Natural Language Processing
自然語言處理

黃瀚萱
Department of Computer Science
National Chengchi University
2020 Fall

Lesson 7

POS Tagging & Sequence Labeling

Schedule

Date	Topic
9/16	Introduction
9/23	Linguistic Essentials
9/30	Collocation
10/7	Language Model
10/14	Performance Evaluation and Word Sense Disambiguation
10/21	Text Classification (HW1 will be assigned)
10/28	Invited Talk: NLP and Cybersecurity (Term Project)
11/4	**POS Tagging**
11/11	Midterm Exam

Schedule

Date	Topic
11/18	Chinese Word Segmentation
11/25	Word Embeddings
12/2	Neural Networks for NLP
12/9	Parsing
12/16	Discourse Analysis
12/23	Invited Talk
12/30	Final Project Presentation I
1/6	Final Project Presentation II
1/13	Final Exam

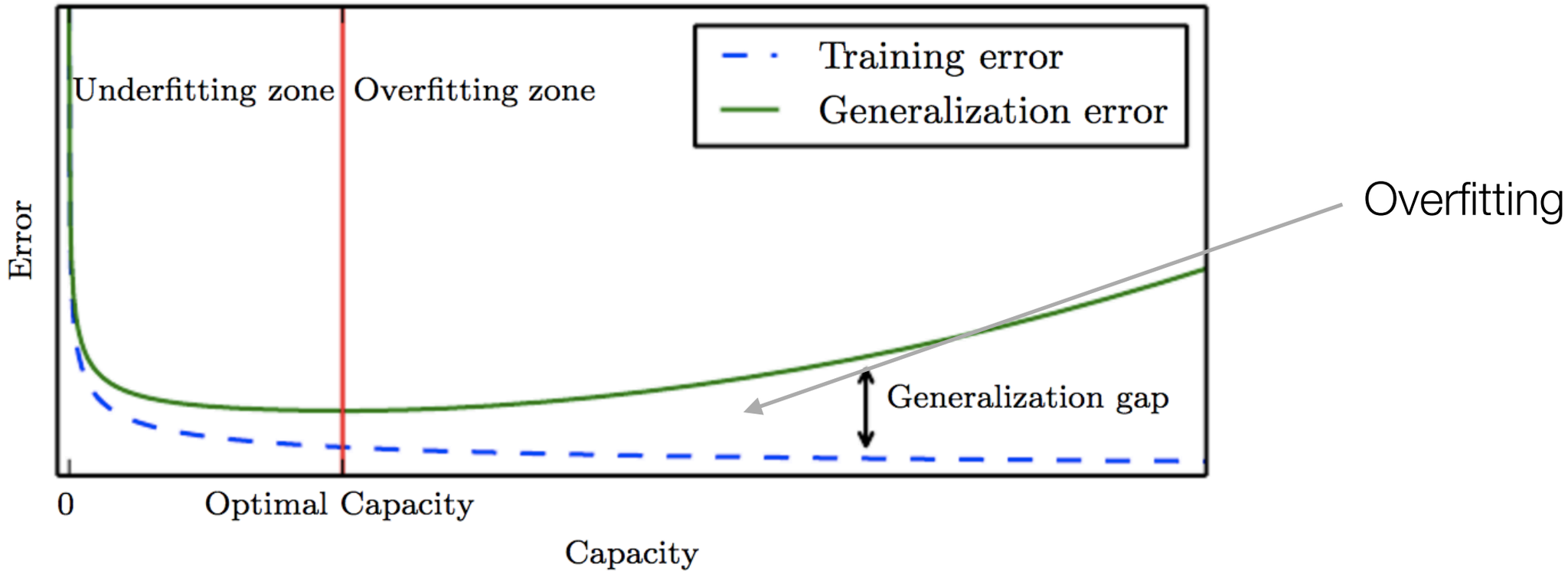
Agenda

- **Overfitting**
- Part-of-speech tags
- Part-of-speech tagging
- Sequence labeling
  - Hidden Markov model
  - Maximum-entropy Markov model
  - Conditional random fields
- Codelab

Overfitting

Training Performance vs Validation Performance

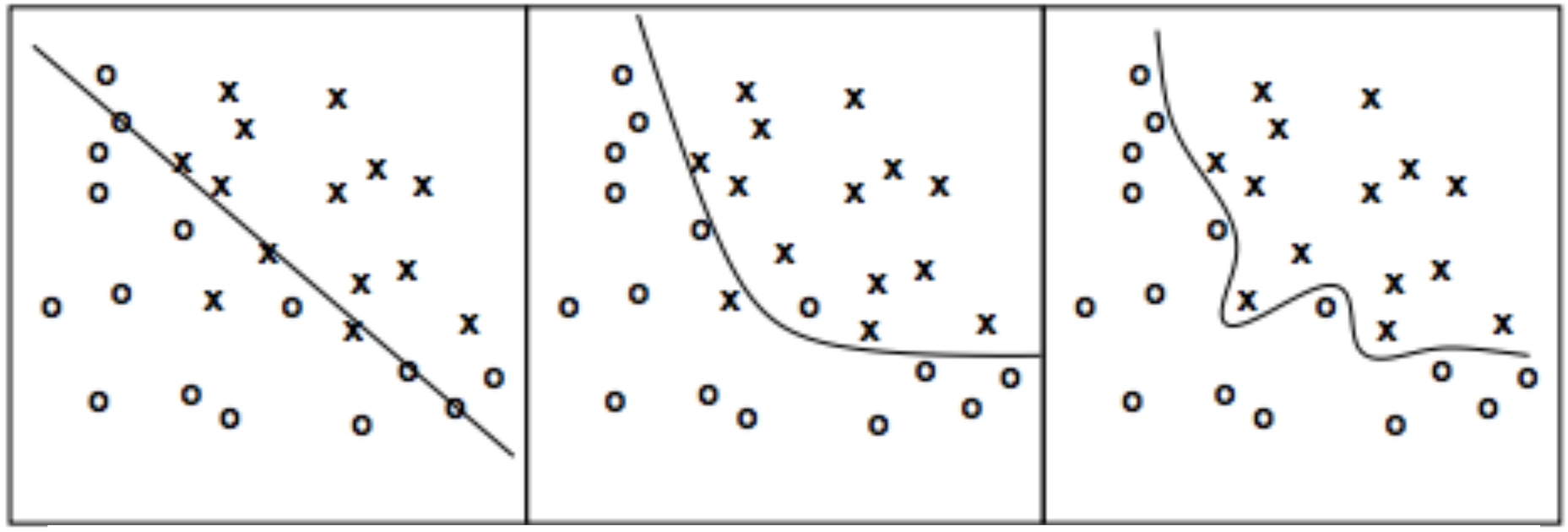
- Training performance
  - Evaluating the model with the training data.
  - Used to measure model's ability to fit the data.
- Testing (validation) performance
  - Evaluating the model with unseen data (in the test/validation set)
  - Used to measure the performance in real applications.



Overfitting

- Training performance >> testing performance

"The most likely hypothesis is the simplest one consistent with the data."

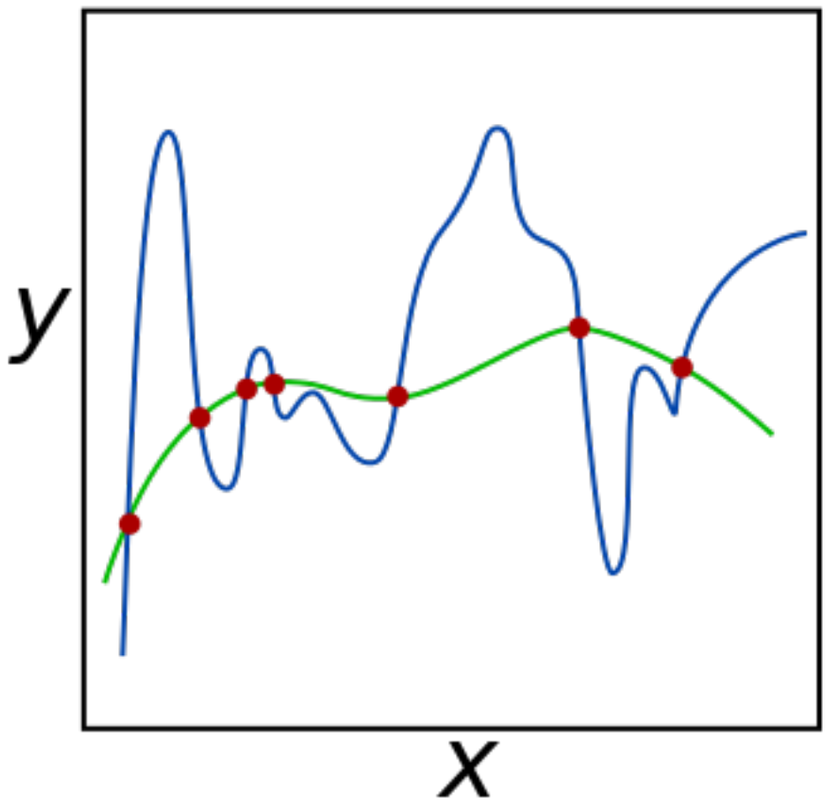


under-fitting good comprise overfitting

Data Sparseness

- Reasons of overfitting
  - Too specific features
  - Too complex model
- Solutions
  - More training data
  - Reduce the complexity of the model
  - Feature selection
  - Regularization

$$D = \{ (x_{1,1}, x_{1,2}, x_{1,3}, ..., x_{1,n}, y_1), (x_{2,1}, x_{2,2}, x_{2,3}, ..., x_{2,n}, y_2), ..., (x_{m,1}, x_{m,2}, x_{m,3}, ..., x_{m,n}, y_m)\}$$
, where $n \gg m$

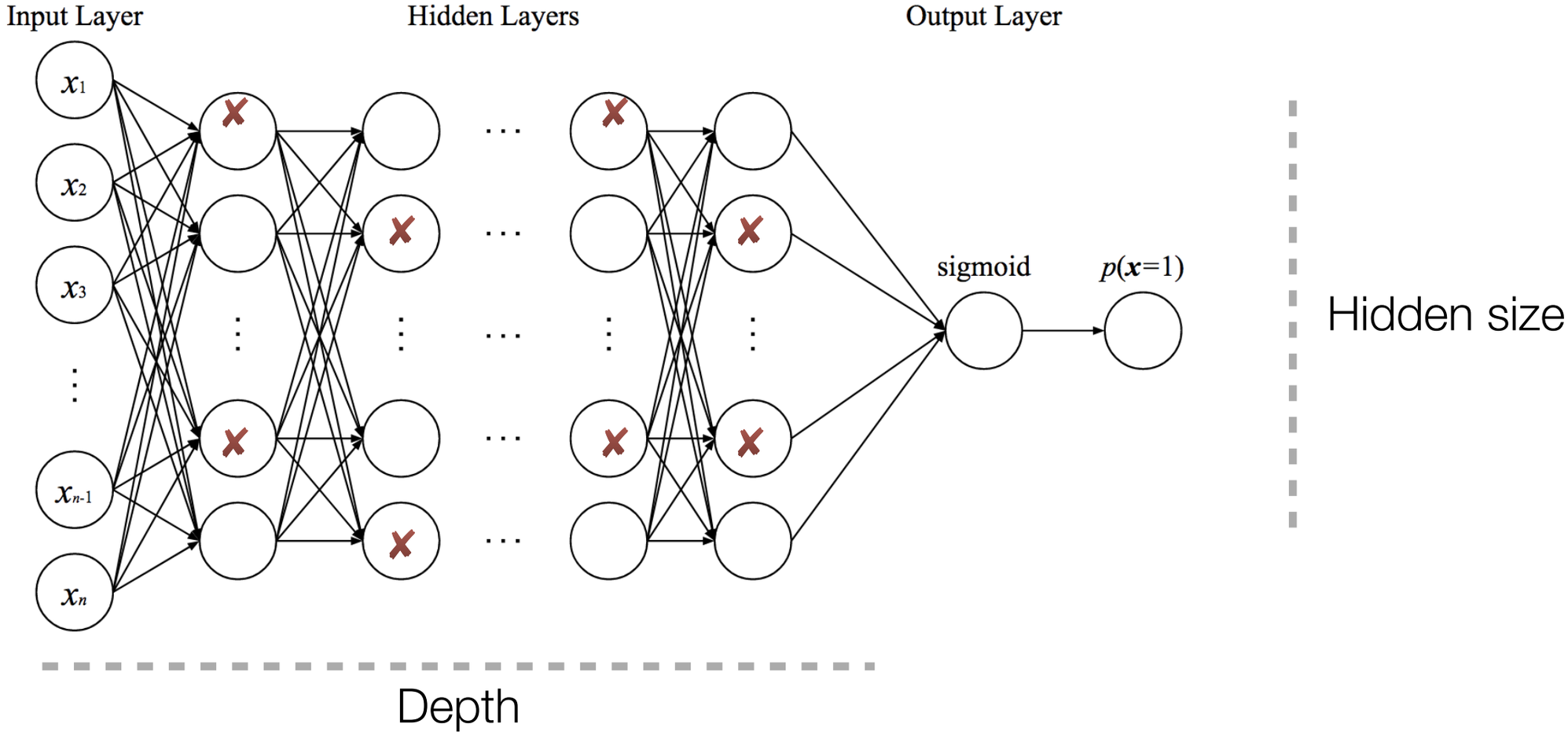


Dealing with Sparsity

- Feature selection
  - Removing unimportant features
- Naive Bayes
  - Smoothing
- Decision tree
  - Pruning
- Logistic regression
  - Increasing regularization by decreasing the hyperparameter c

Dealing with Sparsity

- Neural network
  - Decreasing the depth of the neural network
  - Decreasing the size of hidden layers
  - Dropout



Feature Selection

- Removing low frequency features
  - One-time appearing words
- Verifying the correlation between a feature and the labels.
  - Chi-square test
- Feature selection by model

Reducing the Features with Non-zero Coeffiecents

- The more variables are used, the more likely the model overfits the training data.

