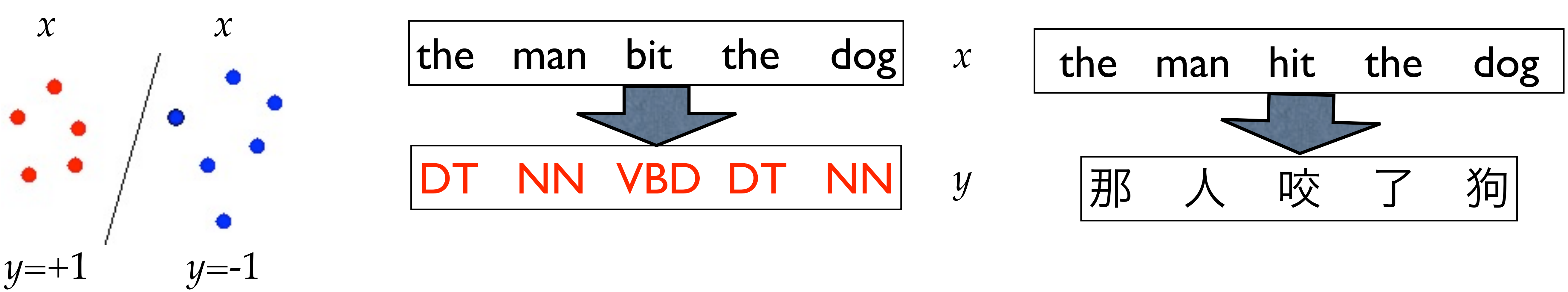
Scalable Large-Margin Structured Learning: Theory and Algorithms



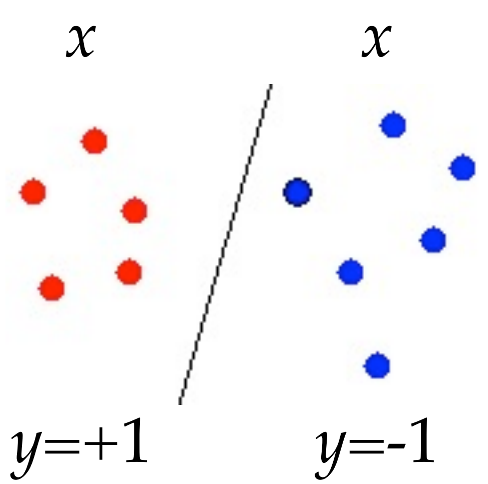
Liang Huang   Kai Zhao   Lemao Liu

The City University of New York (CUNY)

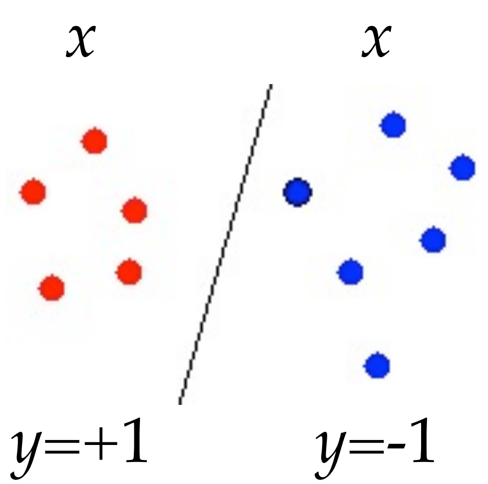
The City University of New York

slides at: http://acl.cs.qc.edu/~lhuang/

What is Structured Prediction?

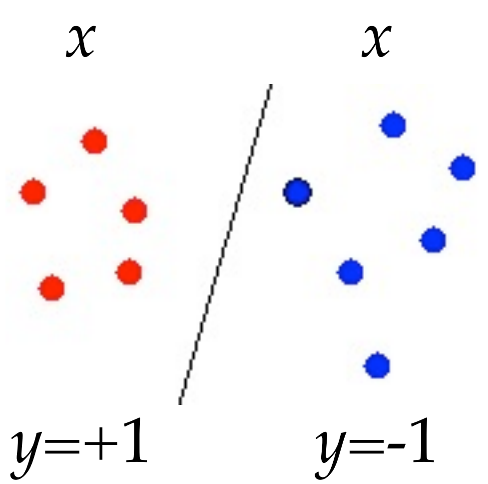


What is Structured Prediction?



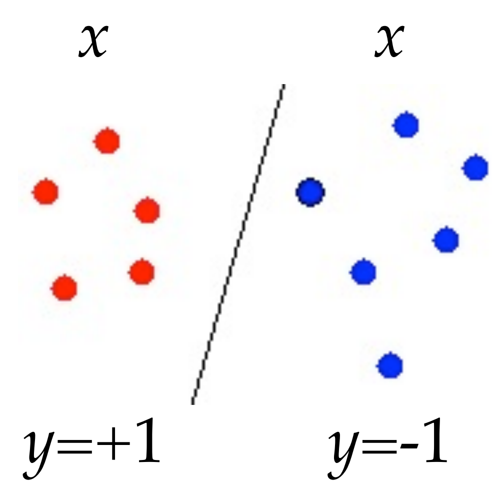
- binary classification: output is binary

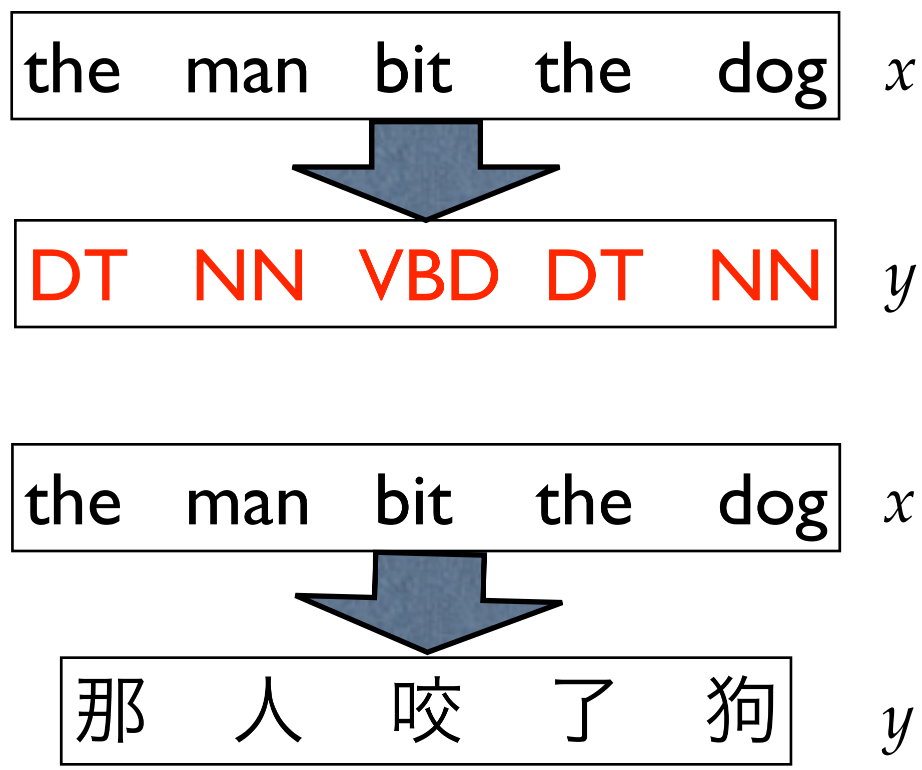
What is Structured Prediction?



- binary classification: output is binary
- multiclass classification: output is a (small) number

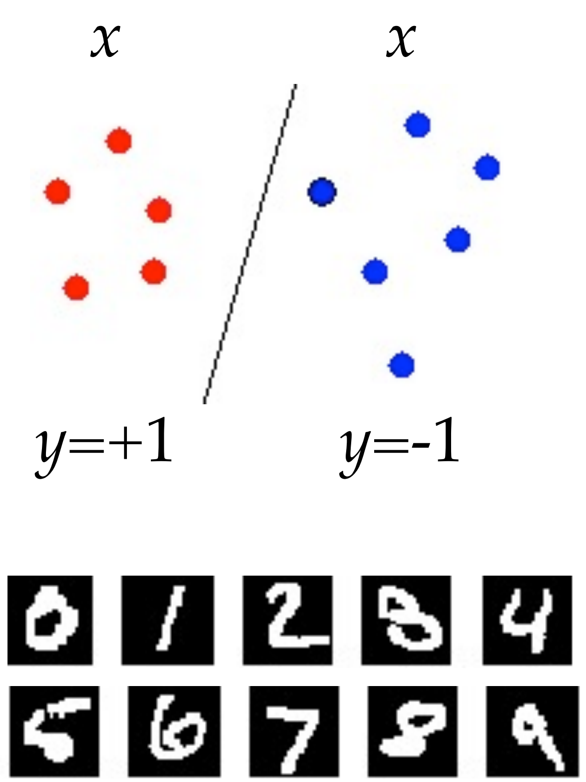
What is Structured Prediction?

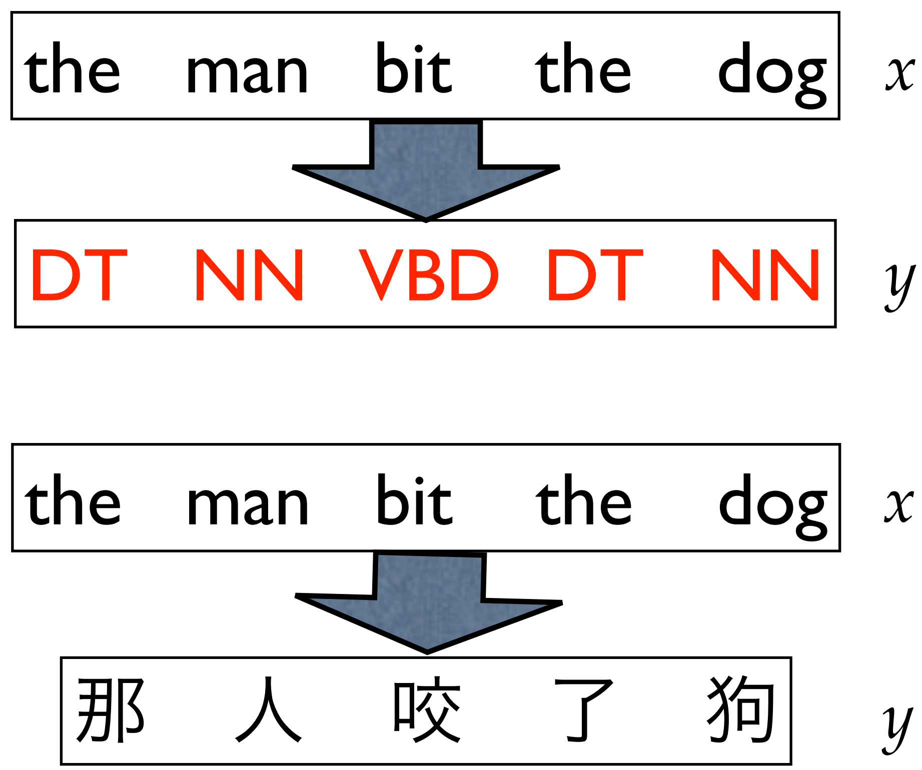


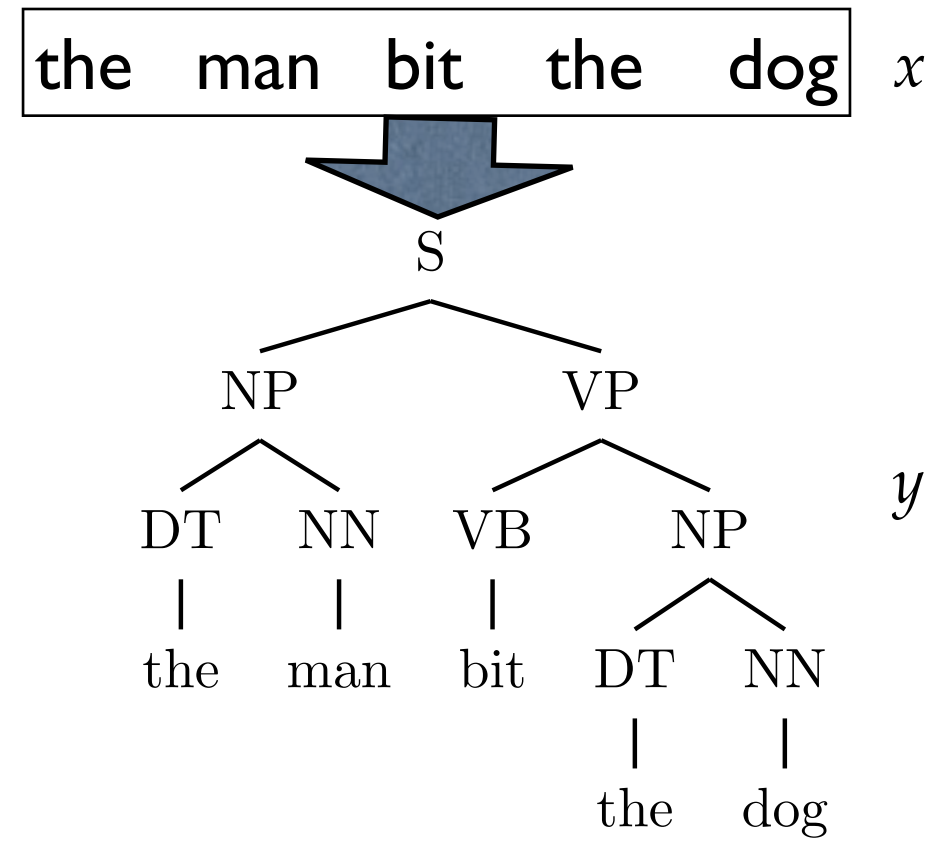


- binary classification: output is binary
- multiclass classification: output is a (small) number
- structured classification: output is a structure (seq., tree, graph)
  - part-of-speech tagging, parsing, summarization, translation

What is Structured Prediction?

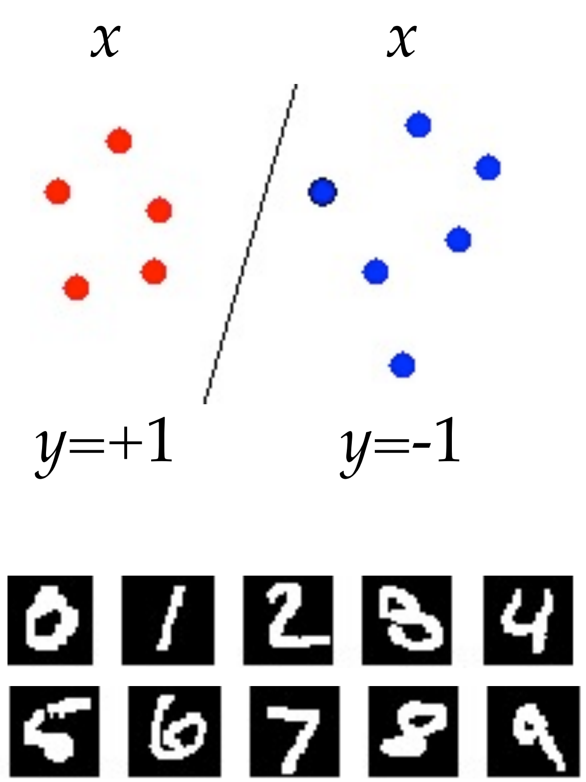


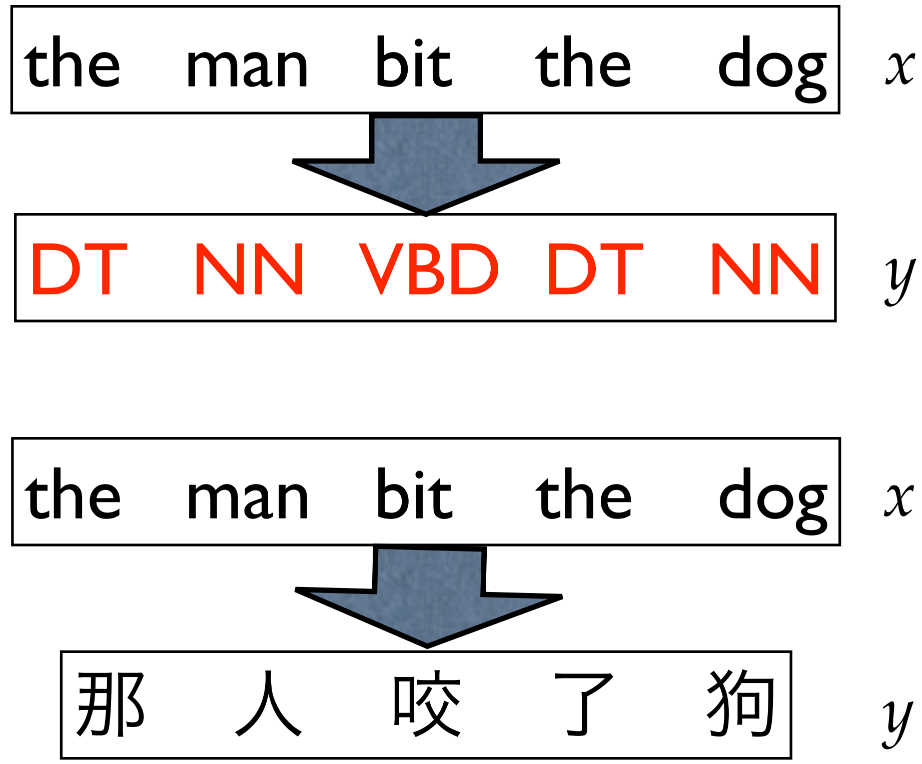


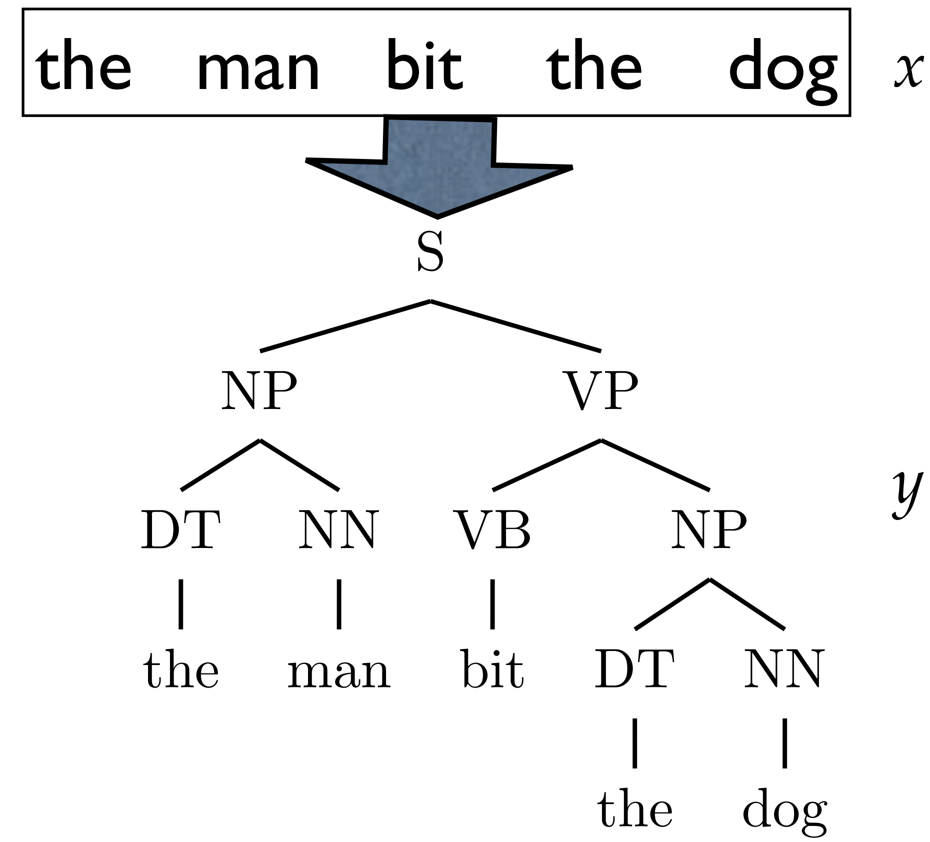


- binary classification: output is binary
- multiclass classification: output is a (small) number
- structured classification: output is a structure (seq., tree, graph)
  - part-of-speech tagging, parsing, summarization, translation

What is Structured Prediction?

