

# 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		38,857	
Ambiguous (2–7 tags)		5,490	11%	8844	18%
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

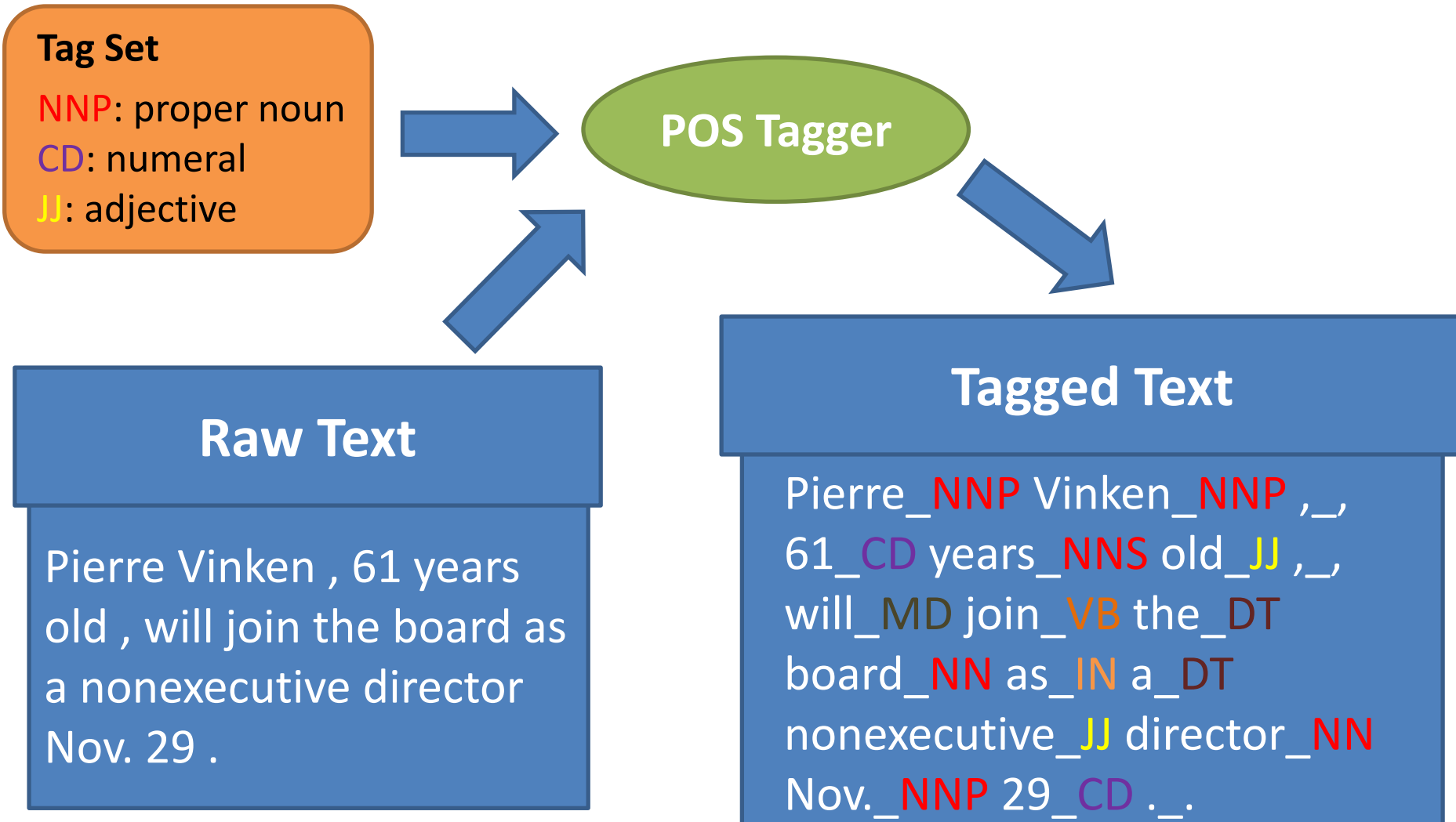
***Rules can be learned  
via inductive learning.***

she PRP  
promised ~~VDN~~, 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*

