Part-of-Speech Tagging & Sequence Labeling

Hongning Wang
CS@UVa

What is POS tagging

Tag Set

NNP: proper noun
CD: numeral
JJ: adjective

POS Tagger

Raw Text

Pierre Vinken , 61 years old, will join the board as a nonexecutive director Nov. 29 .

Tagged Text

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 ._.

Why POS tagging?

- POS tagging is a prerequisite for further NLP analysis
  - Syntax parsing
    - Basic unit for parsing
  - Information extraction
    - Indication of names, relations
  - Machine translation
    - The meaning of a particular word depends on its POS tag
  - Sentiment analysis
    - Adjectives are the major opinion holders
      - Good v.s. Bad, Excellent v.s. Terrible

Challenges in POS tagging

- Words often have more than one POS tag
  - The back door (adjective)
  - On my back (noun)
  - Promised to back the bill (verb)
- Simple solution with dictionary look-up does not work in practice
  - One needs to determine the POS tag for a particular instance of a word from its context

Define a tagset

- We have to agree on a standard inventory of word classes
  - Taggers are trained on a labeled corpora
  - The tagset needs to capture semantically or syntactically important distinctions that can easily be made by trained human annotators

Word classes

- Open classes
  - Nouns, verbs, adjectives, adverbs
- Closed classes
  - Auxiliaries and modal verbs
  - Prepositions, Conjunctions
  - Pronouns, Determiners
  - Particles, Numerals

Public tagsets in NLP

- Brown corpus - Francis and Kucera 1961
  - 500 samples, distributed across 15 genres in rough proportion to the amount published in 1961 in each of those genres
  - 87 tags
- Penn Treebank - Marcus et al. 1993
  - Hand-annotated corpus of Wall Street Journal, 1M words
  - 45 tags, a simplified version of Brown tag set
  - Standard for English now
    - Most statistical POS taggers are trained on this Tagset

How much ambiguity is there?

- Statistics of word-tag pair in Brown Corpus and Penn Treebank

		87-tag Original Brown		45-tag Treebank Brown	
Unambiguous (1 tag)		**44,019**	**11%**	**38,857**	**18%**
Ambiguous (2–7 tags)		**5,490**		**8844**	
Details:	2 tags	4,967		6,731	
	3 tags	411		1621	
	4 tags	91		357	
	5 tags	17		90	
	6 tags	2	(*well*, *beat*)	32	
	7 tags	2	(*still*, *down*)	6	(*well*, *set*, *round*, *open*, *fit*, *down*)
	8 tags			4	(*'s*, *half*, *back*, *a*)
	9 tags			3	(*that*, *more*, *in*)

Is POS tagging a solved problem?

- Baseline
  - Tag every word with its most frequent tag
  - Tag unknown words as nouns
  - Accuracy
    - Word level: 90%
    - Sentence level
      - Average English sentence length 14.3 words
      - $0.9^{14.3} = 22\%$

Accuracy of State-of-the-art POS Tagger
- *Word level: 97%*
- *Sentence level:* $0.97^{14.3} = 65\%$

Building a POS tagger

- Rule-based solution
  1. Take a dictionary that lists all possible tags for each word
  2. Assign to every word all its possible tags
  3. Apply rules that eliminate impossible/unlikely tag sequences, leaving only one tag per word

she PRP
promised ~~VBN~~,VBD
to TO
back VB, JJ, RB, NN!!
the DT
bill NN, ~~VB~~

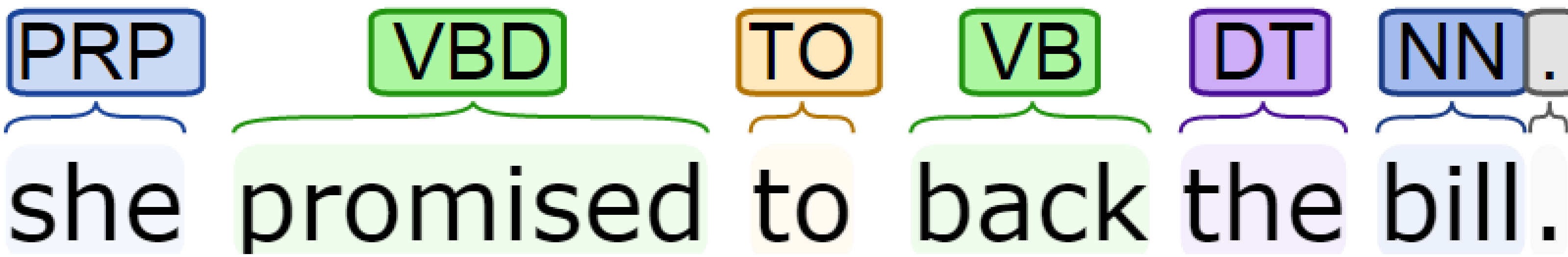
R1: Pronoun should be followed by a past tense verb

R2: Verb cannot follow determiner

Building a POS tagger

- Statistical POS tagging

$$\boldsymbol{t} = \quad t^1 \quad t^2 \quad t^3 \quad t^4 \quad t^5 \quad t^6$$



$$\boldsymbol{w} = \quad w^1 \quad w^2 \quad w^3 \quad w^4 \quad w^5 \quad w^6$$

  – What is the most likely sequence of tags $\boldsymbol{t}$ for the given sequence of words $\boldsymbol{w}$

$$\boldsymbol{t}^* = argmax_{\boldsymbol{t}} p(\boldsymbol{t}|\boldsymbol{w})$$

POS tagging with generative models

- Bayes Rule

$$t^* = argmax_{\boldsymbol{t}} p(\boldsymbol{t}|\boldsymbol{w})$$
$$= argmax_{\boldsymbol{t}} \frac{p(\boldsymbol{w}|\boldsymbol{t})p(\boldsymbol{t})}{p(\boldsymbol{w})}$$
$$= argmax_{\boldsymbol{t}} \boxed{p(\boldsymbol{w}|\boldsymbol{t})}\boxed{p(\boldsymbol{t})}$$