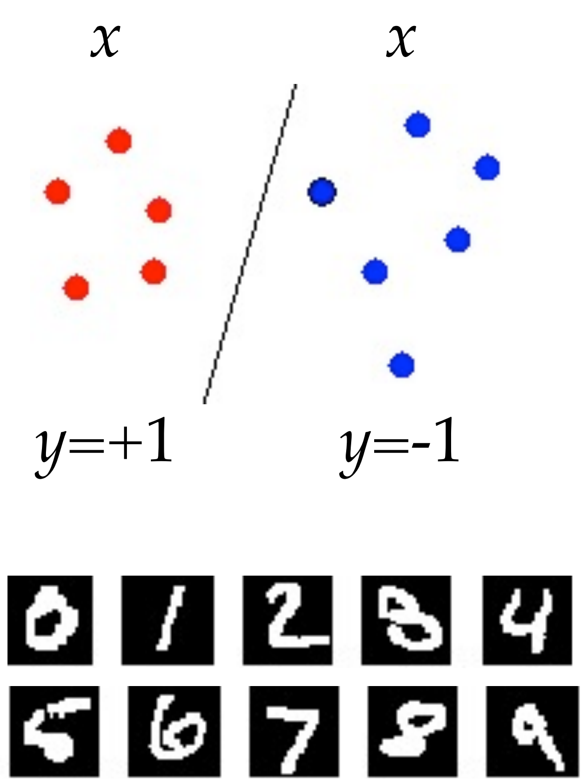


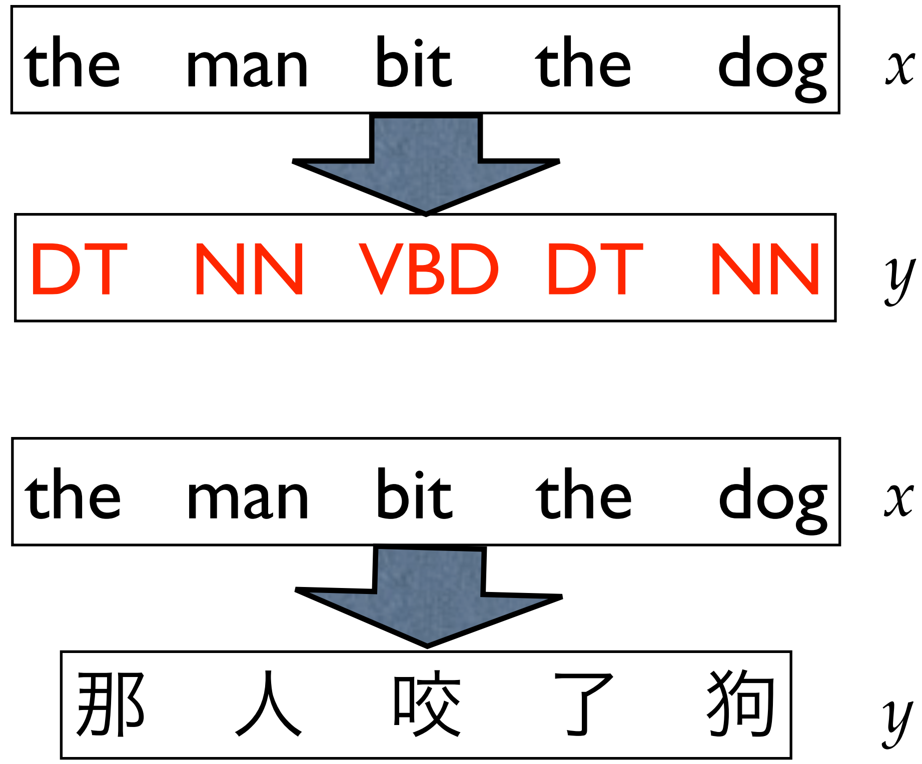


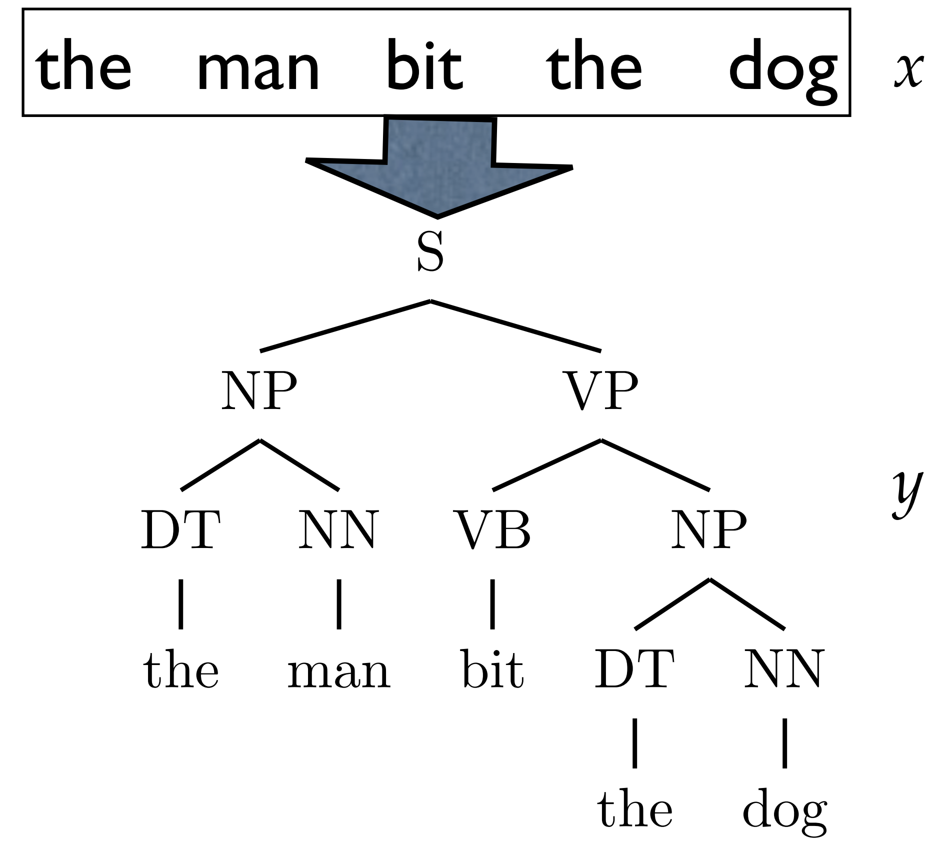


- binary classification: output is binary
- multiclass classification: output is a (small) number
- structured classification: output is a structure (seq., tree, graph)
  - part-of-speech tagging, parsing, summarization, translation
  - exponentially many classes: *search* (inference) efficiency is crucial!

What is Structured Prediction?







- bin
- mu

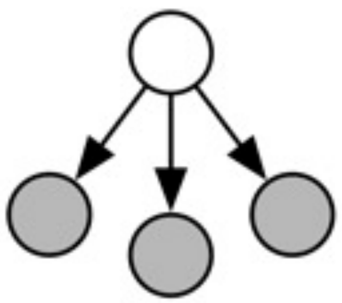
NLP is all about structured prediction!

- structured classification: output is a structure (seq., tree, graph)
  - part-of-speech tagging, parsing, summarization, translation
  - exponentially many classes: *search* (inference) efficiency is crucial!

Examples of Bad Structured Prediction

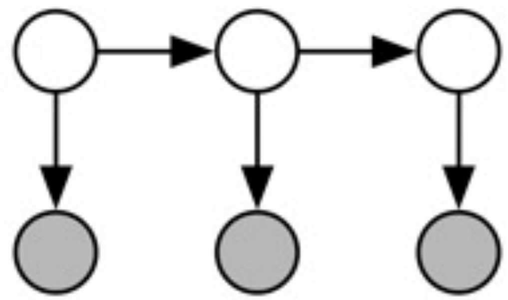
Learning: Unstructured vs. Structured

binary/multiclass



naive bayes

structured learning

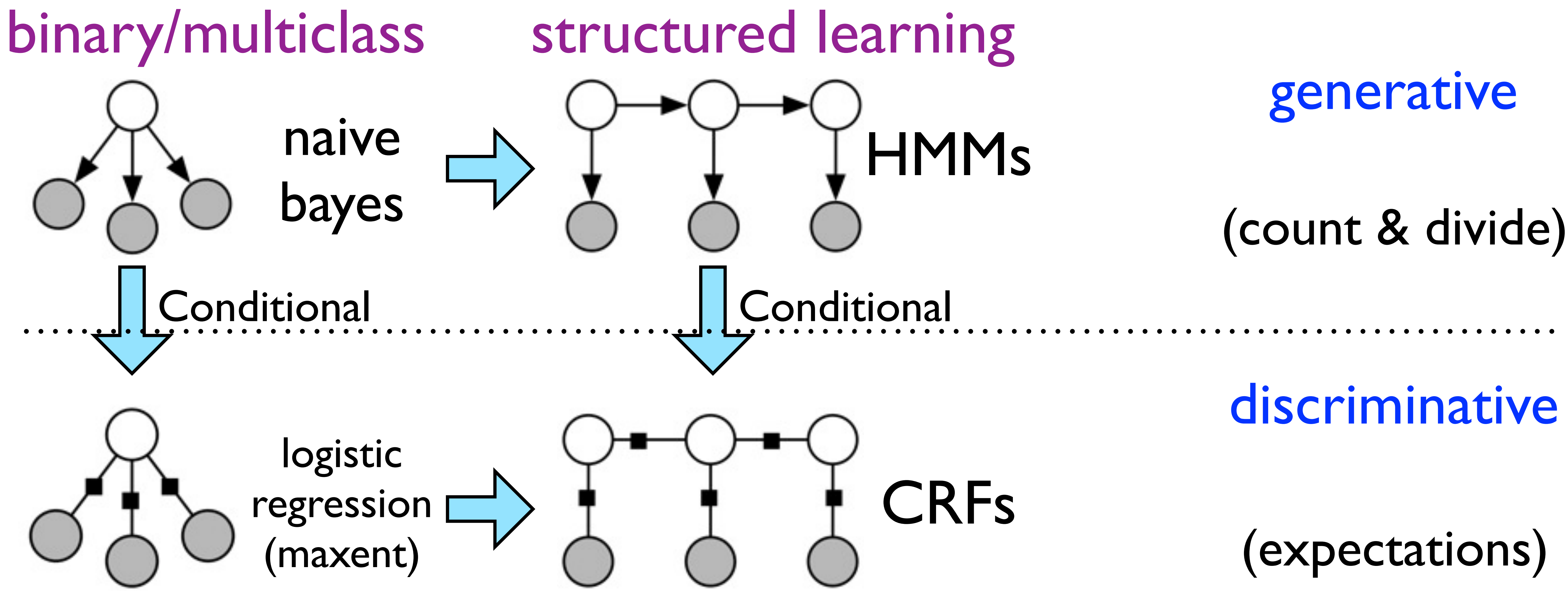


HMMs

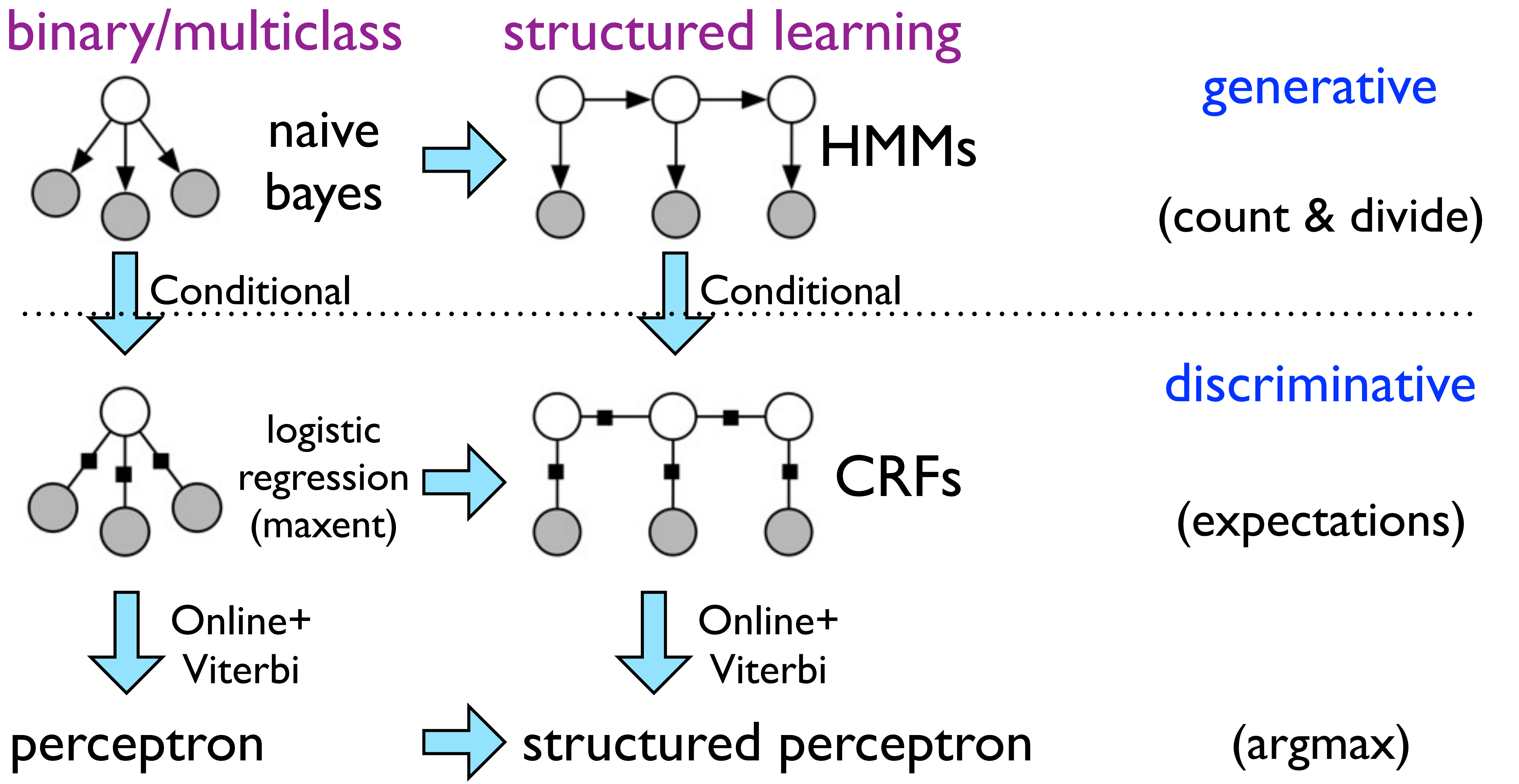
generative

(count & divide)

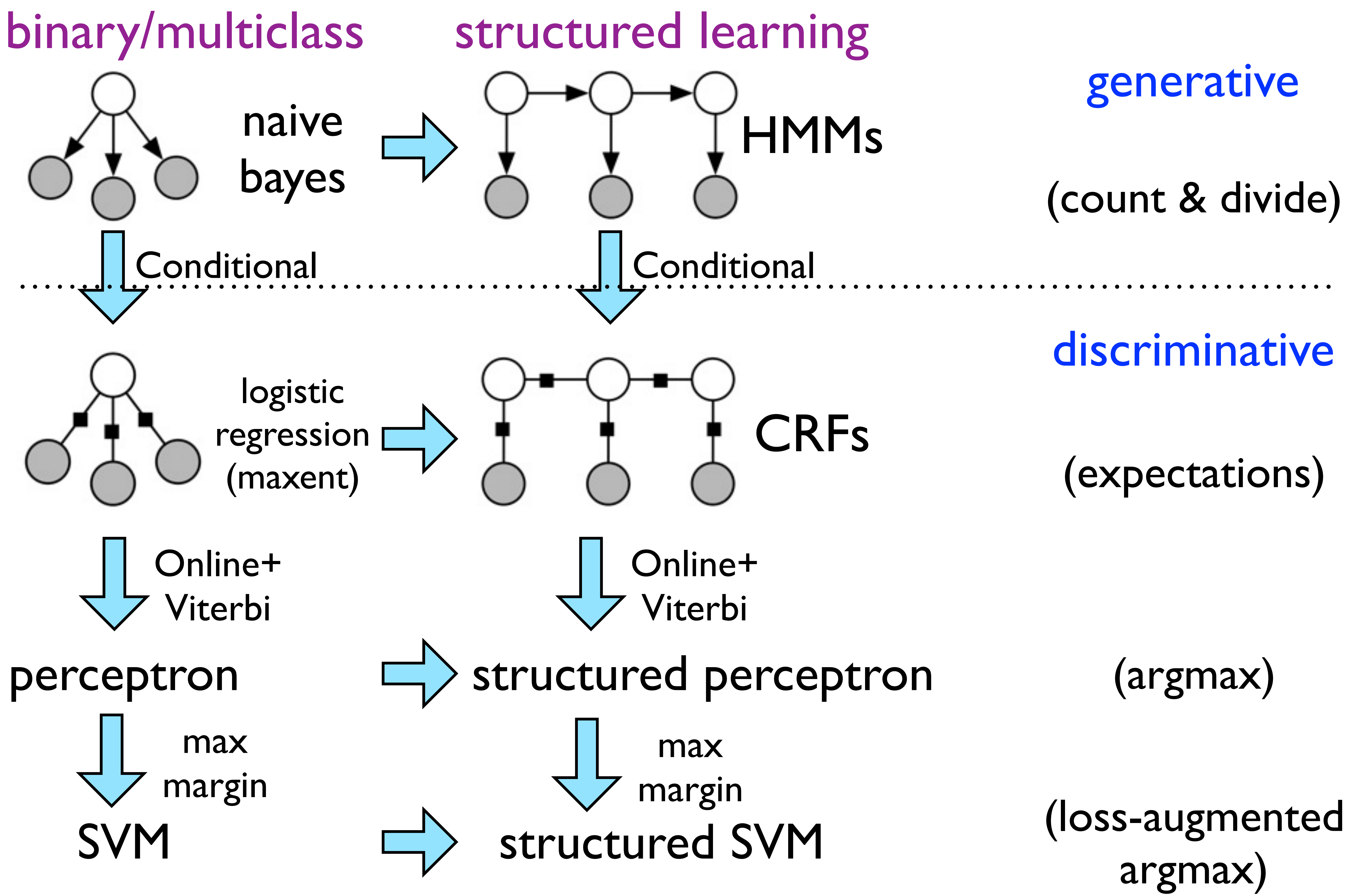
Learning: Unstructured vs. Structured



Learning: Unstructured vs. Structured



Learning: Unstructured vs. Structured



Why Perceptron (Online Learning)?