# Recap: what is POS tagging



# Recap: public tagsets in NLP

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

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

# Recap: 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

***Rules can be learned via inductive learning.***

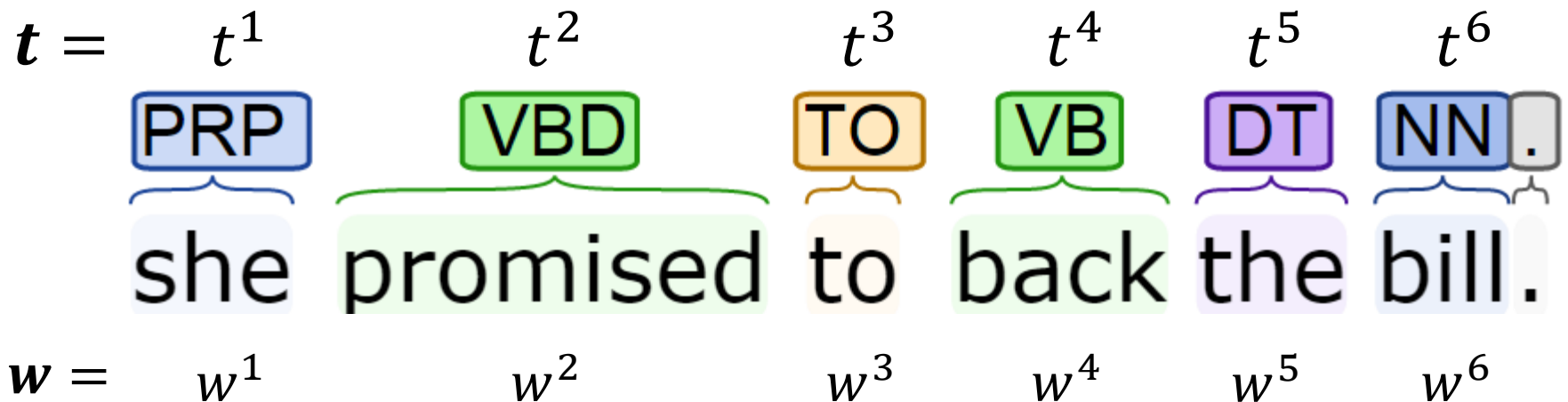
she PRP  
promised ~~VDN~~, VBD  
to TO  
back VB, JJ, RB, NN!!  
the DT  
bill NN, ~~VB~~

*R1: Pronoun should be followed by a past tense verb*

*R2: Verb cannot follow determiner*

# Building a POS tagger

- Statistical POS tagging

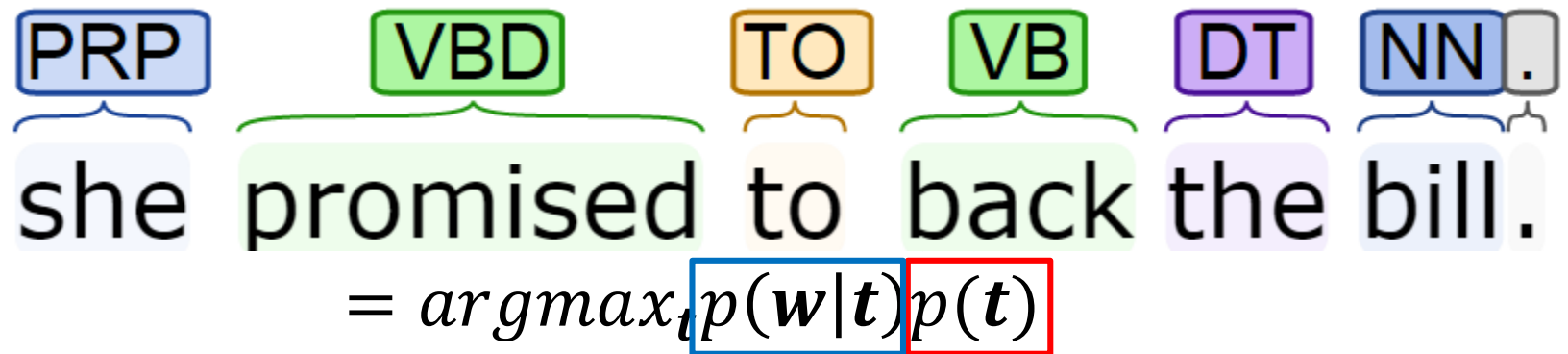


- What is the most **likely** sequence of tags  $t$  for the given sequence of words  $w$

$$t^* = \operatorname{argmax}_t p(t|w)$$

# POS tagging with generative models

- Bayes Rule



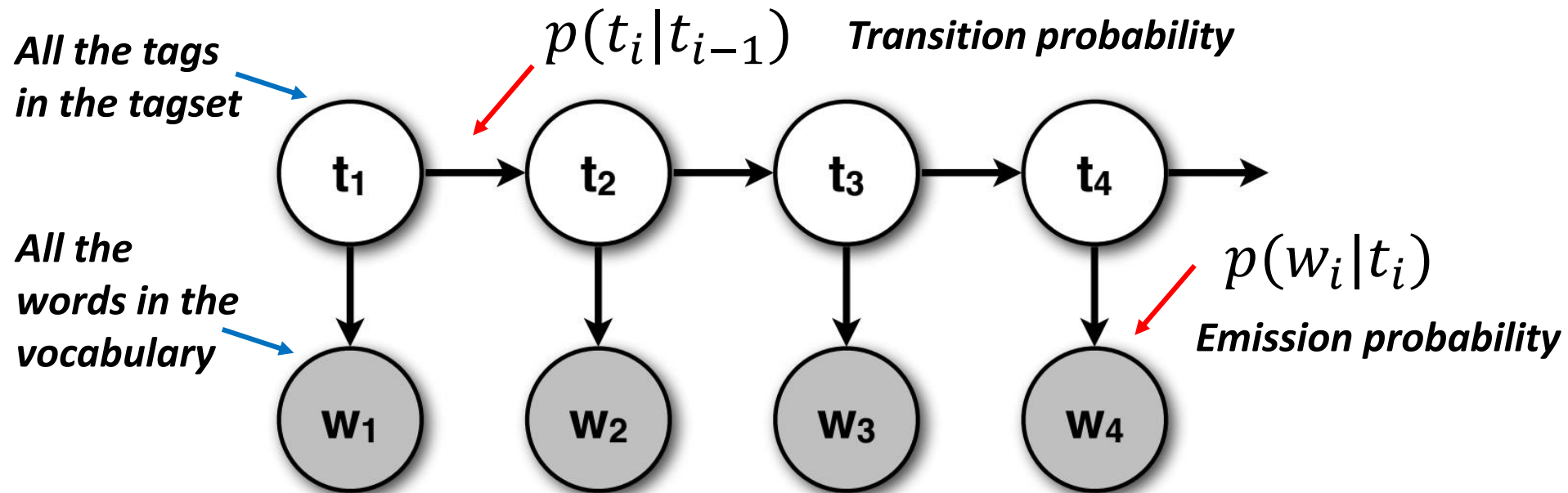
- Joint distribution of tags and words
- Generative model
  - A stochastic process that **first generates the tags**, and then **generates the words based on these tags**