  – Joint distribution of tags and words

  – Generative model

    - A stochastic process that first generates the tags, and then generates the words based on these tags

Hidden Markov models

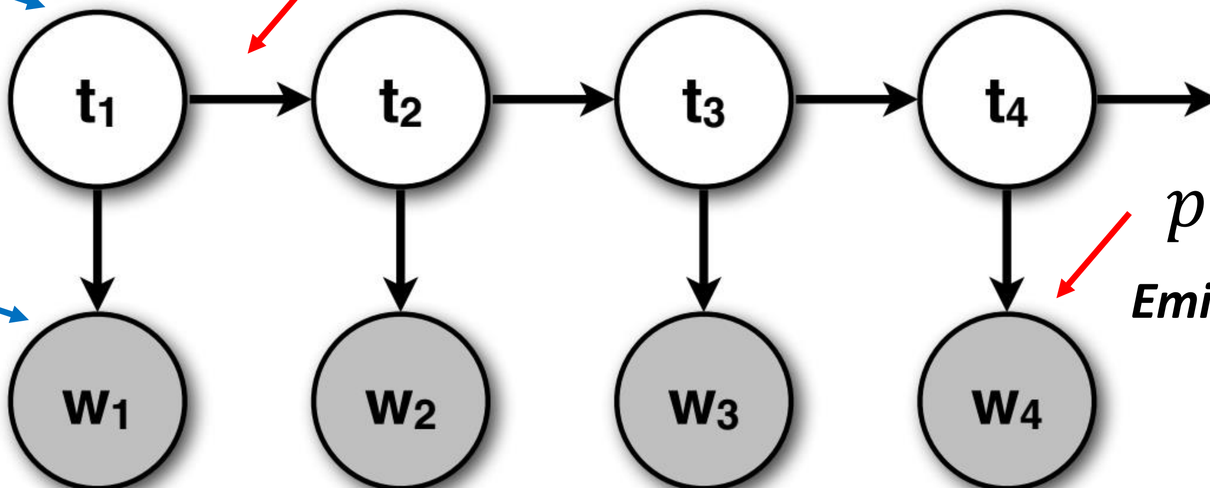
- Two assumptions for POS tagging
  1. Current tag only depends on previous k tags
     - $p(\boldsymbol{t}) = \prod_i p(t_i | t_{i-1}, t_{i-2}, \dots, t_{i-k})$
     - When k=1, it is so-called first-order HMMs
  2. Each word in the sequence depends only on its corresponding tag
     - $p(\boldsymbol{w} | \boldsymbol{t}) = \prod_i p(w_i | t_i)$

Graphical representation of HMMs

$p(t_i|t_{i-1})$ ***Transition probability***

All the tags in the tagset

All the words in the vocabulary



$p(w_i|t_i)$

Emission probability

- Light circle: latent random variables
- Dark circle: observed random variables
- Arrow: probabilistic dependency

Finding the most probable tag sequence

$$\boldsymbol{t}^* = argmax_{\boldsymbol{t}} p(\boldsymbol{t}|\boldsymbol{w})$$

$$= argmax_{\boldsymbol{t}} \prod_i \boxed{p(w_i|t_i)p(t_i|t_{i-1})}$$

- Complexity analysis
  - Each word can have up to T tags
  - For a sentence with N words, there will be up to T^N possible tag sequences
  - Key: explore the special structure in HMMs!