- The more nonzero coefficients are in the model, the more difficult it is to interpret of the model.

- For explanation, a model with 3 features that achieves an accuracy of 90% is better than a model with 1000 features that achieves an accuracy of 92%.

Regularization

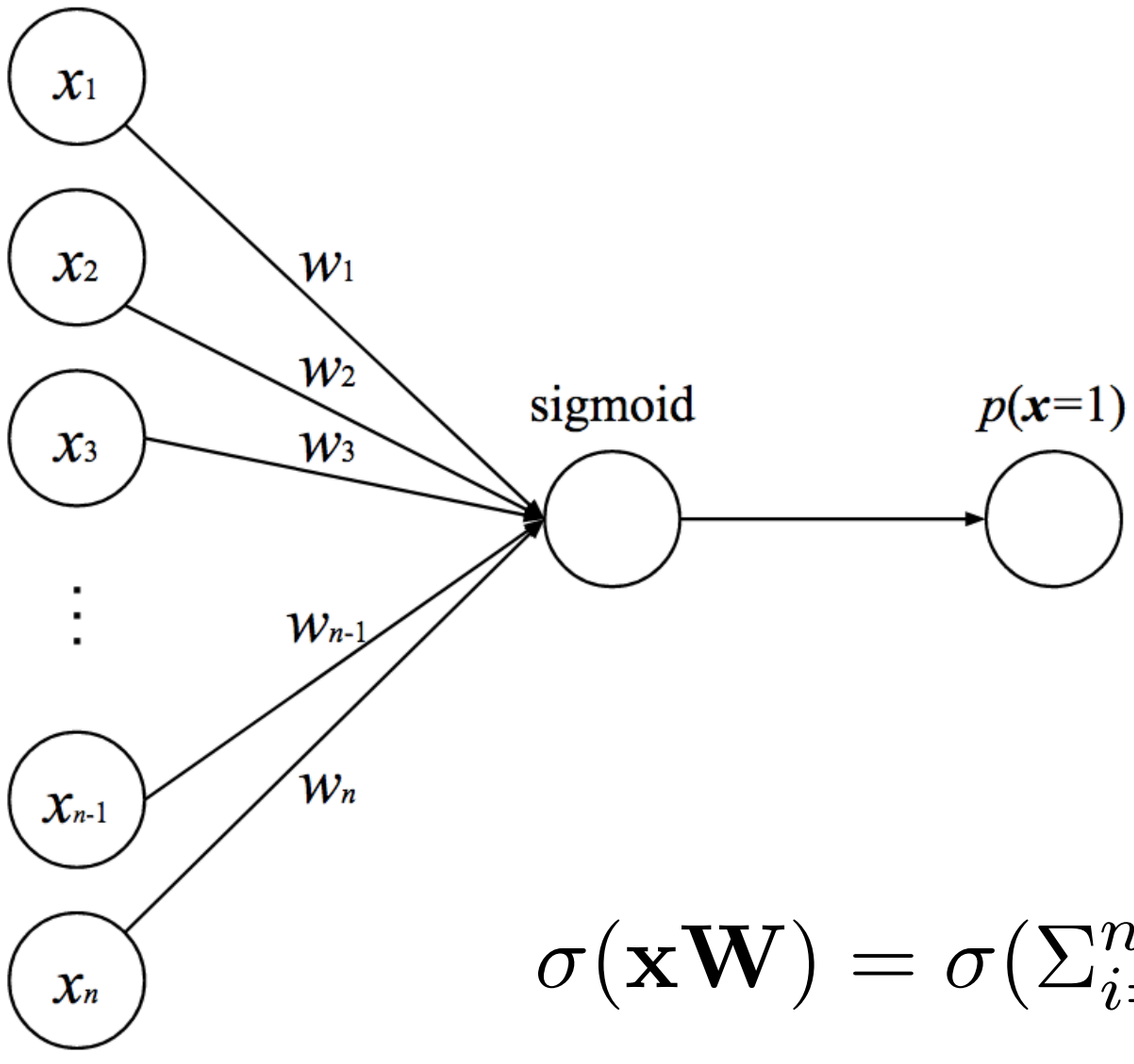
- Regularization is an approach in which we add to the error term a penalty that gets larger as number of nonzero coefficients gets larger.

- Then we minimize the combined error and penalty.

- The more importance we place on the penalty term, the more we discourage large coefficients.

Logistic Regression Model

- The model will be very sensitive to some input x_i with very large $|w_i|$



$$\sigma(\mathbf{xW}) = \sigma(\Sigma_{i=1}^{n} x_i w_i)$$

Lasso Regularization

- Least absolute shrinkage and selection operator
- L_1 norm for regularization
  - Manhattan distance

$$||W||_1 = |w_1| + |w_2| + |w_3| + ... + |w_n|$$

- Loss function with L_1 regularization

$$Loss_{R_{L_1}} = (y - y')^2 - \lambda||W||_1 = (y - y')^2 - \lambda \sum_{i=1}^{n} |w_i|$$

Ridge Regularizatiom

- L_2 norm (squared norm) for regularization
  - Squared Euclidean distance

$$||W||_2^2 = w_1^2 + w_2^2 + w_3^2 + ... + w_n^2$$

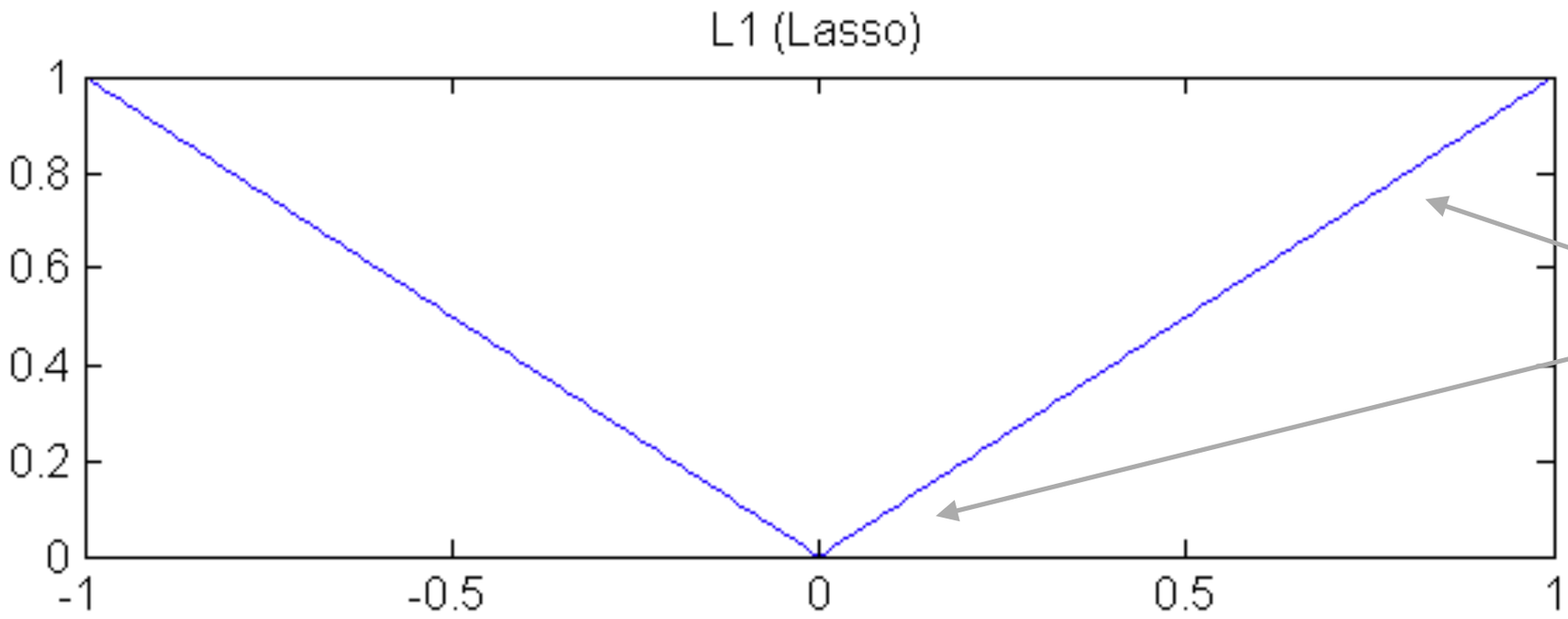
- Loss function with L_2 regularization

$$Loss_{R_{L_2}} = (y - y')^2 - \lambda||W||_2^2 = (y - y')^2 - \lambda \sum_{i=1}^{n} w_i^2$$

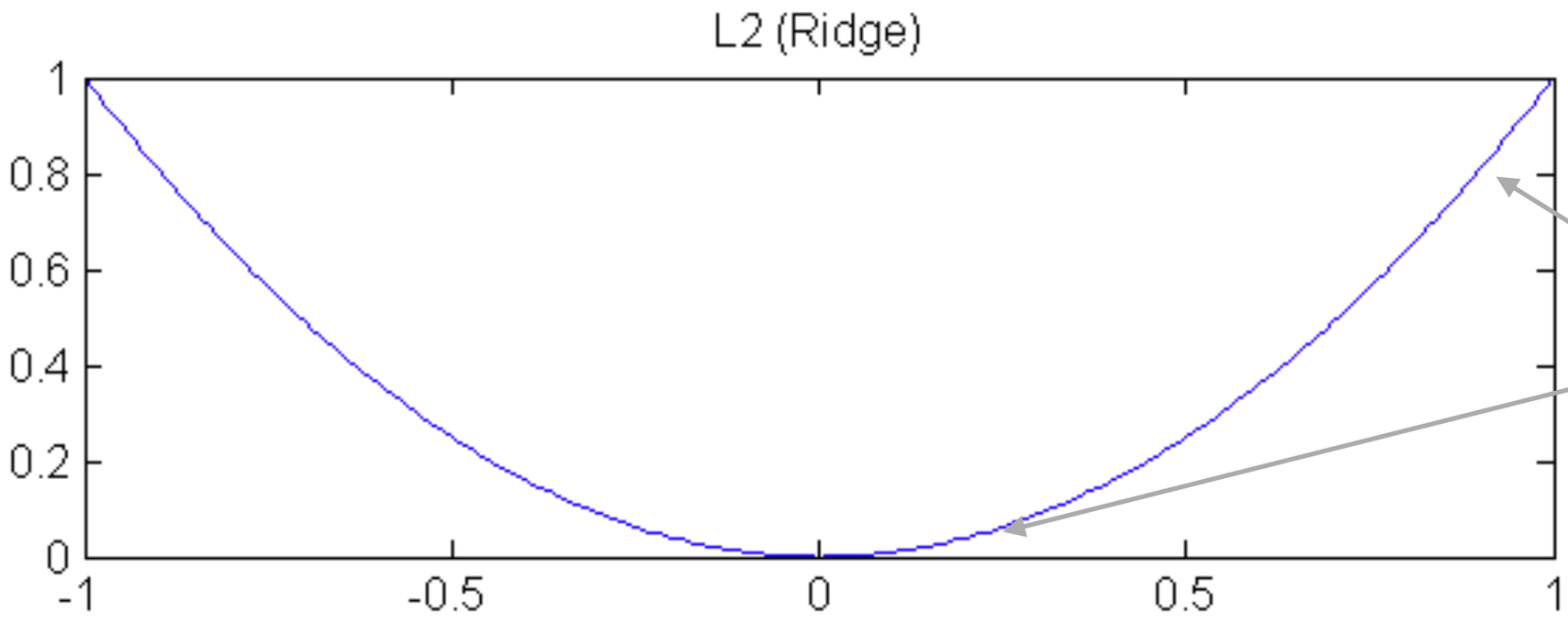
L1 vs L2 Regularization Methods

- L2 severely punishes for high parameter weights, but once the value is close enough to zero, their effect becomes insignificant.
    - Prefer to decrease value of a parameter with high weight by 1 than to decrease the value of ten parameters that already have relatively low weights by 0.1 each.
- L1 punishes **uniformly** for both low and high values.
    - To decrease all the non-zero parameter values toward 0.
    - Resulting a sparse solution (many parameters are 0)