# Hidden Markov models

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

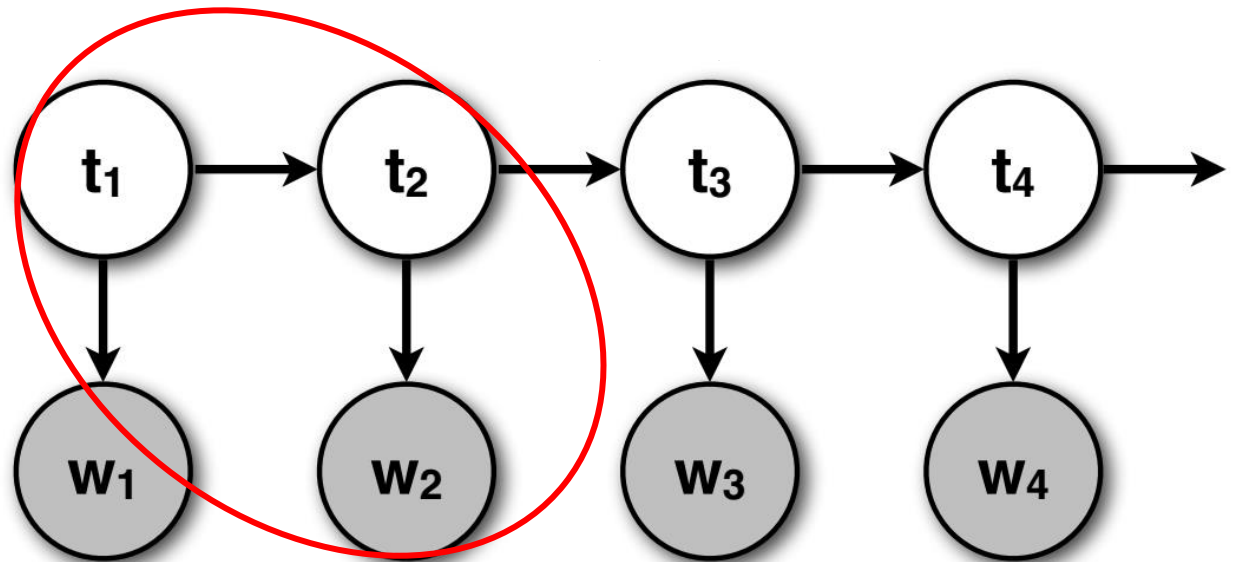


# Graphical representation of HMMs



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

# Finding the most probable tag sequence



- Com
  - 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!

$$t^1 = t_4 t_1 t_3 t_5 t_7$$

$$t^2 = t_4 t_1 t_3 t_5 t_2$$

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



Word  $w_1$  takes tag  $t_4$

# Trellis: a special structure for HMMs

$$t^1 = \underline{t_4} \underline{t_1} \underline{t_3} \underline{t_5} \boxed{t_7}$$

$$t^2 = \underline{t_4} \underline{t_1} \underline{t_3} \underline{t_5} \boxed{t_2}$$

Computation can be reused!

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

Word  $w_1$  takes tag  $t_4$

# 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  $\text{trellis}[j][i]$  from the entries in the previous column  $\text{trellis}[j][i-1]$ 
  - $T[j][i] = P(w_i | t^j) \text{Max}_k \left( T[k][i-1] P(t^j | t_k) \right)$ 
    - Generating the current observation* (blue arrow pointing to  $P(w_i | t^j)$ )
    - The best i-1 tag sequence* (purple arrow pointing to  $T[k][i-1]$ )
    - Transition from the previous best ending tag* (green arrow pointing to  $P(t^j | t_k)$ )

# Viterbi algorithm

Dynamic programming:  $O(T^2N)$ !

$$T[j][i] = P(w_i | t^j) \text{Max}_k (T[k][i-1] P(t^j | t_k))$$

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



Order of computation

# Decode $\operatorname{argmax}_t p(\mathbf{t}|\mathbf{w})$

- 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*

$$T[j][i] = P(w_i | t^j) \operatorname{Max}_k (T[k][i-1] P(t^j | t_k))$$

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

# 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  $w_i t_j$

- $p(t_j | t_i) = \frac{c(t_i, t_j)}{c(t_i)}$

- $p(w_i | t_j) = \frac{c(w_i, t_j)}{c(t_j)}$

*Proper smoothing is necessary!*

# 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

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



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

# BIO Encoding for Shallow Parsing

- Define several new tags
  - B-NP B-VP B-PP: beginning of an “NP”, “VP”, “PP” chunk
  - I-NP I-VP I-PP: 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] .