Trellis: a special structure for HMMs

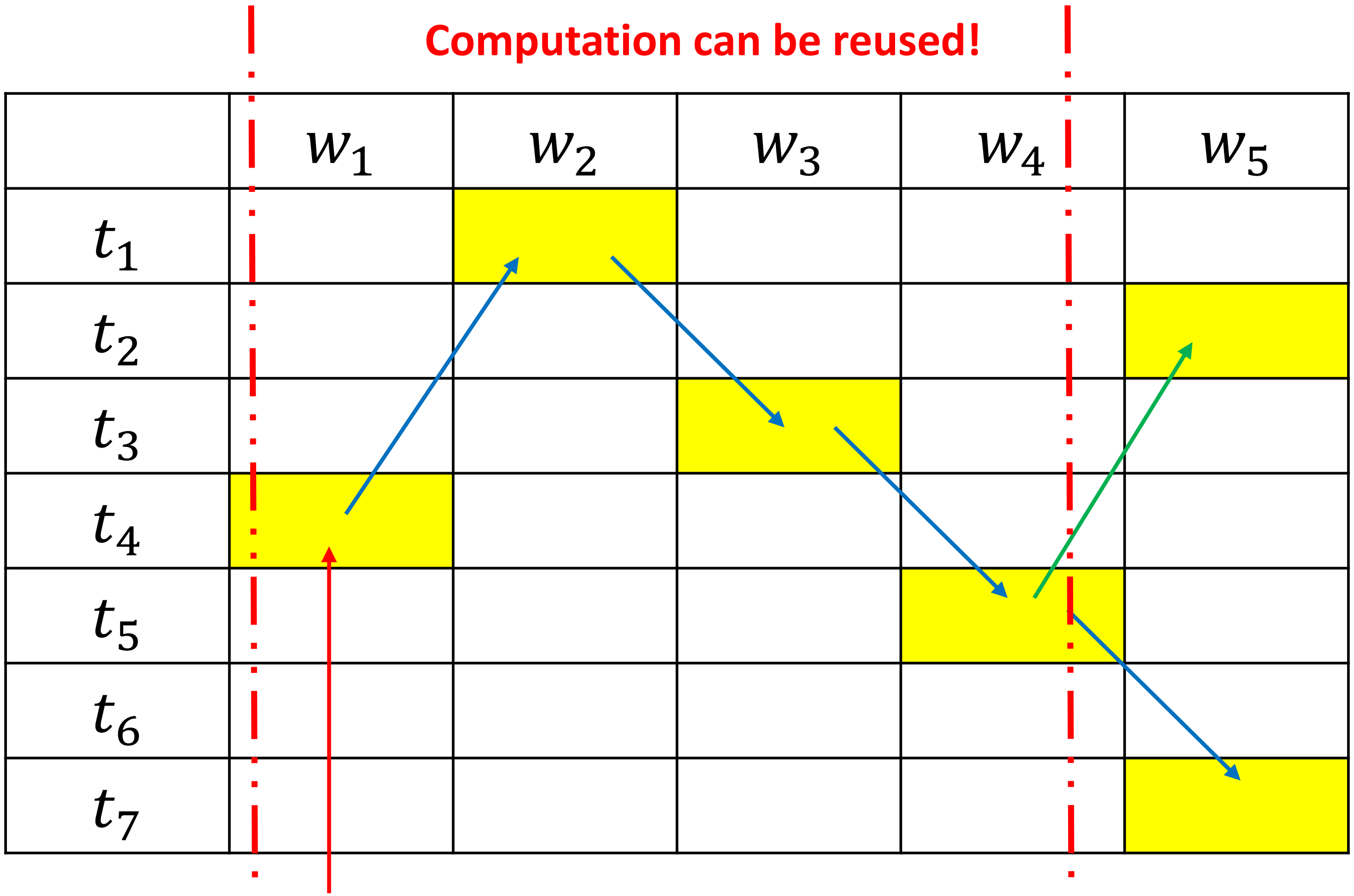
	w_1	w_2	w_3	w_4	w_5
t_1					
t_2					
t_3					
t_4					
t_5					
t_6					
t_7					

Word w_1 takes tag t_4.

Trellis: a special structure for HMMs

$\boldsymbol{t}^{\mathbf{1}} = \underline{t_4 t_1 t_3 t_5} \boxed{t_7}$ $\qquad$ $\boldsymbol{t}^{\mathbf{2}} = \underline{t_4 t_1 t_3 t_5} \boxed{t_2}$

Computation can be reused!



Word w_1 takes tag t_4.

Viterbi algorithm

- Store the best tag sequence for $w_1 \dots w_i$ that ends in t^j in T[j][i]
  - $T[j][i] = \max p(w_1 \dots w_i, t_1 \dots, t_i = t^j)$
- Recursively compute trellis[j][i] from the entries in the previous column trellis[j][i-1]
  - $T[j][i] = P(w_i|t^j) Max_k \left(T[k][i-1] P(t^j|t_k)\right)$

Generating the current observation

The best i-1 tag sequence

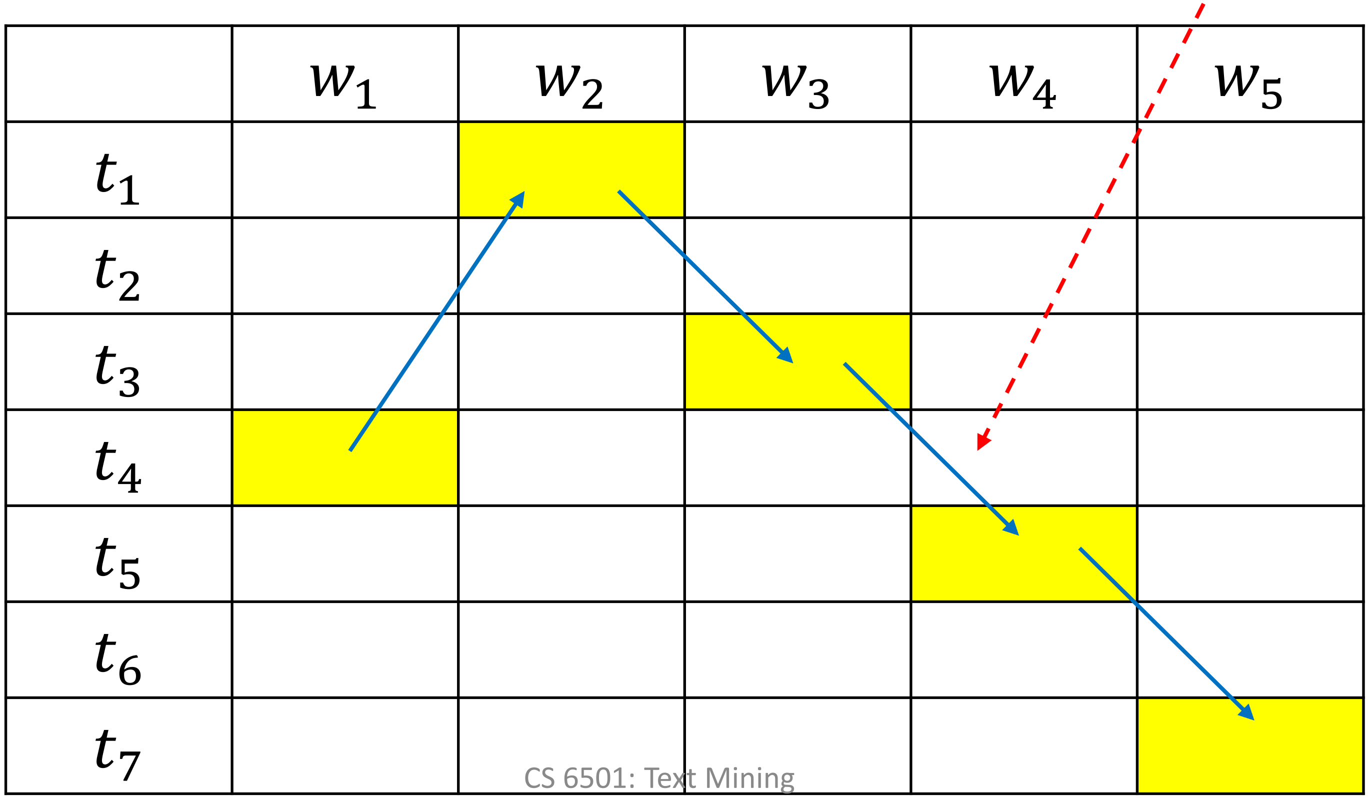
Transition from the previous best ending tag