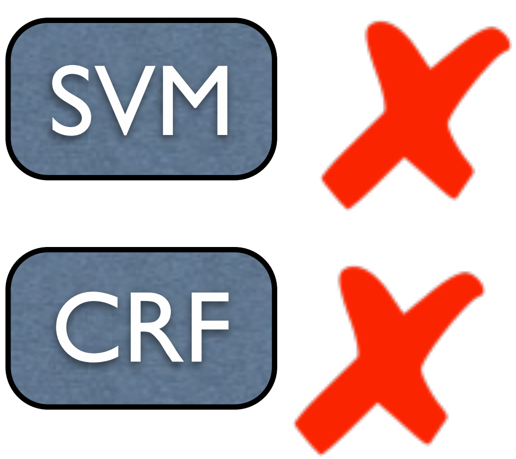
- because we want scalability on big data!
- learning time has to be linear in the number of examples
  - can make only constant number of passes over training data
  - only online learning (perceptron/MIRA) can guarantee this!
  - SVM scales between $O(n^2)$ and $O(n^3)$; CRF no guarantee
- and inference on each example must be super fast
  - another advantage of perceptron: just need argmax

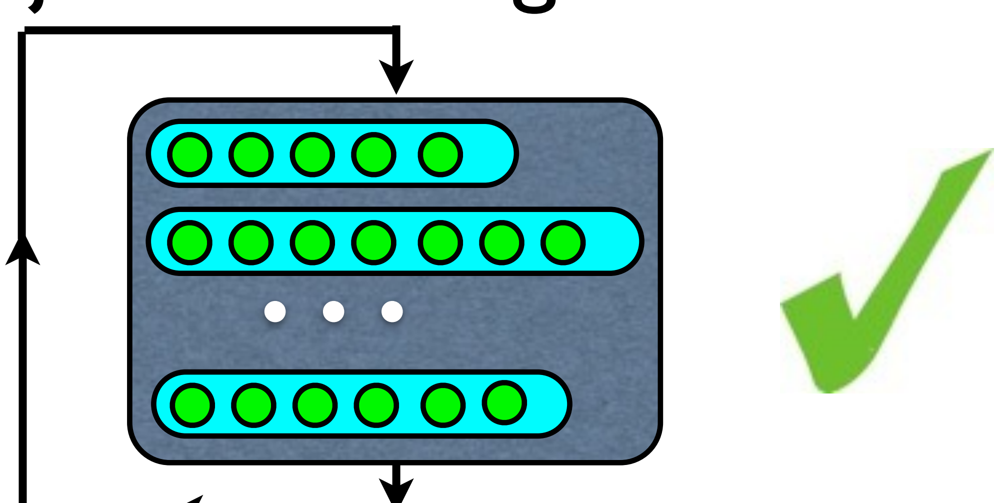
Why Perceptron (Online Learning)?

- because we want scalability on big data!
- learning time has to be linear in the number of examples
  - can make only constant number of passes over training data
  - only online learning (perceptron/MIRA) can guarantee this!
  - SVM scales between $O(n^2)$ and $O(n^3)$; CRF no guarantee
- and inference on each example must be super fast
  - another advantage of perceptron: just need argmax

Why Perceptron (Online Learning)?

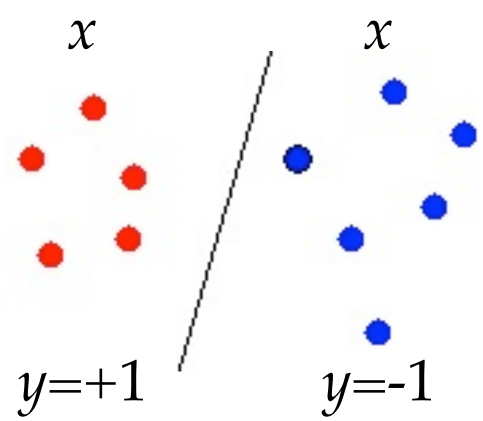
- because we want scalability on big data!
- learning time has to be linear in the number of examples
  - can make only constant number of passes over training data
  - only online learning (perceptron/MIRA) can guarantee this!
  - SVM scales between $O(n^2)$ and $O(n^3)$; CRF no guarantee
- and inference on each example must be super fast
  - another advantage of perceptron: just need argmax



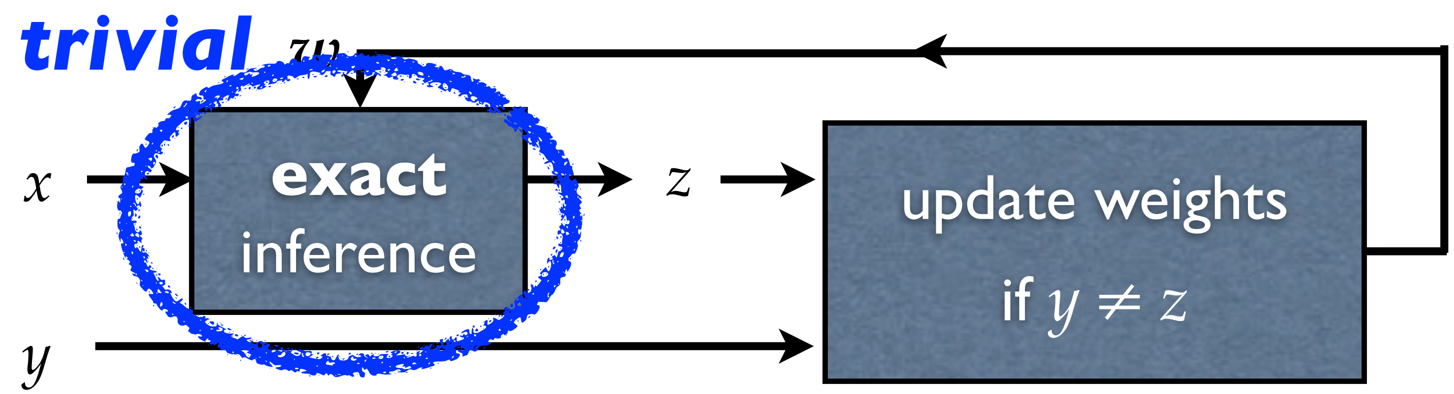


Perceptron: from binary to structured

binary perceptron
(Rosenblatt, 1959)

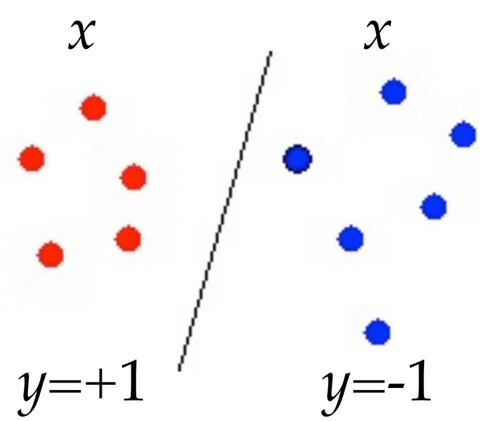


2 classes

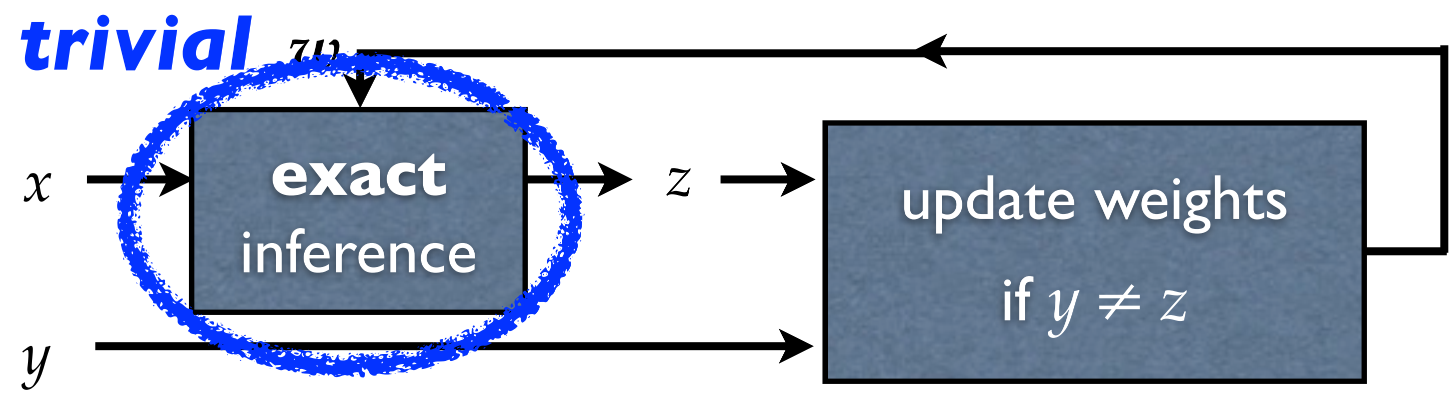


Perceptron: from binary to structured

binary perceptron
(Rosenblatt, 1959)

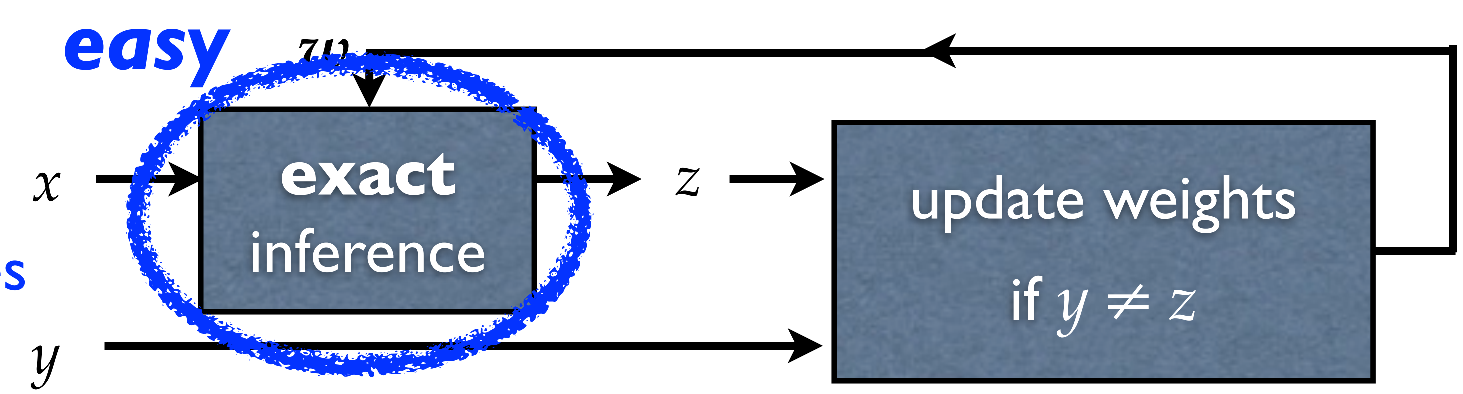


2 classes



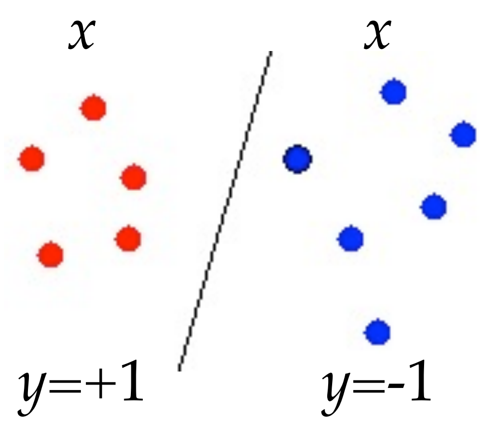
multiclass perceptron
(Freund/Schapire, 1999)

constant
# of classes



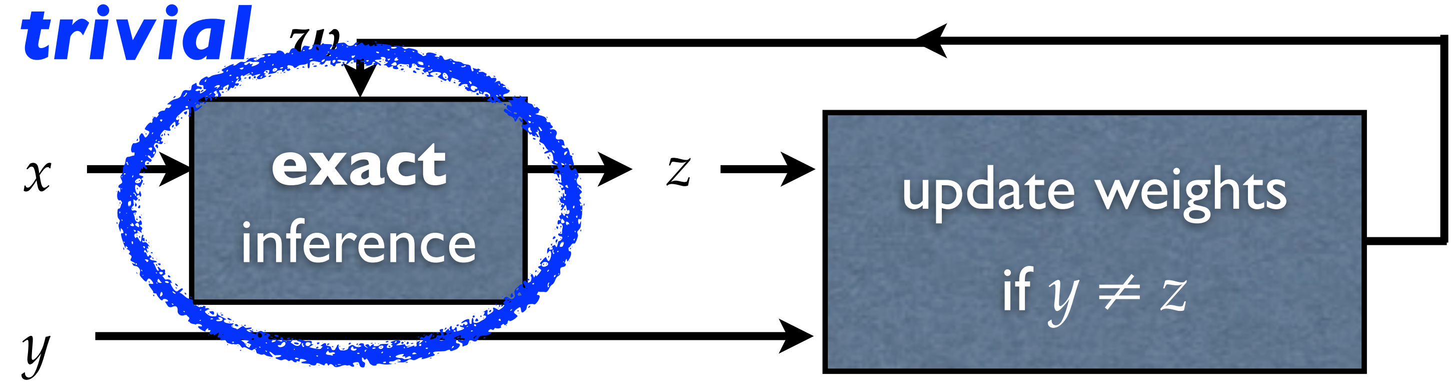
Perceptron: from binary to structured

binary perceptron
(Rosenblatt, 1959)



trivial

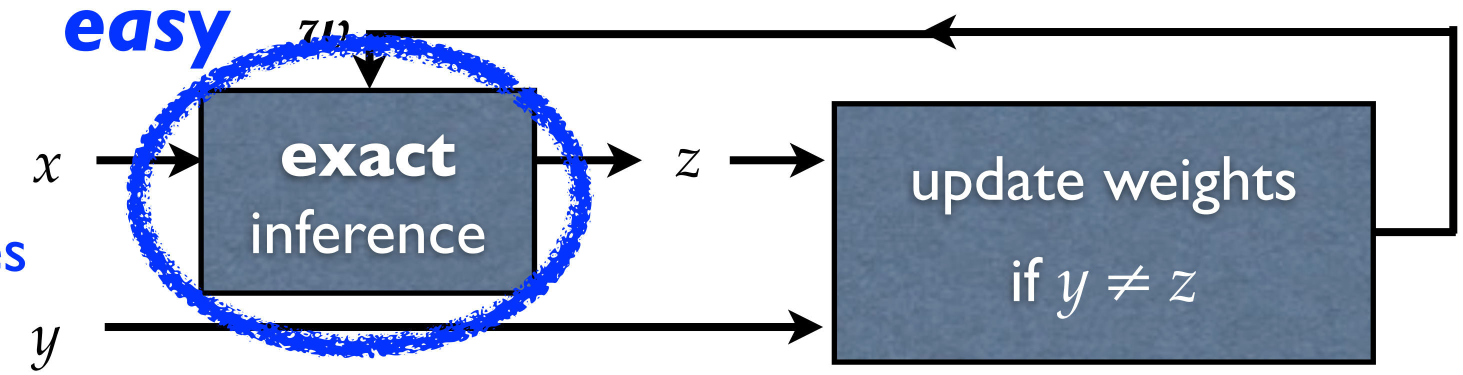
2 classes



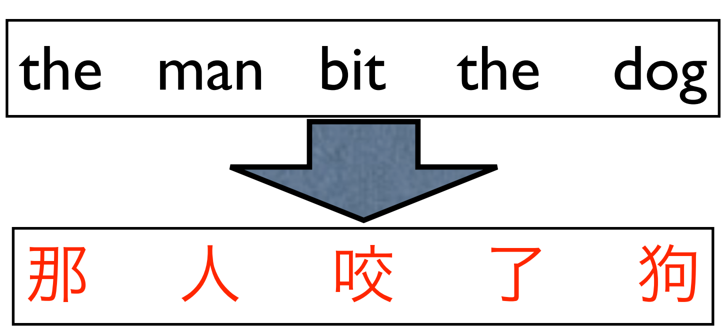
multiclass perceptron
(Freund/Schapire, 1999)

easy

constant
# of classes



structured perceptron
(Collins, 2002)

