A Kernel-based Approach to Sequence Labeling Problems

(Joint work with ***Yuta Tsuboi***)

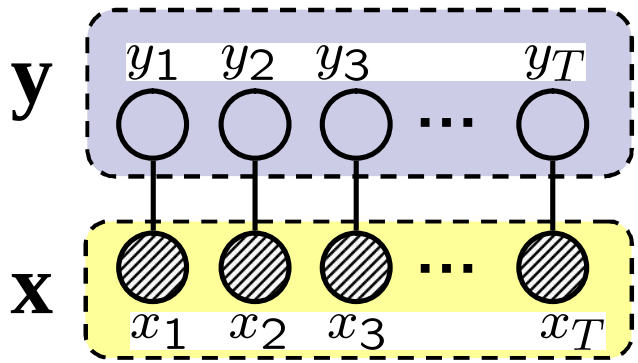
Hisashi Kashima
IBM Research
Tokyo Research Laboratory
hkashima@jp.ibm.com

Outline

- Sequence labeling problems
- Hidden Markov model (HMM)
- Hidden Markov (HM) perceptron
- Marginalized labeling perceptron
- Experiments on natural language processing tasks
- Application to bioinformatics

Sequence Labeling Problems

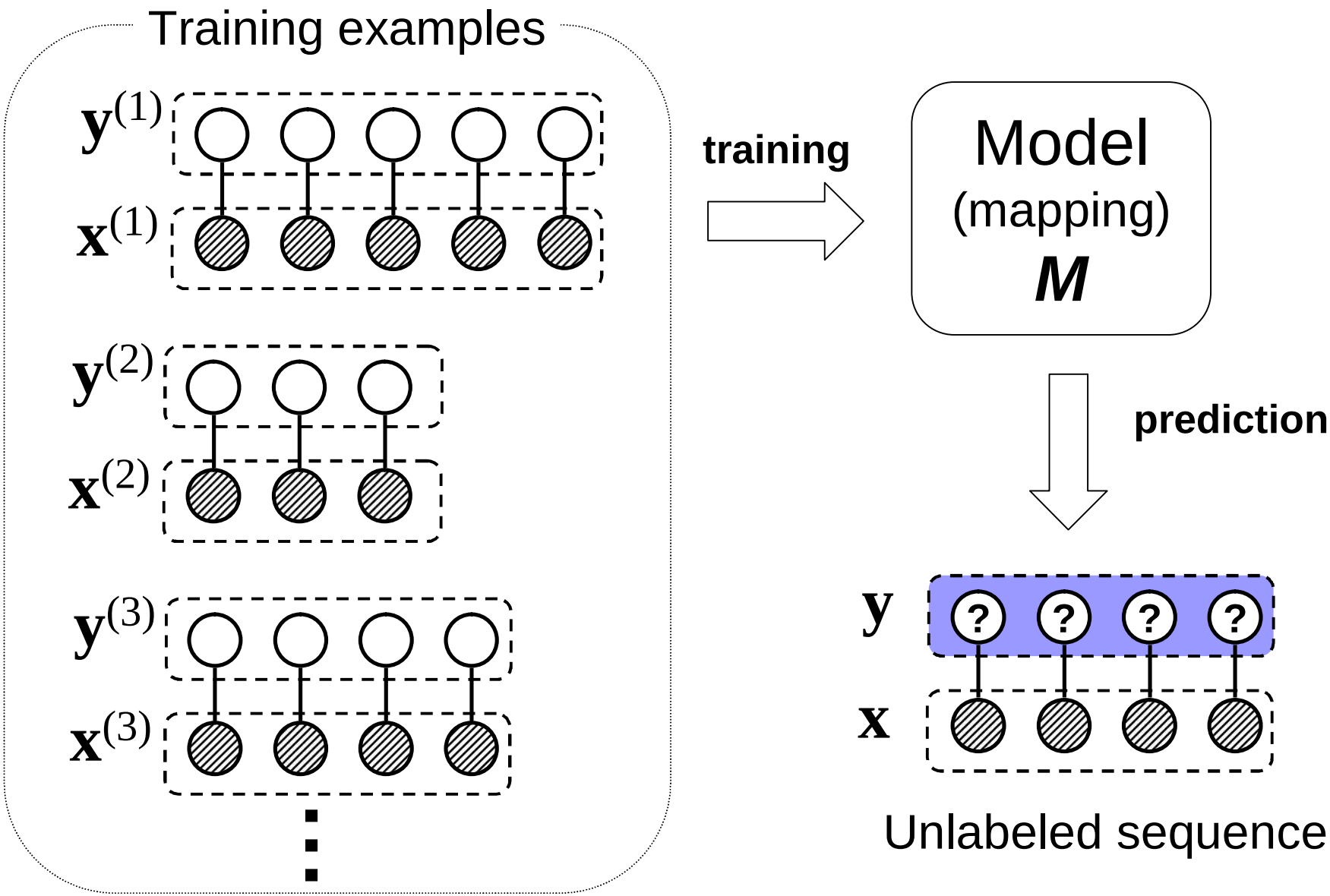
- Mapping $M : \Sigma_x^{|\mathbf{x}|} \rightarrow \Sigma_y^{|\mathbf{x}|}$ (**|x|=|y|=T**)
  - from a sequence of observed variables $\mathbf{x} = (x_1, x_2, x_3, \ldots, x_T)$
  - to a sequence of hidden variables $\mathbf{y} = (y_1, y_2, y_3, \ldots, y_T)$



- Training data
  - Some correctly labeled sequences (pairs of (**x**, **y**))
  - $(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), (\mathbf{x}^{(3)}, \mathbf{y}^{(3)}), \ldots$

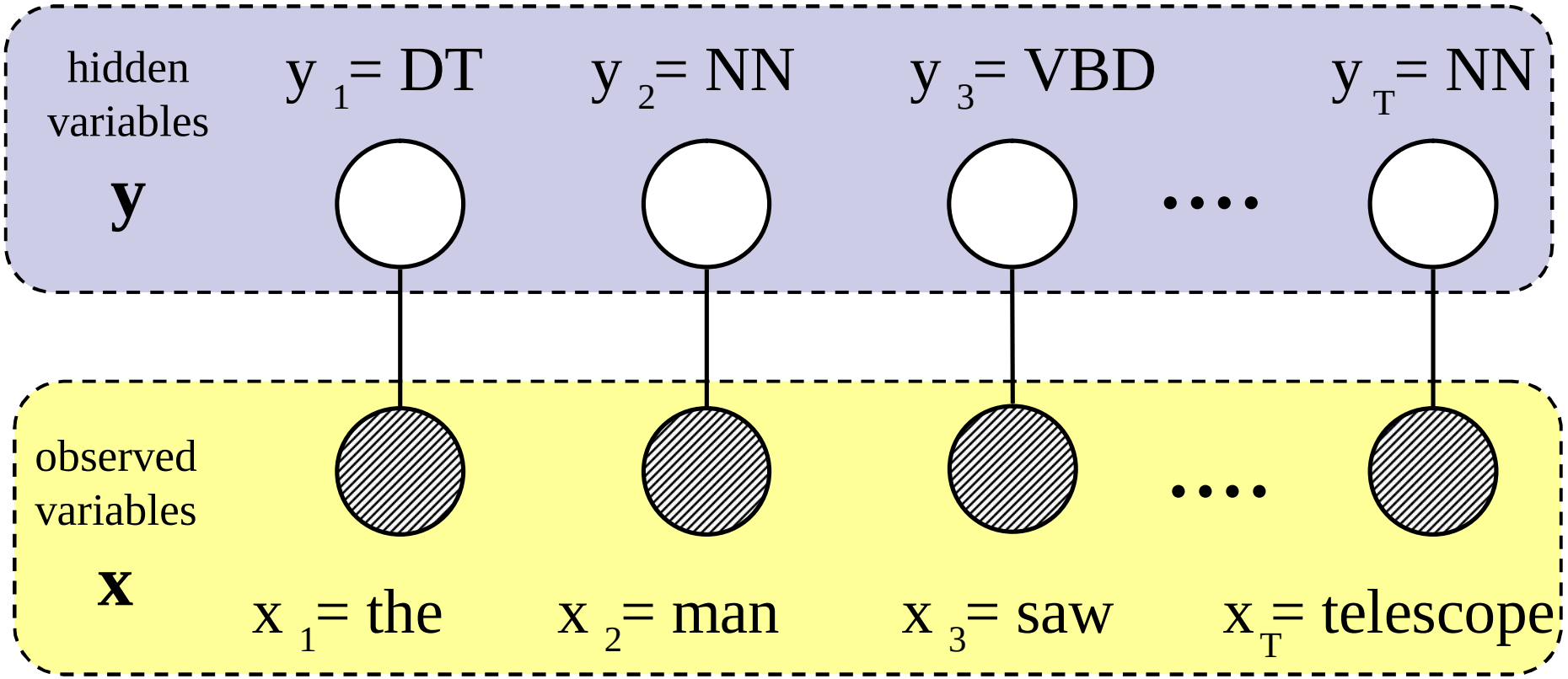
    where $|\mathbf{x}^{(i)}| = |\mathbf{y}^{(i)}| = T^{(i)}$

