Maximum Entropy Markov Models (log-linear model for tagging)

Instructor: Wei Xu

Many slides from Michael Collins

Part-of-Speech Tagging

INPUT:

Profits soared at Boeing Co., easily topping forecasts on Wall Street, as their CEO Alan Mulally announced first quarter results.

OUTPUT:

Profits/N soared/V at/P Boeing/N Co./N ,/, easily/ADV topping/V forecasts/N on/P Wall/N Street/N ,/, as/P their/POSS CEO/N Alan/N Mulally/N announced/V first/ADJ quarter/N results/N ./.

N	=	Noun
V	=	Verb
P	=	Preposition
Adv	=	Adverb
Adj	=	Adjective
...		

Named Entity Recognition

INPUT: Profits soared at Boeing Co., easily topping forecasts on Wall Street, as their CEO Alan Mulally announced first quarter results.

OUTPUT: Profits soared at [Company Boeing Co.], easily topping forecasts on [Location Wall Street], as their CEO [Person Alan Mulally] announced first quarter results.

Named Entity Extraction as Tagging

INPUT:

Profits soared at Boeing Co., easily topping forecasts on Wall Street, as their CEO Alan Mulally announced first quarter results.

OUTPUT:

Profits/NA soared/NA at/NA Boeing/SC Co./CC ,/NA easily/NA topping/NA forecasts/NA on/NA Wall/SL Street/CL ,/NA as/NA their/NA CEO/NA Alan/SP Mulally/CP announced/NA first/NA quarter/NA results/NA ./NA

NA	=	No entity
SC	=	Start Company
CC	=	Continue Company
SL	=	Start Location
CL	=	Continue Location

Our Goal

Training set:
1 Pierre/NNP Vinken/NNP ,/, 61/CD years/NNS old/JJ ,/, will/MD join/VB the/DT board/NN as/IN a/DT nonexecutive/JJ director/NN Nov./NNP 29/CD ./.
2 Mr./NNP Vinken/NNP is/VBZ chairman/NN of/IN Elsevier/NNP N.V./NNP ,/, the/DT Dutch/NNP publishing/VBG group/NN ./.
3 Rudolph/NNP Agnew/NNP ,/, 55/CD years/NNS old/JJ and/CC chairman/NN of/IN Consolidated/NNP Gold/NNP Fields/NNP PLC/NNP ,/, was/VBD named/VBN a/DT nonexecutive/JJ director/NN of/IN this/DT British/JJ industrial/JJ conglomerate/NN ./.
...
38,219 It/PRP is/VBZ also/RB pulling/VBG 20/CD people/NNS out/IN of/IN Puerto/NNP Rico/NNP ,/, who/WP were/VBD helping/VBG Huricane/NNP Hugo/NNP victims/NNS ,/, and/CC sending/VBG them/PRP to/TO San/NNP Francisco/NNP instead/RB ./.

- From the training set, induce a function/algorithm that maps new sentences to their tag sequences.

Overview

- Recap: The Tagging Problem
- Log-linear taggers

Tagging (Sequence Labeling)

- Given a sequence (in NLP, words), assign appropriate labels to each word.
- Many NLP problems can be viewed as sequence labeling:
  - POS Tagging
  - Chunking
  - Named Entity Tagging
- Labels of tokens are dependent on the labels of other tokens in the sequence, particularly their neighbors

Plays	well	with	others.
VBZ	RB	IN	NNS

Log-Linear Models for Tagging

- We have an input sentence $w_{[1:n]} = w_1, w_2, \ldots, w_n$
  (w_i is the i'th word in the sentence)

Log-Linear Models for Tagging

- We have an input sentence $w_{[1:n]} = w_1, w_2, \ldots, w_n$
  (w_i is the i'th word in the sentence)

- We have a tag sequence $t_{[1:n]} = t_1, t_2, \ldots, t_n$
  (t_i is the i'th tag in the sentence)

Log-Linear Models for Tagging

- We have an input sentence $w_{[1:n]} = w_1, w_2, \ldots, w_n$
  (w_i is the i'th word in the sentence)

- We have a tag sequence $t_{[1:n]} = t_1, t_2, \ldots, t_n$
  (t_i is the i'th tag in the sentence)

- We'll use an log-linear model to define

  $$p(t_1, t_2, \ldots, t_n | w_1, w_2, \ldots, w_n)$$

  for any sentence $w_{[1:n]}$ and tag sequence $t_{[1:n]}$ of the same length.
  (Note: contrast with HMM that defines $p(t_1 \ldots t_n, w_1 \ldots w_n)$)