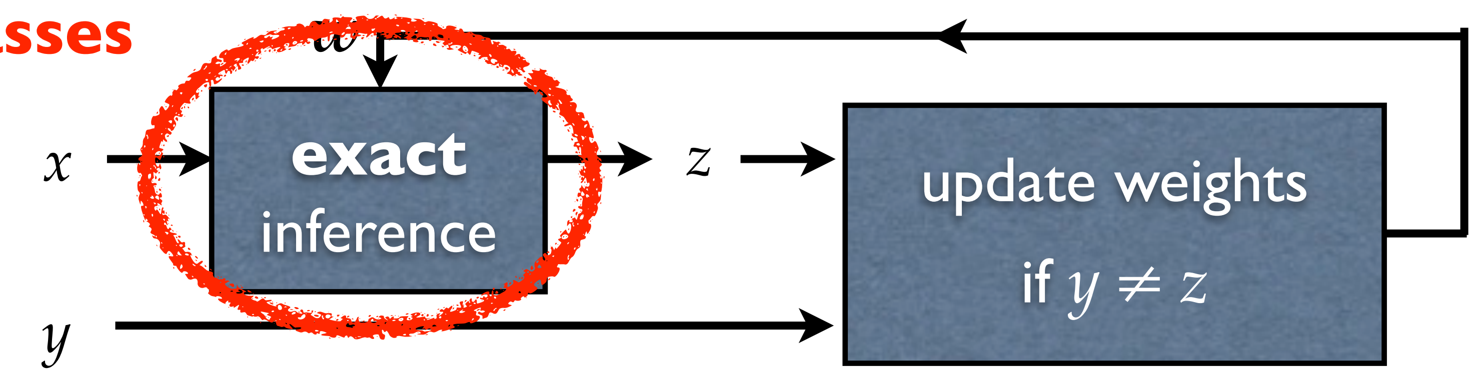


x

y

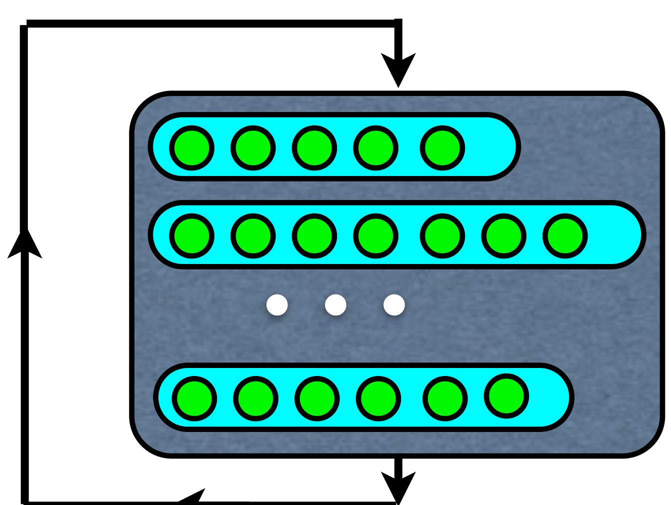
exponential
# of classes

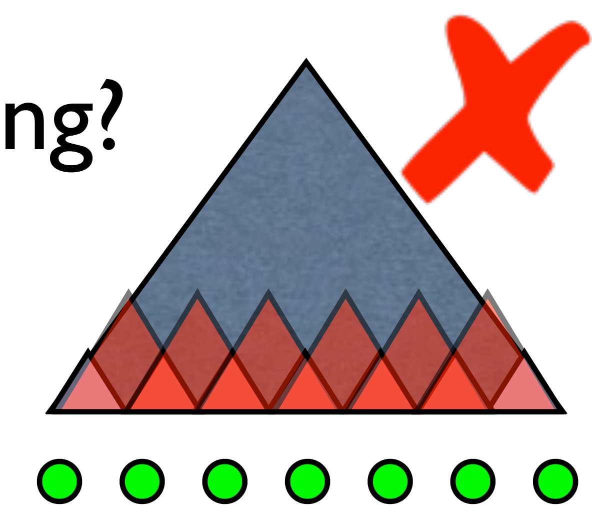
hard



Scalability Challenges

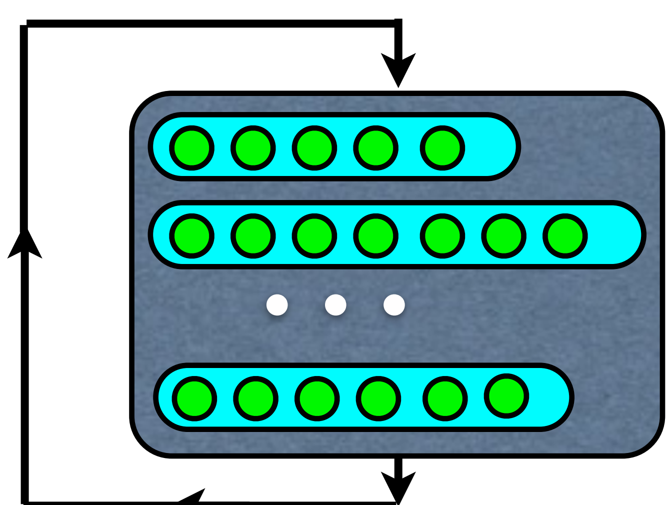
- inference (on one example) is too slow (even w/ DP)
  - can we sacrifice search exactness for faster learning?
  - would inexact search interfere with learning?
  - if so, how should we modify learning?
- even fastest inexact inference is still too slow
  - due to too many training examples
  - can we parallelize online learning?

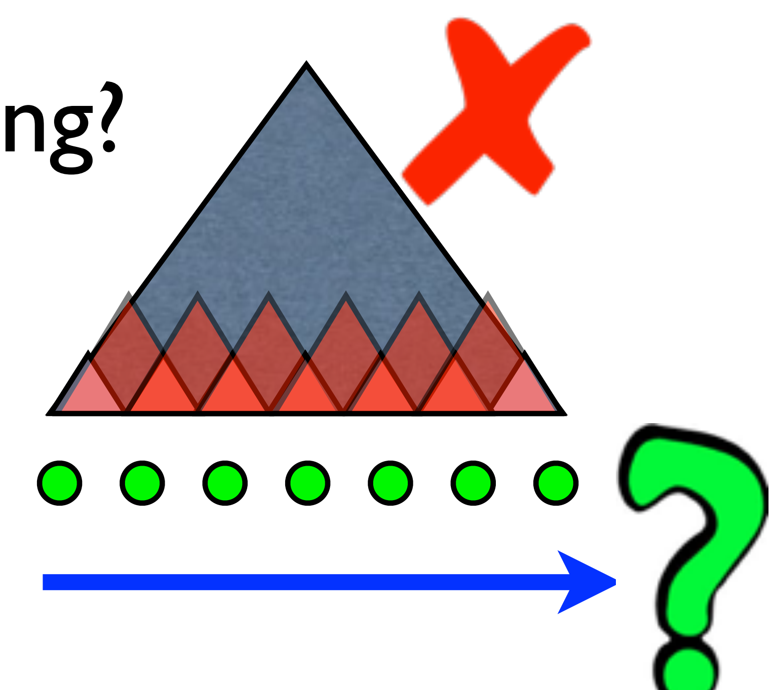




Scalability Challenges

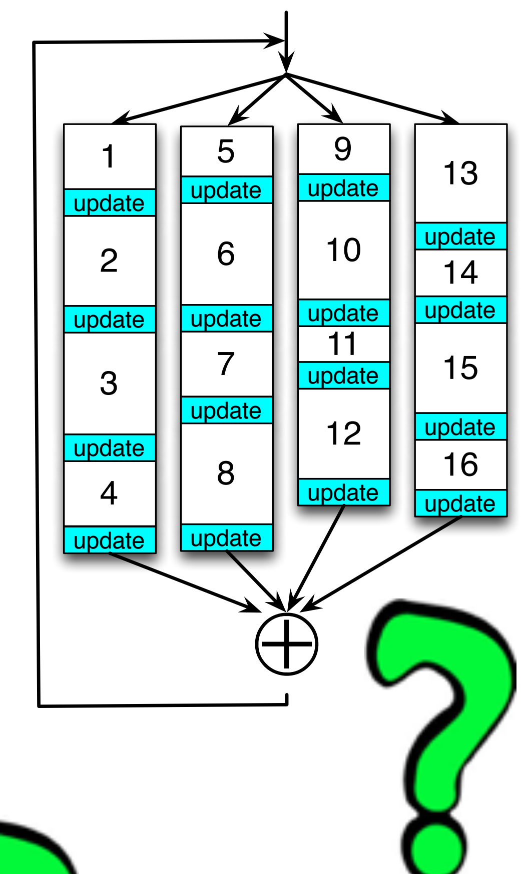
- inference (on one example) is too slow (even w/ DP)
  - can we sacrifice search exactness for faster learning?
  - would inexact search interfere with learning?
  - if so, how should we modify learning?
- even fastest inexact inference is still too slow
  - due to too many training examples
  - can we parallelize online learning?

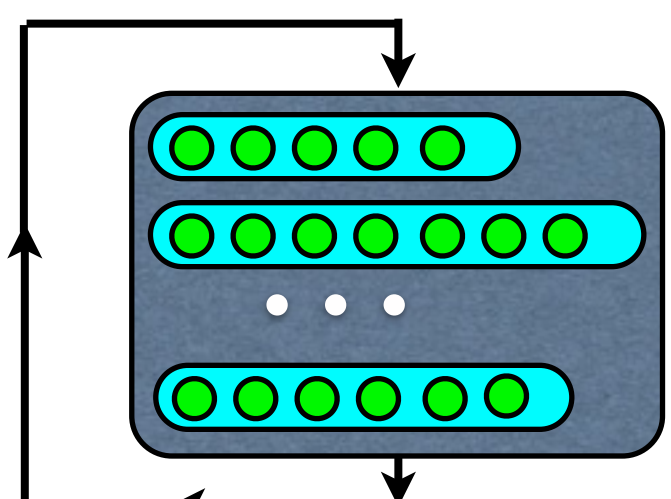


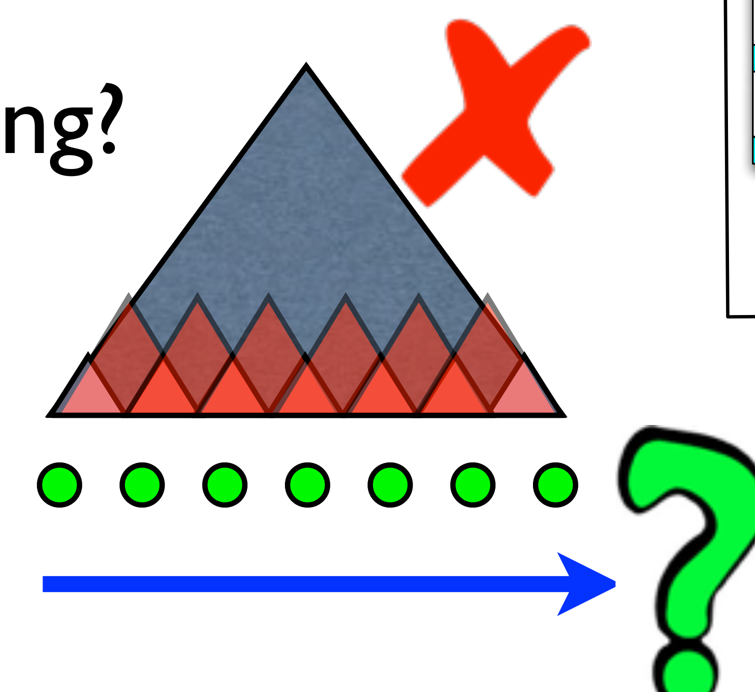


Scalability Challenges

- inference (on one example) is too slow (even w/ DP)
  - can we sacrifice search exactness for faster learning?
  - would inexact search interfere with learning?
  - if so, how should we modify learning?
- even fastest inexact inference is still too slow
  - due to too many training examples
  - can we parallelize online learning?





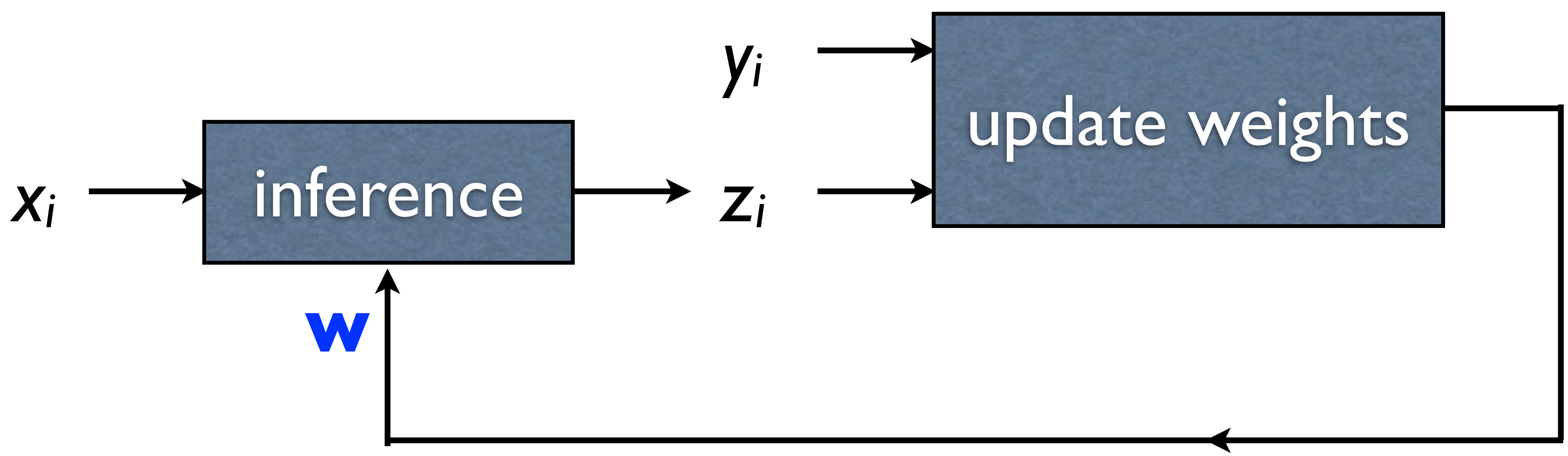


Tutorial Outline

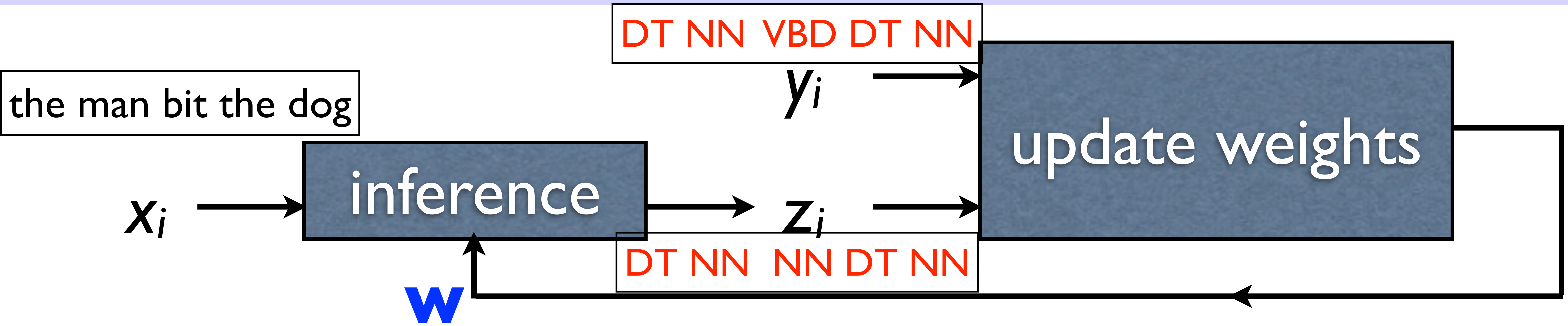
- Overview of Structured Learning
  - Challenges in Scalability
- Structured Perceptron
  - convergence proof
- Structured Perceptron with Inexact Search
- Latent-Variable Structured Perceptron
- Parallelizing Online Learning (Perceptron & MIRA)

Generic Perceptron

- perceptron is the simplest machine learning algorithm
- online-learning: one example at a time
- learning by doing
  - find the best output under the current weights
  - update weights at mistakes



Structured Perceptron



Inputs: Training set (x_i, y_i) for $i = 1 \ldots n$

Initialization: $\mathbf{W} = 0$

Define: $F(x) = \boxed{\operatorname{argmax}_{y \in \mathbf{GEN}(x)}} \mathbf{\Phi}(x, y) \cdot \mathbf{W}$

Algorithm: For $t = 1 \ldots T, i = 1 \ldots n$

$z_i = F(x_i)$

If $(z_i \neq y_i)$ $\boxed{\mathbf{W} \leftarrow \mathbf{W} + \mathbf{\Phi}(x_i, y_i) - \mathbf{\Phi}(x_i, z_i)}$

Output: Parameters $\mathbf{W}$

Example: POS Tagging

y

$\Phi(x, y)$

x

---

z

$\Phi(x, z)$

x

Example: POS Tagging

- gold-standard: DT NN VBD DT NN y
  - the man bit the dog x

$\Phi(x, y)$

- current output: DT NN NN DT NN z
  - the man bit the dog x

$\Phi(x, z)$

- assume only two feature classes
  - tag bigrams t_{i-1} t_i
  - word/tag pairs w_i

Example: POS Tagging

- gold-standard: DT NN VBD DT NN y
  - the man bit the dog x

$\Phi(x, y)$

- current output: DT NN NN DT NN z
  - the man bit the dog x

$\Phi(x, z)$

- assume only two feature classes
  - tag bigrams t_{i-1} t_i
  - word/tag pairs w_i

Example: POS Tagging

- gold-standard: DT NN VBD DT NN y $\Phi(x, y)$
  - the man bit the dog x
- current output: DT NN NN DT NN z $\Phi(x, z)$
  - the man bit the dog x

- assume only two feature classes
  - tag bigrams t_{i-1} t_i
  - word/tag pairs w_i

Example: POS Tagging

- gold-standard: DT NN VBD DT NN y
  - the man bit the dog x

$\Phi(x, y)$

- current output: DT NN NN DT NN z
  - the man bit the dog x

$\Phi(x, z)$

- assume only two feature classes
  - tag bigrams t_{i-1} t_i
  - word/tag pairs w_i