Sequence Labeling Problems



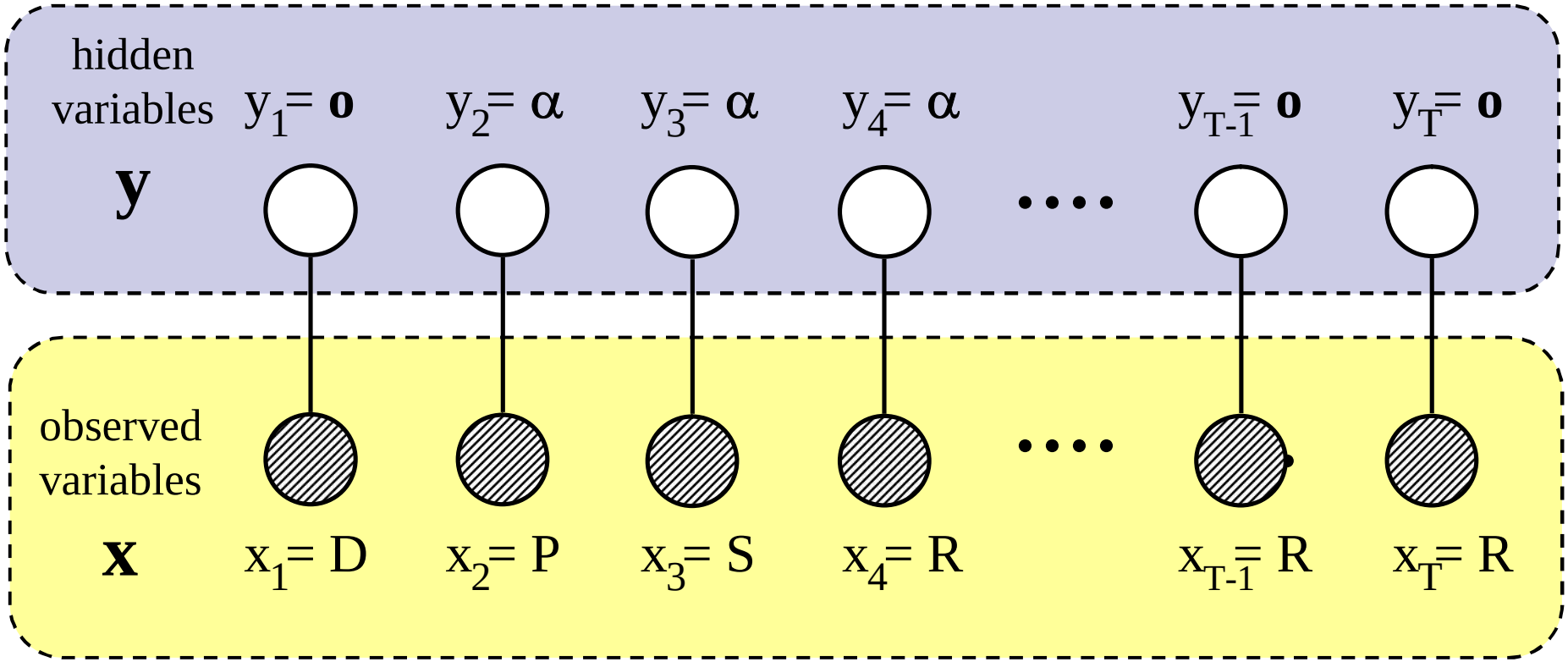
Part-of-Speech Tagging

- Observed variables **x** = sequence of words
  - "the man saw the girl with a telescope."
- Hidden variables **y** =part-of-speech tags
  - assign one of part-of-speech tags to each word



Protein Secondary Structure Prediction

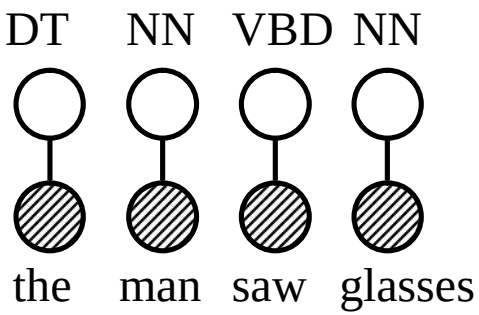
- Observed variables **x** = protein sequence
  - x = (D, P, S, E, …, R, R)
- Hidden variables **y** = secondary structure
  - Assign one of "α" (α-helix), "β" (β-sheet) or "o" (other) to each amino acid



Hidden Markov Models for Labeling Sequences

- Joint probability distribution of (**x**,**y**)

Pr (x, y) = Pr (



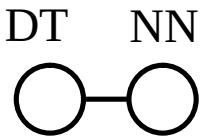
)

- Markov assumption (1st-order)
  - Two types of probabilities
    - Emission probabilities
      - Probability of an observed variable emitted from a hidden variable
      - e.g. Probability of “the” emitted from “DT”
    - Transition probabilities
      - Probability of a hidden variable following another hidden variable
      - e.g. Probability of “NN” following “DT”

emission **Pr (**

)

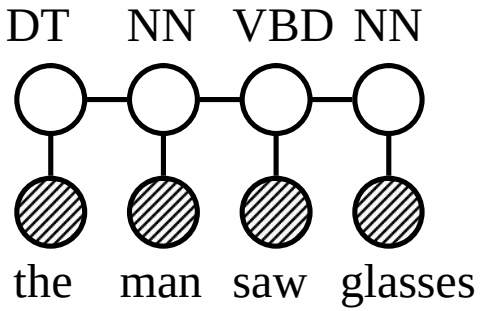
transition **Pr (**



)

Markov Assumption

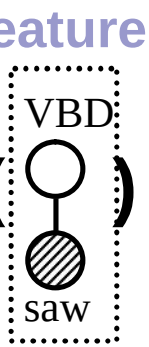
- Likelihood is decomposed into emission probabilities **Pr (**  **)** and transition probabilities **Pr (**  **)**

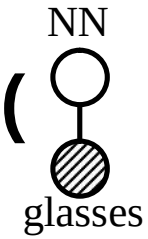
log Pr (

) =

log Pr (

) + log Pr (

) + log Pr (

) + log Pr (

)

Emission probabilities

+ log Pr (

) + log Pr (

) + log Pr (

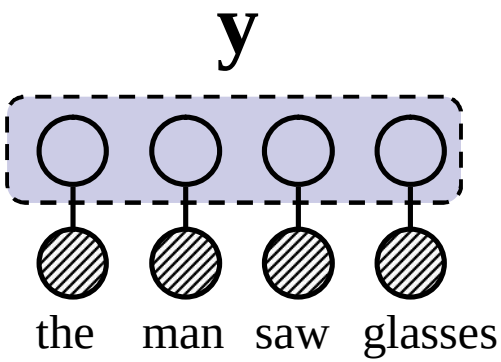
)

Transition probabilities

Prediction (sequence labeling) with HMM

- Given **x**, output **y** that maximizes Pr(**x**, **y**)
  - Among all possible ways of labeling

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \ \mathbf{log\ Pr}\ ($$

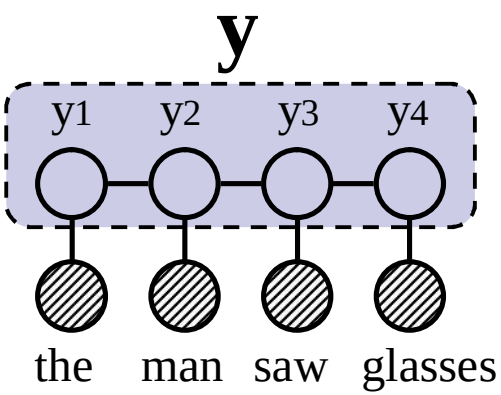


$$)$$

Prediction (sequence labeling) with HMM

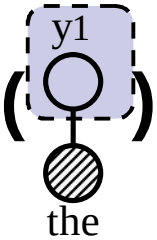
- Combine emission & transition features (= building blocks) to maximize the total probability
- Viterbi algorithm for finding the best hidden labels
  - Dynamic programming

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \ \mathbf{log\ Pr}\Big($$

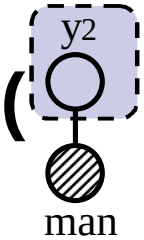


$$\Big) =$$

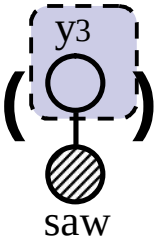
$$\mathbf{log\ Pr}\Big($$



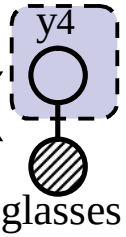
$$\Big) + \mathbf{log\ Pr}\Big($$



$$\Big) + \mathbf{log\ Pr}\Big($$



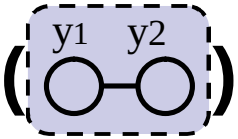
$$\Big) + \mathbf{log\ Pr}\Big($$



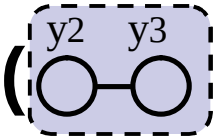
$$\Big)$$

Emission probabilities

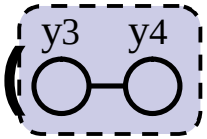
$$+ \mathbf{log\ Pr}\Big($$



$$\Big) + \mathbf{log\ Pr}\Big($$



$$\Big) + \mathbf{log\ Pr}\Big($$



$$\Big)$$

Transition probabilities

Training (Parameter Estimation)

- Maximum likelihood estimation of the joint probability Pr(**x**, **y**)
- Parameters
  - Emission probabilities **Pr (**  **)**
  - Transition probabilities **Pr (**  **)**

  are determined to maximize the joint probability of training data
- Simple
  - count the frequencies of features in training data

Training (Parameter Estimation)

- Hidden Markov models
  - Maximum likelihood estimation of the joint probability of (**x**, **y**)
  - Our purpose: prediction of **y** given **x**
    - Only needs conditional model P(**y**| **x**)
  - MLE of Pr(**x**, **y**) tackles more difficult problem
    - Tends to need many examples

