.

The argmax problem in general

Efficient problem-specific algorithms:

problem	structure	algorithm
constituent parsing	binary trees with context-free features	CYK
dependency parsing	spanning trees with edge features	Chu-Liu-Edmonds
image segmentation	2d with adjacent-pixel features	graph cuts

General algorithm:

- Integer linear programming (ILP)

$$\max_z a^T z \quad \text{s.t. linear constraints on } z \tag{5}$$

- z: indicator of substructures, e.g., $\mathbb{I}\{y_i = \text{article and } y_{i+1} = \text{noun}\}$
- constraints: z must correspond to a valid structure

Conclusion

Multiclass algorithms

- Reduce to binary classification, e.g., OvA, AvA, ECCO
  - Good enough for simple multiclass problems
- Generalize binary classification algorithms using multiclass loss
  - Useful for problems with extremely large output space, e.g., structured prediction
  - Related problems: ranking, multi-label classification