Example: POS Tagging

- gold-standard: DT NN VBD DT NN y
  - the man bit the dog x

$\Phi(x, y)$

- current output: DT NN NN DT NN z
  - the man bit the dog x

$\Phi(x, z)$

- assume only two feature classes
  - tag bigrams t_{i-1} t_i
  - word/tag pairs w_i
- weights ++: (NN,VBD) (VBD, DT) (VBD $\rightarrow$ bit)

Example: POS Tagging

- gold-standard: DT NN VBD DT NN y
  - the man bit the dog x

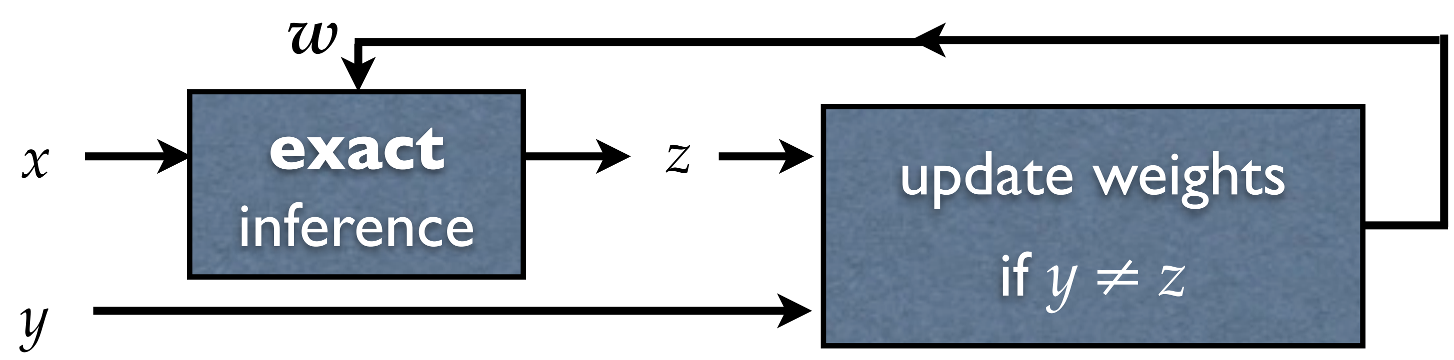
$\Phi(x, y)$

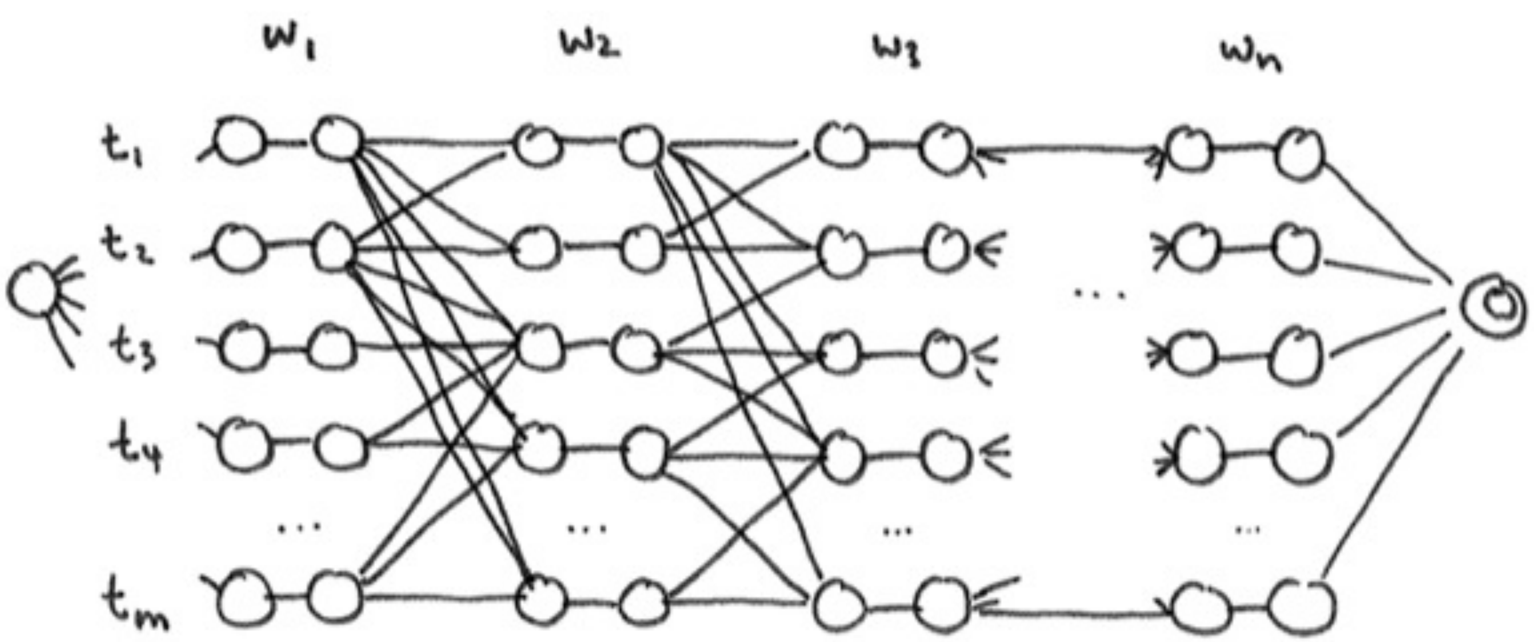
- current output: DT NN NN DT NN z
  - the man bit the dog x

$\Phi(x, z)$

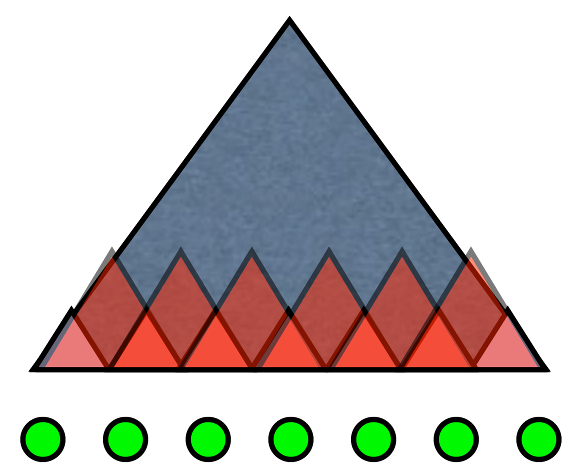
- assume only two feature classes
  - tag bigrams t_{i-1} t_i
  - word/tag pairs w_i
- weights ++: (NN, VBD) (VBD, DT) (VBD $\rightarrow$ bit)
- weights −−: (NN, NN) (NN, DT) (NN $\rightarrow$ bit)

Inference: Dynamic Programming



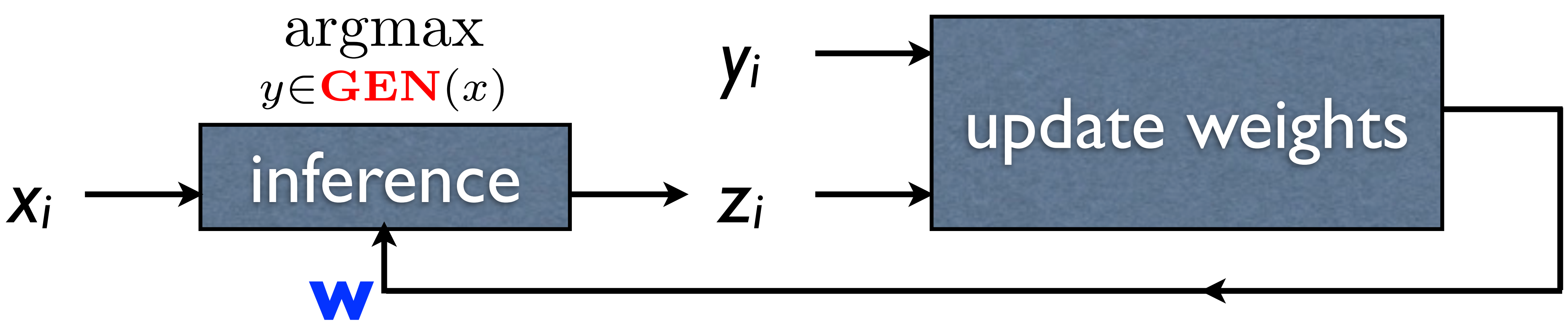


tagging: $O(nT^3)$

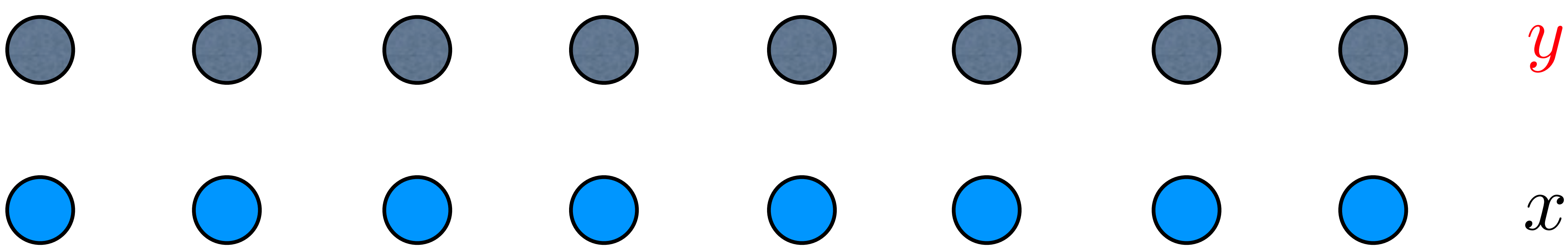


CKY parsing: $O(n^3)$

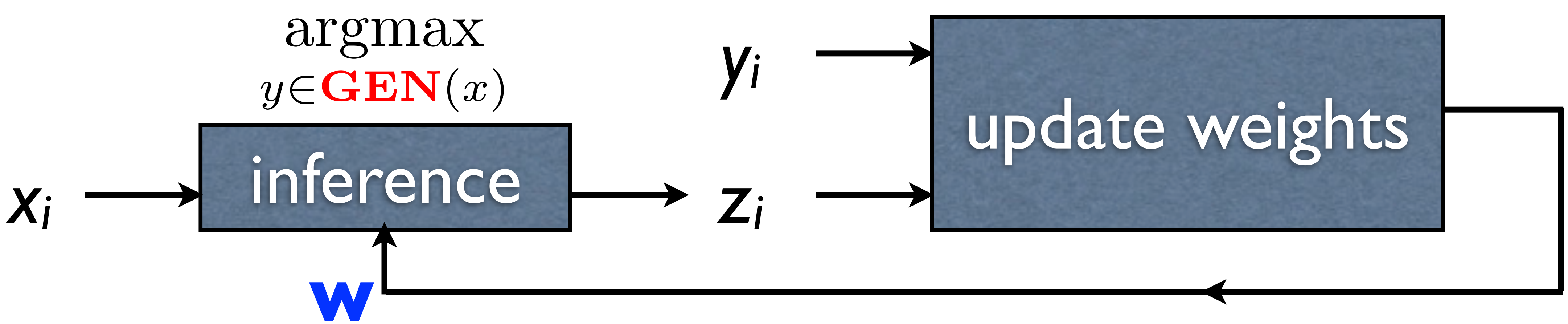
Efficiency vs. Expressiveness



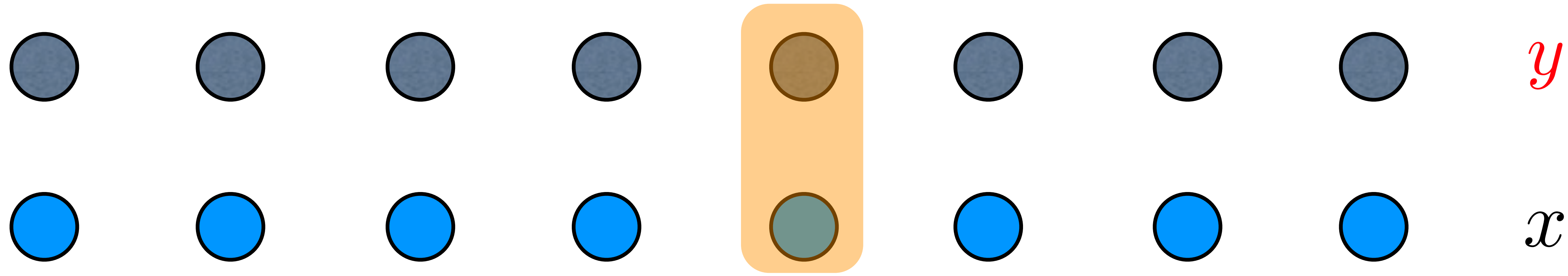
- the inference (argmax) must be efficient
  - either the search space GEN(x) is small, or factored
  - features must be local to y (but can be global to x)
    - e.g. bigram tagger, but look at all input words (cf. CRFs)



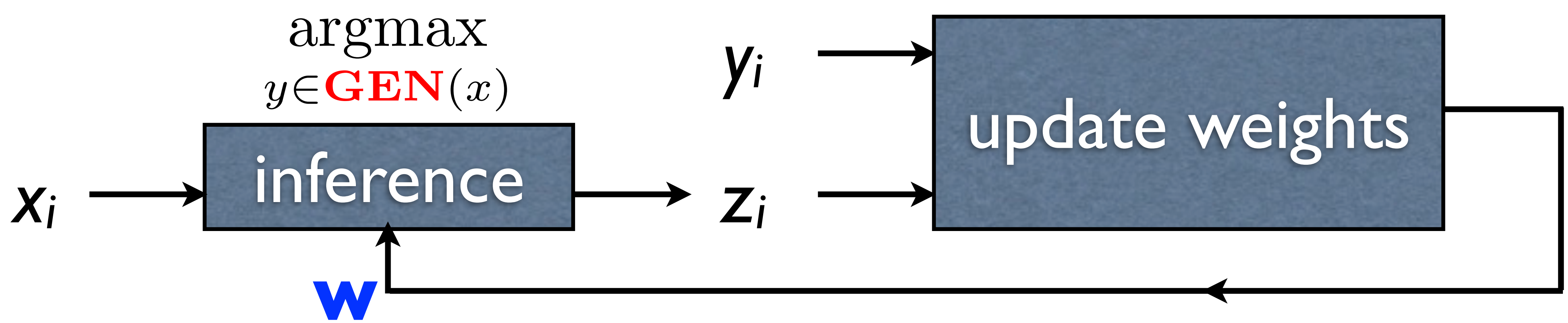
Efficiency vs. Expressiveness



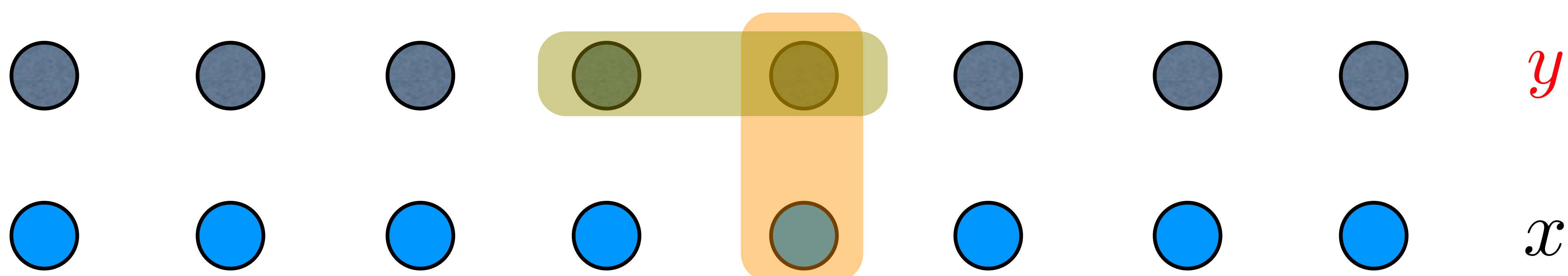
- the inference (argmax) must be efficient
  - either the search space GEN(x) is small, or factored
  - features must be local to y (but can be global to x)
    - e.g. bigram tagger, but look at all input words (cf. CRFs)



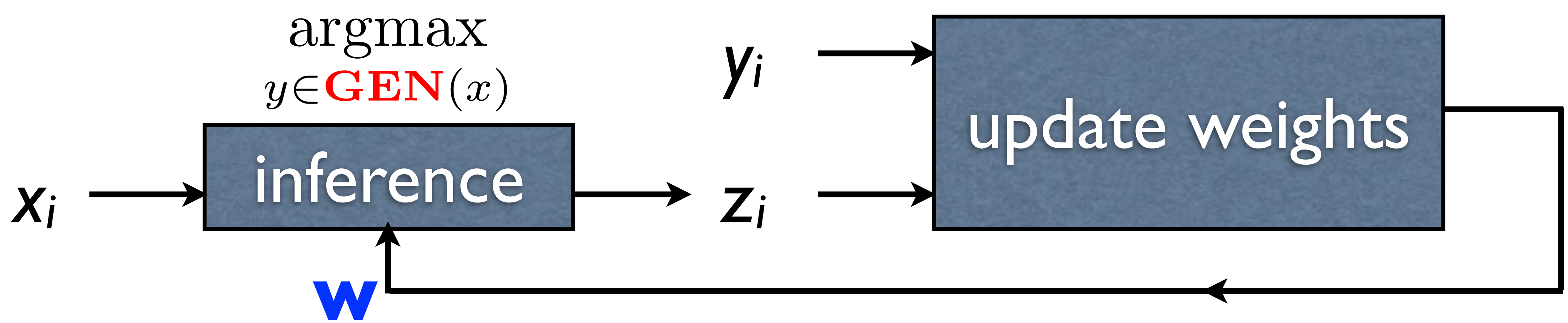
Efficiency vs. Expressiveness



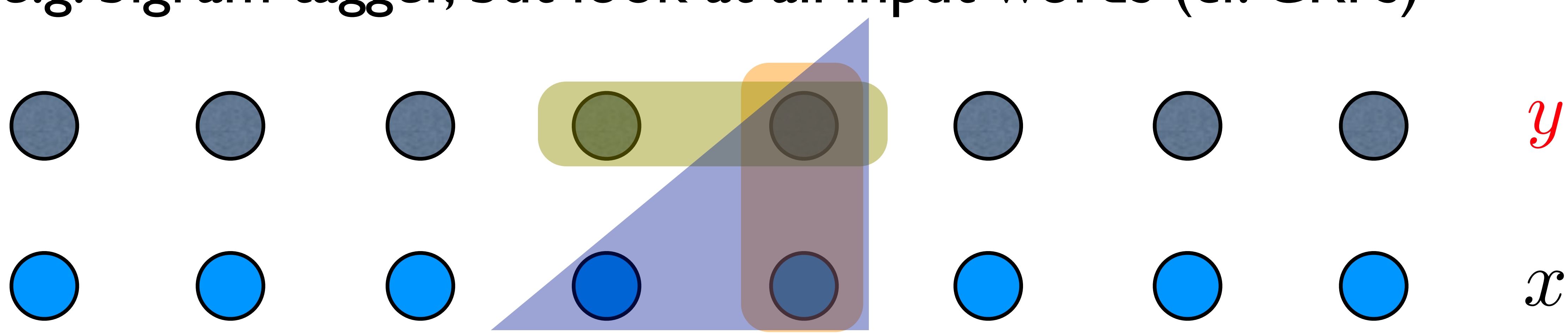
- the inference (argmax) must be efficient
  - either the search space GEN(x) is small, or factored
  - features must be local to y (but can be global to x)
    - e.g. bigram tagger, but look at all input words (cf. CRFs)