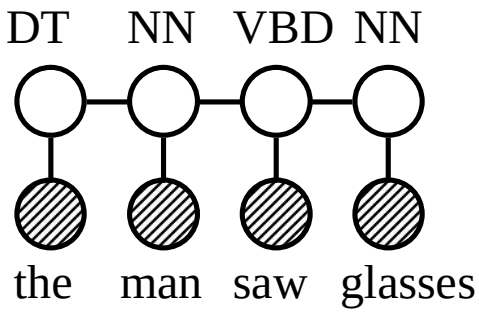
- Conditional models
  - More suited to our purpose
  - Estimate conditional model P(y|x) directly

  - Maximum Entropy Markov Model (MEMM) [McCallum et al, 2000]
  - Conditional Random Field (CRF) [Lafferty et al, 2001]
  - HM-SVM [Altun et al, 2003]
  - **HM-Perceptron** [Collins, 2002]
    - Alternative training algorithm for HMM
    - Online algorithm

Another interpretation of HMM

- Log likelihood can be written as the inner product of weight vector **w** & feature vector $\Phi(\mathbf{x}, \mathbf{y})$

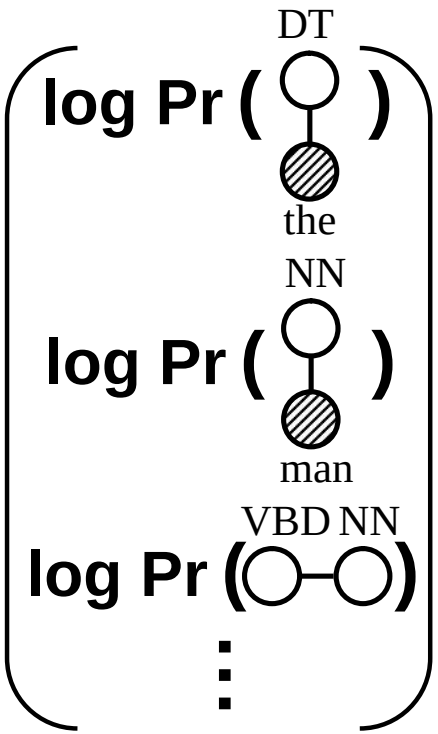
log Pr (



) $= \langle \mathbf{w} \cdot \Phi(\mathbf{x}, \mathbf{y}) \rangle$

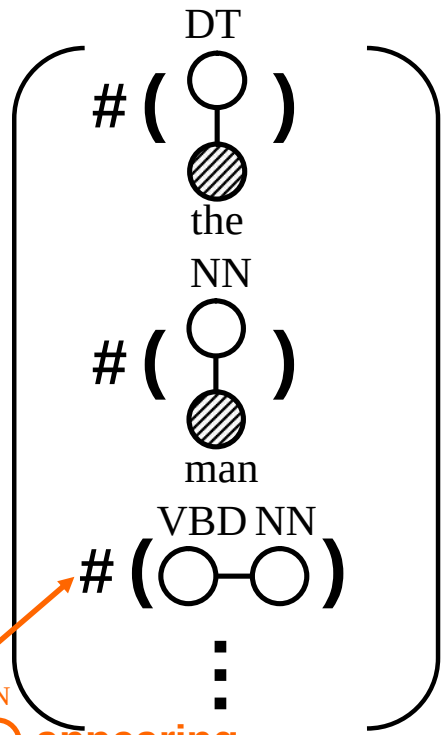
weight vector
(model parameters)

$\mathbf{w} =$



feature vector
(labeled sequence)

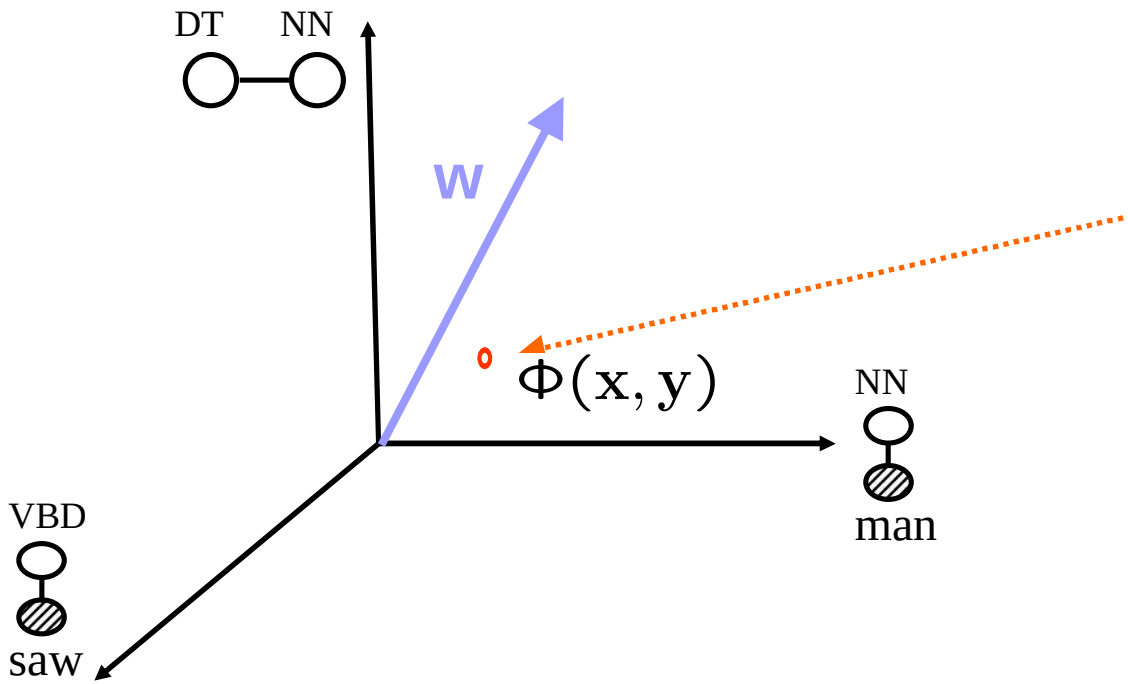
$\Phi(\mathbf{x}, \mathbf{y}) =$



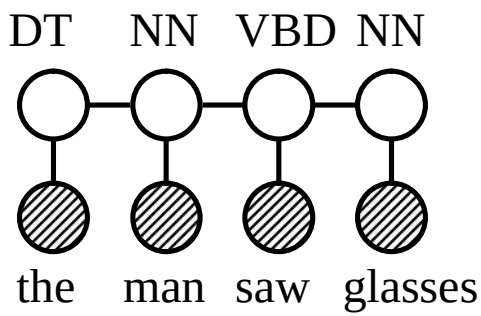
Number of times VBD NN **appearing**

Another interpretation of HMM

- A labeled sequence (**x**,**y**) is mapped into a feature space as a **feature vector** $\Phi(\mathrm{x}, \mathrm{y})$ by using the number of times each feature appears in labeled sequence (**x**,**y**)
- In prediction, feature vectors are projected onto **w**, **y** with the highest $\langle \mathrm{w} \cdot \Phi(\mathrm{x}, \mathrm{y}) \rangle$ is output
- In training, **w** is trained to maximize the joint likelihood of (correctly labeled) training data $\sum_i \langle \mathrm{w} \cdot \Phi(\mathrm{x}^{(i)}, \mathrm{y}^{(i)}) \rangle$



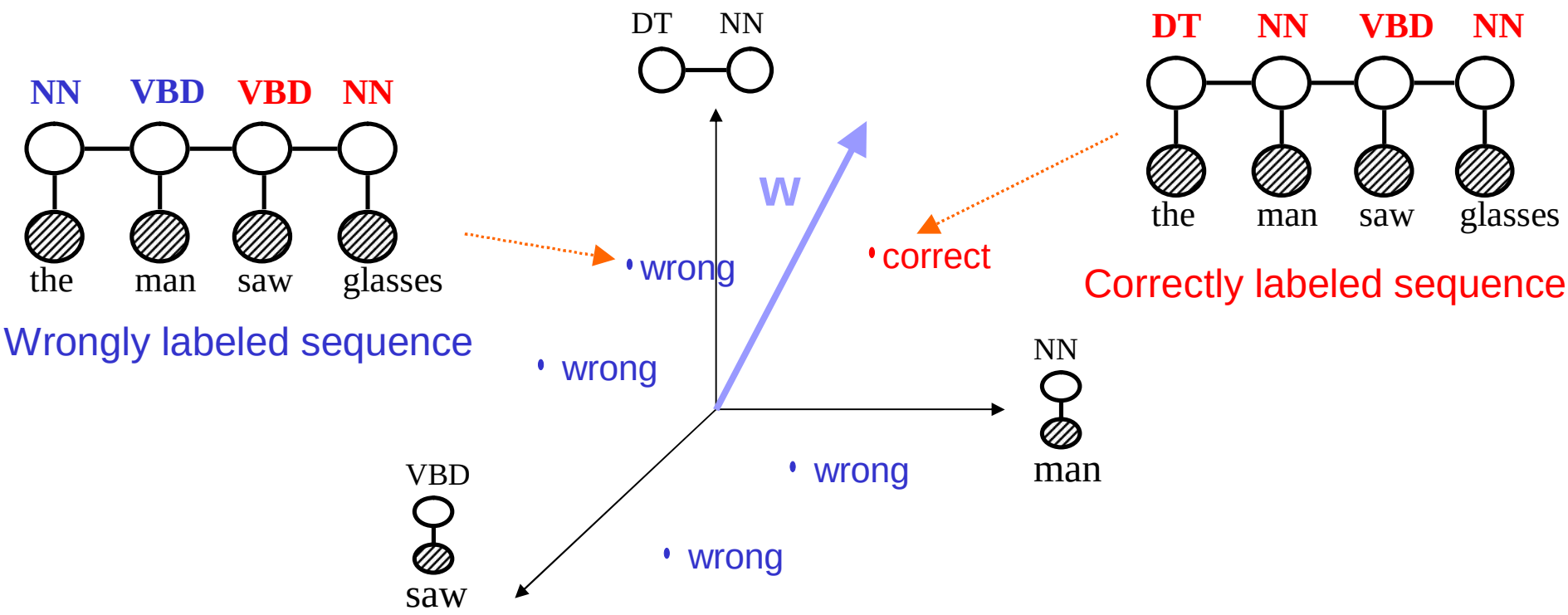
labeled sequence (x, y)



