Maximum Entropy Models

Pawan Goyal

CSE, IIT Kharagpur

Week 4, Lecture 4

Practice Question

Consider the maximum entropy model for POS tagging, where you want to estimate $P(tag|word)$. In a hypothetical setting, assume that *tag* can take the values *D*, *N* and *V* (short forms for Determiner, Noun and Verb). The variable *word* could be any member of a set V of possible words, where V contains the words *a*, *man*, *sleeps*, as well as additional words. The distribution should give the following probabilities

- $P(D|a) = 0.9$
- $P(N|man) = 0.9$
- $P(V|sleeps) = 0.9$
- $P(D|word) = 0.6$ for any word other than *a*, *man* or *sleeps*
- $P(N|word) = 0.3$ for any word other than *a*, *man* or *sleeps*
- $P(V|word) = 0.1$ for any word other than *a*, *man* or *sleeps*

It is assumed that all other probabilities, not defined above could take any values such that $\sum_{tag} P(tag|word) = 1$ is satisfied for any word in V.

- Define the features of your maximum entropy model that can model this distribution. Mark your features as f_1, f_2 and so on. Each feature should have the same format as explained in the class. [**Hint:** 6 Features should make the analysis easier]
- For each feature f_i, assume a weight λ_i. Now, write expression for the following probabilities in terms of your model parameters
  - $P(D|cat)$
  - $P(N|laughs)$
  - $P(D|man)$
- What value do the parameters in your model take to give the distribution as described above. (i.e. $P(D|a) = 0.9$ and so on. *You may leave the final answer in terms of equations*)

Features for POS Tagging (Ratnaparakhi, 1996)

The specific word and tag context available to a feature is

$$h_i = \{w_i, w_{i+1}, w_{i+2}, w_{i-1}, w_{i-2}, t_{i-1}, t_{i-2}\}$$

Features for POS Tagging (Ratnaparakhi, 1996)

The specific word and tag context available to a feature is

$$h_i = \{w_i, w_{i+1}, w_{i+2}, w_{i-1}, w_{i-2}, t_{i-1}, t_{i-2}\}$$

Example: $f_j(h_i, t_i) = 1$ if $suffix(w_i) = \text{"}ing\text{''} \& t_i = VBG$

Example Features

Condition	Features	
w_i is not rare	$w_i = X$	$\& \ t_i = T$
w_i is rare	X is prefix of w_i, $\|X\| \leq 4$	$\& \ t_i = T$
	X is suffix of w_i, $\|X\| \leq 4$	$\& \ t_i = T$
	w_i contains number	$\& \ t_i = T$
	w_i contains uppercase character	$\& \ t_i = T$
	w_i contains hyphen	$\& \ t_i = T$
$\forall \ w_i$	$t_{i-1} = X$	$\& \ t_i = T$
	$t_{i-2}t_{i-1} = XY$	$\& \ t_i = T$
	$w_{i-1} = X$	$\& \ t_i = T$
	$w_{i-2} = X$	$\& \ t_i = T$
	$w_{i+1} = X$	$\& \ t_i = T$
	$w_{i+2} = X$	$\& \ t_i = T$

Example Features

Word:	the	stories	about	well-heeled	communities	and	developers
Tag:	DT	NNS	IN	JJ	NNS	CC	NNS
Position:	1	2	3	4	5	6	7

Example Features

Word:	the	stories	about	well-heeled	communities	and	developers
Tag:	DT	NNS	IN	JJ	NNS	CC	NNS
Position:	1	2	3	4	5	6	7

$w_i =$ about	$\& \ t_i =$ IN
$w_{i-1} =$ stories	$\& \ t_i =$ IN
$w_{i-2} =$ the	$\& \ t_i =$ IN
$w_{i+1} =$ well-heeled	$\& \ t_i =$ IN
$w_{i+2} =$ communities	$\& \ t_i =$ IN
$t_{i-1} =$ NNS	$\& \ t_i =$ IN
$t_{i-2}t_{i-1} =$ DT NNS	$\& \ t_i =$ IN