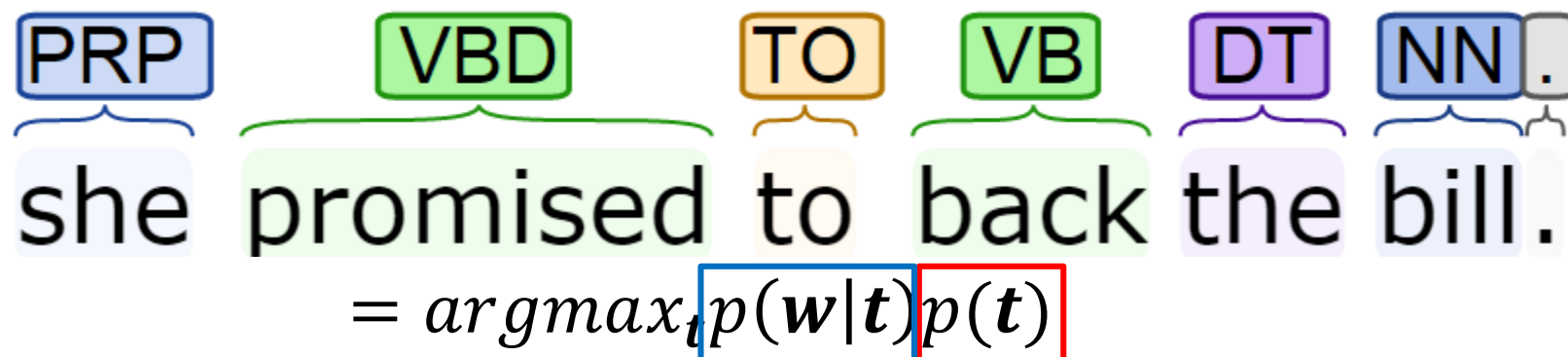
POS Tagging with restricted Tagset?

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



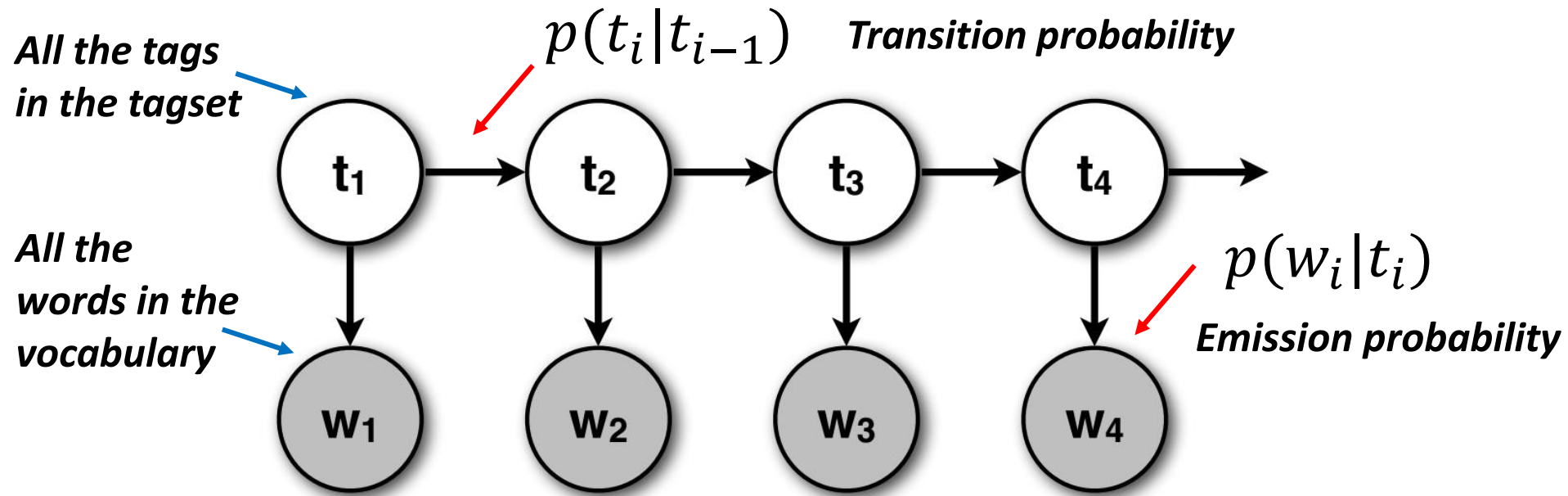
# Recap: POS tagging with generative models

- Bayes Rule



- 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**

# Recap: graphical representation of HMMs



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

# Recap: decode $\operatorname{argmax}_t p(\mathbf{t}|\mathbf{w})$

- 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*

$$T[j][i] = P(w_i | t^j) \operatorname{Max}_k (T[k][i-1] P(t^j | t_k))$$

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

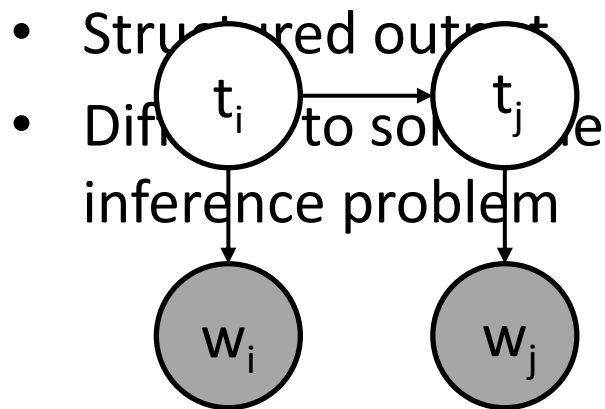
# Sequence labeling

- 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
    - $\mathbf{t}^* = \operatorname{argmax}_{\mathbf{t}} p(\mathbf{t}|\mathbf{w})$