Dynamic programming: $O(TN)$!

Decode $argmax_{\boldsymbol{t}} p(\boldsymbol{t}, \boldsymbol{w})$

- Take the highest scoring entry in the last column of the trellis

Keep backpointers in each trellis to keep track of the most probable sequence



Train an HMMs tagger

- Parameters in an HMMs tagger
  - Transition probability: $p(t_i|t_j), T \times T$
  - Emission probability: $p(w|t), V \times T$
  - Initial state probability: $p(t|\pi), T \times 1$

For the first tag in a sentence

Train an HMMs tagger

- Maximum likelihood estimator
  - Given a labeled corpus, e.g., Penn Treebank
  - Count how often we have the pair of $t_i t_j$ and $w_i t_j$
    - $p(t_i|t_j) = \frac{c(t_i,t_j)}{c(t_j)}$
    - $p(w_i|t_j) = \frac{c(w_i,c_j)}{c(t_j)}$

Public POS taggers

- Brill's tagger
  - http://www.cs.jhu.edu/~brill/
- TnT tagger
  - http://www.coli.uni-saarland.de/~thorsten/tnt/
- Stanford tagger
  - http://nlp.stanford.edu/software/tagger.shtml
- SVMTool
  - http://www.lsi.upc.es/~nlp/SVMTool/
- GENIA tagger
  - http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/tagger/
- More complete list at
  - http://www-nlp.stanford.edu/links/statnlp.html#Taggers

Let’s take a look at other NLP tasks

- Noun phrase (NP) chunking
  - Task: identify all non-recursive NP chunks

```
Pierre Vinken , 61 years old , will join IBM 's board
as a nonexecutive director Nov. 29 .
```

```
[NP Pierre Vinken] , [NP 61 years] old , will join
[NP IBM] 's [NP board] as [NP a nonexecutive director]
[NP Nov. 2] .
```

The BIO encoding

- Define three new tags
  - B-NP: beginning of a noun phrase chunk
  - I-NP: inside of a noun phrase chunk
  - O: outside of a noun phrase chunk

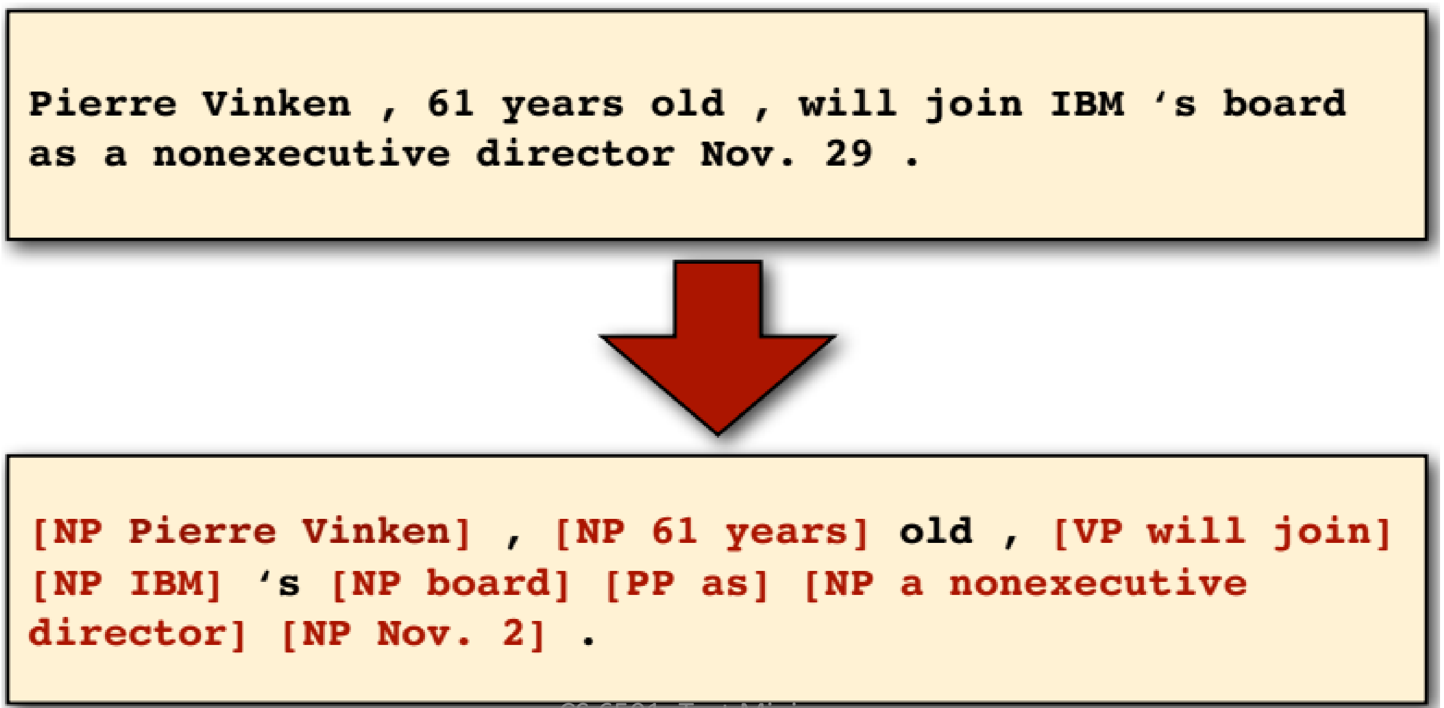
```
[NP Pierre Vinken] , [NP 61 years] old , will join
[NP IBM] 's [NP board] as [NP a nonexecutive director]
[NP Nov. 2] .
```