Hidden Markov (HM) Perceptron [Collins, 2002]

- Training algorithm of **w** suited for conditional prediction
- **w** is trained so that correctly labeled sequence has higher score than all wrongly labeled sequences
  - For each training example $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$,

$$\underbrace{\langle \mathbf{w} \cdot \Phi(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) \rangle}_{\text{correct labeling}} > \underbrace{\langle \mathbf{w} \cdot \Phi(\mathbf{x}^{(i)}, \mathbf{y}^{\text{wrong}}) \rangle}_{\text{wrong labeling}} \text{ for } \forall \mathbf{y}^{\text{wrong}} \neq \mathbf{y}^{(i)}$$



Algorithm of HM-Perceptron

- Online algorithm
  - Processes training examples one by one
- Prediction
  - Given $\mathbf{x}^{(i)}$, output the prediction for $\mathbf{y}^{(i)}$ with the highest score according to the current $\mathbf{w}$
    - Viterbi algorithm

$$\hat{\mathbf{y}}^{(i)}(\mathbf{x}) = \underset{\mathbf{y}}{\operatorname{argmax}} \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$$

Among
all possible ways
of labeling for **y**

- Update **w** when the prediction is wrong
  - If $\hat{\mathbf{y}}^{(i)} \neq \mathbf{y}^{(i)}$, (prediction for the i-th example is wrong)

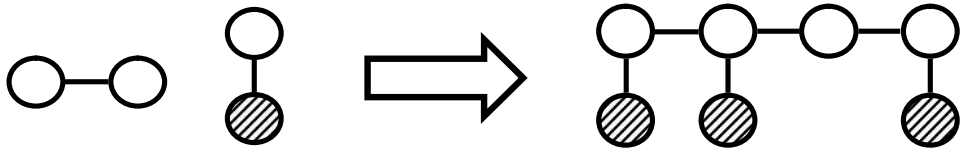
$$\mathbf{w}^{new} = \mathbf{w}^{old} + \Phi(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) - \Phi(\mathbf{x}^{(i)}, \hat{\mathbf{y}}^{(i)})$$

True labeling

Wrong labeling

HM-Perceptron with Large Features

- Arbitrarily **large features** to incorporate wider contexts
  - idioms, motifs,…



- Two problems
  1. Dimension of feature vector $\Phi(\mathbf{x}, \mathbf{y})$ becomes high
  2. Computational complexity of label prediction

$$\hat{\mathbf{y}}(\mathbf{x}) = \operatorname*{argmax}_{\mathbf{y}} \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$$

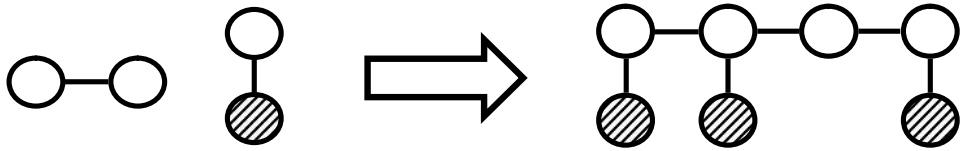
depends exponentially on the max number of hidden variables contained in features

- Viterbi algorithm

⇨ HM-perceptron cannot handle arbitrarily large features

HM-Perceptron with Large Features

- Arbitrarily **large features** to incorporate wider contexts
  - idioms, motifs,…



- Two problems
  1. Dimension of feature vector $\Phi(\mathbf{x}, \mathbf{y})$ becomes too high
  2. Computational complexity of label prediction

$$\hat{\mathbf{y}}(\mathbf{x}) = \operatorname*{argmax}_{\mathbf{y}} \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$$

depends exponentially on the max number of hidden variables contained in features

- Viterbi algorithm

⇨ HM-perceptron cannot handle arbitrarily large features

HM-Perceptron in Dual Form

$$\mathbf{w}^{new} = \mathbf{w}^{old} + \Phi(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) - \Phi(\mathbf{x}^{(i)}, \hat{\mathbf{y}}^{(i)})$$

- **w** is represented as a linear combination of feature vectors

$$\mathbf{w} = \sum_{j=1}^{|Example|} \sum_{\tilde{\mathbf{y}}} \underbrace{\alpha_j(\tilde{\mathbf{y}})}_{\text{New weight}} \Phi(\mathbf{x}^{(j)}, \tilde{\mathbf{y}})$$

- Prediction
  - Given **x**, output the prediction of **y** with the highest score

$$\begin{aligned} \hat{\mathbf{y}}(\mathbf{x}) &= \operatorname*{argmax}_{\mathbf{y}} \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle \text{ (primal form)} \\ &= \operatorname*{argmax}_{\mathbf{y}} \sum_{j=1}^{|Example|} \sum_{\tilde{\mathbf{y}}} \alpha_j(\tilde{\mathbf{y}}) \underbrace{\langle \Phi(\mathbf{x}^{(j)}, \tilde{\mathbf{y}}), \Phi(\mathbf{x}, \mathbf{y}) \rangle}_{\text{Inner product}} \text{ (dual form)} \end{aligned}$$

- Update
  - If $\hat{\mathbf{y}}^{(i)} \neq \mathbf{y}^{(i)}$, (prediction for the i-th example is wrong)
    - $\alpha_i^{new}(\mathbf{y}^{(i)}) = \alpha_i^{old}(\mathbf{y}^{(i)}) + 1$
    - $\alpha_i^{new}(\hat{\mathbf{y}}^{(i)}) = \alpha_i^{old}(\hat{\mathbf{y}}^{(i)}) - 1,$

Kernel Methods

- Substitute inner products by “kernel function”

$$K((\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y}')) = \langle \Phi(\mathbf{x},\mathbf{y}), \Phi(\mathbf{x}',\mathbf{y}')\rangle$$

- Dimension of feature space does not matter
  - As long as we have algorithms to compute inner product (=kernel function) without explicit construction of feature vectors

$$\begin{aligned}\hat{\mathbf{y}}(\mathbf{x}) &= \operatorname*{argmax}_{\mathbf{y}} \langle \mathbf{w}, \Phi(\mathbf{x},\mathbf{y})\rangle \text{ (primal form)} \\ &= \operatorname*{argmax}_{\mathbf{y}} \sum_{j=1}^{|Example|} \sum_{\tilde{\mathbf{y}}} \alpha_j(\tilde{\mathbf{y}}) K(\Phi(\mathbf{x}^{(j)},\tilde{\mathbf{y}}), \Phi(\mathbf{x},\mathbf{y})) \text{ (dual form)}\end{aligned}$$

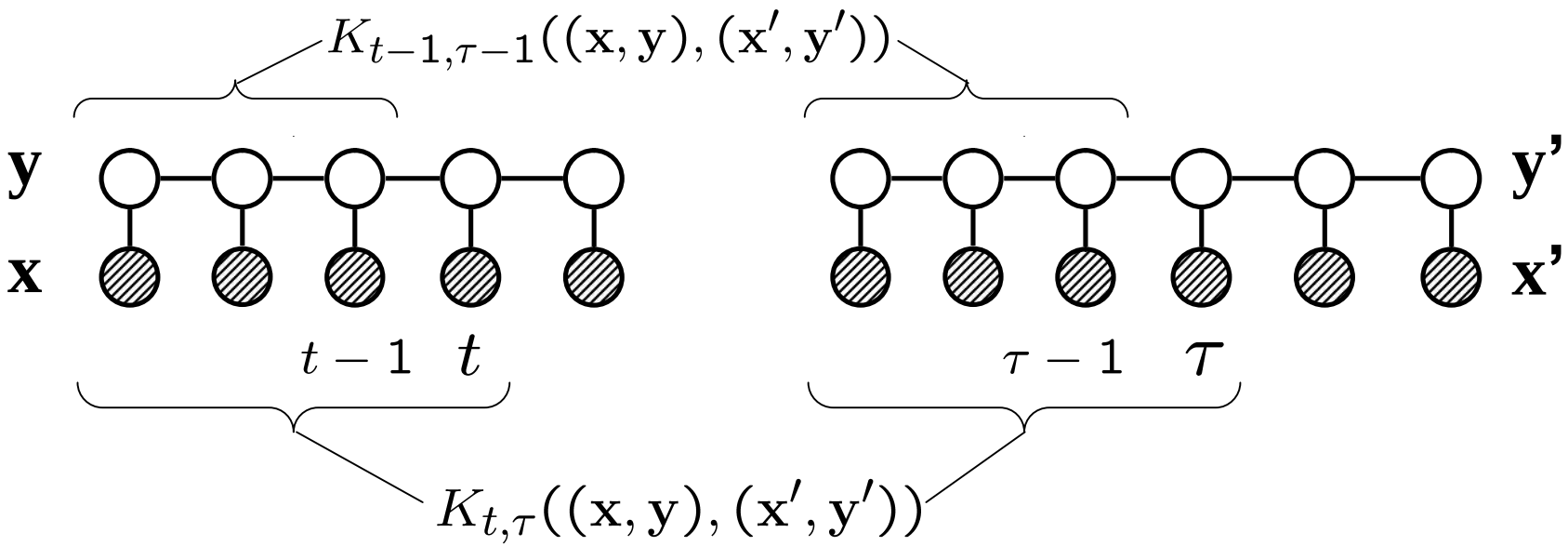
Kernel for Labeled Sequences

- Recursive computation by dynamic programming in O(|**x**||**x**'|)
  - Kernel function is decomposed into kernel functions of pairs of positions

$$K((\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y}')) = \sum_{t=1}^{|\mathbf{x}|}\sum_{\tau=1}^{|\mathbf{x}'|} K_{t,\tau}((\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y}'))$$