L1 vs L2 Regularization Methods



Uniform penalty for each value



Higher penalty for the higher value

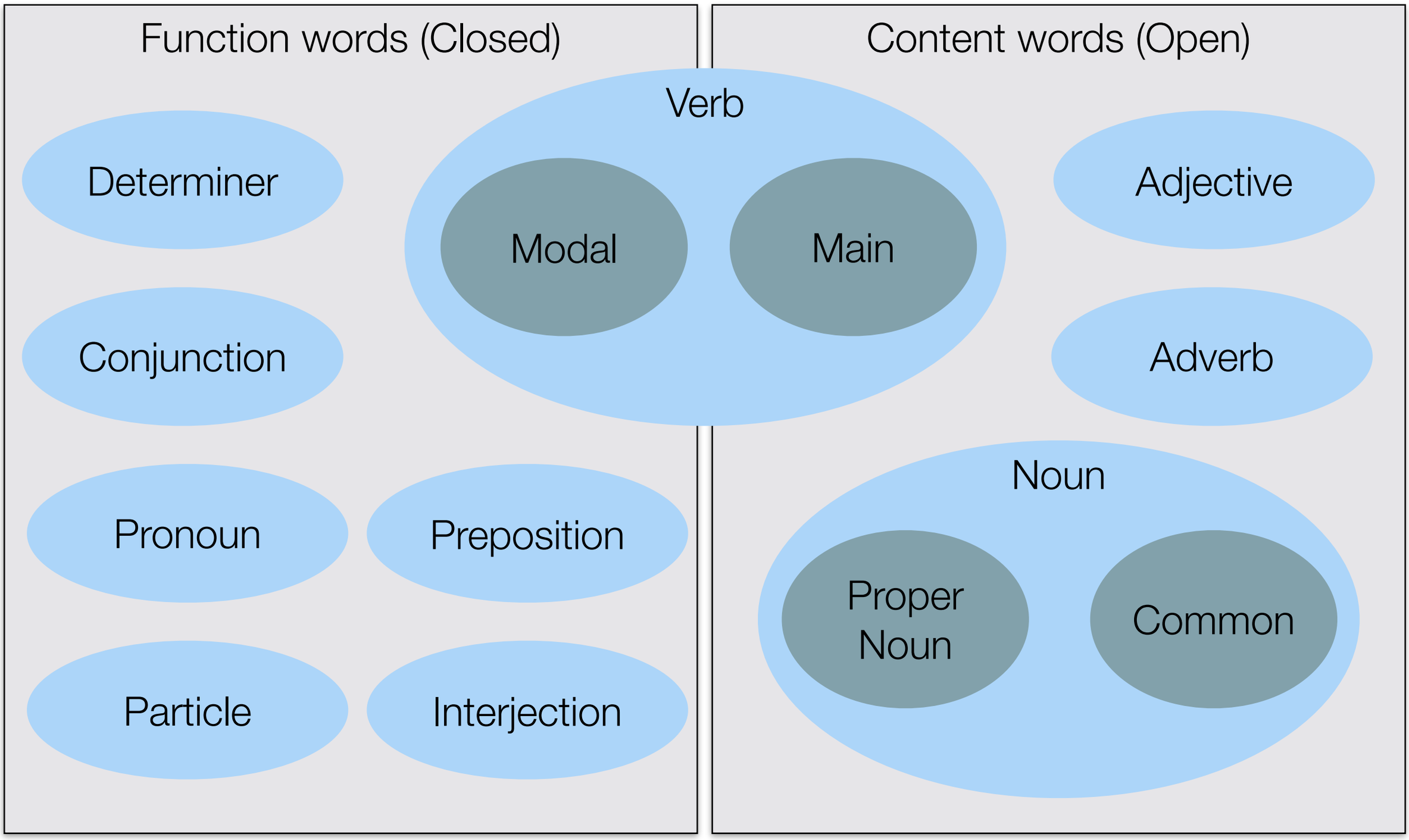
Elastic-Net

- A regularization method that combines both L_1 and L_2

$$Loss_{R_{EN}} = (y - y')^2 - (\lambda_1 ||W||_1 + \lambda_2 ||W||_2^2)$$

Part-of-Speech Tagging

Word Classes



Open Classes vs. Closed Classes

- Closed
  - Determiners: a, an, the
  - Pronouns: I, us, me, she, her, he, him, they, them
  - Prepositions: in, on, of, with, over, by…
- Open
  - Nouns
  - Verbs
  - Adjectives
  - Adverbs

Words with Multiple Part-of-Speech

- Word often have multiple part-of-speech
- POS tagging is performed to determine the POS tag for each word in a sentence.

Example	POS of ***back***
The **back** door	Adjective
On my **back**	Noun
Win the voters **back**	Adverb
Promised to **back** the bill	Verb

Full Part-of-Speech Tags

CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential *there*
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	*to*
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

Sizes of Tag Sets

- Penn Treebank, most widely used in computational work, is a simplified version of the Brown tag set.
- Many tag sets for other languages have also been developed.

Set	Number of Tags
Brown	179
Penn	45
CLAWS1	132
CLAWS2	166
CLAWS c5	62
London-Lund	197

Different Tag Sets

Words	CLAWS c5	Brown	Penn TB	ICE
She	PNP	PPS	PRP	PRON
was	VBD	BEDZ	VBD	AUX
told	VVN	VBN	VBN	V
that	CJT	CS	IN	CONJUNC
the	AT0	AT	DT	ART
journey	NN1	NN	NN	N
might	VM0	MD	MD	AUX
kill	VVI	VB	VB	V
her	PNP	PRO	PRP	PRON
.	PUN	.	.	PUNC

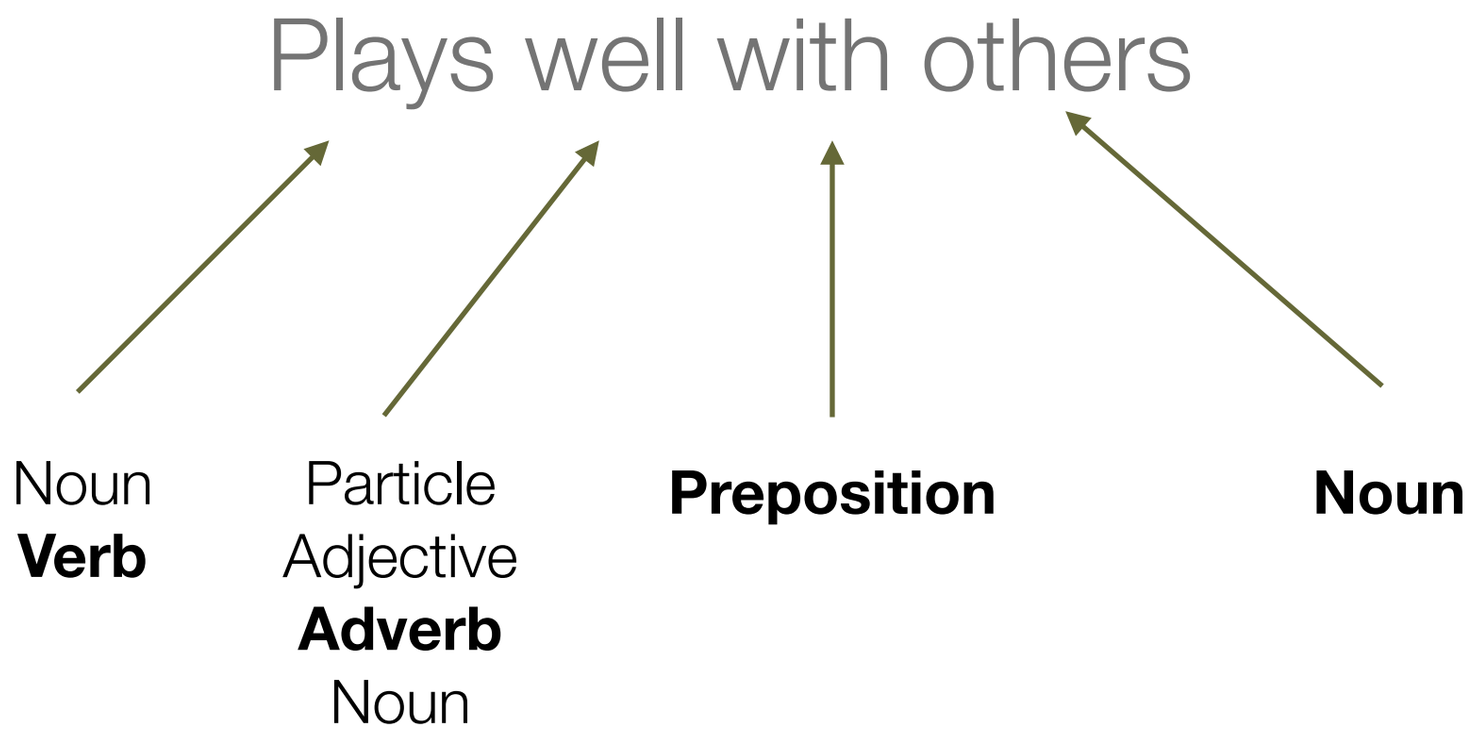
Universal Tagsets

- The intersection of part-of-speech in despite of specific languages.

Open class words	Closed class words	Other
Adjective	Pre/Post-position	Punctuation Mark
Adverb	Auxiliary verb	Symbol
Interjection	Coordinating Conjunction	Other
Noun	Determiner	
Proper Noun	Numeral	
Verb	Particle	
	Pronoun	
	Subordinating Conjunction	

Ambiguity of POS Tagging

- POS tagging is not a trivial task since it is inherent ambiguity.



Difficulty of POS Tagging

- About 40% of words are ambiguous
- The POS ambiguous words tend to be common words

Example	POS of ***that***
I can't believe **that** he's only 17.	Conjunction
I've got **that** pain in my back again.	Determiner
It costs **that** much!	Adverb
The people **that** live next door	Pronoun

Information for POS Tagging

- Information used to resolve the ambiguity of POS tagging
  - Knowledge of contextual words
    - Determiners are usually followed by nouns
- POS distribution of the word
  - *man* is less likely to be used as a verb

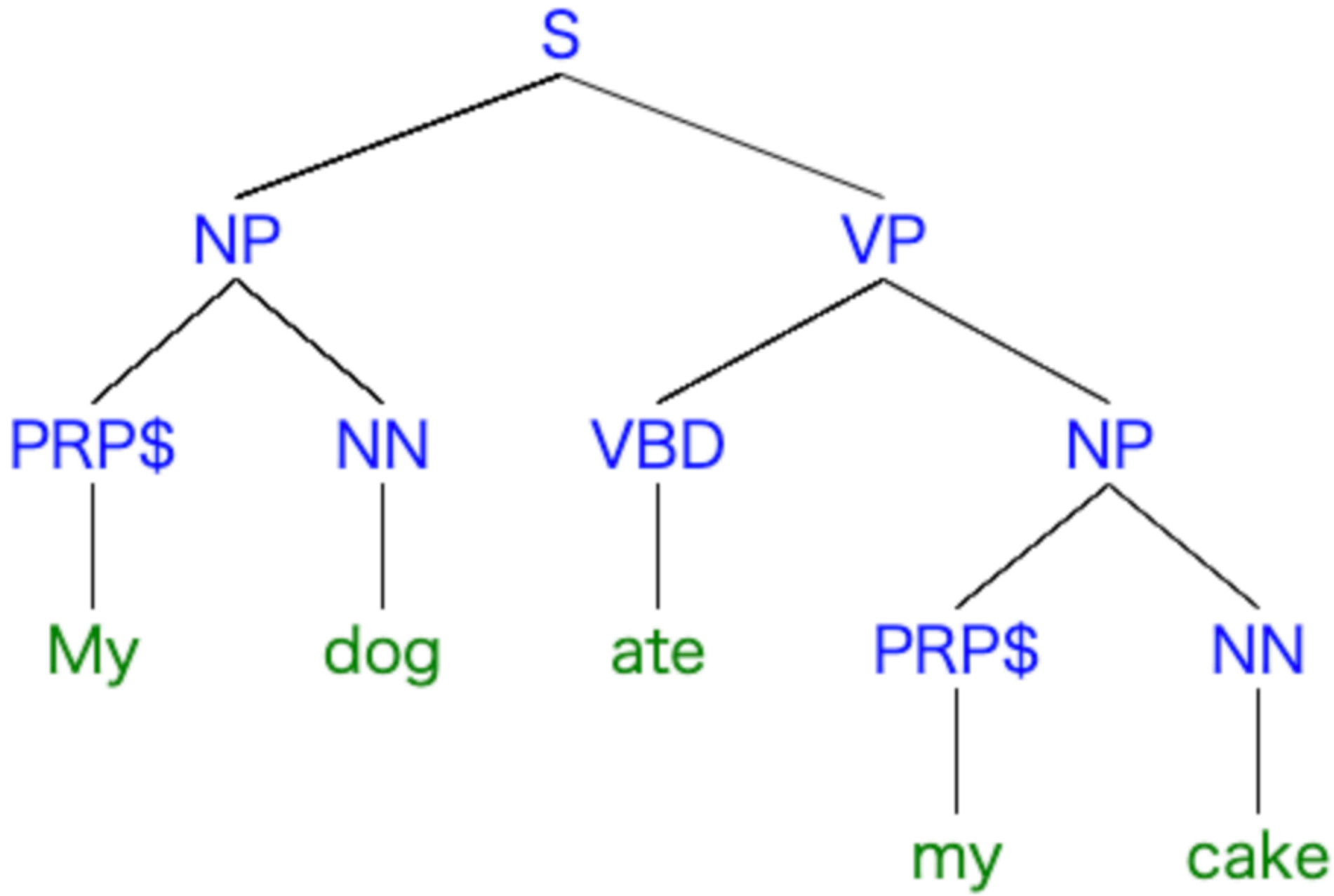
Wordform Information

- The capitalization/subword information of a word provides useful information.
    - Uppercased words: **A**merica -> Proper noun
    - Prefixes: **un**familiar -> Adjective
    - Suffixes: serious**ly** -> Adverb