POS Tagging with restricted Tagset?

```
Pierre_B-NP Vinken_I-NP ,_O 61_B-NP years_I-NP
old_O ,_O will_O join_O IBM_B-NP 's_O board_B-NP as_O
a_B-NP nonexecutive_I-NP director_I-NP Nov._B-NP
29_I-NP ._O
```

Another NLP task

- Shallow parsing
  - Task: identify all non-recursive NP, verb (“VP”) and preposition (“PP”) chunks



BIO Encoding for Shallow Parsing

- Define several new tags
  - B-NP B-VP B-PP: beginning of an NP, "VP", "PP" chunk
  - I-NP: inside of an NP, "VP", "PP" chunk
  - O: outside of any chunk

```
[NP Pierre Vinken] , [NP 61 years] old , [VP will join]
[NP IBM] 's [NP board] [PP as] [NP a nonexecutive
director] [NP Nov. 2] .
```

POS Tagging with restricted Tagset?

```
Pierre_B-NP Vinken_I-NP ,_O 61_B-NP years_I-NP
old_O ,_O will_B-VP join_I-VP IBM_B-NP 's_O board_B-NP
as_B-PP a_B-NP nonexecutive_I-NP director_I-NP Nov._B-
NP 29_I-NP ._O
```

Yet Another NLP task

- Named Entity Recognition
  - Task: identify all mentions of named entities (people, organizations, locations, dates)

```
Pierre Vinken , 61 years old , will join IBM 's board
as a nonexecutive director Nov. 29 .
```

```
[PERS Pierre Vinken] , 61 years old , will join
[ORG IBM] 's board as a nonexecutive director
[DATE Nov. 2] .
```

BIO Encoding for NER

- Define many new tags
  - B-PERS, B-DATE,...: beginning of a mention of a person/date...
  - I-PERS, B-DATE,...: inside of a mention of a person/date...
  - O: outside of any mention of a named entity

```
[PERS Pierre Vinken] , 61 years old , will join
[ORG IBM] 's board as a nonexecutive director
[DATE Nov. 2] .
```

POS Tagging with restricted Tagset?

```
Pierre_B-PERS Vinken_I-PERS ,_O 61_O years_O old_O ,_O
will_O join_O IBM_B-ORG 's_O board_O as_O a_O
nonexecutive_O director_O Nov._B-DATE 29_I-DATE ._O
```

Sequence labeling

- Many NLP tasks are sequence labeling tasks
  – Input: a sequence of tokens/words
  – Output: a sequence of corresponding labels
    - E.g., POS tags, BIO encoding for NER
  – Solution: finding the most probable label sequence for the given word sequence
    - $\boldsymbol{t}^* = argmax_{\boldsymbol{t}} p(\boldsymbol{t}|\boldsymbol{w})$

Comparing to traditional classification problem

Sequence labeling

- $\boldsymbol{t}^* = argmax_{\boldsymbol{t}} p(\boldsymbol{t}|\boldsymbol{w})$
  - $\boldsymbol{t}$ is a vector/matrix
- Dependency between both $(\boldsymbol{t}, \boldsymbol{w})$ and $(\boldsymbol{t}, \boldsymbol{t})$
- Structured output
- Difficult to solve the inference problem