# Comparing to traditional classification problem

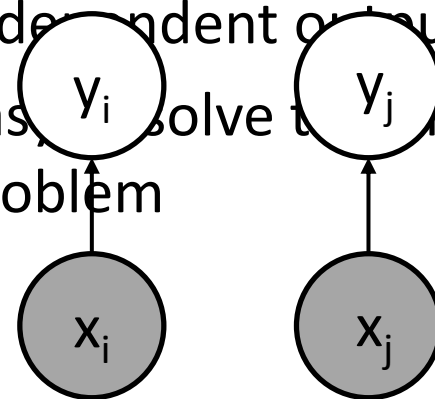
## Sequence labeling

- $\mathbf{t}^* = \operatorname{argmax}_{\mathbf{t}} p(\mathbf{t} | \mathbf{w})$ 
  - $\mathbf{t}$  is a vector/matrix
- Dependency between both  $(\mathbf{t}, \mathbf{w})$  and  $(t_i, t_j)$



## Traditional classification

- $y = \operatorname{argmax}_y p(y | \mathbf{x})$ 
  - $y$  is a single label
- Dependency only within  $(y, \mathbf{x})$
- Independent output
- Easy to solve the inference problem



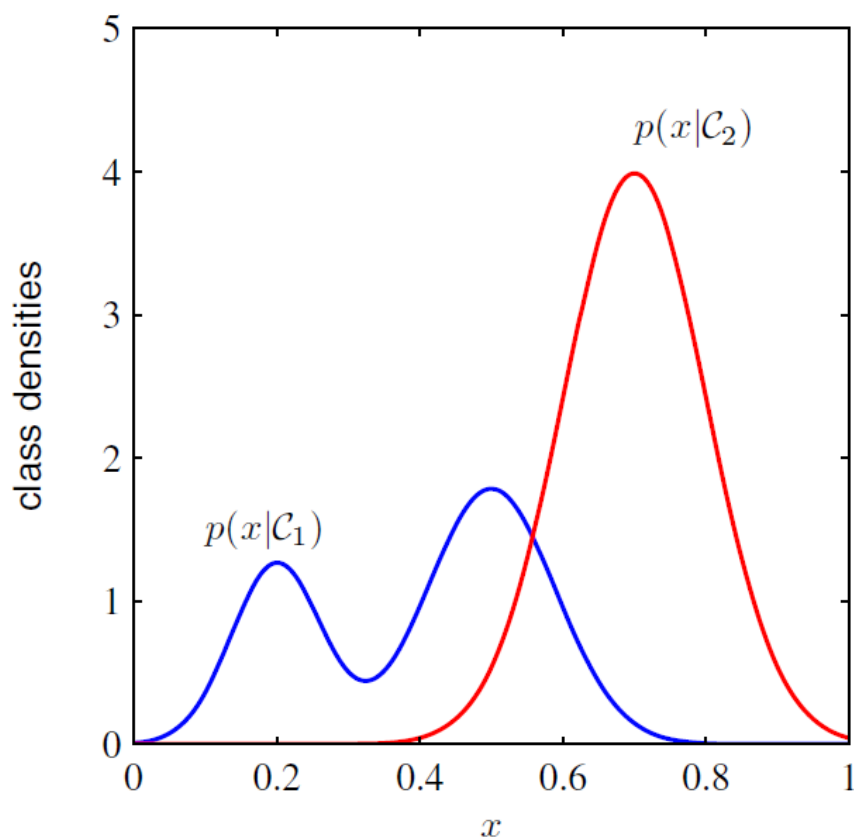
# Two modeling perspectives

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

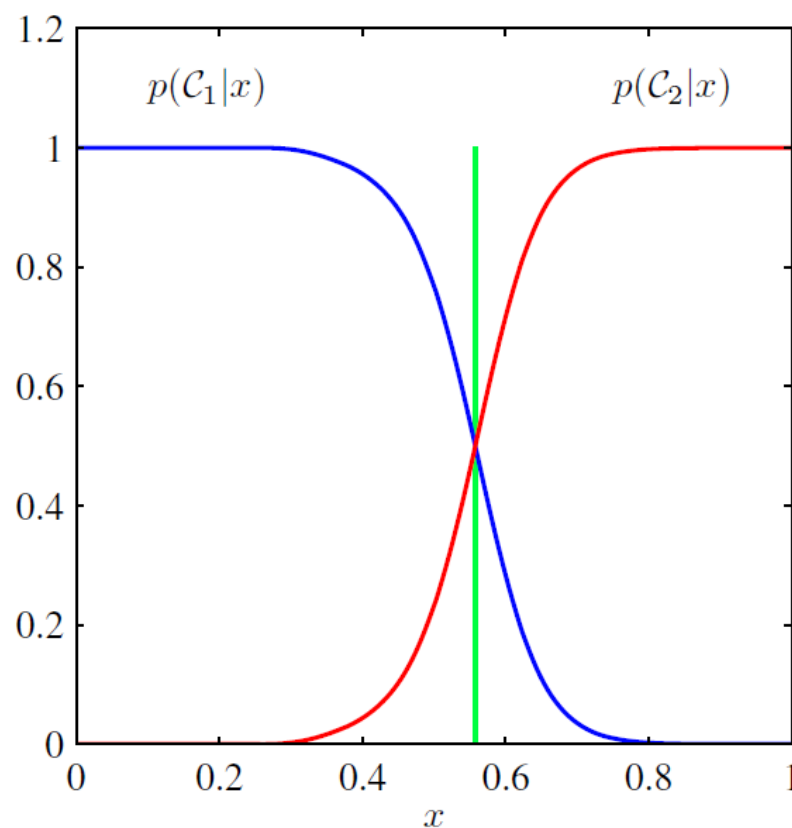
# Generative V.S. discriminative models

- Binary classification as an example