Log-Linear Models for Tagging

- We have an input sentence $w_{[1:n]} = w_1, w_2, \ldots, w_n$
  (w_i is the i'th word in the sentence)

- We have a tag sequence $t_{[1:n]} = t_1, t_2, \ldots, t_n$
  (t_i is the i'th tag in the sentence)

- We'll use an log-linear model to define

$$p(t_1, t_2, \ldots, t_n | w_1, w_2, \ldots, w_n)$$

  for any sentence $w_{[1:n]}$ and tag sequence $t_{[1:n]}$ of the same length.
  (Note: contrast with HMM that defines $p(t_1 \ldots t_n, w_1 \ldots w_n)$)

- Then the most likely tag sequence for $w_{[1:n]}$ is

$$t^*_{[1:n]} = \mathrm{argmax}_{t_{[1:n]}} p(t_{[1:n]} | w_{[1:n]})$$

How to model $p(t_{[1:n]}|w_{[1:n]})$?

A Trigram Log-Linear Tagger:

$p(t_{[1:n]}|w_{[1:n]}) = \prod_{j=1}^{n} p(t_j \mid w_1 \ldots w_n, t_1 \ldots t_{j-1})$ Chain rule

How to model $p(t_{[1:n]}|w_{[1:n]})$?

A Trigram Log-Linear Tagger:

$$p(t_{[1:n]}|w_{[1:n]}) = \prod_{j=1}^{n} p(t_j \mid w_1 \ldots w_n, t_1 \ldots t_{j-1})$$ Chain rule

$$= \prod_{j=1}^{n} p(t_j \mid w_1, \ldots, w_n, t_{j-2}, t_{j-1})$$

Independence assumptions

- We take $t_0 = t_{-1} = *$

How to model $p(t_{[1:n]}|w_{[1:n]})$?

A Trigram Log-Linear Tagger:

$$p(t_{[1:n]}|w_{[1:n]}) = \prod_{j=1}^{n} p(t_j \mid w_1 \ldots w_n, t_1 \ldots t_{j-1})$$ Chain rule

$$= \prod_{j=1}^{n} p(t_j \mid w_1, \ldots, w_n, t_{j-2}, t_{j-1})$$

Independence assumptions

- We take $t_0 = t_{-1} = *$
- Independence assumption: each tag only depends on previous two tags

$$p(t_j|w_1, \ldots, w_n, t_1, \ldots, t_{j-1}) = p(t_j|w_1, \ldots, w_n, t_{j-2}, t_{j-1})$$

An Example

Hispaniola/NNP quickly/RB became/VB an/DT important/JJ base/?? from which Spain expanded its empire into the rest of the Western Hemisphere .

- There are many possible tags in the position ??

$\mathcal{Y} = \{\text{NN, NNS, Vt, Vi, IN, DT, } \ldots\}$

Representation: Histories

- A **history** is a 4-tuple $\langle t_{-2}, t_{-1}, w_{[1:n]}, i \rangle$
- t_{-2}, t_{-1} are the previous two tags.
- $w_{[1:n]}$ are the n words in the input sentence.
- i is the index of the word being tagged
- $\mathcal{X}$ is the set of all possible histories

Hispaniola/NNP quickly/RB became/VB an/DT important/JJ base/?? from which Spain expanded its empire into the rest of the Western Hemisphere .

- $t_{-2}, t_{-1} =$ DT, JJ
- $w_{[1:n]} = \langle Hispaniola, quickly, became, \ldots, Hemisphere, . \rangle$
- $i = 6$

An Example (continued)

- $\mathcal{X}$ is the set of all possible histories of form $\langle t_{-2}, t_{-1}, w_{[1:n]}, i \rangle$
- $\mathcal{Y} = \{\text{NN, NNS, Vt, Vi, IN, DT}, \ldots\}$
- We have m features $f_k : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ for $k = 1 \ldots m$

For example:

$$f_1(h, t) = \begin{cases} 1 & \text{if current word } w_i \text{ is base and } t = \texttt{Vt} \\ 0 & \text{otherwise} \end{cases}$$

analogy to e(base|Vt) in HMMs

$$f_2(h, t) = \begin{cases} 1 & \text{if current word } w_i \text{ ends in } \texttt{ing} \text{ and } t = \texttt{VBG} \\ 0 & \text{otherwise} \end{cases}$$

difficult for HMMs

$$\ldots$$

$f_1(\langle \text{JJ, DT}, \langle \text{ Hispaniola}, \ldots \rangle, 6 \rangle, \text{Vt}) = 1$
$f_2(\langle \text{JJ, DT}, \langle \text{ Hispaniola}, \ldots \rangle, 6 \rangle, \text{Vt}) = 0$
$\ldots$