Applications of POS Tagging

- Most used for subsequence applications
  - Machine translation:
    - Reordering of adjectives and nouns (Spanish to English)
  - Text-to-speech
    - The pronouncement of a word may be different according to its POS.
    - present [/prɪˈzent/ as verb] vs present [/ˈprezənt/ as noun and adjective]
- Providing fundamental information for syntactic/dependency parsers.

POS for Constitution Parsing



Models for Sequence Labeling

- Given a sequence $\boldsymbol{x}$, we would like to predict the label y_i for each element x_i in $\boldsymbol{x}$.

  - $\boldsymbol{x} = \{x_1, x_2, \ldots, x_T\}$ is a sequence of T words forming a sentence.

  - y_i is the part-of-speech tag of word x_i

  - $\boldsymbol{y} = \{y_1, y_2, \ldots, y_T\}$, the sequence of POS tags for $\boldsymbol{x}$

Probabilistic Sequence Labeling

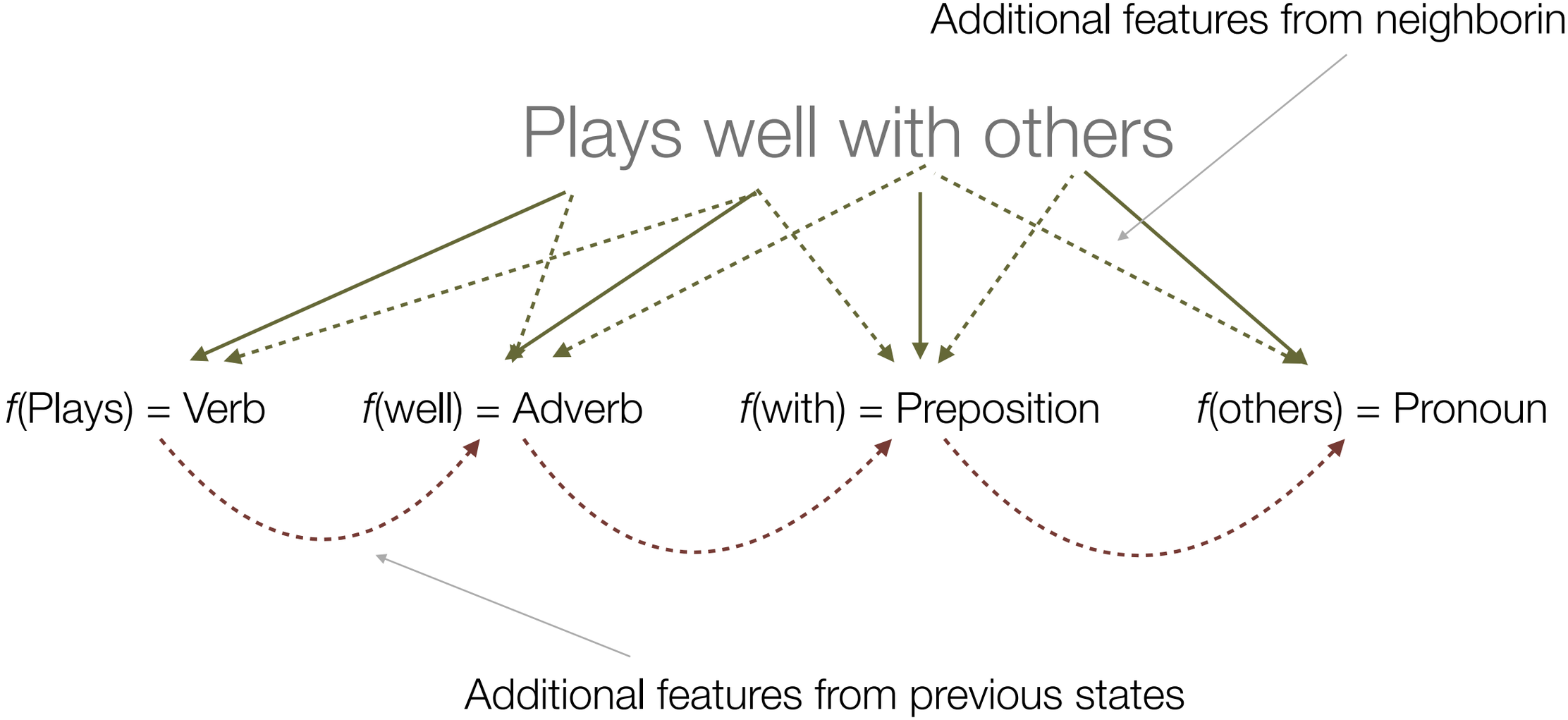
- Finding the sequence ***y*** with the highest probability given the input sequence ***x***.

$$\bar{\mathbf{y}} = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x})$$

- How to estimate $P(\boldsymbol{y}|\boldsymbol{x})$?
  - Hidden Markov Model
  - Maximum-entropy Markov model
  - Conditional random fields

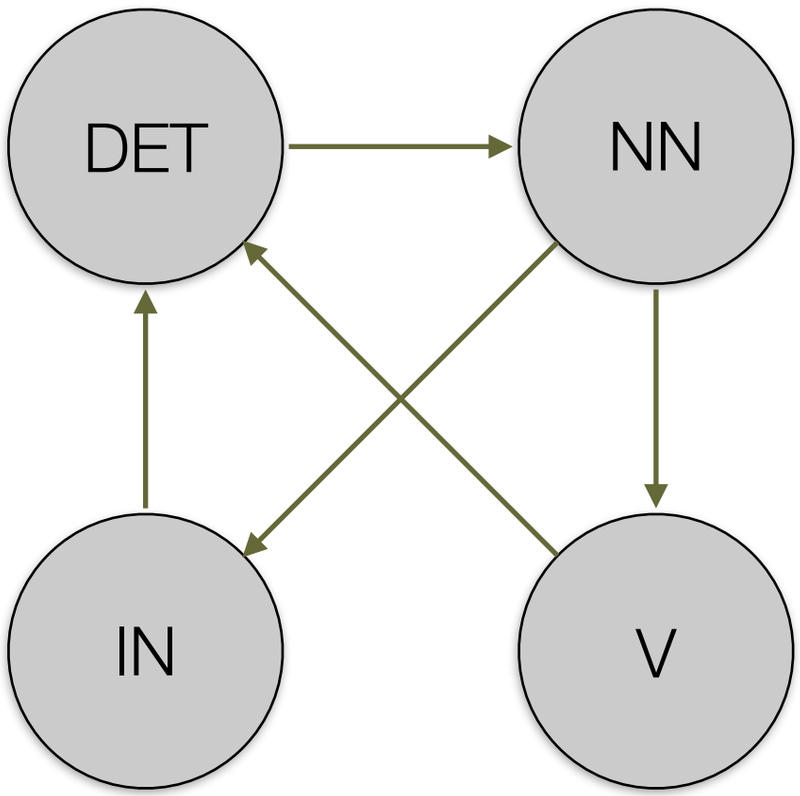
Sequence Labeling as Classification

- Sequence labeling is a classification task that classifies each element of a sequence by considering the information of its neighboring elements.



Hidden Markov Models

Markov Model



S → NP VP
NP → DET NN
NP → NP PP
VP → VP PP
VP → V NP
PP → IN NP
DET → the | a | an
NN → cat | mouse | run
V → ate | slept | run
IN → in | of | to

Limited horizon: $P(y_{t+1} = S_i|y_1, y_2, ..., y_t) = P(y_{t+1} = S_i|y_t)$

Time invariant: $P(y_{t+1} = S_i|y_t) = P(y_2 = S_i|y_1)$

First Order Markov Model

- Th state at time t+1 depends only on the state at time t.

$$P(y_{t+1} = S_i | y_1, y_2, ..., y_t) = P(y_{t+1} = S_i | y_t)$$

- The part of speech tag of x_{t+1} completely determined by x_t

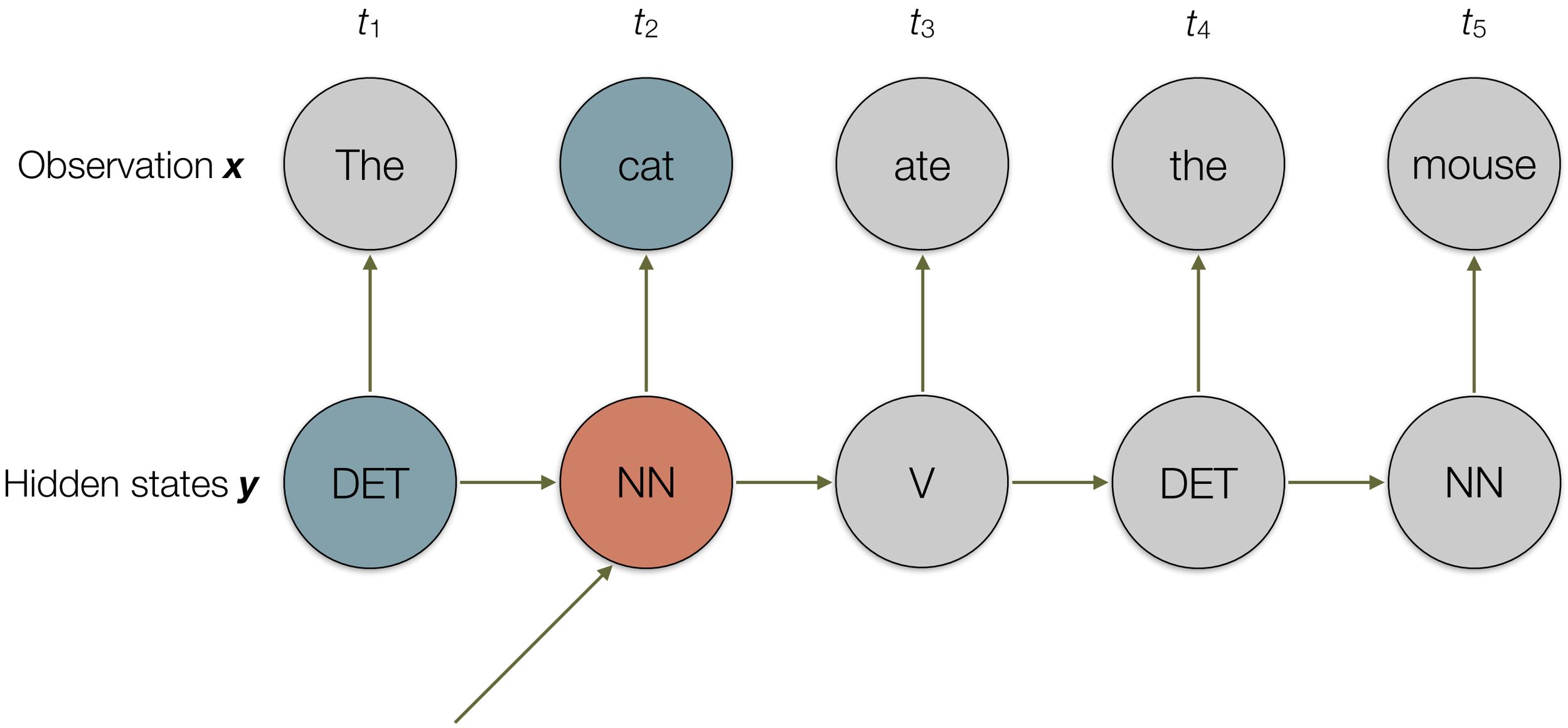
Second Order Markov Model

- Th state at time *t*+1 depends only on the state at time *t*.