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  $p(\mathbf{w}|\mathbf{t})$  and  $p(\mathbf{t})$
- Flexible, can be used in unsupervised learning

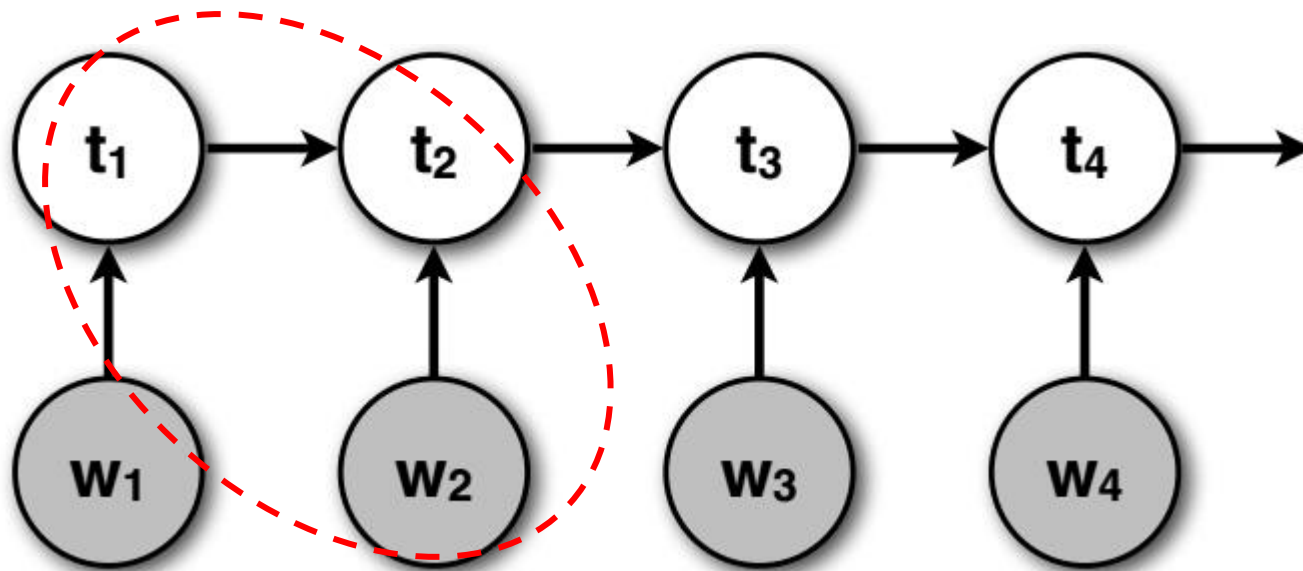
## Discriminative

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



# Maximum entropy Markov models

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



# Design features

- Emission-like features

- Binary feature functions

- $f_{\text{first-letter-capitalized-NNP}}(\text{China}) = 1$
    - $f_{\text{first-letter-capitalized-VB}}(\text{know}) = 0$

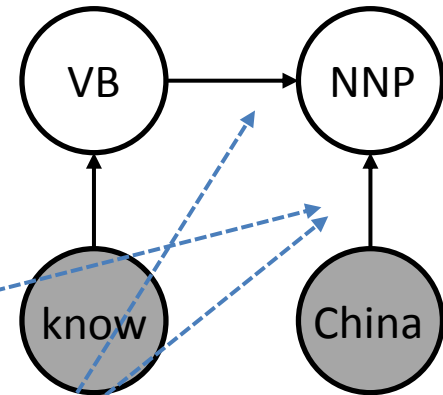
- Integer (or real-valued) feature functions

- $f_{\text{number-of-vowels-NNP}}(\text{China}) = 2$

- Transition-like features

- Binary feature functions

- $f_{\text{first-letter-capitalized-VB-NNP}}(\text{China}) = 1$



Not necessarily  
independent features!