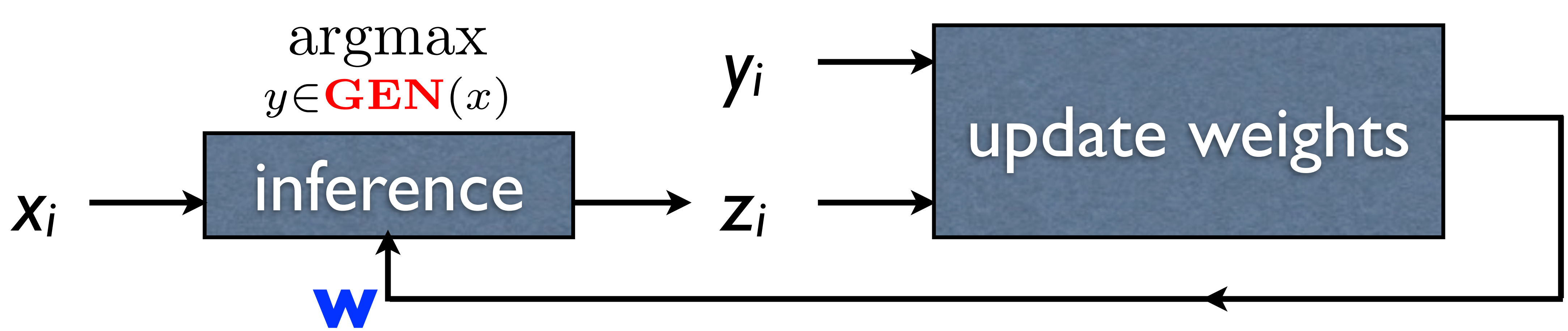
Efficiency vs. Expressiveness



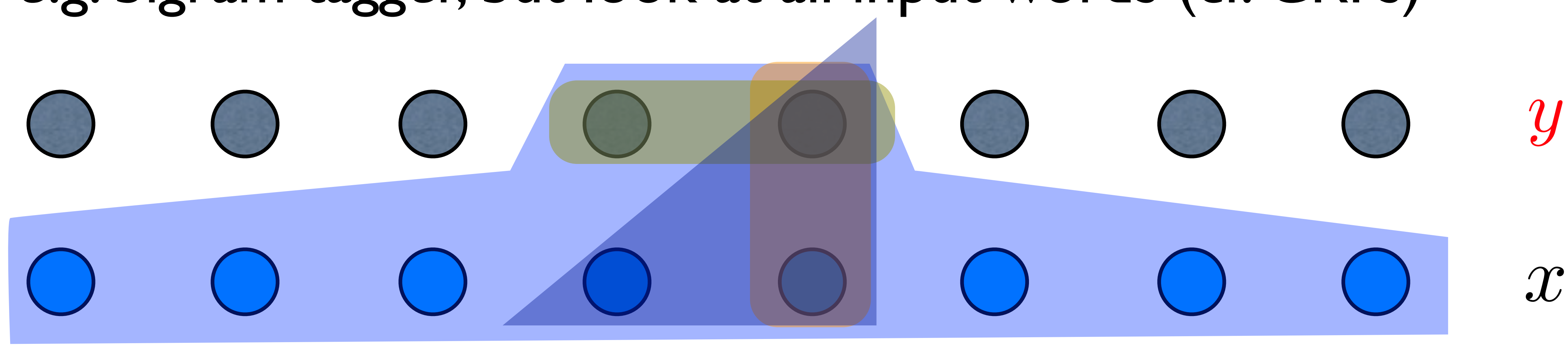
- the inference (argmax) must be efficient
  - either the search space GEN(x) is small, or factored
  - features must be local to y (but can be global to x)
    - e.g. bigram tagger, but look at all input words (cf. CRFs)



Efficiency vs. Expressiveness



- the inference (argmax) must be efficient
  - either the search space GEN(x) is small, or factored
  - features must be local to y (but can be global to x)
    - e.g. bigram tagger, but look at all input words (cf. CRFs)



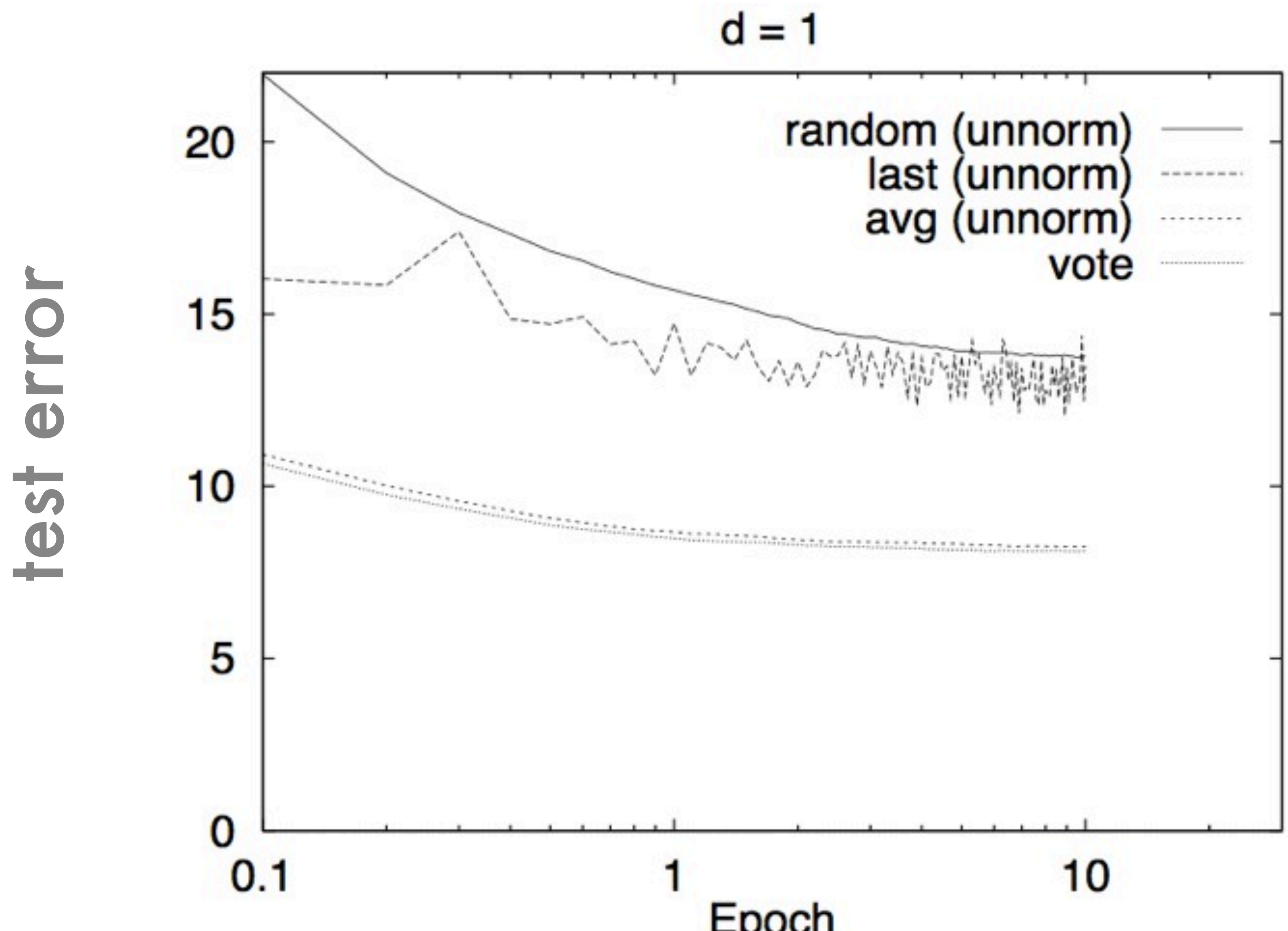
Averaged Perceptron

- much more stable and accurate results
- approximation of voted perceptron (large-margin) (Freund & Schapire, 1999)

Inputs:	Training set (x_i, y_i) for $i = 1 \ldots n$
Initialization:	$\mathbf{W}_0 = 0$
Define:	$F(x) = \mathrm{argmax}_{y \in \mathbf{GEN}(x)} \mathbf{\Phi}(x, y) \cdot \mathbf{W}$
Algorithm:	For $t = 1 \ldots T$, $i = 1 \ldots n$ $z_i = F(x_i)$ If $(z_i \neq y_i)$ $\mathbf{W}_{j+1} \leftarrow \mathbf{W}_j + \mathbf{\Phi}(x_i, y_i) - \mathbf{\Phi}(x_i, z_i)$
Output:	Parameters $\mathbf{W} = \sum_j \mathbf{W}_j$

Averaging => Large Margin

- much more stable and accurate results
- approximation of voted perceptron (large-margin) (Freund & Schapire, 1999)



Efficient Implementation of Averaging

- naive implementation (running sum) doesn't scale
- very clever trick from Daume (2006, PhD thesis)

Algorithm AVERAGEDSTRUCTUREDPERCEPTRON$(x_{1:N}, y_{1:N}, I)$

$\boldsymbol{w}_0 \leftarrow \langle 0, \ldots, 0 \rangle$
$\boldsymbol{w}_a \leftarrow \langle 0, \ldots, 0 \rangle$
$c \leftarrow 1$
for $i = 1 \ldots I$ **do**
    for $n = 1 \ldots N$ **do**
        $\hat{y}_n \leftarrow \arg\max_{y \in \mathcal{Y}} \boldsymbol{w}_0{}^\top \Phi(x_n, y_n)$
        if $y_n \neq \hat{y}_n$ **then**
            $\boldsymbol{w}_0 \leftarrow \boldsymbol{w}_0 + \Phi(x_n, y_n) - \Phi(x_n, \hat{y}_n)$
            $\boldsymbol{w}_a \leftarrow \boldsymbol{w}_a + c\Phi(x_n, y_n) - c\Phi(x_n, \hat{y}_n)$
        end if
        $c \leftarrow c + 1$
    end for
end for
return $\boldsymbol{w}_0 - \boldsymbol{w}_a / c$

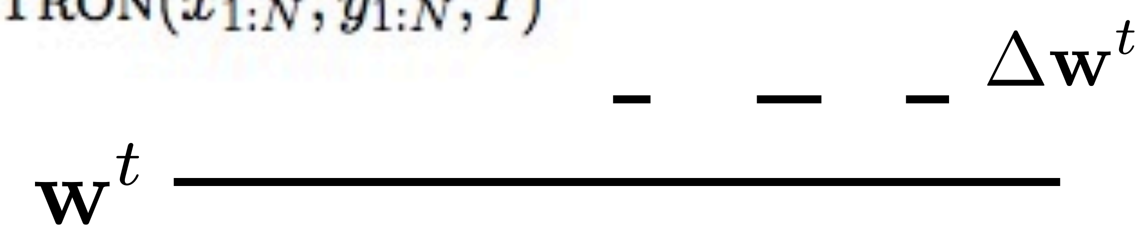


Figure 2.3: The averaged structured perceptron learning algorithm.

Efficient Implementation of Averaging

- naive implementation (running sum) doesn't scale
- very clever trick from Daume (2006, PhD thesis)

Algorithm AVERAGEDSTRUCTUREDPERCEPTRON$(x_{1:N}, y_{1:N}, I)$

$\boldsymbol{w}_0 \leftarrow \langle 0, \ldots, 0 \rangle$
$\boldsymbol{w}_a \leftarrow \langle 0, \ldots, 0 \rangle$
$c \leftarrow 1$
for $i = 1 \ldots I$ **do**
  for $n = 1 \ldots N$ **do**
    $\hat{y}_n \leftarrow \arg\max_{y \in \mathcal{Y}} \boldsymbol{w}_0{}^\top \Phi(x_n, y_n)$
    if $y_n \neq \hat{y}_n$ **then**
      $\boldsymbol{w}_0 \leftarrow \boldsymbol{w}_0 + \Phi(x_n, y_n) - \Phi(x_n, \hat{y}_n)$
      $\boldsymbol{w}_a \leftarrow \boldsymbol{w}_a + c\Phi(x_n, y_n) - c\Phi(x_n, \hat{y}_n)$
    end if
    $c \leftarrow c + 1$
  end for
end for
return $\boldsymbol{w}_0 - \boldsymbol{w}_a / c$

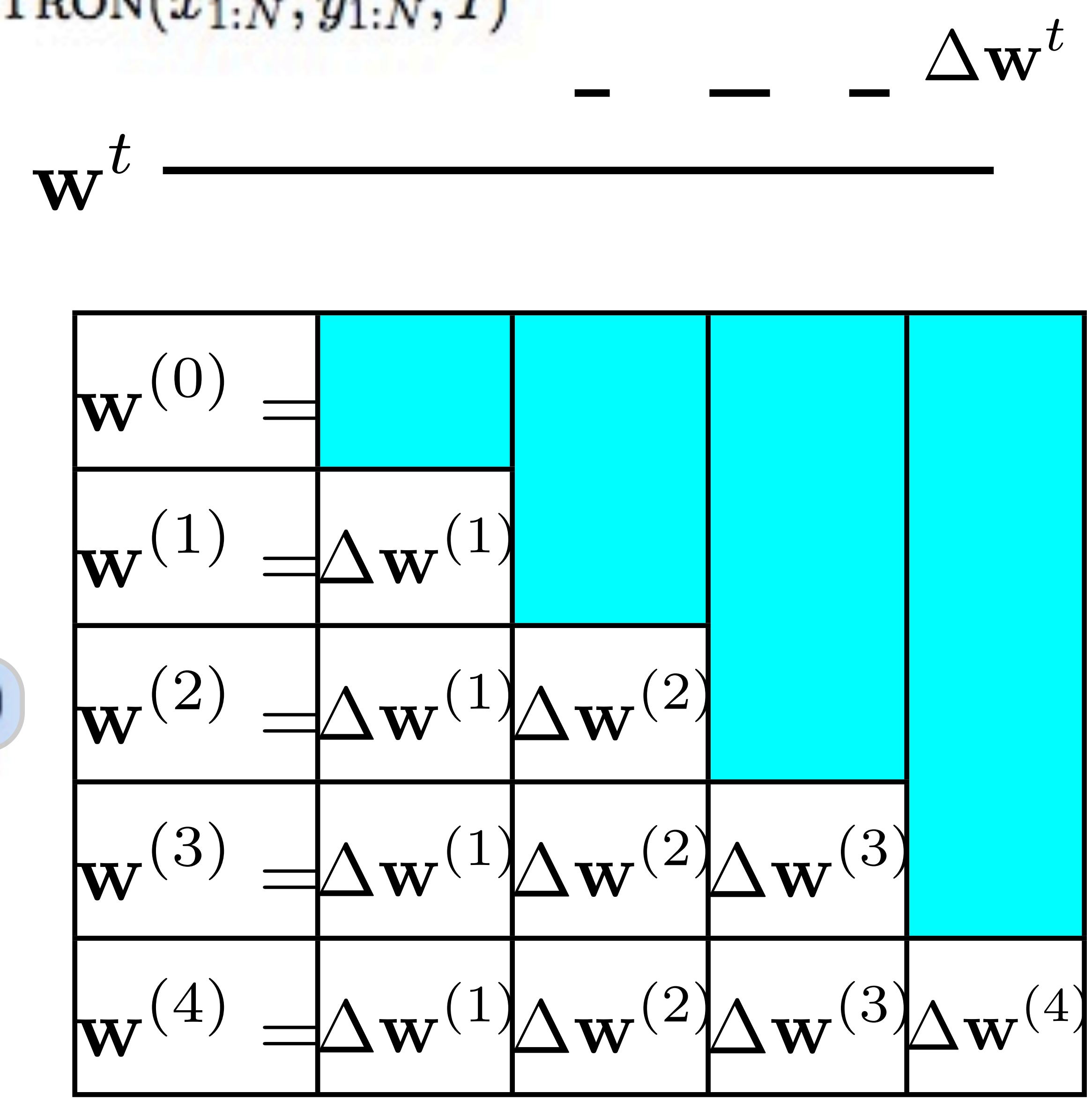


Figure 2.3: The averaged structured perceptron learning algorithm.