$$P(y_{t+1} = S_i | y_1, y_2, ..., y_t) = P(y_{t+1} = S_i | y_{t-1}, y_t)$$

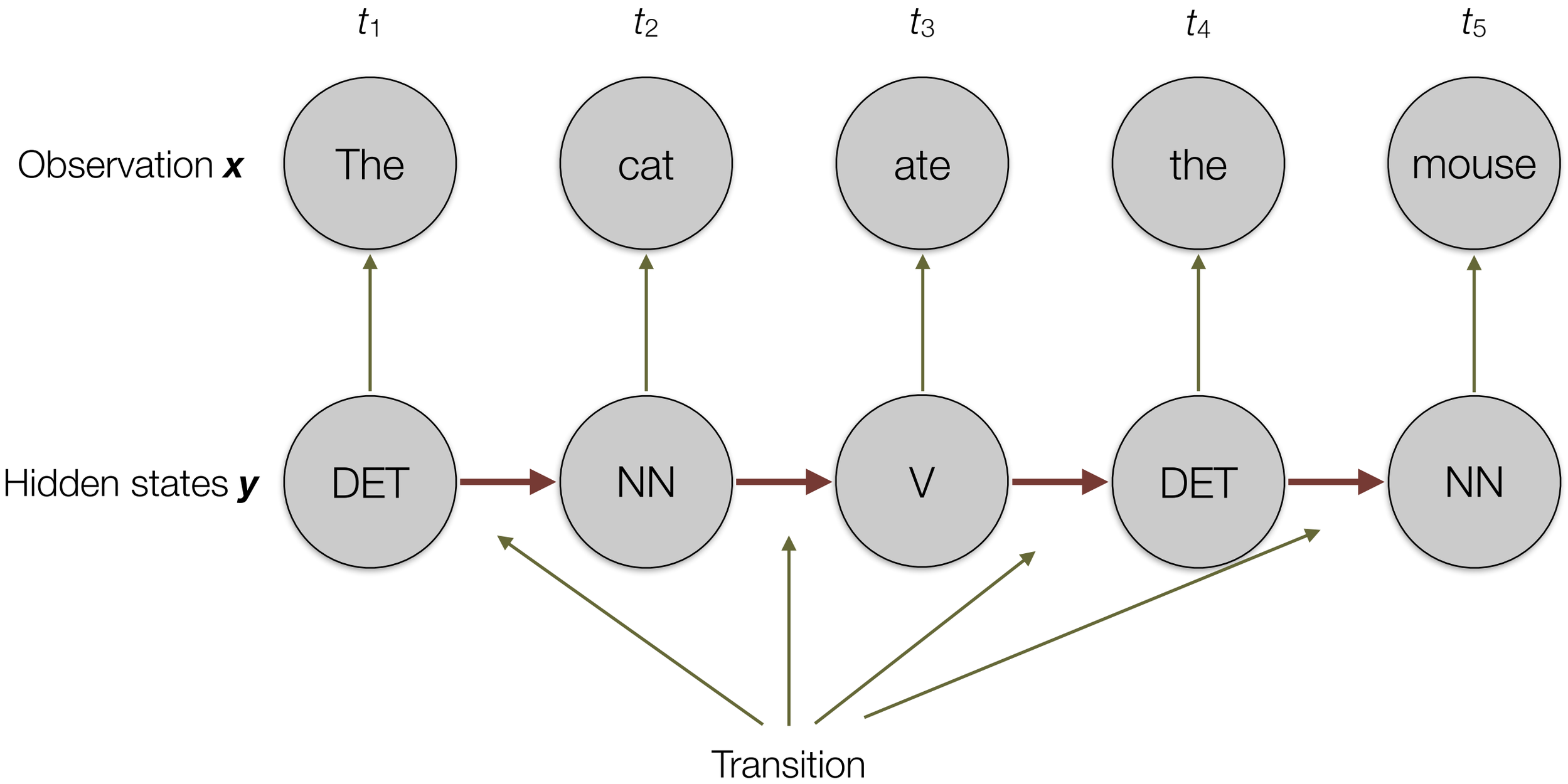
- The part of speech tag of x_{t+1} completely determined by x_{t-1} *and* x_t

Hidden Markov Model



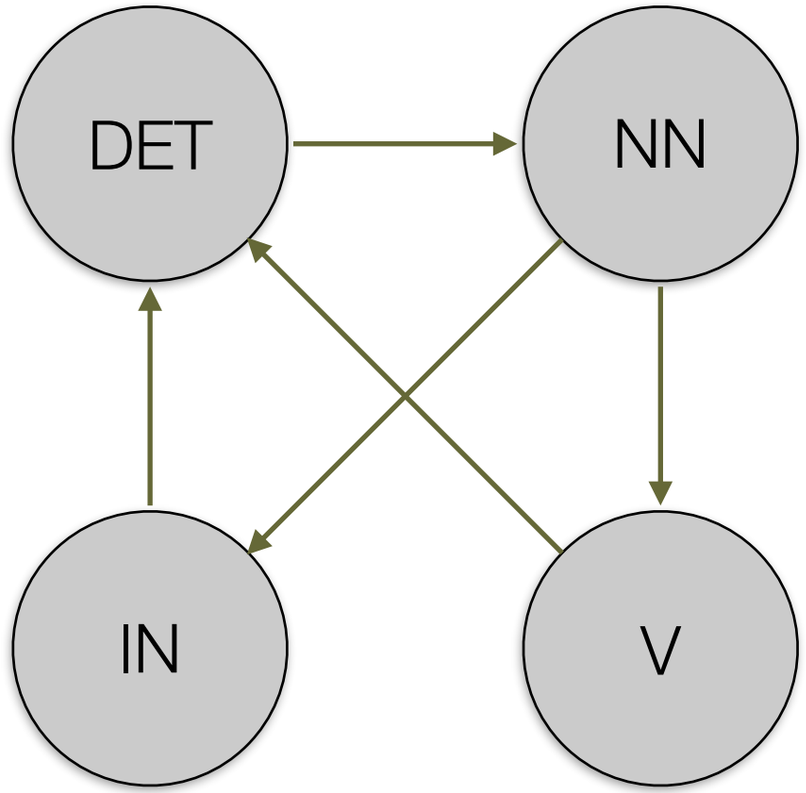
Each hidden state y_t is determined by its previous state y_{t-1} and its observation x_t

Transition between Hidden States



Transition Probabilities

a	DET	IN	V	NN
DET	0	0	0	1
IN	1	0	0	0
V	1	0	0	0
NN	0	0.3	0.7	0

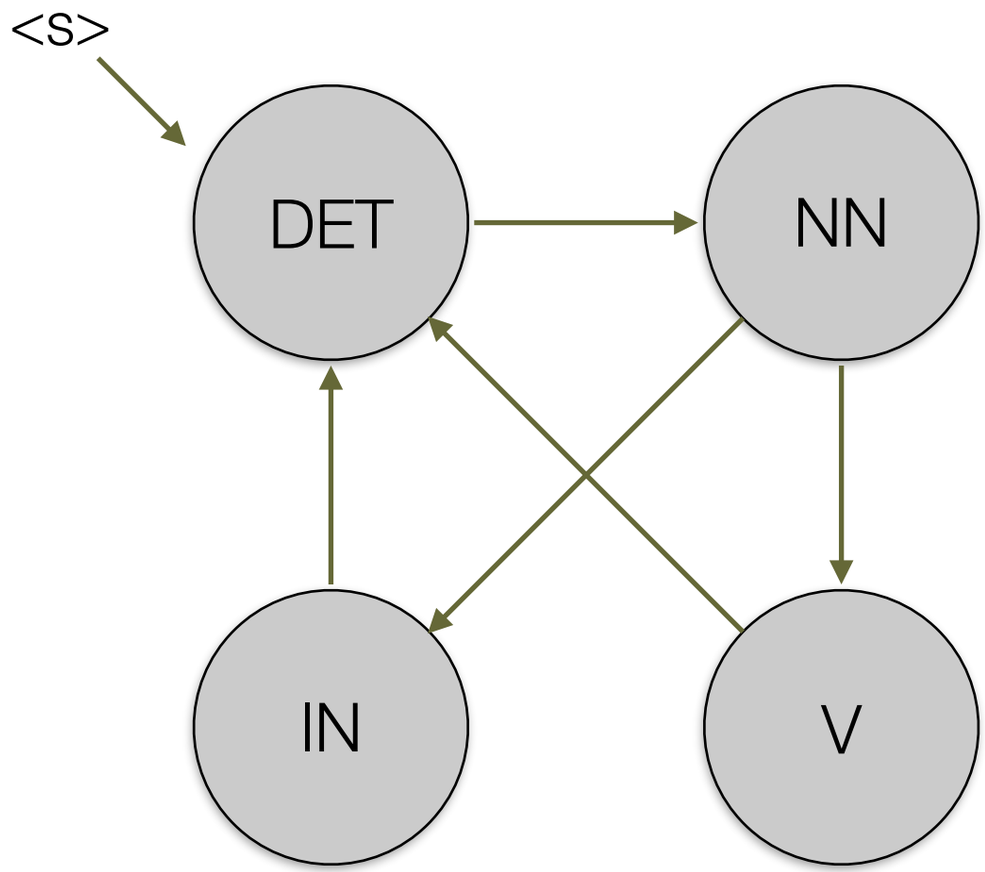


$$a_{ij} = P(y_{t+1} = S_j | y_t = S_i)$$

$$a_{ij} \geq 0 \text{ and } \sum_{j=1}^{M} a_{ij} = 1$$

Initial Probabilities

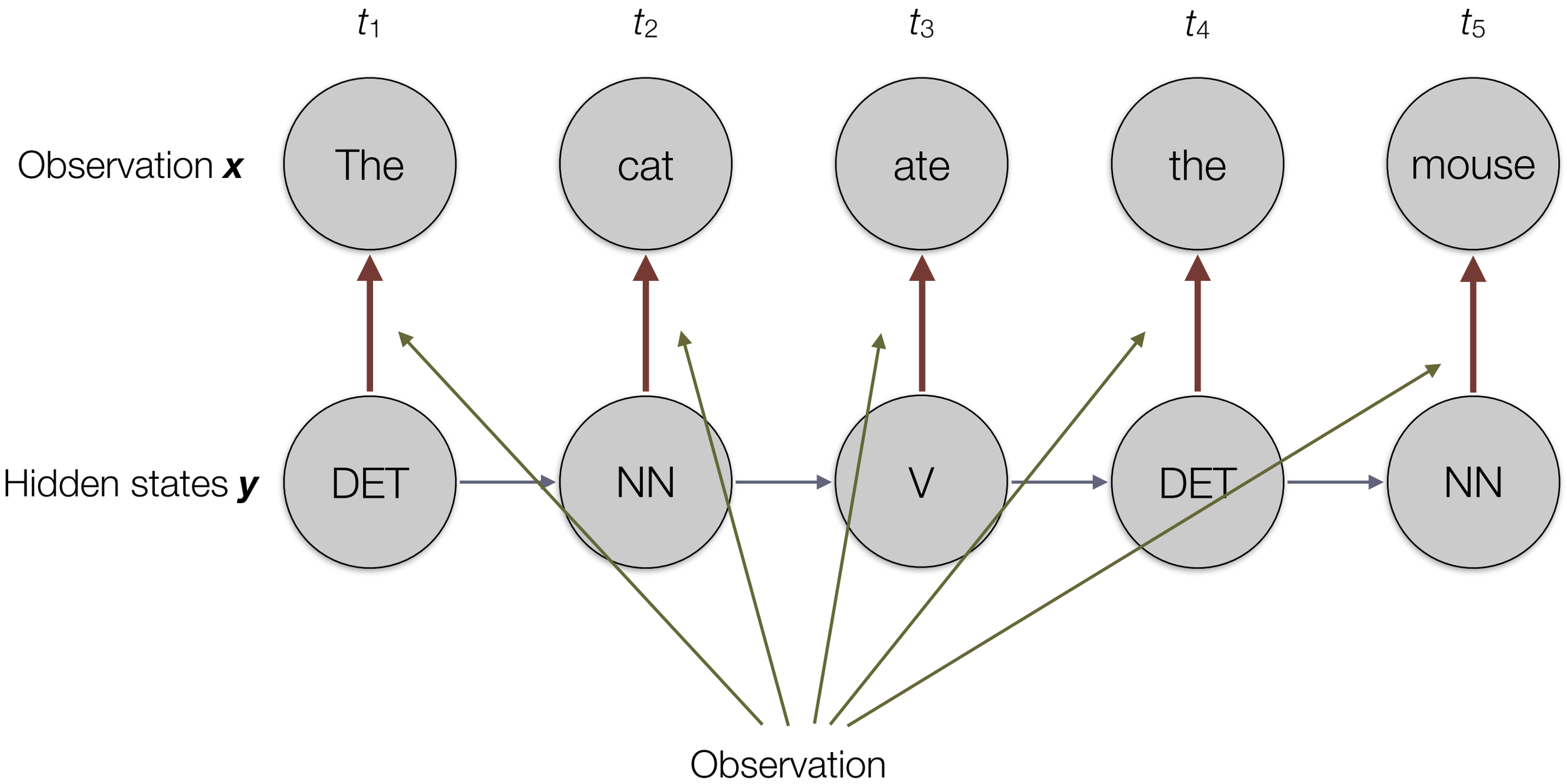
DET	1
IN	0
V	0
NN	0



$$\pi_i = P(y_1 = S_i)$$

$$\sum_{i=1}^{M} \pi_i = 1$$

Observation Probabilities



Observation Probabilities

	DET	NN	V	IN
the	0.5	0	0	0
a	0.4	0	0	0
an	0.1	0	0	0
cat	0	0.5	0	0
mouse	0	0.4	0	0
runs	0	0.1	0.3	0
ate	0	0	0.4	0
slept	0	0	0.3	0
in	0	0	0	0.3
of	0	0	0	0.4
to	0	0	0	0.3