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

- Associate a real-valued weight  $\lambda$  to each specific type of feature function
  - $\lambda_k$  for  $f_{\text{first-letter-capitalized-NNP}}(w)$
- Define a scoring function  $f(t_i, t_{i-1}, w_i) = \sum_k \lambda_k f_k(t_i, t_{i-1}, w_i)$
- 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

$$\begin{aligned} p(\mathbf{t}|\mathbf{w}) &= \prod_i p(t_i|w_i, t_{i-1}) \\ &= \frac{\prod_i \exp(f(t_i, t_{i-1}, w_i))}{\sum_{\mathbf{t}} \prod_i \exp(f(t_i, t_{i-1}, w_i))} \end{aligned}$$

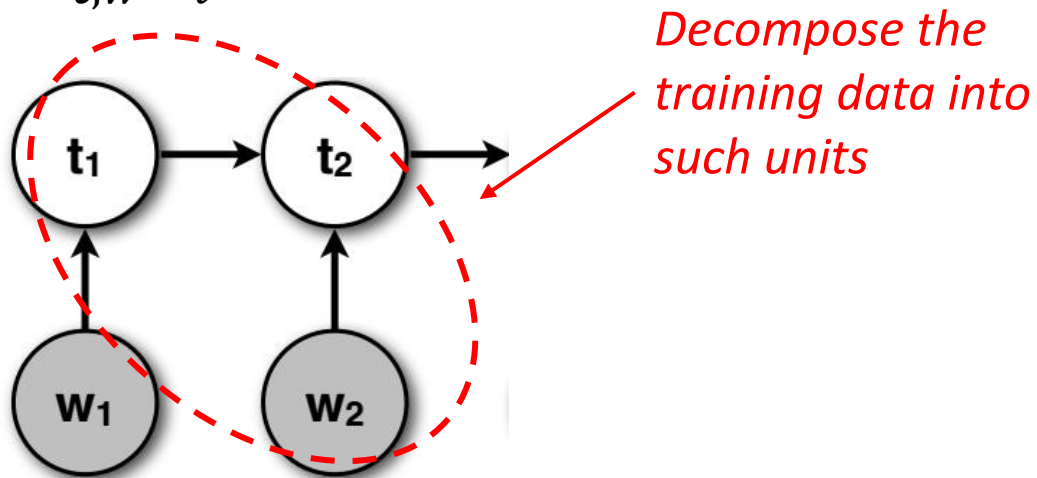
- It is a log-linear model
  - $-\log p(\mathbf{t}|\mathbf{w}) = \sum_i f(t_i, t_{i-1}, w_i) - \mathcal{C}(\lambda)$
- Viterbi algorithm can be used to decode the most probable label sequence solely based on  $\sum_i f(t_i, t_{i-1}, w_i)$

Constant only related to  $\lambda$

# Parameter estimation

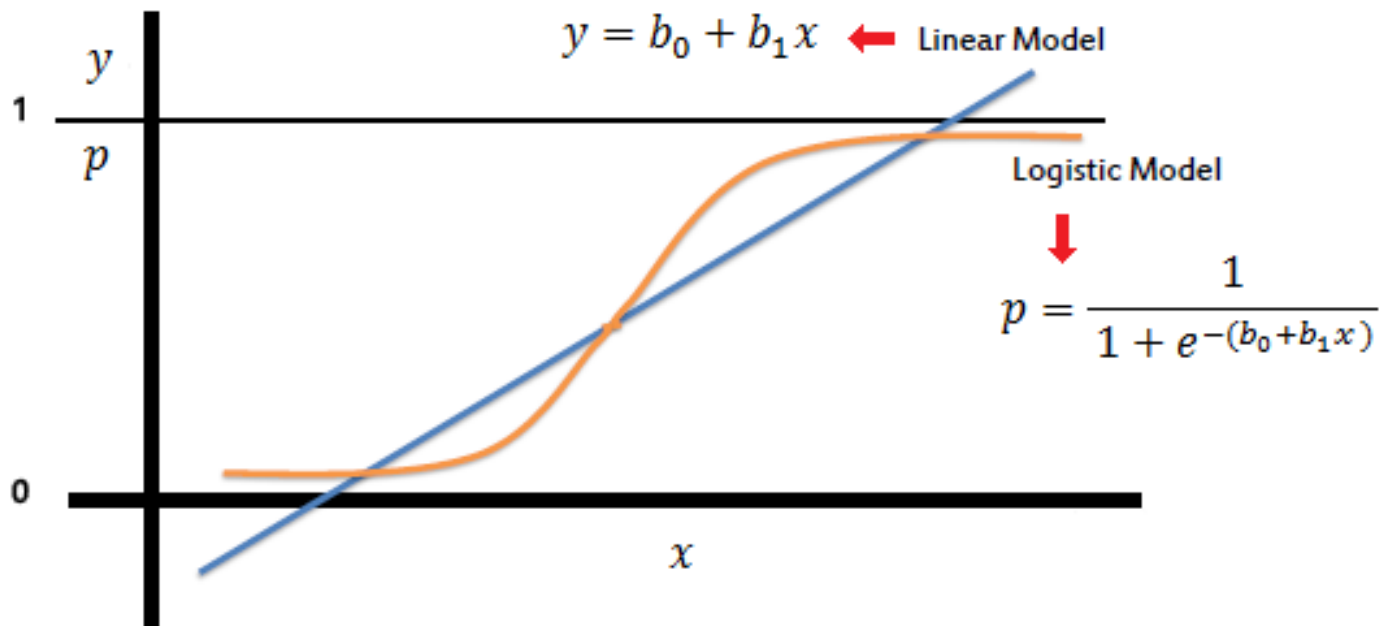
- Maximum likelihood estimator can be used in a similar way as in HMMs

$$\begin{aligned} -\lambda^* &= \operatorname{argmax}_{\lambda} \sum_{t,w} \log p(\mathbf{t}|w) \\ &= \operatorname{argmax}_{\lambda} \sum_{t,w} \sum_i f(t_i, t_{i-1}, w_i) - C(\lambda) \end{aligned}$$