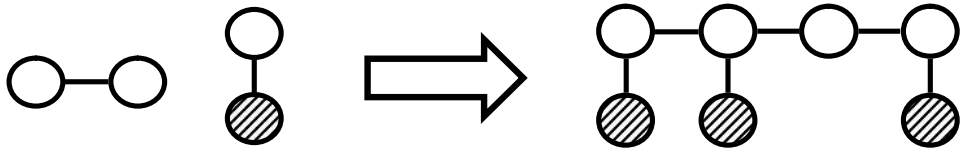
  - kernel function of each pair of positions is recursively defined by the kernel function of the previous pair of positions

$$K_{t,\tau}((\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y}')) = \Big(K_{t-1,\tau-1}((\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y}')) + 1\Big)\cdot k(y_t, y'_\tau)\cdot\Big(k(x_t, x'_\tau) + 1\Big)$$



HM-Perceptron with Large Features

- Arbitrarily **large features** to incorporate wider contexts
  - idioms, motifs,…



- Two problems
  1. Dimension of feature vector $\Phi(\mathbf{x}, \mathbf{y})$ becomes too high
  2. Computational complexity of label prediction

$$\hat{\mathbf{y}}(\mathbf{x}) = \underset{\mathbf{y} \in \Sigma_y^{|\mathbf{x}|}}{\operatorname{argmax}} \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$$

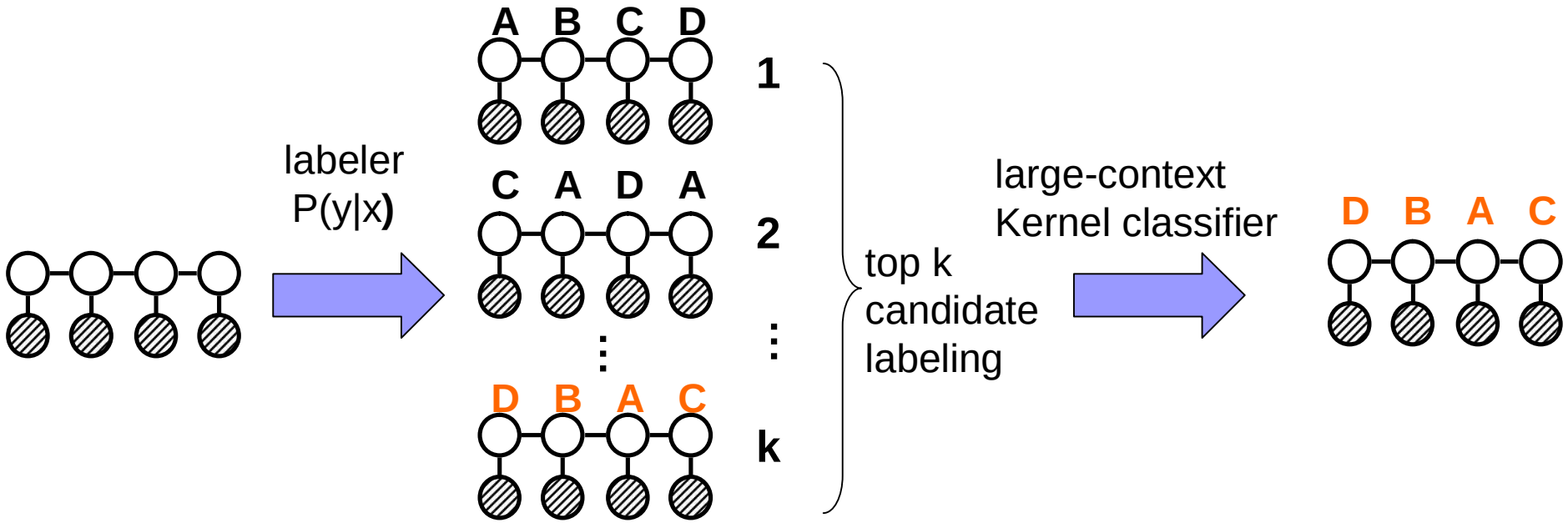
depends exponentially on the max number of hidden variables contained in features

- Viterbi algorithm

⇨ HM-perceptron cannot handle arbitrarily large features

Re-ranking [Collins, 2000]

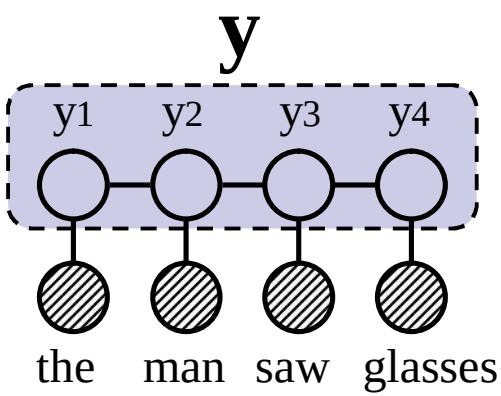
- Avoid to use Viterbi
- Generate top *k* candidates by standard labeling algorithms (HMM/HM-perceptron)
- Re-rank the candidates by using large features
  - No need to use Viterbi algorithm
- Correct labeling is not guaranteed to be included in the candidates



Point-wise label prediction [Kakade et al. ,2002]

- Avoid to use Viterbi algorithm
- Sequential prediction
  - All labels at once

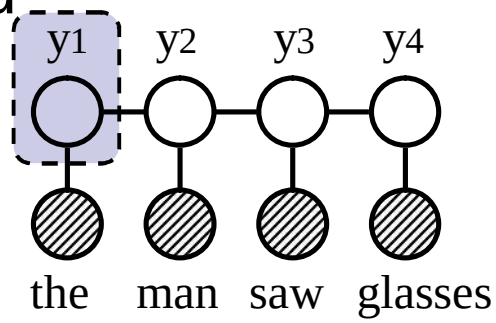
$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \ \mathbf{log\ Pr}\Big($$



$$\Big)$$

- Point-wise prediction
  - Each of hidden variables is predicted independently by using marginalized likelihood

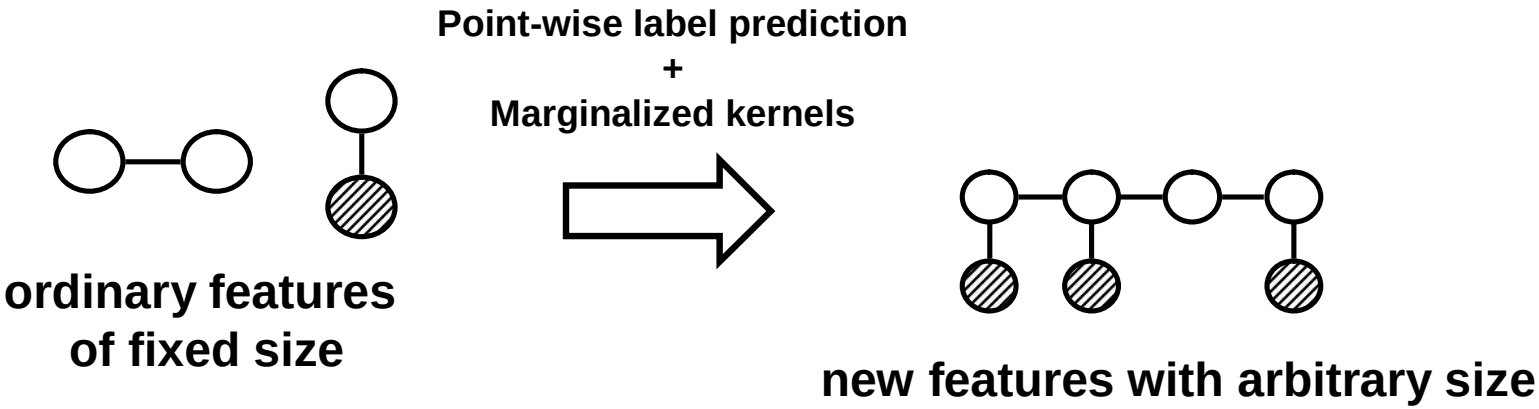
$$\hat{y}_1 = \underset{y_1}{\operatorname{argmax}} \sum_{y_2, y_3, y_4} \mathbf{log\ Pr}\Big($$



$$\Big)$$

Marginalized Labeling Perceptron (Proposed Method)

- Can incorporate arbitrary size features efficiently
  - Dual representation allows to use (marginalized) **kernels**
  - **Point-wise label prediction** [Kakade et al. ,2002]
    - No need to use Viterbi
      since each hidden variable is predicted independently



Marginalized Labeling Perceptron (primal)

- Marginalized Labeling Perceptron

$$\hat{y}_t(\mathbf{x}) = \underset{\tilde{y}_t}{\operatorname{argmax}} \left\langle \mathbf{w}, \sum_{\mathbf{y}:y_t=\tilde{y}_t} P(\mathbf{y}|\mathbf{x})\Phi(\mathbf{x},\mathbf{y}) \right\rangle$$

Point-wise prediction (no Viterbi !)