DET → the | a | an
NN → cat | mouse | runs
V → ate | slept | runs
IN → in | of | to

$$b_j(w) = P(x_t = w | y_t = S_j)$$

Elements of a Hidden Markov Model

- *M* hidden states

$$S = \{S_1, S_2, S_3, ..., S_{|M|}\}$$

- Word set consisting of |*W*| words

$$W = \{w_1, w_2, w_3, ..., w_{|W|}\}$$

- Transition probabilities $R^{|M| \times |M|}$

$$\mathbf{A} = [a_{ij}] \text{ where } a_{ij} = P(y_{t+1} = S_j | y_t = S_i)$$

- Observation probabilities $R^{|M| \times |W|}$

$$\mathbf{B} = [b_j(w)] \text{ where } b_j(w) = P(x_t = w | y_t = S_j)$$

- Initial probabilities

$$\mathbf{\Pi} = [\pi_i] \text{ where } \pi_i = P(y_1 = S_i)$$

$$\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$$

Basic Tasks of HMMs

- Given a model λ, evaluate the probability of any given observation sequence $\boldsymbol{x} = \{x_1, x_2, x_3, \ldots, x_T\}$
  - **Evaluate** $P(\boldsymbol{x}|\lambda)$
- Given a model λ and an observation sequence $\boldsymbol{x} = \{x_1, x_2, x_3, \ldots, x_T\}$, predict the hidden states with the highest probability of generating $\boldsymbol{x}$.
  - **Predict** $\boldsymbol{y}$**'** that maximizes $P(\boldsymbol{y}|\boldsymbol{x}, \lambda)$.
- Given a training set of n observation sequences $\boldsymbol{X} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \boldsymbol{x}^3, \ldots, \boldsymbol{x}^n\}$, learn the model that maximizes the probability of generating $\boldsymbol{X}$.
  - **Learn** λ' that maximizes $P(\boldsymbol{X}|\lambda)$

Predicting the State Sequence

- Finding the state sequence $\boldsymbol{y} = \{y_1, y_2, y_3, \ldots, y_T\}$ with the highest probability of generating the observation sequence $\boldsymbol{x} = \{x_1, x_2, x_3, \ldots, x_T\}$ given the model λ.

- Define as $\delta_t(i)$ as the probability of the highest probability path at time t that accounts for the first t observations and ends in S_i

$$\delta_t(i) = \max_{y_1, y_2, \ldots, y_{t-1}} P(y_1, y_2, ..., y_{t-1}, y_t = S_i, x_1, x_2, ..., x_t | \lambda)$$

- Calculate $\delta_t(i)$ recursively with the Viterbi algorithm

Visible Markov Model

- Learning the parameters from labeled data directly.

$$\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$$

- Transition probabilities $R^{|M| \times |M|}$

$$\mathbf{A} = [a_{ij}] \text{ where } a_{ij} = P(y_{t+1} = S_j | y_t = S_i)$$

- Observation probabilities $R^{|M| \times |W|}$

$$\mathbf{B} = [b_j(w)] \text{ where } b_j(w) = P(x_t = w | y_t = S_j)$$

- Initial probabilities

$$\mathbf{\Pi} = [\pi_i] \text{ where } \pi_i = P(y_1 = S_i)$$

Johnson 2007

Parameters of HMMs

A	DET	IN	V	NN
DET	0	0	0	1
IN	1	0	0	0
V	0.8	0.2	0	0
NN	0	0.2	0.7	0.1

DET	1
IN	0
V	0
NN	0

B	DET	NN	V	IN
the	0.5	0	0	0
a	0.4	0	0	0
an	0.1	0	0	0
cat	0	0.5	0	0
mouse	0	0.4	0	0
runs	0	0.1	0.3	0
ate	0	0	0.4	0
slept	0	0	0.3	0
in	0	0	0	0.3
of	0	0	0	0.4
to	0	0	0	0.3

Parameters of HMMs

- The relation between the POS of the current word and that of its previous word.

A	DET	IN	V	NN
DET	0	0	0	1
IN	1	0	0	0
V	1	0	0	0
NN	0	0.3	0.7	0

$$a_{ij} = P(S_j|S_i) = \frac{C(S_i S_j)}{C(S_i)}$$

$C(S_i)$: Number of occurrences of S_i

$C(S_i S_j)$: Number of occurrences of S_i followed by S_j

Parameters of HMMs

- The POS of the word at the beginning of a sentence.

DET	1
IN	0
V	0
NN	0

$$\pi_i = P(y_1 = S_i) = \frac{C(y_1^k = S_i)}{N}$$

y_1^k : POS tag of the first word in the sentence k in the dataset

N : Number of sentences in the dataset

Parameters of HMMs

- The distribution of POS tags of a word.

B	DET	NN	V	IN
the	0.5	0	0	0
a	0.4	0	0	0
an	0.1	0	0	0
cat	0	0.5	0	0
mouse	0	0.4	0	0
runs	0	0.1	0.3	0
ate	0	0	0.4	0
slept	0	0	0.3	0
in	0	0	0	0.3
of	0	0	0	0.4
to	0	0	0	0.3

$$b_j(w) = P(w|S_j) = \frac{C(w, S_j)}{C(S_j)}$$

$C(w, S_j)$: Number of occurrences of the word w labeled with the POS tag S_j

w : A word in the vocabulary

Information for POS Tagging

- Information used to resolve the ambiguity of POS tagging
  - Knowledge of contextual words
    - Determiners are usually followed by nouns

$$a_{det,noun} > a_{det,verb}$$

$$\pi_{det} > \pi_{verb}$$

- POS distribution of the word
  - *man* is less likely to be used as a verb

$$b_{noun}(\text{man}) > b_{verb}(\text{man})$$

Viterbi Algorithm

Viterbi Algorithm

$$\delta_t(i) = \max_{y_1, y_2, \ldots, y_{t-1}} P(y_1, y_2, ..., y_{t-1}, y_t = S_i, x_1, x_2, ..., x_t | \lambda)$$

1. Initialization:

$$\delta_1(i) = \pi_i b_i(x_1)$$

← Base case for $t = 1$

$$\psi_1(i) = 0$$

← No previous state

2. Recursion:

$$\delta_t(j) = \max_i^M \delta_{t-1}(i) a_{ij} \cdot b_j(x_t)$$

← General case for $t > 1$

$$\psi_t(j) = \arg\max_i^M \delta_{t-1}(i) a_{ij} \cdot b_j(x_t)$$

← Best i for $\delta_t(j)$ denoting the best previous state for $y_t = S_j$

3. Termination:

$$P(\mathbf{y}|\mathbf{x}, \lambda) = \max_i^M \delta_T(i)$$

$$\bar{\mathbf{y}} = \arg\max_i^M \delta_T(i)$$

$$\bar{y}_t = \psi_{t+1}(\bar{y}_{t+1}), t = T-1, T-2, ..., 1$$

A	DET	IN	V	NN
DET	0	0	0	1
IN	1	0	0	0
V	0.8	0.2	0	0
NN	0	0.2	0.7	0.1

DET	1
IN	0
V	0
NN	0

B	DET	NN	V	IN
the	0.5	0	0	0
a	0.4	0	0	0
an	0.1	0	0	0
cat	0	0.5	0	0
mouse	0	0.4	0	0
runs	0	0.1	0.3	0
ate	0	0	0.4	0
slept	0	0	0.3	0
in	0	0	0	0.3
of	0	0	0	0.4
to	0	0	0	0.3

	the	cat	runs	to	the	mouse
DET	0.5	0	0	0	0.001575	0
IN	0	0	0	0.00315	0	0
V	0	0	0.0525	0	0	0
NN	0	0.25	0.0025	0	0	0.00063

Maximum-Entropy Markov Model

Limitation of HMMs

- The assumption of HMM is very limited
  - y_t is determined by only y_{t-1} and x_t.
- Many kind of information are missing
  - x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2}
  - Position of t (the first word, the second word, the last word, etc)
  - Subword information of x_t such as its suffix and prefix

Maximum-Entropy Markov Model

- Maximum-entropy model is actually the multi-class logistic regression model.

- Maximum-entropy Markov model

- Markov model with logistic regression classifier

Information of entire ***x*** can be used at any time *t*

$$P_{MEMM}(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{T} P(y_t|y_{t-1}, \mathbf{x})$$

← Logistic regression classifier

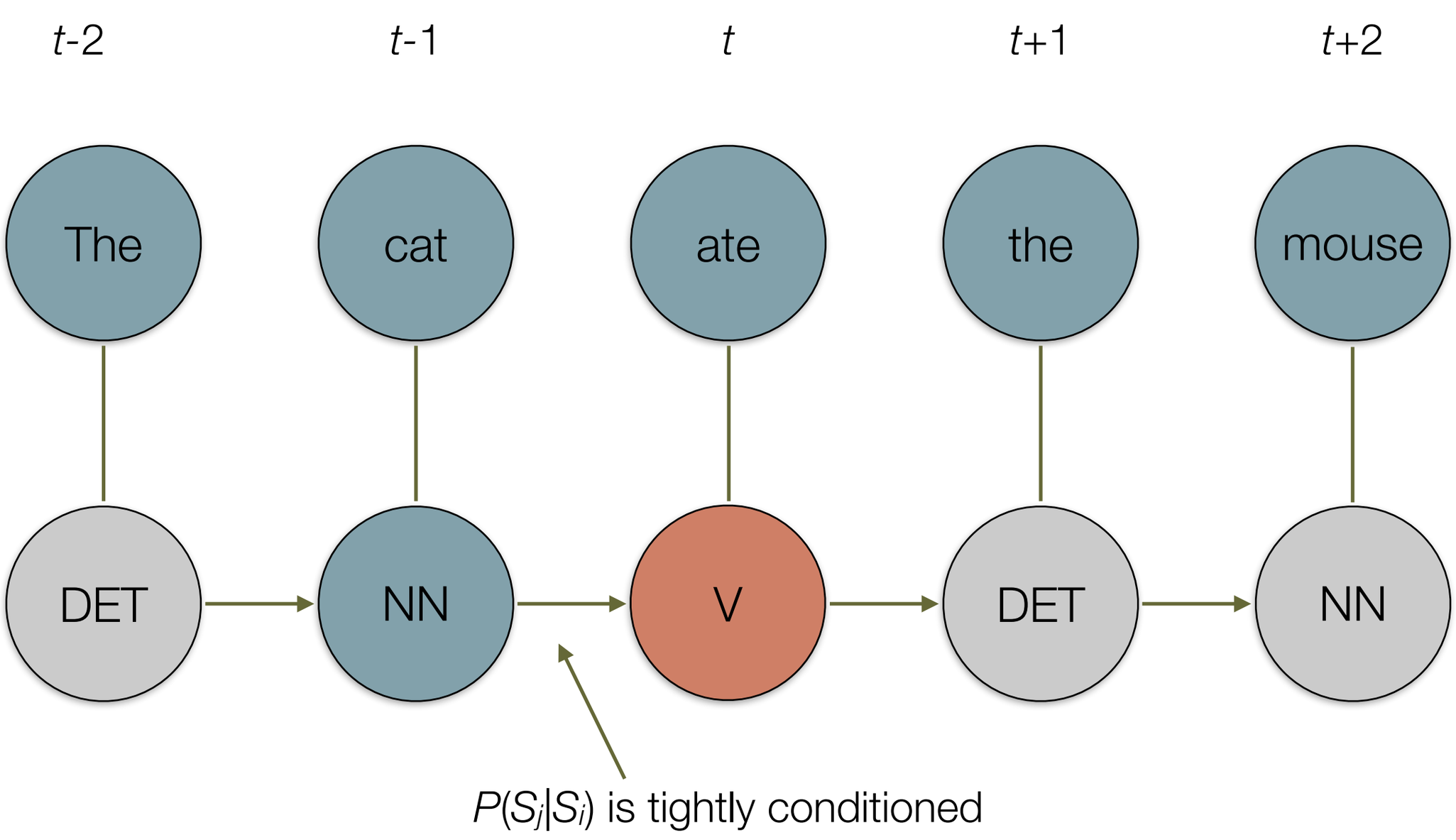
$$P(y_t|y_{t-1}, \mathbf{x}) = \frac{1}{Z_t(y_{t-1}, \mathbf{x})} \exp\left[\sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t)\right]$$