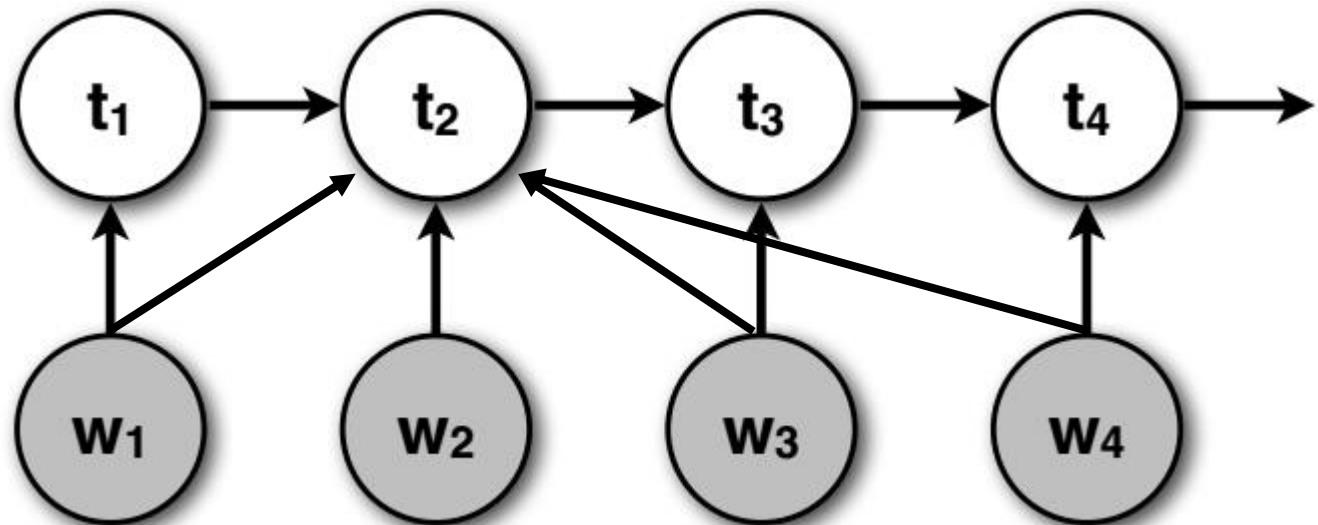
# Why maximum entropy?

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



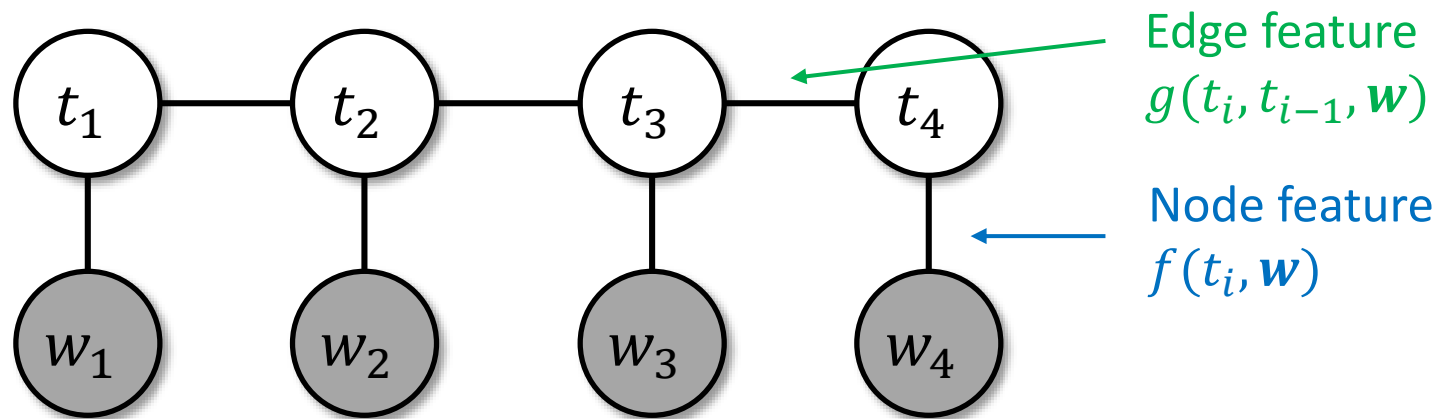
# A little bit more about MEMMs

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



# Conditional random field

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





# What you should know

- Definition of POS tagging problem
  - Property & challenges
- Public tag sets
- Generative model for POS tagging
  - HMMs
- General sequential labeling problem
- Discriminative model for sequential labeling
  - MEMMs

# Today's reading

- Speech and Language Processing
  - Chapter 5: Part-of-Speech Tagging
  - Chapter 6: Hidden Markov and Maximum Entropy Models
  - Chapter 22: Information Extraction (optional)