Marginalized feature vector <u>with fixed label at *t*</u> (incorporates all candidates)

Feature vector of arbitrary size features

Pre-trained labeler with small size features (e.g. HMM, HM-Perceptron, MEMM, CRF, …)

- HM-Perceptron

$$\hat{\mathbf{y}}(\mathbf{x}) = \underset{\mathbf{y}}{\operatorname{argmax}} \langle \mathbf{w}, \Phi(\mathbf{x},\mathbf{y}) \rangle$$

Find the best hidden sequence among all possible ways of labeling for *y* (needs Viterbi !)

Learning Marginalized Labeling Perceptron (primal)

- Marginalized Labeling Perceptron

$$\mathbf{w}^{new} = \mathbf{w}^{old} + \sum_{\mathbf{y}:y_t=y_t^{(i)}} P(\mathbf{y}|\mathbf{x})\Phi(\mathbf{x},\mathbf{y}) - \sum_{\mathbf{y}:y_t=\hat{y}_t^{(i)}} P(\mathbf{y}|\mathbf{x})\Phi(\mathbf{x},\mathbf{y})$$

$$\Longrightarrow \quad \mathbf{w} = \sum_{j=1}^{|Example|} \sum_{\tau=1}^{|\mathbf{x}^{(j)}|} \sum_{\tilde{y}_\tau} \alpha_{j\tau}(\tilde{y}_\tau) \sum_{\mathbf{y}:y_\tau=\tilde{y}_\tau} P(\mathbf{y}|\mathbf{x})\Phi(\mathbf{x}^{(j)},\mathbf{y})$$

- HM-Perceptron

$$\mathbf{w}^{new} = \mathbf{w}^{old} + \Phi(\mathbf{x}^{(i)},\mathbf{y}^{(i)}) - \Phi(\mathbf{x}^{(i)},\hat{\mathbf{y}}^{(i)})$$

$$\Longrightarrow \quad \mathbf{w} = \sum_{j=1}^{|Example|} \sum_{\tilde{\mathbf{y}}} \alpha_j(\tilde{\mathbf{y}})\Phi(\mathbf{x}^{(j)},\tilde{\mathbf{y}})$$

Marginalized Labeling Perceptron (dual)

- Dual representation
  - Kernel methods to handle arbitrary size features efficiently
- Prediction

$$\hat{y}_t(\mathbf{x}) = \operatorname*{argmax}_{\tilde{y}_t} \sum_{j=1}^{|Examples|} \sum_{\tau=1}^{|\mathbf{x}^{(j)}|} \sum_{\bar{y}_\tau} \alpha_{j\tau}(\bar{y}_\tau) K(\mathbf{x}^{(j)}, \mathbf{x}, \tau, t, \bar{y}_\tau, \tilde{y}_t)$$

  - Marginalized inner products = Marginalized kernel

$$K(\mathbf{x}, \mathbf{x}', t, \tau, \tilde{y}_t, \tilde{y}'_\tau) = \sum_{\mathbf{y}: y_t = \tilde{y}_t} \sum_{\mathbf{y}': y'_\tau = \tilde{y}'_\tau} P(\mathbf{y}|\mathbf{x}) P(\mathbf{y}'|\mathbf{x}') \langle \Phi(\mathbf{x}, \mathbf{y}), \Phi(\mathbf{x}', \mathbf{y}') \rangle$$

Positions whose labels are fixed

fixed labels at *t* & *t*

Marginalize over all possible ways of labeling of y & y' with fixed labels at *t* & τ

Inner product between two labeled sequences

- Update
  - $\alpha_{it}^{new}(y_t^{(i)}) = \alpha_{it}^{old}(y_t^{(i)}) + 1$
  - $\alpha_{it}^{new}(\hat{\mathbf{y}}^{(i)}) = \alpha_{it}^{old}(\hat{y}_t^{(i)}) - 1$

Computation of Marginalized Kernels

- Given two sequences of observed variables **x** & **x'**
  - Compute marginalized kernels for all pairs of positions *t* & ***τ*** at once
  - Kernel decomposition for efficient computation

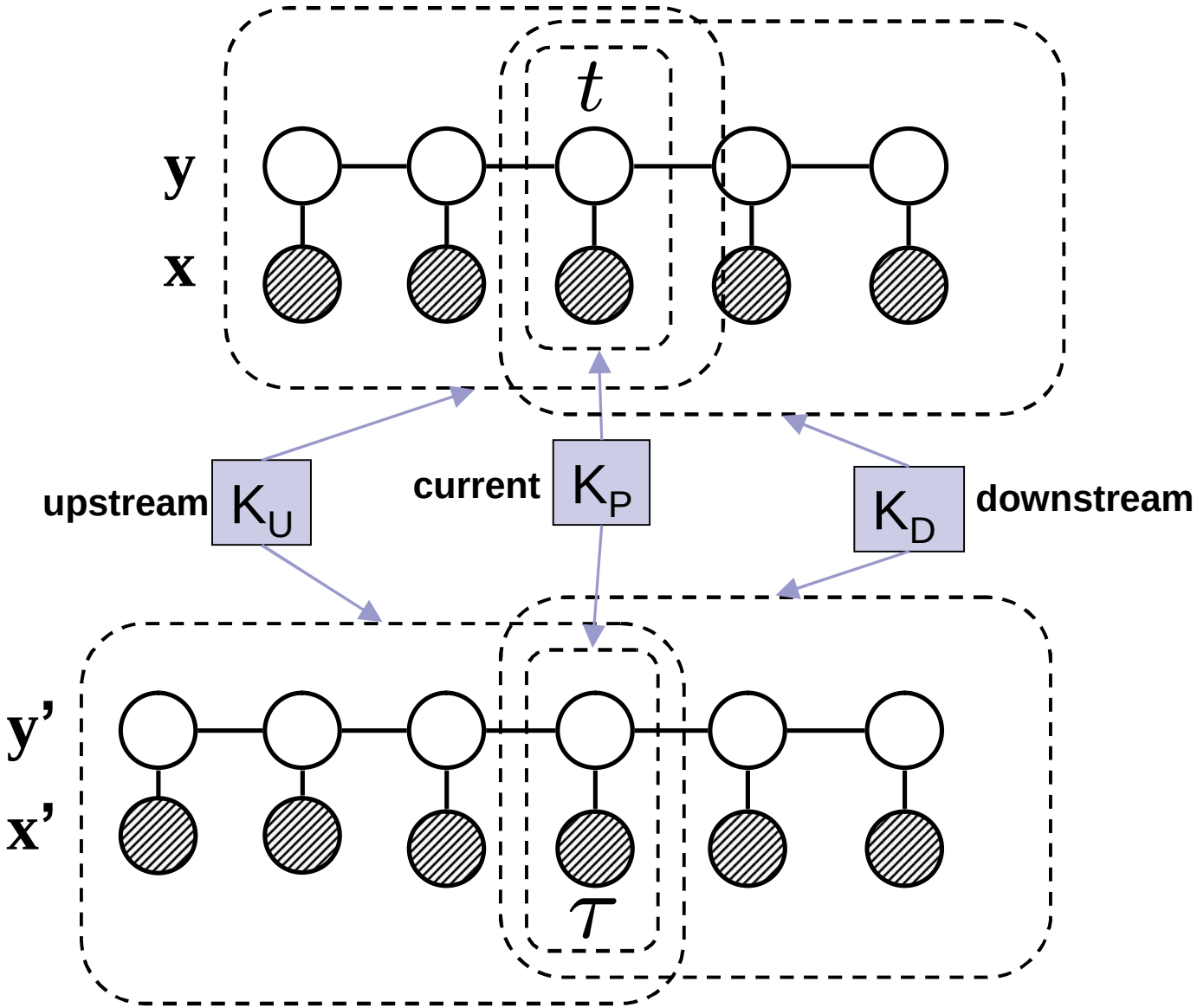
$$
\begin{aligned}
K(\mathbf{x}, \mathbf{x}', t, \tau, \tilde{y}_t, \tilde{y}'_\tau) &= \sum_{\mathbf{y}: y_t = \tilde{y}_t} \sum_{\mathbf{y}': y'_\tau = \tilde{y}'_\tau} P(\mathbf{y}|\mathbf{x}) P(\mathbf{y}'|\mathbf{x}') \langle \Phi(\mathbf{x}, \mathbf{y}), \Phi(\mathbf{x}', \mathbf{y}') \rangle \\
&= \underbrace{K_U(\mathbf{x}, \mathbf{x}', t, \tau)}_{\text{up-stream kernel}} \cdot \underbrace{K_P(\mathbf{x}, \mathbf{x}', t, \tau, \tilde{y}_t, \tilde{y}'_\tau)}_{\text{current position kernel}} \cdot \underbrace{K_D(\mathbf{x}, \mathbf{x}', t, \tau)}_{\text{down-stream kernel}}
\end{aligned}
$$

Each kernel is recursively computed by dynamic programming

  - Computational Complexity O(|x| |x'|)

Kernel Decomposition (Sequences)