Normalization term

Weight of the feature *k*

Occurrence of the feature *k*

Maximum-Entropy Markov Model



Future states cannot affect the posterior distribution over earlier states

Features of Markov Maximum Entropy Model

- Without the assumption of statistical independency
- Various (dependent) features for each position *t* can be considered.
  - $f_i(yt, y_{t\text{-}1}, x_t)$: 1 if x_t is in the uppercase and y_t is Proper Noun
  - $f_j(y_t, y_{t\text{-}1}, x_t)$: 1 if $x_{t\text{-}1}$ ends without 's', x_t ends with 's', $y_{t\text{-}1}$ is Noun, and y_t is Verb

$$P_{MEMM}(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{T} P(y_t|y_{t-1}, \mathbf{x})$$

$$P(y_t|y_{t-1}, \mathbf{x}) = \frac{1}{Z_t(y_{t-1}, \mathbf{x})} \exp\left[\sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t)\right]$$

Weight of each feature is obtain with MLE

Conditional Random Fields

- Most popular model for sequence labeling

Information of entire ***x*** can be used at any time *t*

$$P_{CRF}(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{T} P(y_t|y_{t-1}, \mathbf{x})$$

← Logistic regression classifier

$$P(y_t|y_{t-1}, \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left[\sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t)\right]$$

Normalization term sum over labels of an entire sequence

Weight of the feature *k*

Occurrence of the feature *k*

Key difference between MMEM and CRF

MEMM vs CRF

$$P_{MEMM}(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{T} P(y_t|y_{t-1}, \mathbf{x})$$

$$P(y_t|y_{t-1}, \mathbf{x}) = \frac{1}{Z_t(y_{t-1}, \mathbf{x})} \exp\left[\sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t)\right]$$

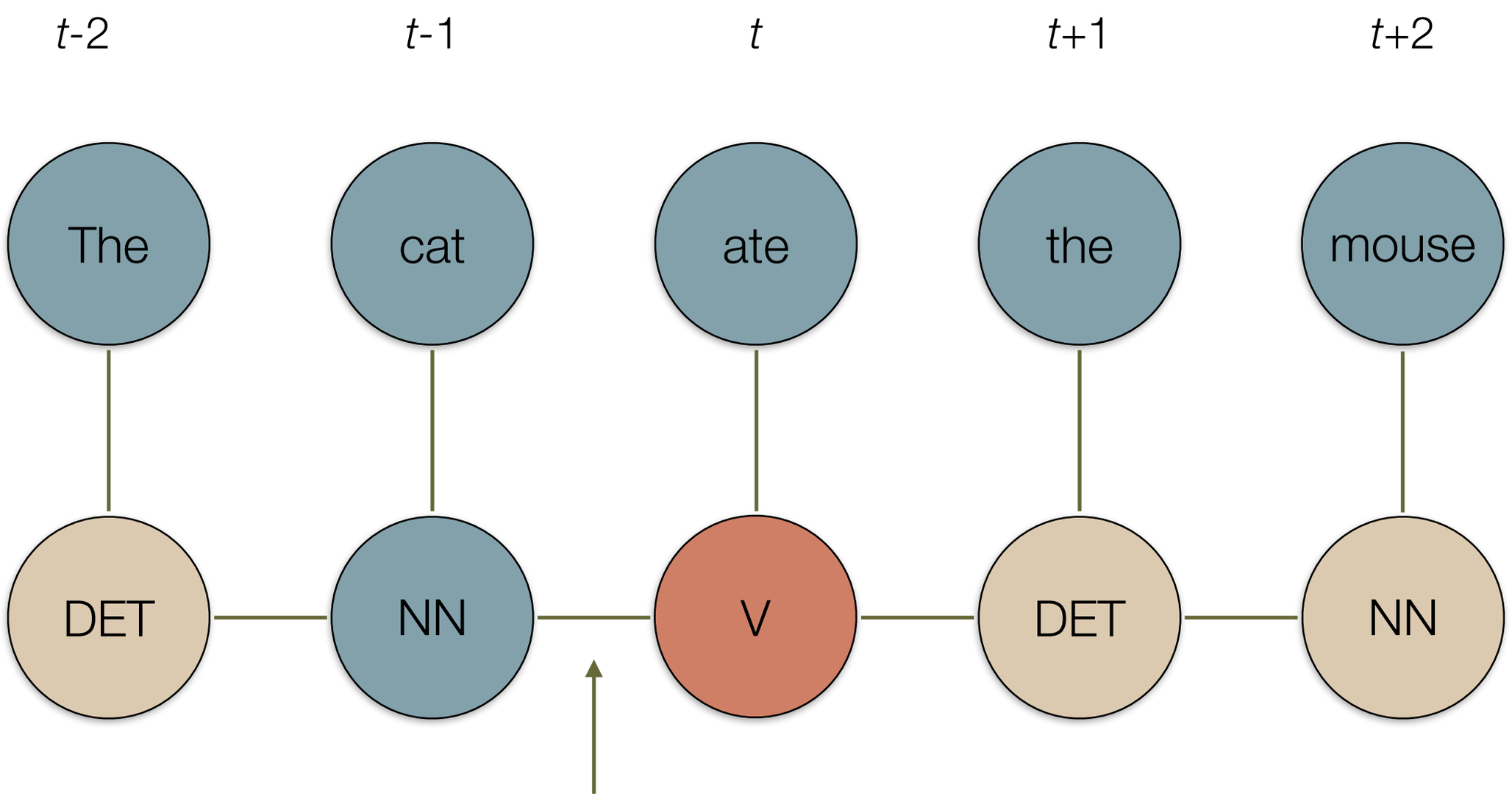
Normalization term of the sequence at ***t***

$$P_{CRF}(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{T} P(y_t|y_{t-1}, \mathbf{x})$$

$$P(y_t|y_{t-1}, \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left[\sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t)\right]$$

Normalization term sum over labels of an entire sequence ***x***

Conditional Random Fields



Decoding by the Viterbi Algorithm

$$P_{CRF}(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{T} P(y_t|y_{t-1}, \mathbf{x})$$

$$P(y_t|y_{t-1}, \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left[\sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t)\right]$$

$$\bar{\mathbf{y}} = \arg\max P_{CRF}(\mathbf{y}|\mathbf{x})$$

max(P(IN | V, $\mathbf{x}$), P(IN | NN, $\mathbf{x}$)) P(DET | IN, $\mathbf{x}$)

	<s>	the	cat	runs	to	the	mouse
<s>	1	0	0	0	0	0	0
DET	0	1	0	0	0	0.098	0
IN	0	0	0	0	0.098	0	0
V	0	0	0	0.49	0	0	0
NN	0	0	1	0.03	0	0	0.098

Feature Engineering for MEMM and CRFs

Sequence Labeling

x
My
dog
ate
my
cake
.

y
PRP
NN
VBD
PRP
NN
PU

Features for Sequence Labeling

x
My
dog
ate
my
cake
.

x_t	x_{t-1}	x_{t+1}	x_{t-1}, x_t	x_t, x_{t+1}	Capital Initial	Punct	Begin	End	Shape
my	<s>	dog	<s>, my	my dog	1	0	1	0	Xx
dog	my	ate	My dog	dog ate	0	0	0	0	xxx
ate	dog	my	dog ate	ate my	0	0	0	0	xxx
my	ate	cake	ate my	my cake	0	0	0	0	xx
cake	my	.	my cake	cake .	0	0	0	0	xxxx
.	cake	</s>	cake .	. </s>	0	1	0	1	.

y
PRP
NN
VBD
PRP
NN
PU

Feature Templates

x
My
dog
ate
my
cake
.

Templates	Patterns	Examples
Unigrams	x_t x_{t-1} x_{t-2} x_{t-3} ... x_{t+1} x_{t+2} x_{t+3} ...	ate dog My <s> my cake .
Bigrams	$x_t x_{t+1}$ $x_{t-1} x_t$ $x_{t-2} x_{t-1}$ $x_{t+1} x_{t+2}$ ...	ate/my dog/ate My/dog my/cake
Trigram	$x_{t-1} x_t x_{t+1}$ $x_t x_{t+1} x_{t+2}$ $x_{t-2} x_{t-1} x_t$ $x_{t-3} x_{t-2} x_{t-1}$ $x_{t+1} x_{t+2} x_{t+3}$	dog/ate/my ate/my/cake my/dog/ate <s>/my/dog my/cake/.
Variance	$x_{t-1} x_{t+1}$ $x_{t-2} x_{t+1}$ $x_{t-1} x_{t+2}$	dog/my My/my dog/cake

In addition to the character level, more information could be added as features.

Feature Extraction for CRFs

- Extracting features for a sequence (an instance)

```
def extract_sent_features(x):
    sent_features = []
    for i in range(len(x)):
        sent_features.append(extract_char_features(x, i))
    return sent_features
```

- Extracting features for each unit
  - Word for POS tagging or NER recognition
  - Character for Chinese word segmentation

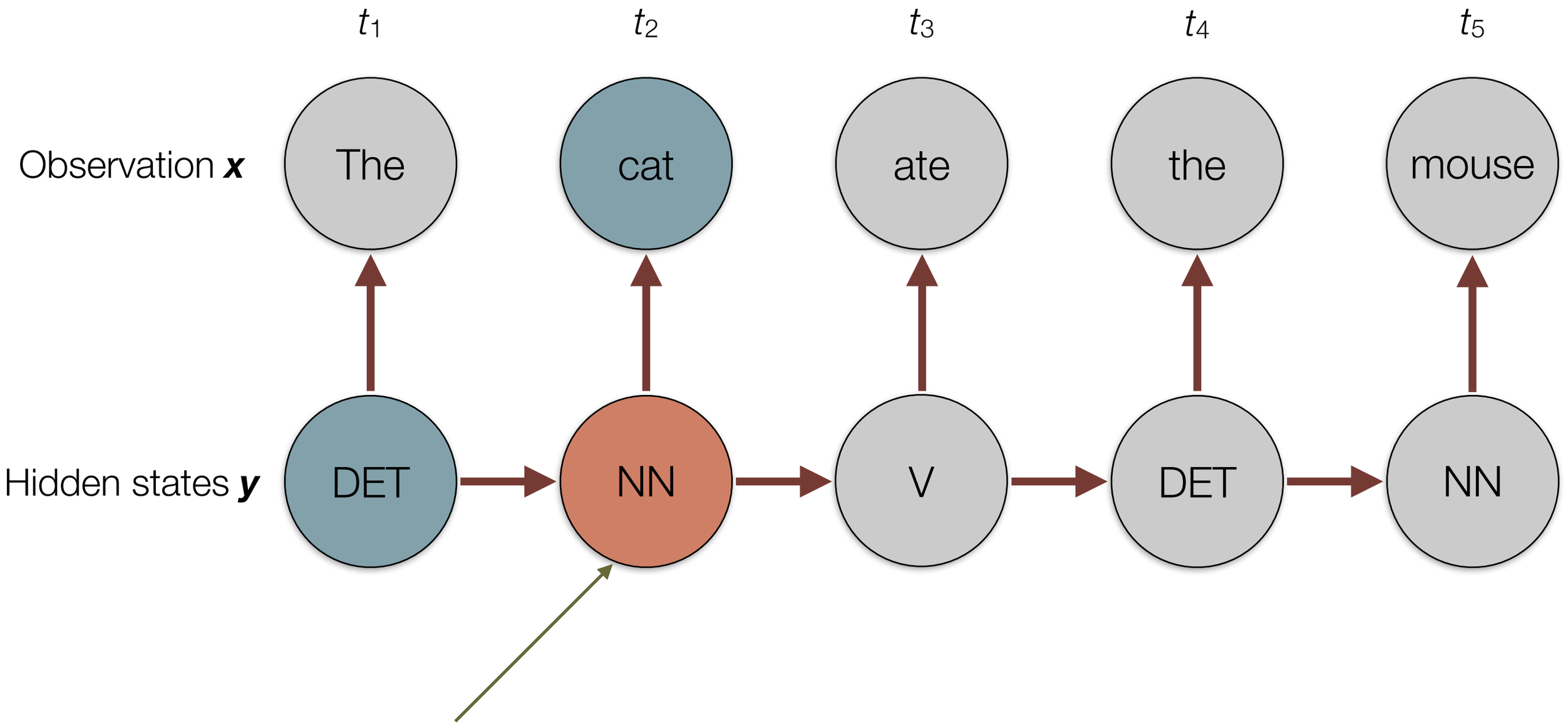
```
def extract_char_features(sent, position):
    char_features = {}
    for i in range(-3, 4):
        if len(sent) > position + i >= 0:
            char_features['char_at_%d' % i] = sent[position + i]
    return char_features
```