Traditional classification

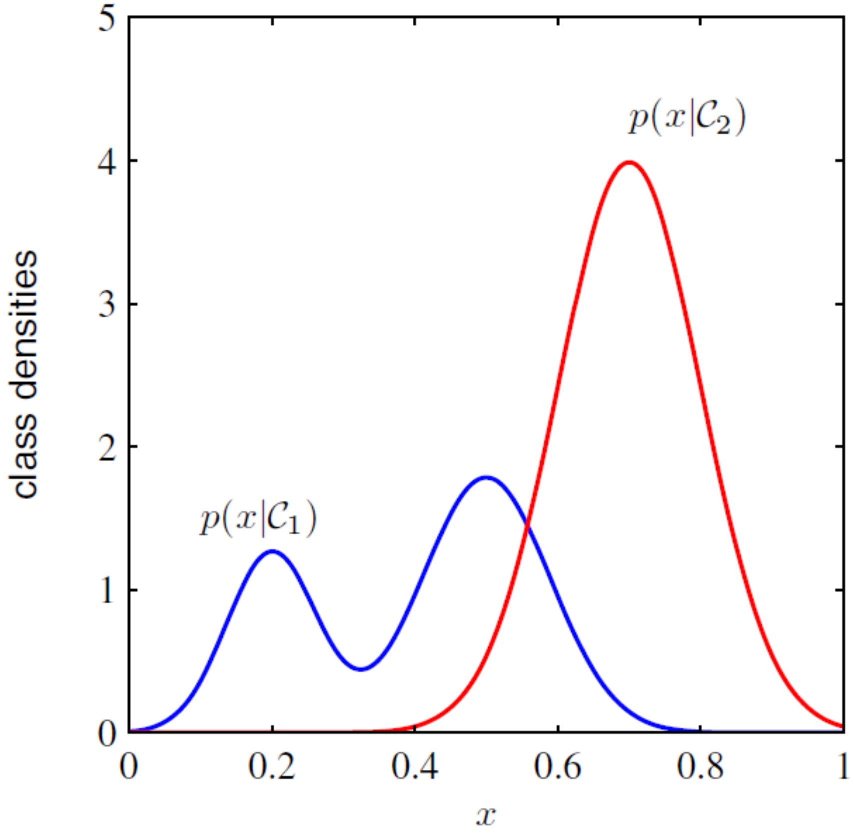
- $y = argmax_y p(y|\boldsymbol{x})$
  - y is a single label
- Dependency only within $(\boldsymbol{t}, \boldsymbol{w})$
- Independent output
- Easy to solve the inference problem

Two modeling perspectives

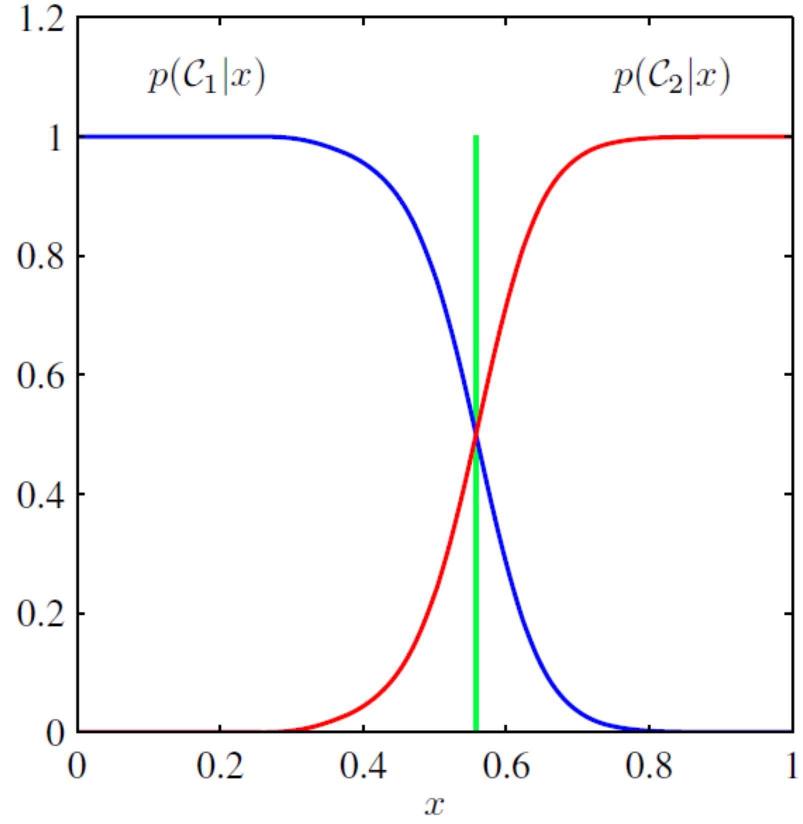
- Generative models
  - Model the joint probability of labels and words
  - $\boldsymbol{t}^* = argmax_{\boldsymbol{t}} p(\boldsymbol{t}|\boldsymbol{w}) = argmax_{\boldsymbol{t}} p(\boldsymbol{w}|\boldsymbol{t}) p(\boldsymbol{t})$
- Discriminative models
  - Directly model the conditional probability of labels given the words
  - $\boldsymbol{t}^* = argmax_{\boldsymbol{t}} p(\boldsymbol{t}|\boldsymbol{w}) = f(\boldsymbol{t}, \boldsymbol{w})$