Training the Log-Linear Model

- To train a log-linear model, we need a training set (x_i, y_i) for $i = 1 \ldots n$. Then search for

$$v^* = \text{argmax}_v \left(\underbrace{\sum_i \log p(y_i | x_i; v)}_{Log-Likelihood} - \underbrace{\frac{\lambda}{2} \sum_k v_k^2}_{Regularizer} \right)$$

- Training set is simply all history/tag pairs seen in the training data

The Viterbi Algorithm

Problem: for an input $w_1 \ldots w_n$, find

$$\arg\max_{t_1 \ldots t_n} p(t_1 \ldots t_n \mid w_1 \ldots w_n)$$

We assume that p takes the form

$$p(t_1 \ldots t_n \mid w_1 \ldots w_n) = \prod_{i=1}^{n} q(t_i | t_{i-2}, t_{i-1}, w_{[1:n]}, i)$$

(In our case $q(t_i | t_{i-2}, t_{i-1}, w_{[1:n]}, i)$ is the estimate from a log-linear model.)

The Viterbi Algorithm

- Define n to be the length of the sentence
- Define

$$r(t_1 \dots t_k) = \prod_{i=1}^{k} q(t_i | t_{i-2}, t_{i-1}, w_{[1:n]}, i)$$

- Define a dynamic programming table

$$\pi(k, u, v) = \text{maximum probability of a tag sequence ending in tags } u, v \text{ at position } k$$

that is,

$$\pi(k, u, v) = \max_{\langle t_1, \dots, t_{k-2} \rangle} r(t_1 \dots t_{k-2}, u, v)$$

A Recursive Definition

Base case:

$$\pi(0, *, *) = 1$$

Recursive definition:
For any $k \in \{1 \ldots n\}$, for any $u \in \mathcal{S}_{k-1}$ and $v \in \mathcal{S}_k$:

$$\pi(k, u, v) = \max_{t \in \mathcal{S}_{k-2}} \left(\pi(k-1, t, u) \times q(v|t, u, w_{[1:n]}, k) \right)$$

where $\mathcal{S}_k$ is the set of possible tags at position k

The Viterbi Algorithm with Backpointers

Input: a sentence $w_1 \ldots w_n$, log-linear model that provides $q(v|t, u, w_{[1:n]}, i)$ for any tag-trigram t, u, v, for any $i \in \{1 \ldots n\}$
Initialization: Set $\pi(0, *, *) = 1$.
Algorithm:

- For $k = 1 \ldots n$,
  - For $u \in \mathcal{S}_{k-1}$, $v \in \mathcal{S}_k$,

$$
\begin{aligned}
\pi(k, u, v) &= \max_{t \in \mathcal{S}_{k-2}} \left(\pi(k-1, t, u) \times q(v|t, u, w_{[1:n]}, k)\right) \\
bp(k, u, v) &= \arg\max_{t \in \mathcal{S}_{k-2}} \left(\pi(k-1, t, u) \times q(v|t, u, w_{[1:n]}, k)\right)
\end{aligned}
$$

- Set $(t_{n-1}, t_n) = \arg\max_{(u,v)} \pi(n, u, v)$
- For $k = (n-2) \ldots 1$, $t_k = bp(k+2, t_{k+1}, t_{k+2})$
- **Return** the tag sequence $t_1 \ldots t_n$

FAQ Segmentation: McCallum et. al

- McCallum et. al compared HMM and log-linear taggers on a *FAQ Segmentation* task

- Main point: in an HMM, modeling

$$p(word|tag)$$

  is difficult in this domain

FAQ Segmentation: McCallum et. al

```
    <head>X-NNTP-POSTER: NewsHound v1.33
    <head>
    <head>Archive name: acorn/faq/part2
    <head>Frequency: monthly
    <head>
<question>2.6) What configuration of serial cable should I use
  <answer>
  <answer>   Here follows a diagram of the necessary connections
  <answer>programs to work properly. They are as far as I know t
  <answer>agreed upon by commercial comms software developers fo
  <answer>
  <answer>   Pins 1, 4, and 8 must be connected together inside
  <answer>is to avoid the well known serial port chip bugs. The
```

FAQ Segmentation: Line Features

```
begins-with-number
begins-with-ordinal
begins-with-punctuation
begins-with-question-word
begins-with-subject
blank
contains-alphanum
contains-bracketed-number
contains-http
contains-non-space
contains-number
contains-pipe
contains-question-mark
ends-with-question-mark
first-alpha-is-capitalized
indented-1-to-4
```