Perceptron vs. CRFs

- perceptron is online and Viterbi approximation of CRF
- simpler to code; faster to converge; ~same accuracy

CRFs
(Lafferty et al, 2001)

$$\sum_{(x,y)\in D} \sum_{z\in \mathbf{GEN}(x)} \frac{\exp(\mathbf{w}\cdot\Phi(x,z))}{Z(x)}$$

$$\text{for } (x,y)\in D,\ \underset{z\in\mathbf{GEN}(x)}{\mathbf{argmax}}\ \mathbf{w}\cdot\Phi(x,z)$$

structured perceptron
(Collins, 2002)

Perceptron vs. CRFs

- perceptron is online and Viterbi approximation of CRF
- simpler to code; faster to converge; ~same accuracy

CRFs
(Lafferty et al, 2001)

online

$$\boxed{\sum_{(x,y)\in D}} \sum_{z\in \mathbf{GEN}(x)} \frac{\exp(\mathbf{w}\cdot\Phi(x,z))}{Z(x)}$$

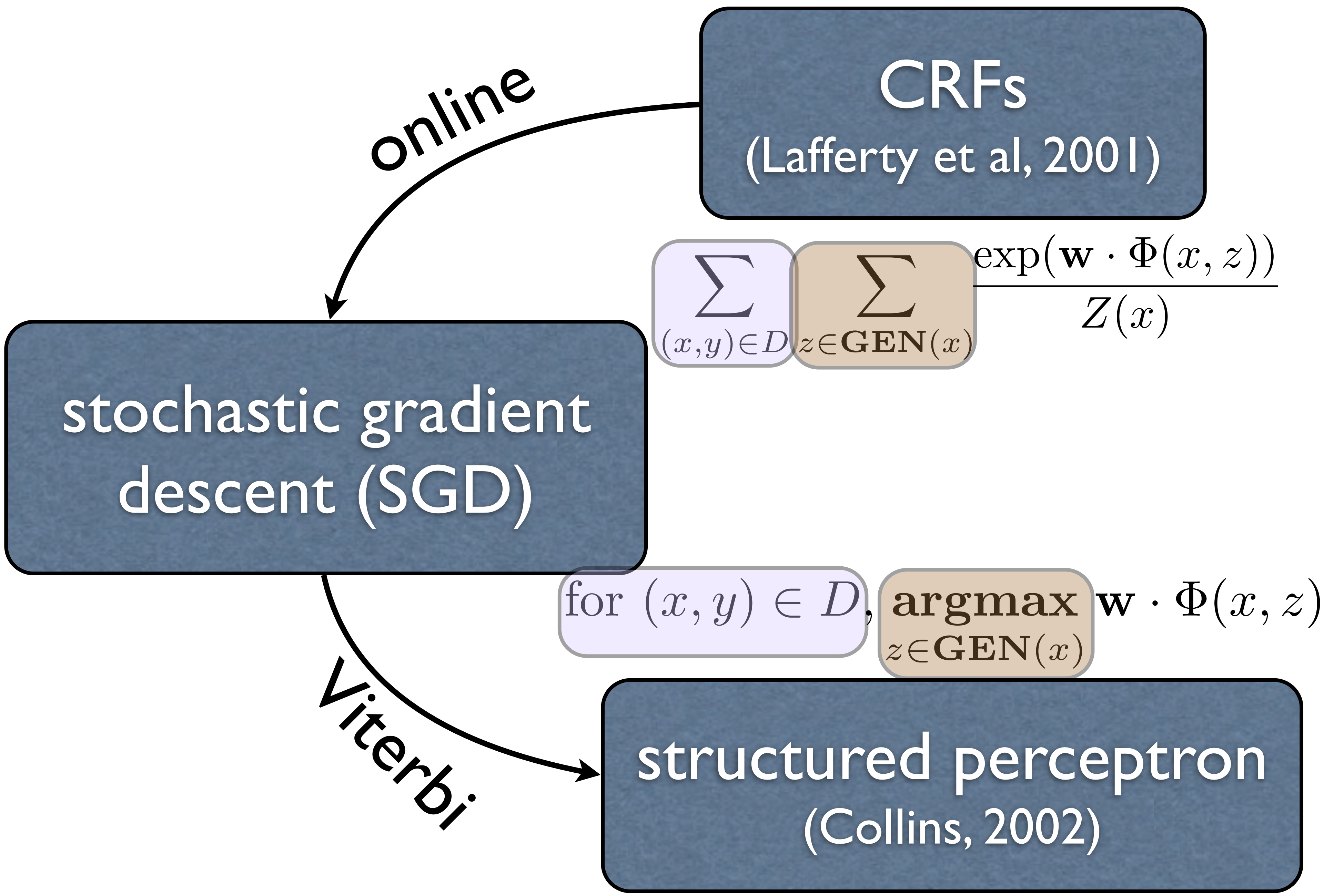
stochastic gradient descent (SGD)

$$\boxed{\text{for } (x,y)\in D}, \ \mathop{\mathbf{argmax}}_{z\in \mathbf{GEN}(x)} \mathbf{w}\cdot\Phi(x,z)$$

structured perceptron
(Collins, 2002)

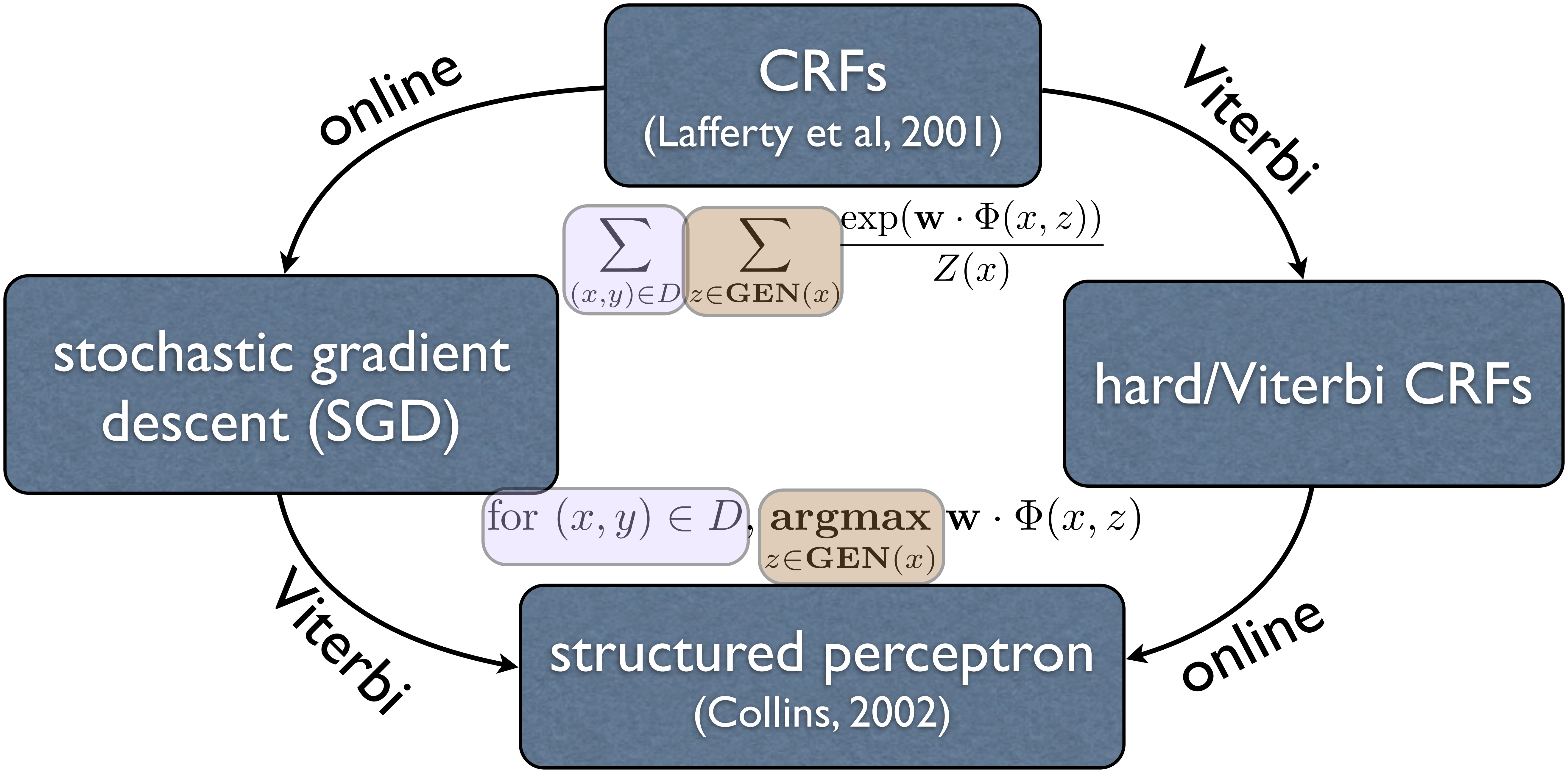
Perceptron vs. CRFs

- perceptron is online and Viterbi approximation of CRF
- simpler to code; faster to converge; ~same accuracy



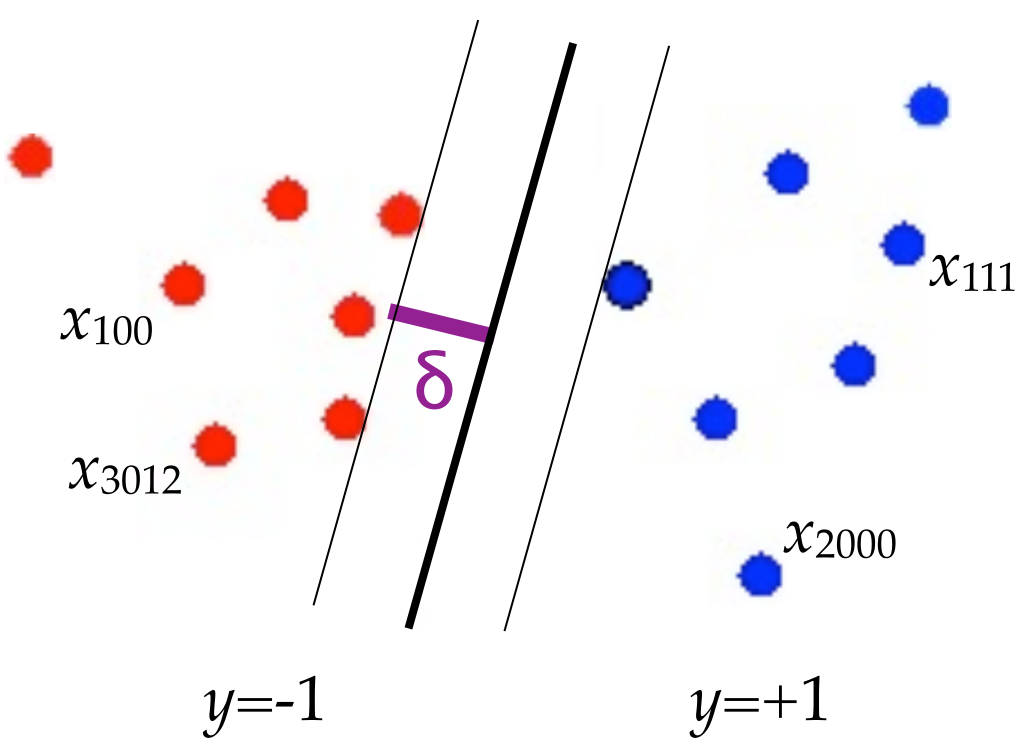
Perceptron vs. CRFs

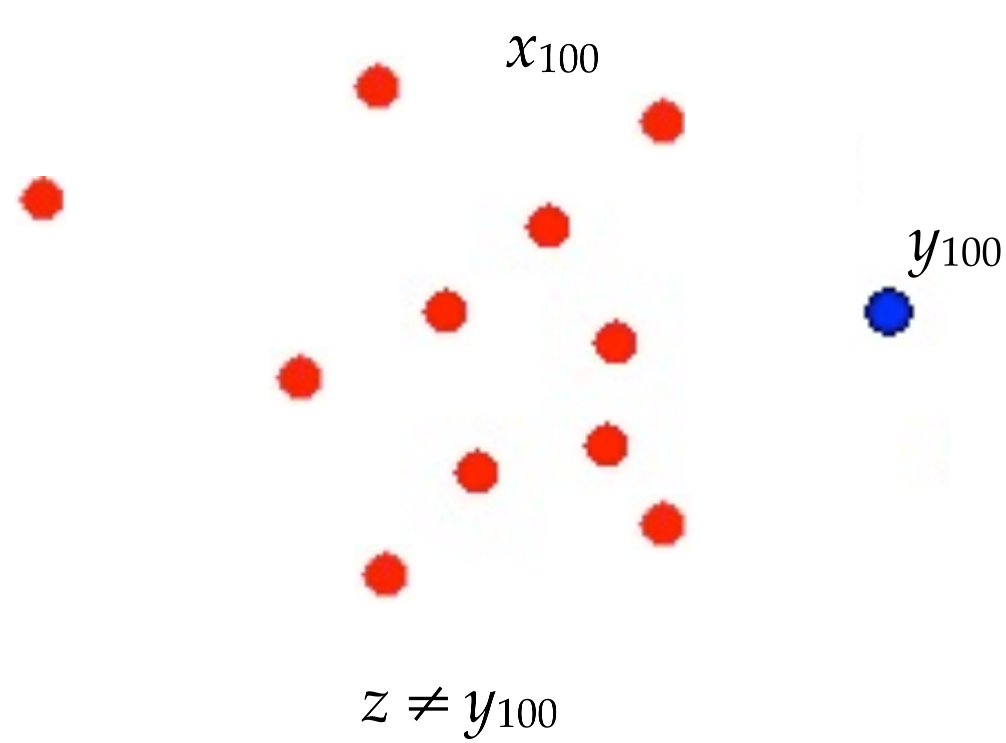
- perceptron is online and Viterbi approximation of CRF
- simpler to code; faster to converge; ~same accuracy



Perceptron Convergence Proof

- binary classification: converges iff. data is separable
- structured prediction: converges iff. data is separable
  - there is an oracle vector that correctly labels all examples
  - one vs the rest (correct label better than all incorrect labels)
- theorem: if separable, then **# of updates $\leq$ R^2 / δ^2** R: diameter

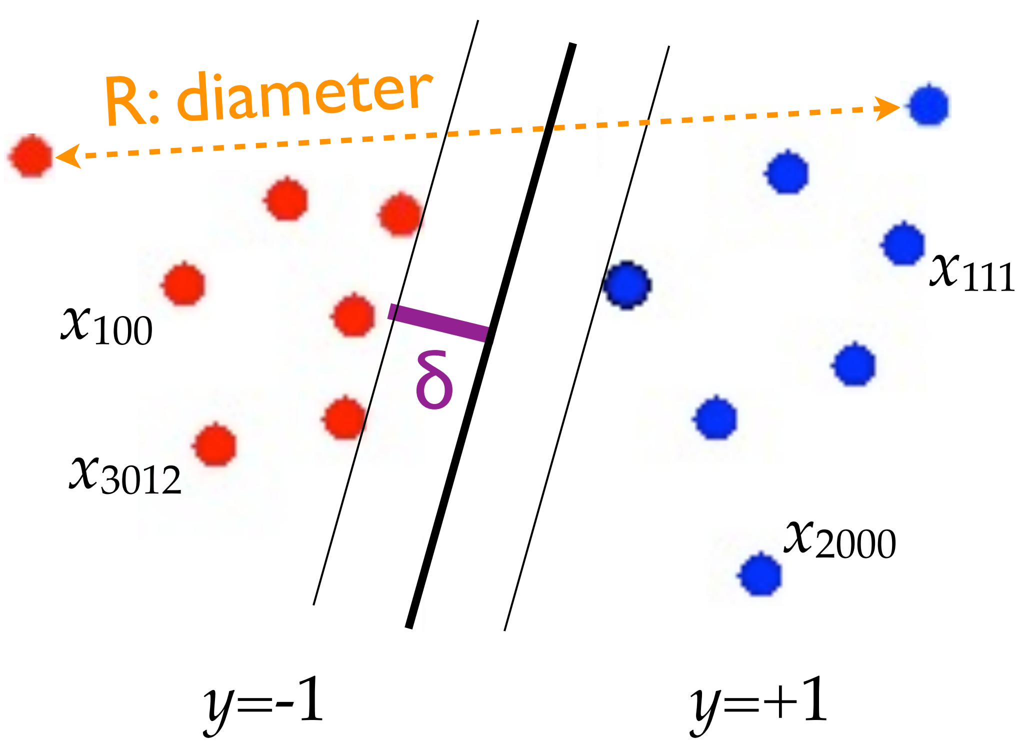


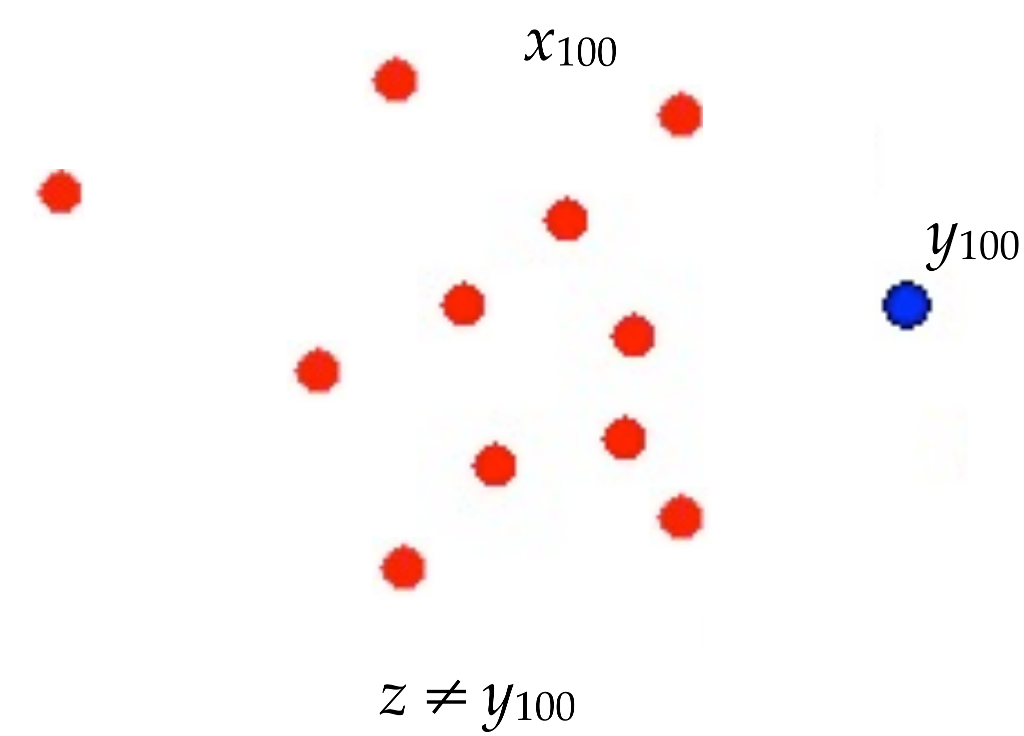


Novikoff => Freund & Schapire => Collins
1962 1999 2002

Perceptron Convergence Proof

- binary classification: converges iff. data is separable
- structured prediction: converges iff. data is separable
  - there is an oracle vector that correctly labels all examples
  - one vs the rest (correct label better than all incorrect labels)
- theorem: if separable, then **# of updates ≤ R^2 / δ^2** R: diameter