Generative V.S. discriminative models

Generative Model's view



Discriminative Model's view



Generative V.S. discriminative models

Generative

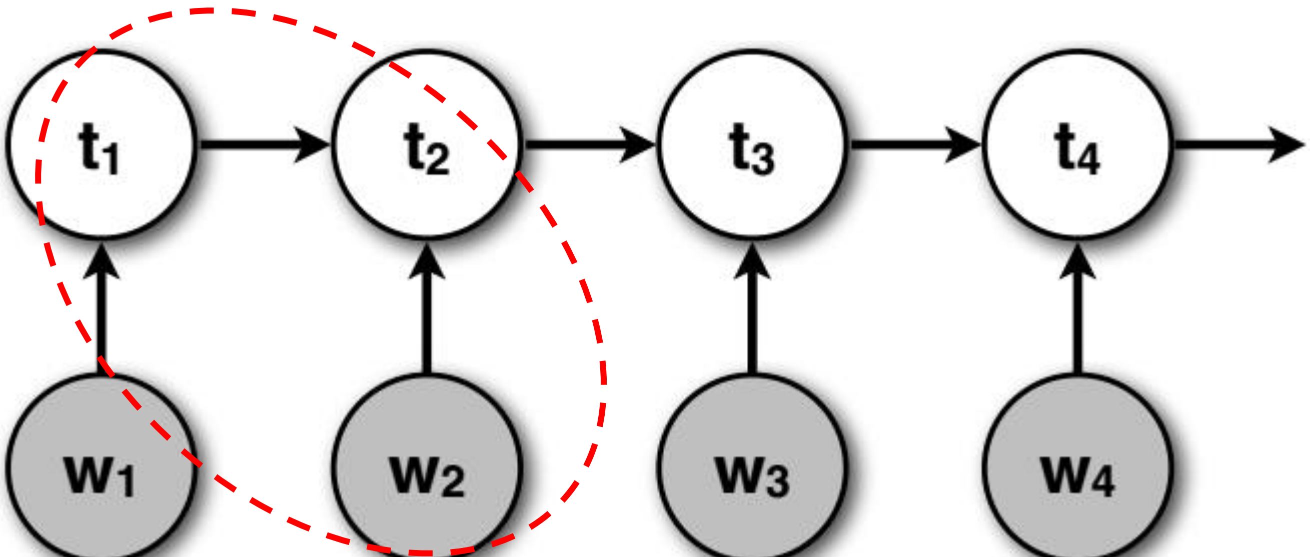
- Specifying joint distribution
  - Full probabilistic specification for all the random variables
- Dependence assumption has to be specified for $p(\boldsymbol{w}|\boldsymbol{t})$ and $p(\boldsymbol{t})$
- Flexible, can be used in unsupervised learning

Discriminative

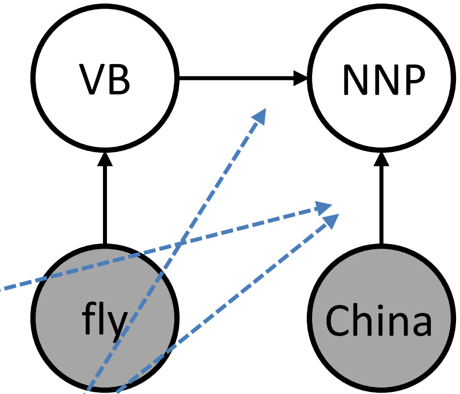
- Specifying conditional distribution
  - Only explain the target variable
- Arbitrary features can be incorporated for modeling $p(\boldsymbol{t}|\boldsymbol{w})$
- Need training data, only suitable for (semi-) supervised learning

Maximum entropy Markov models

- MEMMs are discriminative models of the labels $\boldsymbol{t}$ given the observed input sequence $\boldsymbol{w}$
  - $p(\boldsymbol{t}|\boldsymbol{w}) = \prod_i p(t_i|w_i, t_{i-1})$



Design features



- Emission-like features
  - Binary feature functions
    - $f_{\text{first-letter-capitalized-}\textbf{NNP}}(\text{China}) = 1$
    - $f_{\text{first-letter-capitalized-}\textbf{VB}}(\text{fly}) = 0$
  - Integer (or real-valued) feature functions
    - $f_{\text{number-of-vowels-}\textbf{NNP}}(\text{China}) = 2$
- Transition-like features
  - Binary feature functions
    - $f_{\text{first-letter-capitalized-}\textbf{NNP-VB}}(\text{China}) = 1$

Not necessarily independent features!

Parameterization of $p(t_i|w_i, t_{i-1})$

- Associate a real-valued weight λ to each specific type of feature function
  - λ_k for $f_{\text{first-letter-capitalized-}\textbf{NNP}}(w)$
- Define a scoring function $f(t_i, t_{i-1}, w_i) = \sum_k \lambda_k f_k(w_i)$
- Naturally $p(t_i|w_i, t_{i-1}) \propto \exp f(t_i, t_{i-1}, w_i)$
  - Recall the basic definition of probability
    - $P(x) > 0$
    - $\sum_x p(x) = 1$

Parameterization of MEMMs

$$p(\boldsymbol{t}|\boldsymbol{w}) = \prod_i p(t_i|w_i, t_{i-1})$$
$$\propto \prod_i \exp(f(t_i, t_{i-1}, w_i))$$

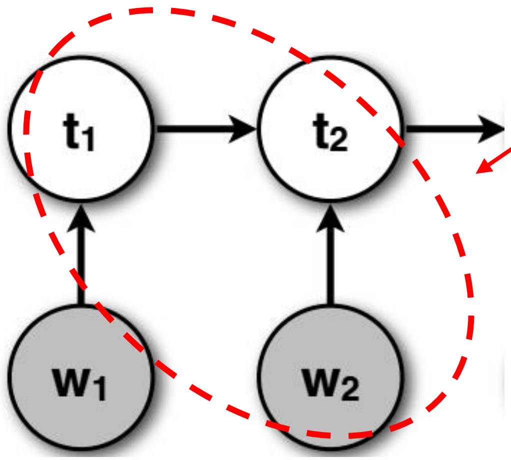
- It is a log-linear model

  Constant only related to λ

  $-\log p(\boldsymbol{t}|\boldsymbol{w}) = \sum_i f(t_i, t_{i-1}, w_i) - C(\boldsymbol{\lambda})$
- Viterbi algorithm can be used to decode the most probable label sequence solely based on $\sum_i f(t_i, t_{i-1}, w_i)$