- **Analogous to Forward-Backward algorithm for HMM**



Experiments: Named Entity Recognition

- NLP version of secondary structure prediction

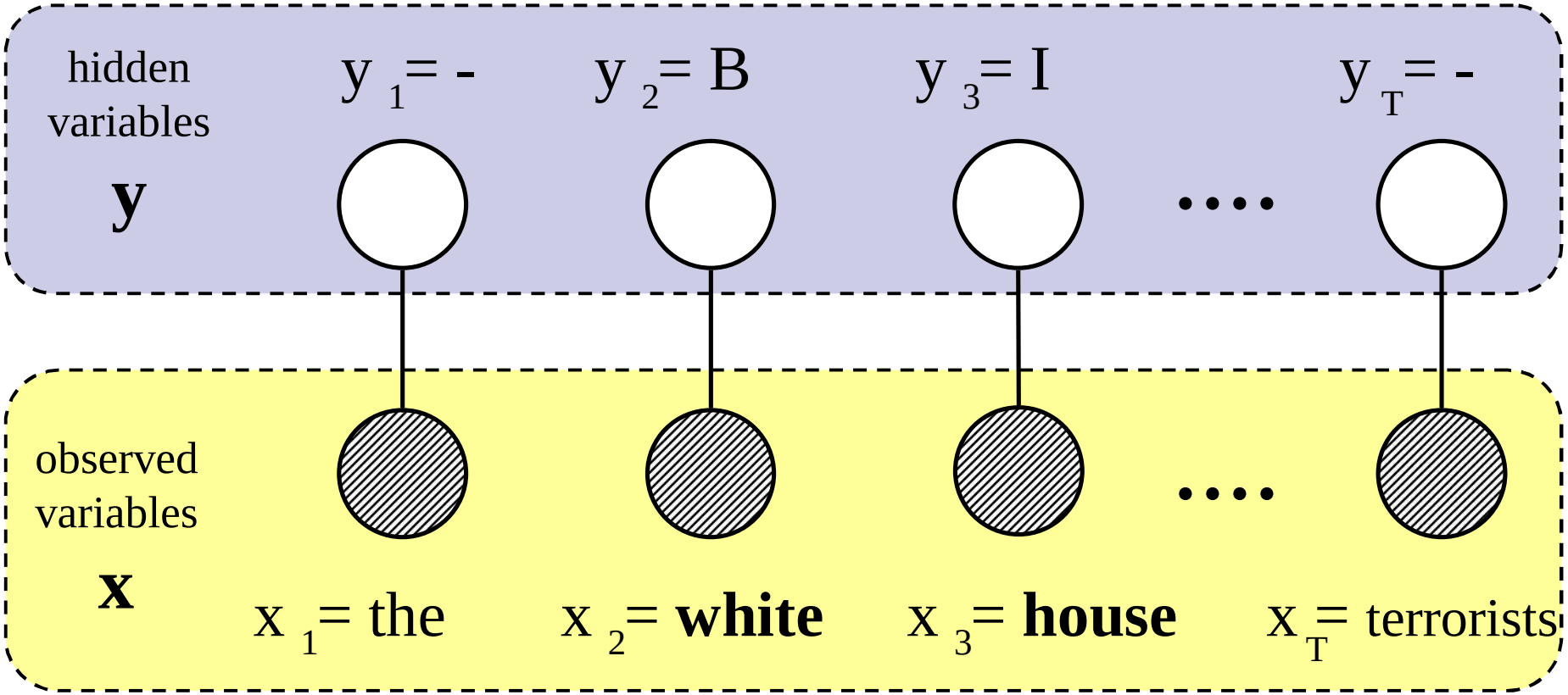
- CoNLL2002 data
  - 300 sentences (8,500 words)
  - 9 labels to indicate *person name*, *organization name*, *place*, …

- Sales log data
  - 184 sentences (3,500 words)
  - a sales log written by sales representatives (written in Japanese)
    - *"They ported their server to a **linux cluster** for its **availability**"*
  - 12 labels to indicate *product name*, *company name*, *reason for buying the product*,…

- Comparison of two methods
  - HM-Perceptron
    - window size = 3
  - Marginalized labeling perceptron with sequence kernel

Named Entity Recognition

- Given **x** = sequence of words
  - "The white house was attacked by the terrorists"
- predict **y** = named entities
  - beginning (B), inside (I), other (-)



Results (5-fold cross validation)

- Uniform distributions as the priors P(**y**|**x**) for marginalized kernels
- CoNLL2002 data

	Precision	Recall	F1
HM-Perceptron	23.8% (14.6)	17.9% (3.0)	18.6 (5.2)
Sequence Kernel	**49.0**% (6.0)	**23.1**% (8.1)	**30.5** (6.7)

(standard deviation)

- Sales log

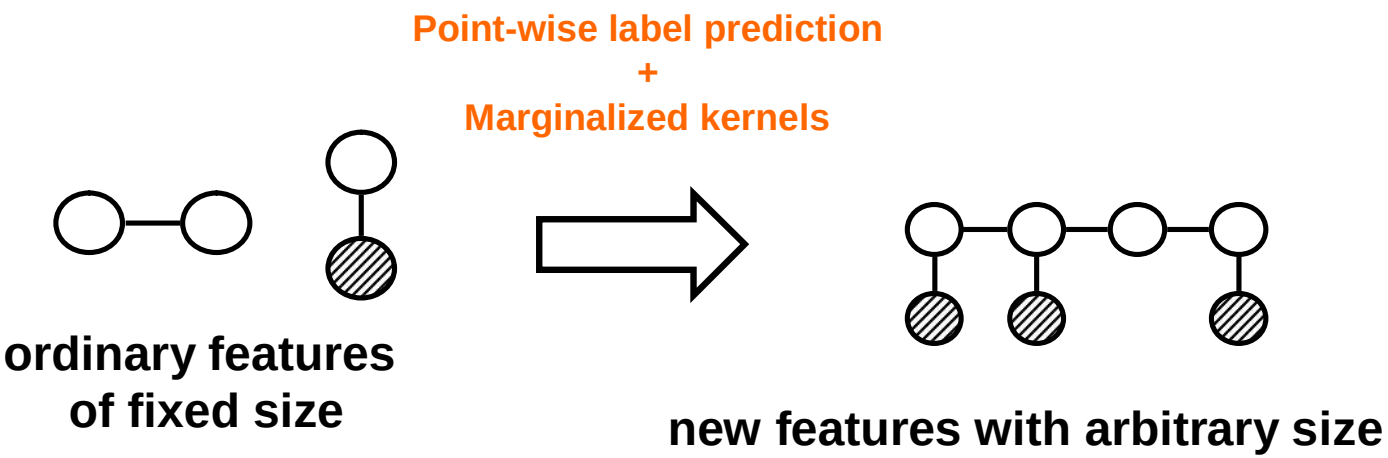
	Precision	Recall	F1
HM-Perceptron	51.5%(8.5)	24.0%(21.4)	28.9(20.3)
Sequence Kernel	**52.2**% (9.5)	**29.6**% (4.0)	**37.5**(4.1)

(standard deviation)

- Precision = #(correctly predicted NE labels) / #(predicted NE labels)
- Recall = #(correctly predicted NE labels) / #(true NE labels)
- F1 = 2・Specificity・Sensitivity / (Specificity + Sensitivity)

Summary

- Marginalized labeling perceptron
  - Labeling learning algorithm for sequence labeling
  - Can handle features with arbitrarily many hidden variables by using
    - Point-wise label prediction → avoids Viterbi
    - Marginalized kernels
      → avoids explicit handling of high dimensional feature vectors

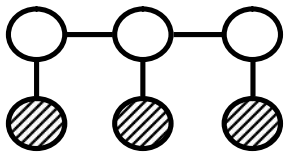


Application in Bioinformatics

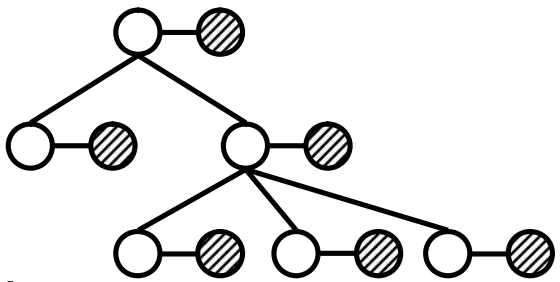
- Protein secondary structure prediction
  - Predict alpha-helix/beta-strand regions in protein sequences
- Gene finding
  - Predict gene regions in DNA sequences
- Phosphorylation site prediction
  - Predict phosphorylated sites in protein sequences
- Information extraction from biomedical texts
  - Protein name recognition

Marginalized Kernels for Labeling Structured Data

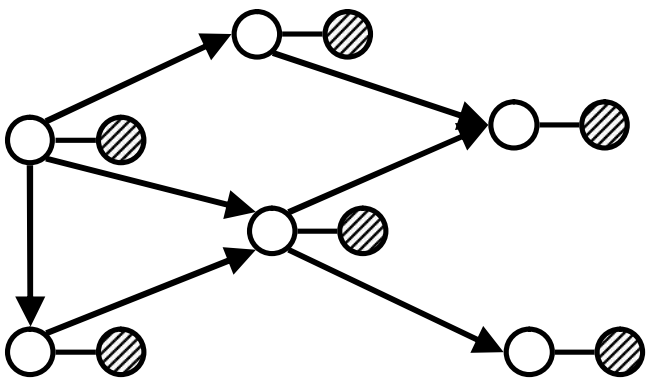
- $$K(\mathbf{x}, \mathbf{x}', t, \tau, \tilde{y}_t, \tilde{y}'_\tau) = \sum_{\mathbf{y}:y_t=\tilde{y}_t} \sum_{\mathbf{y}':y'_\tau=\tilde{y}'_\tau} P(\mathbf{y}|\mathbf{x}) P(\mathbf{y}'|\mathbf{x}') \langle \Phi(\mathbf{x}, \mathbf{y}), \Phi(\mathbf{x}', \mathbf{y}') \rangle$$
- 3 kernels
  - Sequence labeling
    - Sequence features