FAQ Segmentation: The Log-Linear Tagger

```
    <head>X-NNTP-POSTER: NewsHound v1.33
    <head>
    <head>Archive name: acorn/faq/part2
    <head>Frequency: monthly
    <head>
<question>2.6) What configuration of serial cable should I use

            Here follows a diagram of the necessary connections
```

$\Rightarrow$ "tag=question;prev=head;begins-with-number"
"tag=question;prev=head;contains-alphanum"
"tag=question;prev=head;contains-nonspace"
"tag=question;prev=head;contains-number"
"tag=question;prev=head;prev-is-blank"

FAQ Segmentation: An HMM Tagger

```
<question>2.6) What configuration of serial cable should I use
```

- First solution for $p(word \mid tag)$:

$p($"2.6) What configuration of serial cable should I use" $\mid$ question$) =$
$e(\text{ 2.6)} \mid \text{question})\times$
$e(What \mid \text{question})\times$
$e(configuration \mid \text{question})\times$
$e(of \mid \text{question})\times$
$e(serial \mid \text{question})\times$
$\ldots$

- i.e. have a **language model** for each tag

FAQ Segmentation: McCallum et. al

- Second solution: first map each sentence to string of features:

  `<question>2.6) What configuration of serial cable should I use`

  $\Rightarrow$

  `<question>begins-with-number contains-alphanum contains-nonspace contains-number prev-is-blank`

- Use a language model again:

$p($"2.6) What configuration of serial cable should I use" $|$ question$) =$
$e($begins-with-number $|$ question$)\times$
$e($contains-alphanum $|$ question$)\times$
$e($contains-nonspace $|$ question$)\times$
$e($contains-number $|$ question$)\times$
$e($prev-is-blank $|$ question$)\times$

FAQ Segmentation: Results

Method	Precision	Recall
ME-Stateless	0.038	0.362
TokenHMM	0.276	0.140
FeatureHMM	0.413	0.529
MEMM	0.867	0.681

- Precision and recall results are for recovering segments

FAQ Segmentation: Results

Method	Precision	Recall
ME-Stateless	0.038	0.362
TokenHMM	0.276	0.140
FeatureHMM	0.413	0.529
MEMM	0.867	0.681

- Precision and recall results are for recovering segments
- ME-stateless is a log-linear model that treats every sentence seperately (no dependence between adjacent tags)

FAQ Segmentation: Results

Method	Precision	Recall
ME-Stateless	0.038	0.362
TokenHMM	0.276	0.140
FeatureHMM	0.413	0.529
MEMM	0.867	0.681

- Precision and recall results are for recovering segments
- ME-stateless is a log-linear model that treats every sentence seperately (no dependence between adjacent tags)
- TokenHMM is an HMM with first solution we've just seen

FAQ Segmentation: Results

Method	Precision	Recall
ME-Stateless	0.038	0.362
TokenHMM	0.276	0.140
FeatureHMM	0.413	0.529
MEMM	0.867	0.681

- Precision and recall results are for recovering segments
- ME-stateless is a log-linear model that treats every sentence seperately (no dependence between adjacent tags)
- TokenHMM is an HMM with first solution we've just seen
- FeatureHMM is an HMM with second solution we've just seen

FAQ Segmentation: Results

Method	Precision	Recall
ME-Stateless	0.038	0.362
TokenHMM	0.276	0.140
FeatureHMM	0.413	0.529
MEMM	0.867	0.681

- Precision and recall results are for recovering segments
- ME-stateless is a log-linear model that treats every sentence seperately (no dependence between adjacent tags)
- TokenHMM is an HMM with first solution we've just seen
- FeatureHMM is an HMM with second solution we've just seen
- MEMM is a log-linear trigram tagger (MEMM stands for "Maximum-Entropy Markov Model")

Summary

- Key ideas in log-linear taggers:
  - Decompose
    $$p(t_1 \dots t_n | w_1 \dots w_n) = \prod_{i=1}^{n} p(t_i | t_{i-2}, t_{i-1}, w_1 \dots w_n)$$
  - Estimate
    $$p(t_i | t_{i-2}, t_{i-1}, w_1 \dots w_n)$$
    using a log-linear model
  - For a test sentence $w_1 \dots w_n$, use the Viterbi algorithm to find
    $$\arg\max_{t_1 \dots t_n} \left(\prod_{i=1}^{n} p(t_i | t_{i-2}, t_{i-1}, w_1 \dots w_n) \right)$$
- Key advantage over HMM taggers: flexibility in the features they can use