$w_{i-1} =$ about	$\& \ t_i =$ JJ
$w_{i-2} =$ stories	$\& \ t_i =$ JJ
$w_{i+1} =$ communities	$\& \ t_i =$ JJ
$w_{i+2} =$ and	$\& \ t_i =$ JJ
$t_{i-1} =$ IN	$\& \ t_i =$ JJ
$t_{i-2}t_{i-1} =$ NNS IN	$\& \ t_i =$ JJ
prefix(w_i)=w	$\& \ t_i =$ JJ
prefix(w_i)=we	$\& \ t_i =$ JJ
prefix(w_i)=wel	$\& \ t_i =$ JJ
prefix(w_i)=well	$\& \ t_i =$ JJ
suffix(w_i)=d	$\& \ t_i =$ JJ
suffix(w_i)=ed	$\& \ t_i =$ JJ
suffix(w_i)=led	$\& \ t_i =$ JJ
suffix(w_i)=eled	$\& \ t_i =$ JJ
w_i contains hyphen	$\& \ t_i =$ JJ

Search Algorithm

Conditional Probability

Given a sentence $\{w_1, \dots, w_n\}$, a tag sequence candidate $\{t_1, \dots, t_n\}$ has conditional probability:

$$P(t_1, \dots, t_n | w_1 \dots, w_n) = \prod_{i=1}^{n} p(t_i | x_i)$$

Search Algorithm

Conditional Probability

Given a sentence $\{w_1, \ldots, w_n\}$, a tag sequence candidate $\{t_1, \ldots, t_n\}$ has conditional probability:

$$P(t_1, \ldots, t_n | w_1 \ldots, w_n) = \prod_{i=1}^{n} p(t_i | x_i)$$

A *Tag Dictionary* is used, which, for each known word, lists the tags that it has appeared with in the training set.

Search Algorithm

Let $W = \{w_1, \ldots, w_n\}$ be a test sentence, s_{ij} be the jth highest probability tag sequence up to and including word w_i.

Search Algorithm

Let $W = \{w_1, \ldots, w_n\}$ be a test sentence, s_{ij} be the jth highest probability tag sequence up to and including word w_i.

Search description

- Generate tags for w_1, find top N, set $s_{1j}, 1 \le j \le N$, accordingly.

Search Algorithm

Let $W = \{w_1, \ldots, w_n\}$ be a test sentence, s_{ij} be the jth highest probability tag sequence up to and including word w_i.

Search description

- Generate tags for w_1, find top N, set $s_{1j}, 1 \leq j \leq N$, accordingly.
- Initialize $i = 2$
  - Initialize $j = 1$
  - Generate tags for w_i, given $s_{(i-1)j}$ as previous tag context, and append each tag to $s_{(i-1)j}$ to make a new sequence
  - $j = j + 1$, repeat if $j \leq N$

Search Algorithm

Let $W = \{w_1, \ldots, w_n\}$ be a test sentence, s_{ij} be the jth highest probability tag sequence up to and including word w_i.

Search description

- Generate tags for w_1, find top N, set $s_{1j}, 1 \le j \le N$, accordingly.
- Initialize $i = 2$
  - Initialize $j = 1$
  - Generate tags for w_i, given $s_{(i-1)j}$ as previous tag context, and append each tag to $s_{(i-1)j}$ to make a new sequence
  - $j = j + 1$, repeat if $j \le N$
- Find N highest probability sequences generated by above loop, set s_{ij} accordingly

Search Algorithm

Let $W = \{w_1, \ldots, w_n\}$ be a test sentence, s_{ij} be the jth highest probability tag sequence up to and including word w_i.

Search description

- Generate tags for w_1, find top N, set $s_{1j}, 1 \leq j \leq N$, accordingly.
- Initialize $i = 2$
    - Initialize $j = 1$
    - Generate tags for w_i, given $s_{(i-1)j}$ as previous tag context, and append each tag to $s_{(i-1)j}$ to make a new sequence
    - $j = j + 1$, repeat if $j \leq N$
- Find N highest probability sequences generated by above loop, set s_{ij} accordingly
- $i = i + 1$, repeat if $i \leq n$

Search Algorithm

Let $W = \{w_1, \ldots, w_n\}$ be a test sentence, s_{ij} be the jth highest probability tag sequence up to and including word w_i.

Search description

- Generate tags for w_1, find top N, set $s_{1j}, 1 \leq j \leq N$, accordingly.
- Initialize $i = 2$
  - Initialize $j = 1$
  - Generate tags for w_i, given $s_{(i-1)j}$ as previous tag context, and append each tag to $s_{(i-1)j}$ to make a new sequence
  - $j = j + 1$, repeat if $j \leq N$
- Find N highest probability sequences generated by above loop, set s_{ij} accordingly
- $i = i + 1$, repeat if $i \leq n$
- Return highest probability sequence s_{n1}

A Good Reference

Berger et al., *A Maximum Entropy Approach to Natural Language Processing*, Computational Linguistics, Vol. 22, No. 1.