Parameter estimation

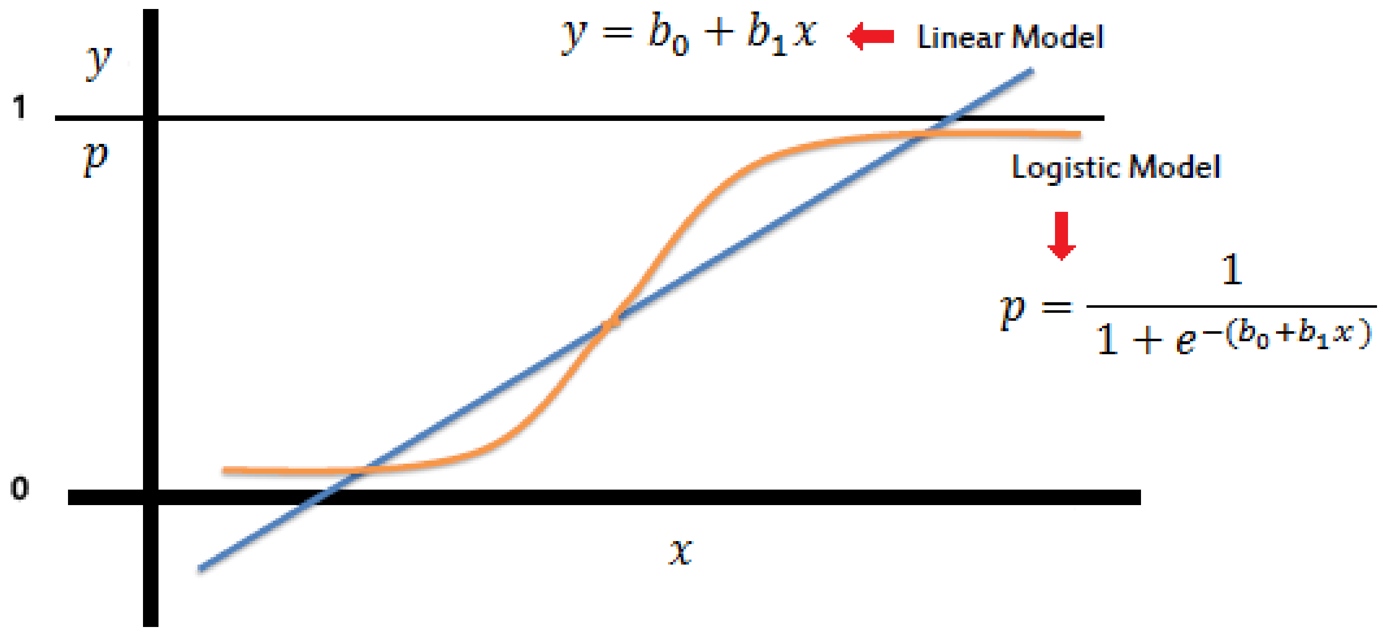
- Maximum likelihood estimator can be used in a similar way as in HMMs
  - $\lambda^* = argmax_\lambda \sum_{t,w} \log p(t|w)$
    $= argmax_\lambda \sum_{t,w} \sum_i f(t_i, t_{i-1}, w_i) - C(\boldsymbol{\lambda})$



Decompose the training data into such units

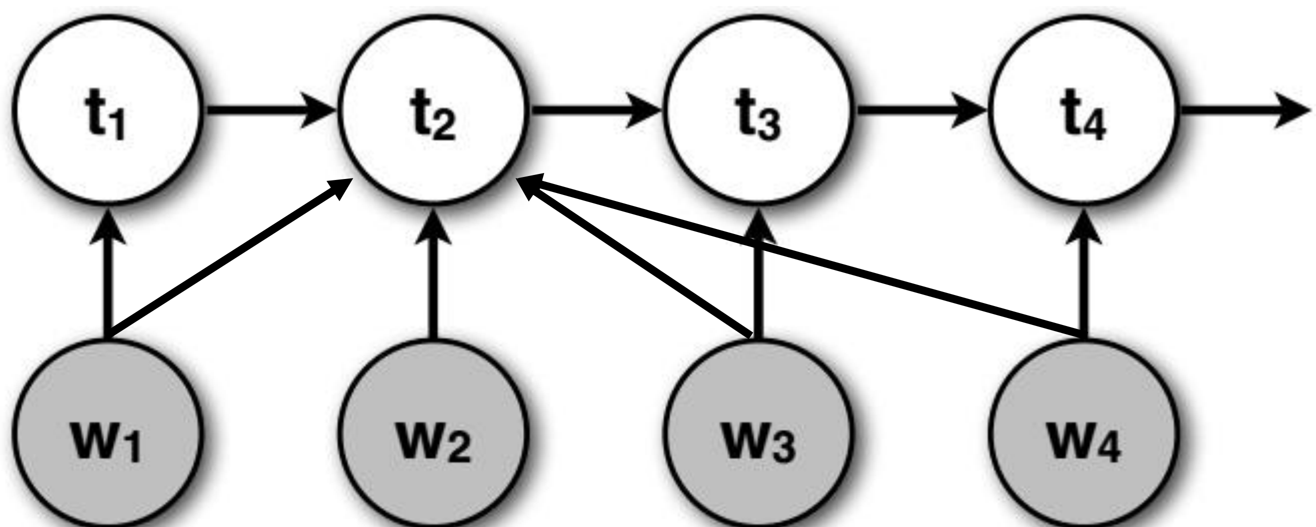
Why maximum entropy?

- We will explain this in detail when discussing the Logistic Regression models



A little bit more about MEMMs

- Emission features can go across multiple observations
  - $f(t_i, t_{i-1}, w_i) = \sum_k \lambda_k f_k(w_i, \boldsymbol{w})$
  - Especially useful for shallow parsing and NER tasks



Conditional random field

- A more advanced model for sequence labeling
  - Model global dependency
  - $p(t|w) \propto \prod_i \exp(\sum_k \lambda_k f_k(t_i, \boldsymbol{w}) + \sum_l \eta_l g_l(t_i, t_{i-1}, \boldsymbol{w}))$