Only unigram features x_{t-3}, x_{t-2}, x_{t-1}, x_t, x_{t+1}, x_{t+2}, x_{t+3}

Summary

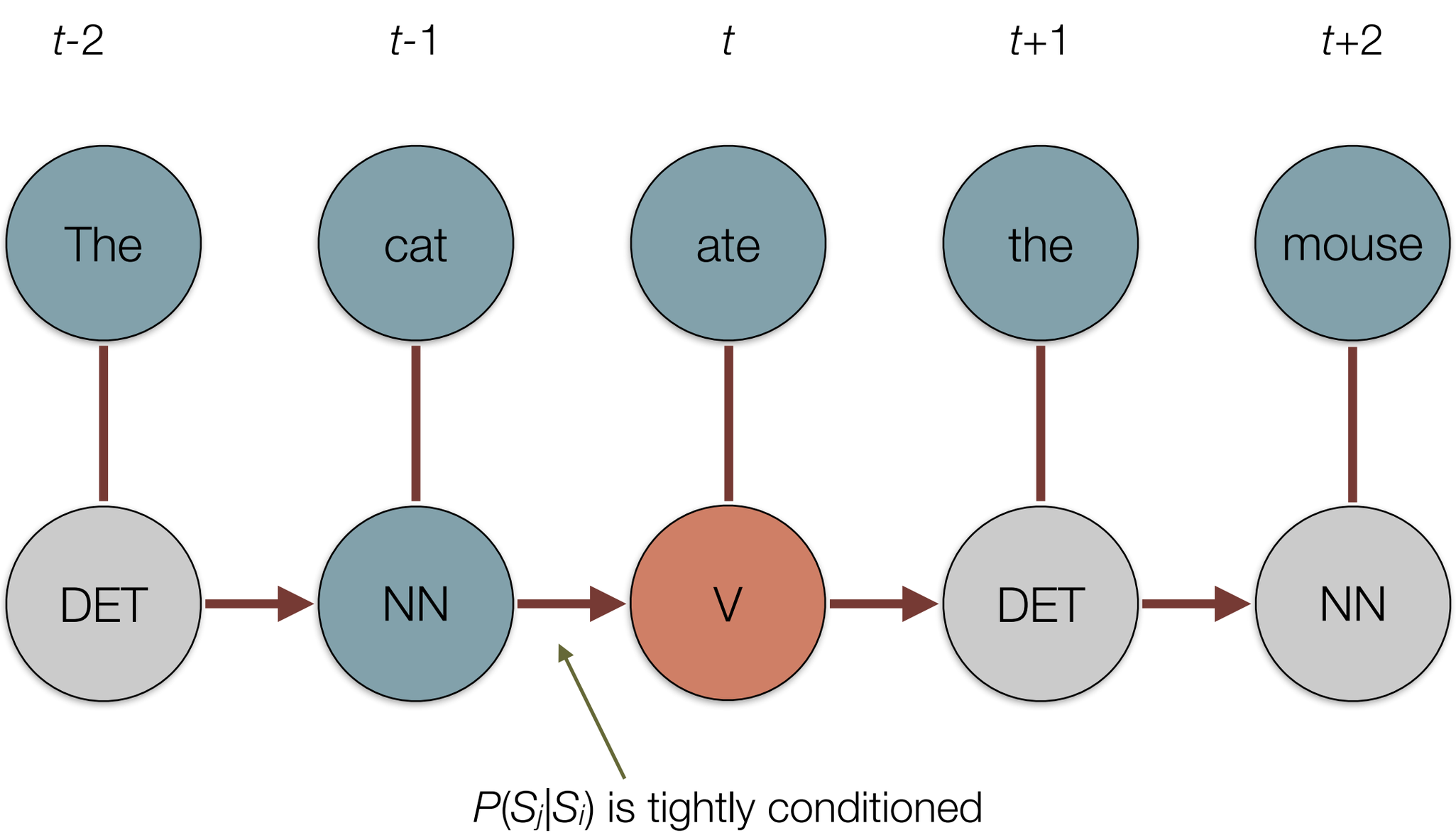
Model	Pros	Cons
HMM	• Simple to train • Friendly for unsupervised learning	y_t only depends on y_{t-1} and x_t
MEMM	• Information of entire ***x*** can be used to decide y_t • Local optimal • Better performances	High complexity of training
CRF	• Information of entire ***x*** can be used to decide y_t • Global optimal • Even better than MEMM	Higher complexity of training

Hidden Markov Model



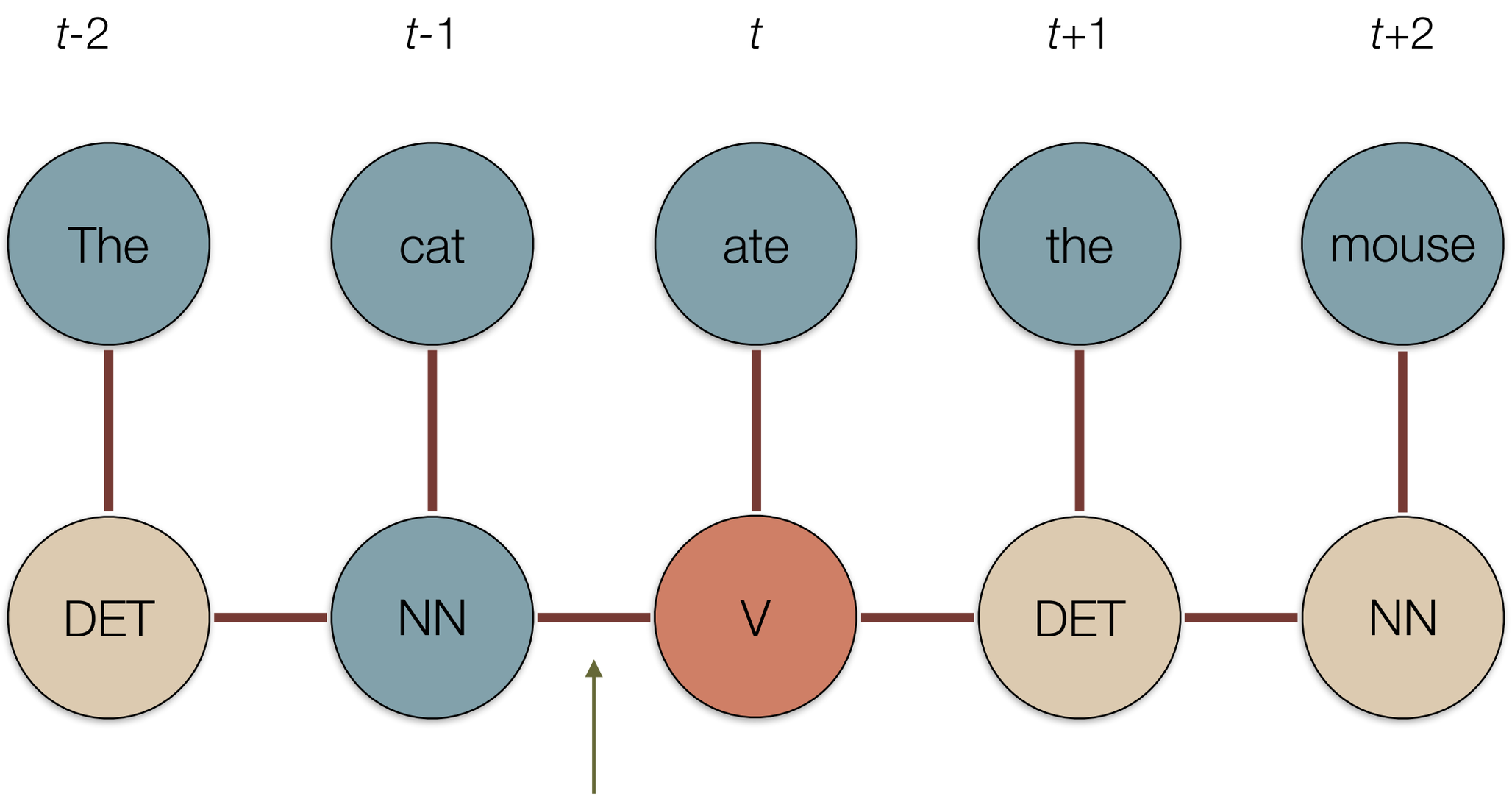
Each hidden state y_t is determined by its previous state y_{t-1} and its observation x_t

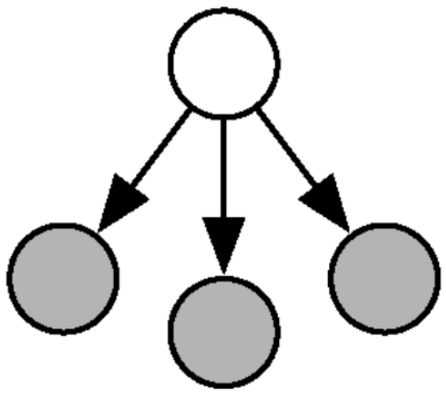
Maximum-Entropy Markov Model



Future states cannot affect the posterior distribution over earlier states

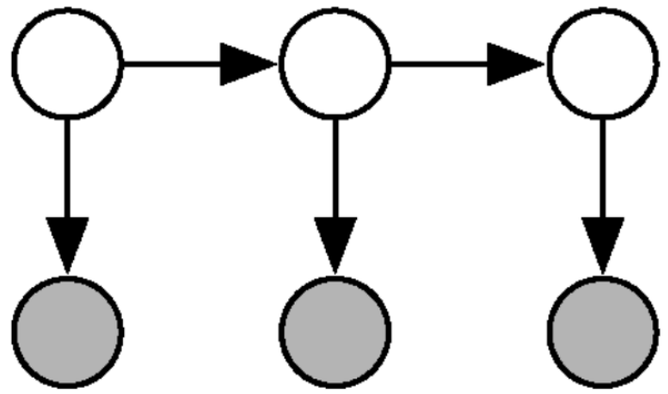
Conditional Random Fields





Naive Bayes

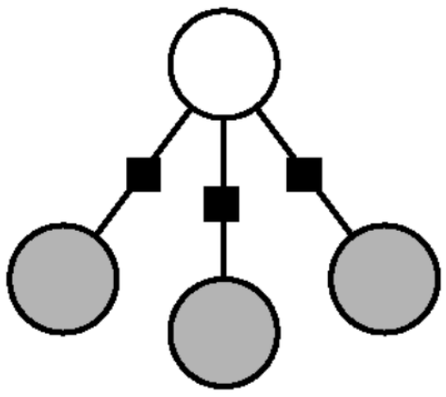
SEQUENCE



HMMs

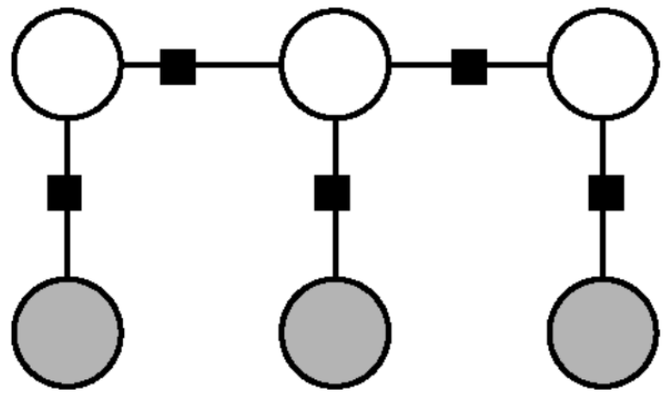
CONDITIONAL

CONDITIONAL



Logistic Regression

SEQUENCE



Linear-chain CRFs

Sutton & McCallum, 2011