  - Ordered tree labeling
    - Tree features



  - Directed acyclic graph (DAG) labeling
    - Path features



Outline

- Sequence labeling problems
- Hidden Markov model (HMM)
- Hidden Markov (HM) perceptron
- Marginalized labeling perceptron
- Experiments on natural language processing tasks
- Application to bioinformatics

Thank you

Conditional Models

- conditional model

$$\Pr(\mathbf{y}|\mathbf{x}) = \frac{\exp\langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y})\rangle}{\sum_{\tilde{\mathbf{y}}} \exp\langle \mathbf{w}, \Phi(\mathbf{x}, \tilde{\mathbf{y}})\rangle}$$

- MLE of conditional models
  - (sequential objective function)

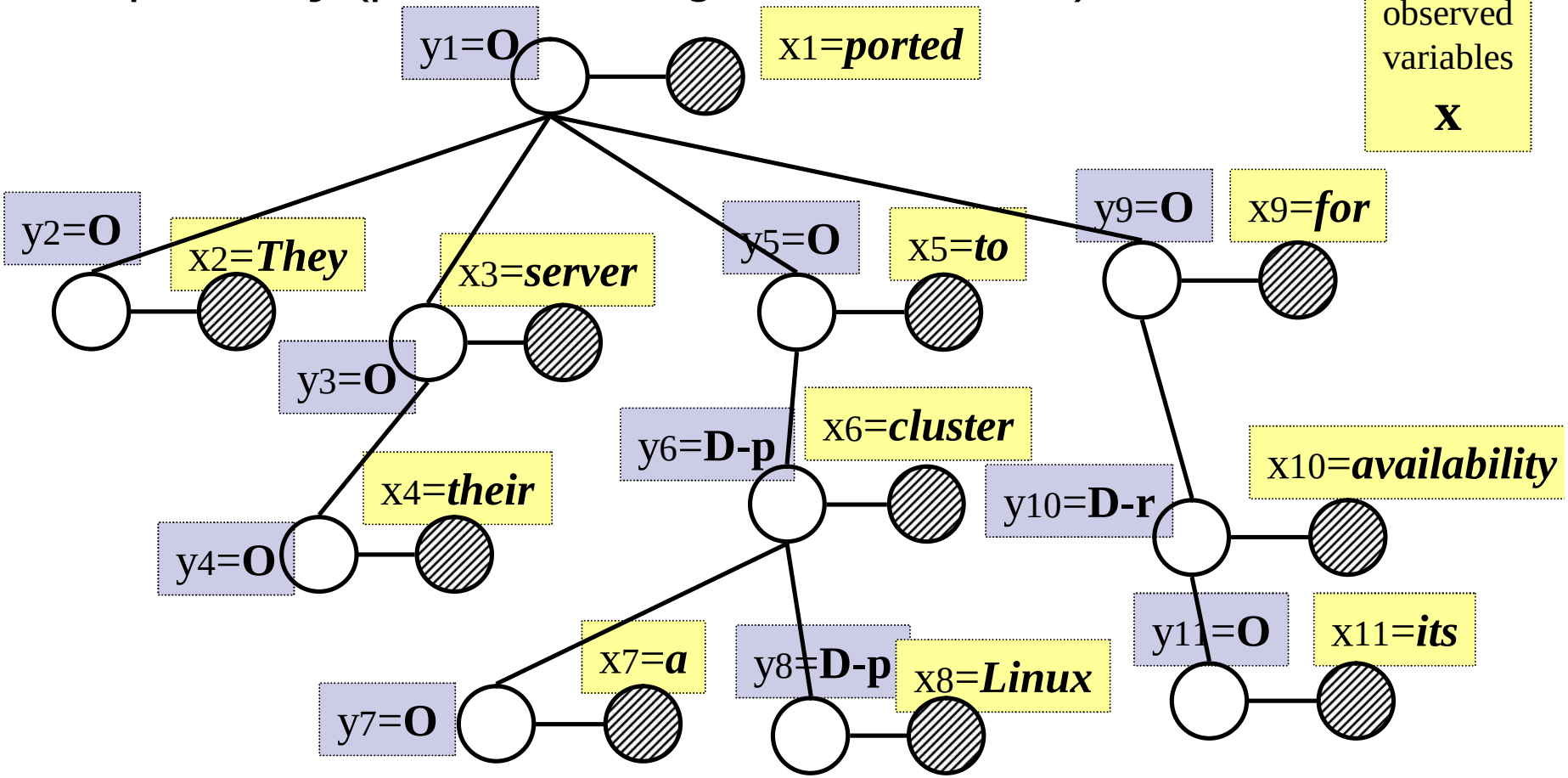
$$\hat{\mathbf{w}} = \operatorname*{argmax}_{\mathbf{w}} \sum_i \Pr(\mathbf{y}^{(i)}|\mathbf{x}^{(i)})$$

  - (point-wise objective function)

$$\hat{\mathbf{w}} = \operatorname*{argmax}_{\mathbf{w}} \sum_i \sum_{t=1}^{T^{(i)}} \sum_{\mathbf{y}:y_t=y_t^{(i)}} \Pr(\mathbf{y}|\mathbf{x}^{(i)})$$

Tree Labeling Problem

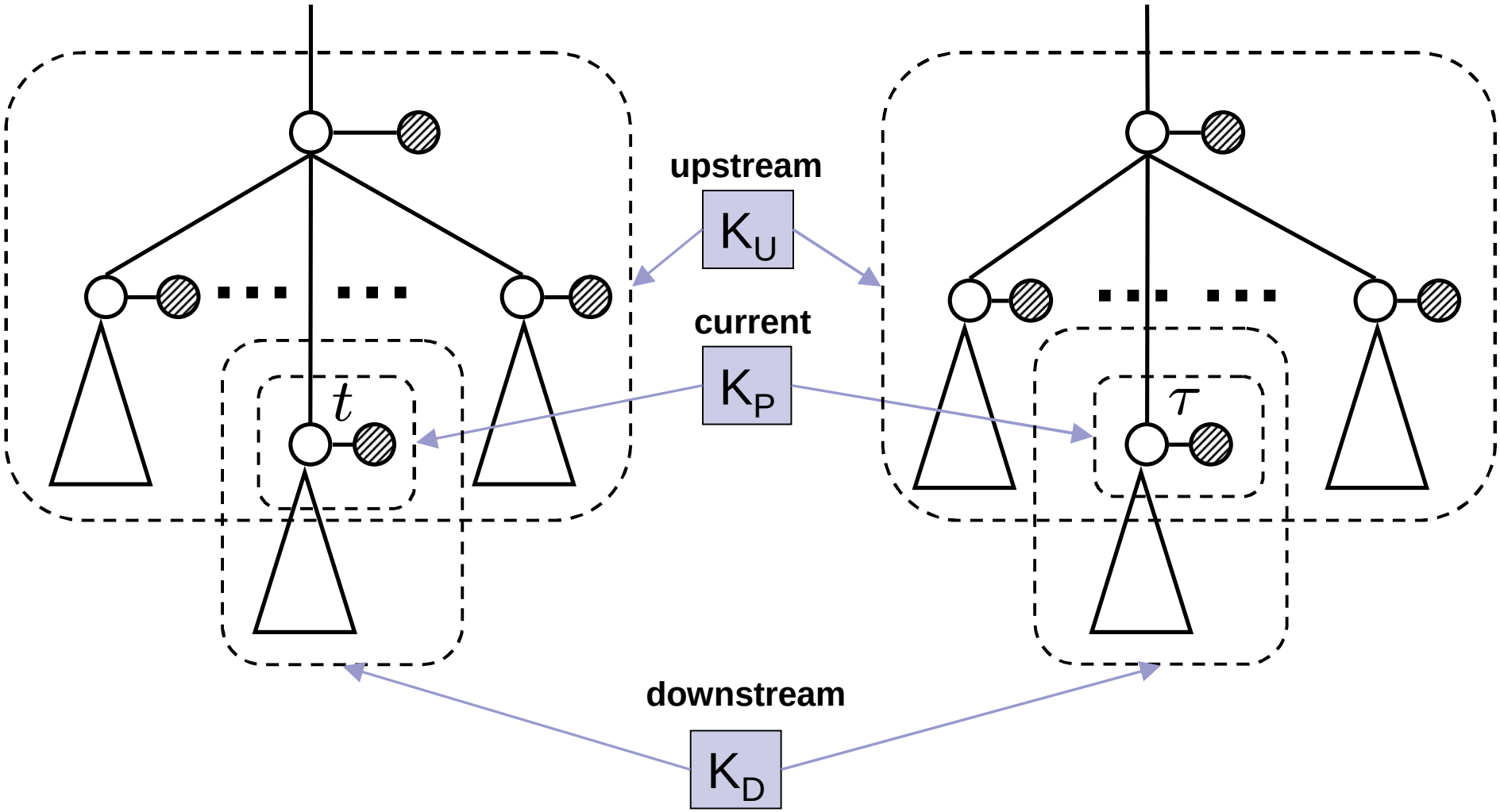
- Information Extraction from parse trees
  - Given **x** (parse tree for text), predict **y** (products bought, reason, etc.)



"They ported their server to a linux cluster for its availability"

Kernel Decomposition (Ordered Trees)

- **Analogous to Inside-Outside algorithm for probabilistic CFG**



Kernel Decomposition (DAG)