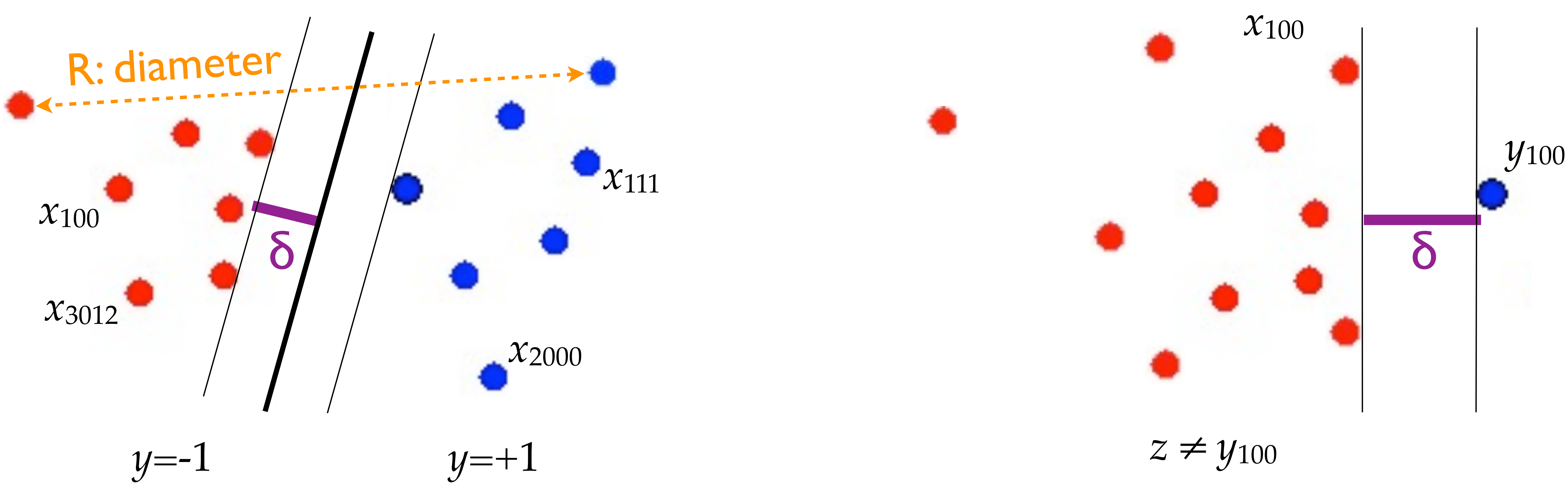




Novikoff => Freund & Schapire => Collins
1962 1999 2002

Perceptron Convergence Proof

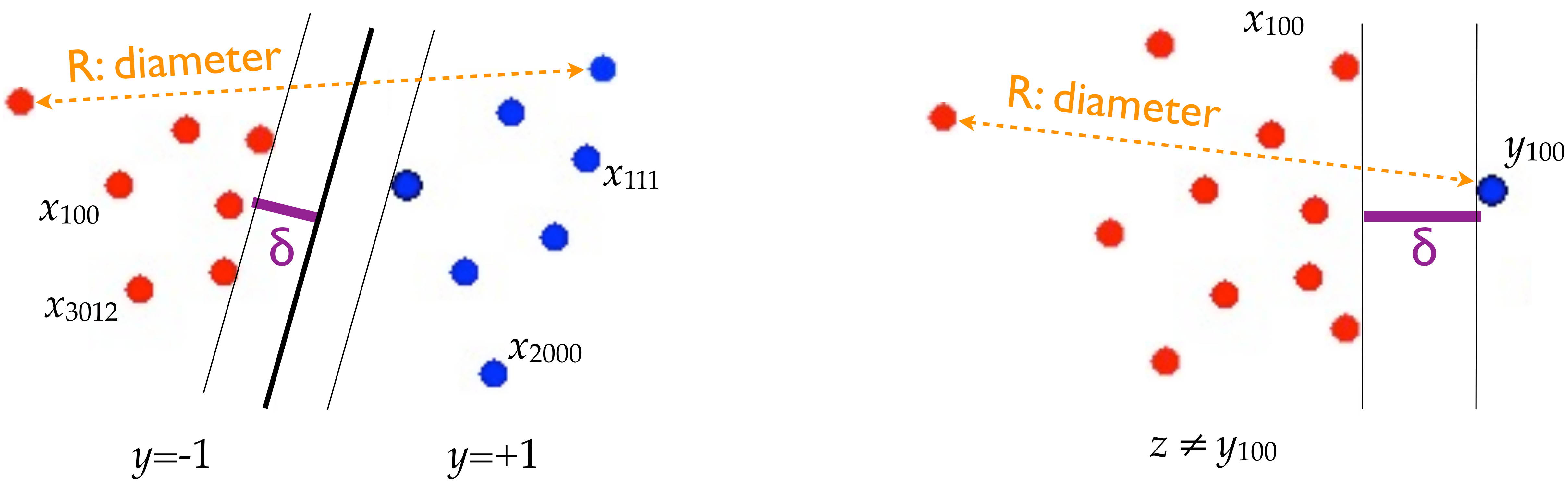
- binary classification: converges iff. data is separable
- structured prediction: converges iff. data is separable
  - there is an oracle vector that correctly labels all examples
  - one vs the rest (correct label better than all incorrect labels)
- theorem: if separable, then **# of updates ≤ R^2 / δ^2** R: diameter



Novikoff => Freund & Schapire => Collins
1962 1999 2002

Perceptron Convergence Proof

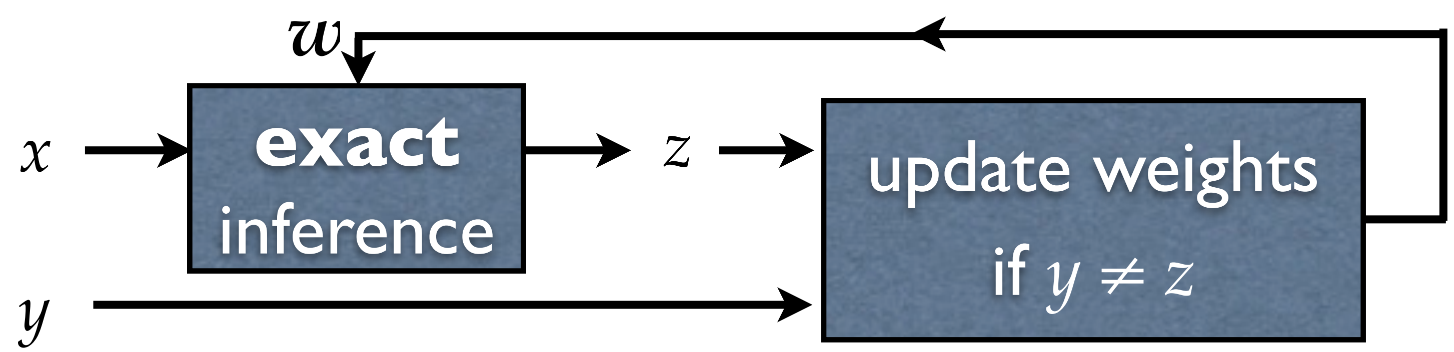
- binary classification: converges iff. data is separable
- structured prediction: converges iff. data is separable
  - there is an oracle vector that correctly labels all examples
  - one vs the rest (correct label better than all incorrect labels)
- theorem: if separable, then **# of updates ≤ R^2 / δ^2**   R: diameter

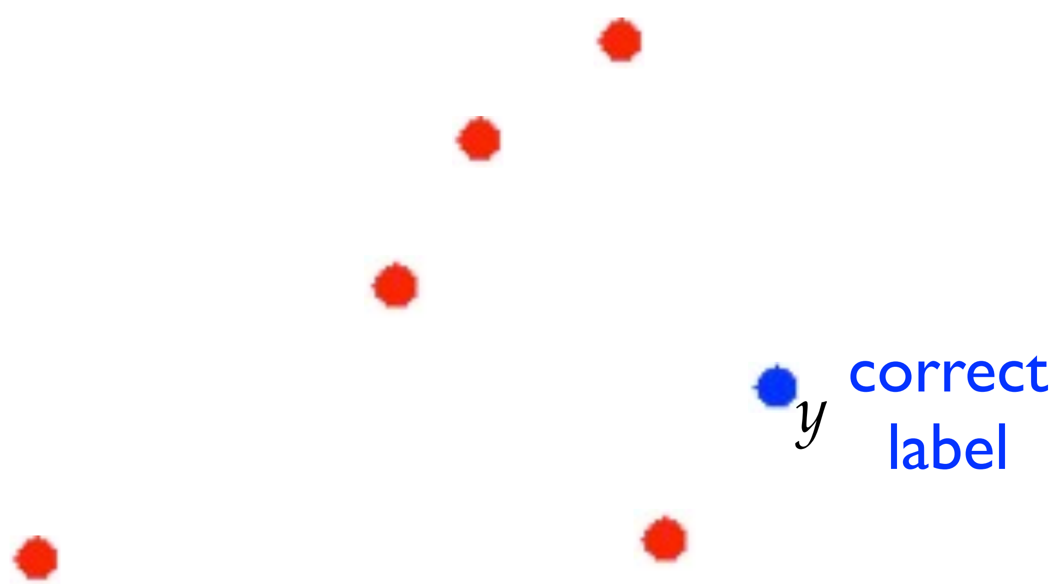


Novikoff => Freund & Schapire => Collins
1962 1999 2002

Geometry of Convergence Proof pt 1

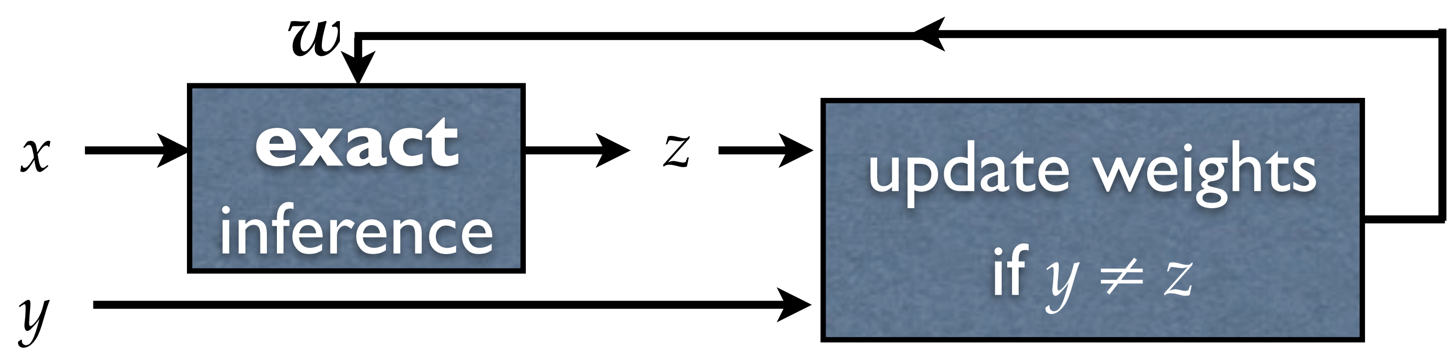
repeat
  for each example (x, y) **in** D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\Phi(x, y, z)$
until converged

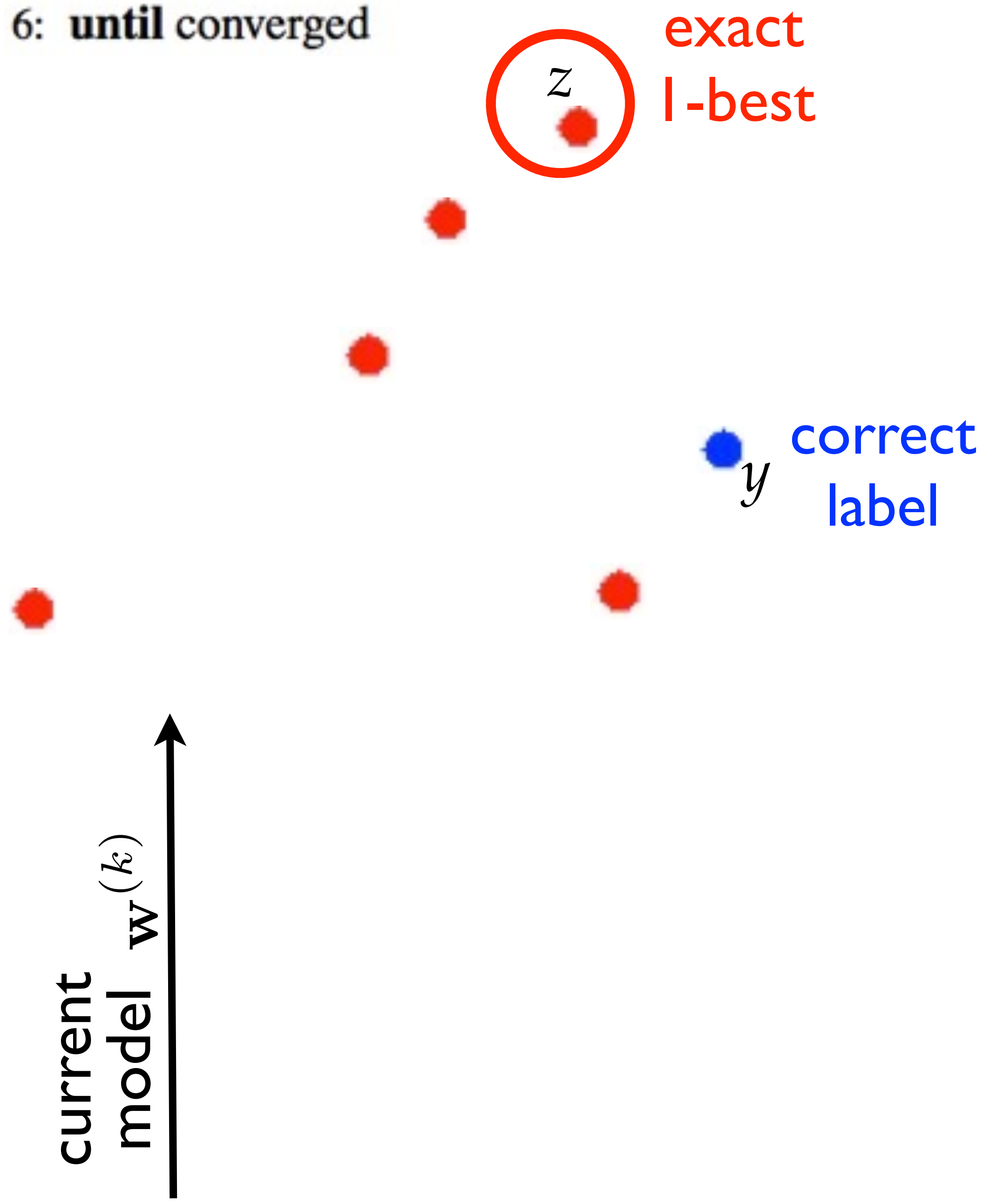




Geometry of Convergence Proof pt 1

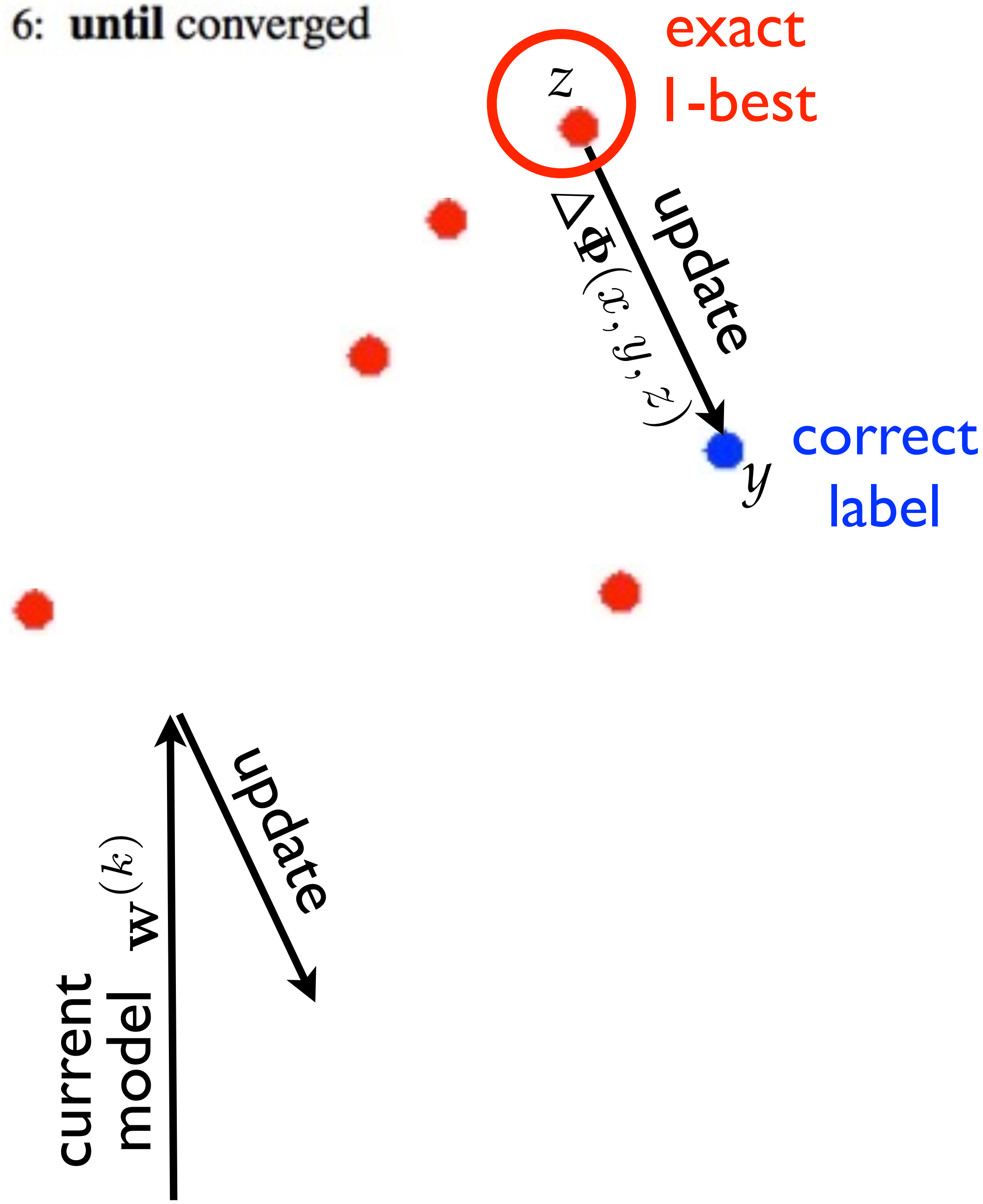
repeat
  for each example (x, y) in D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged

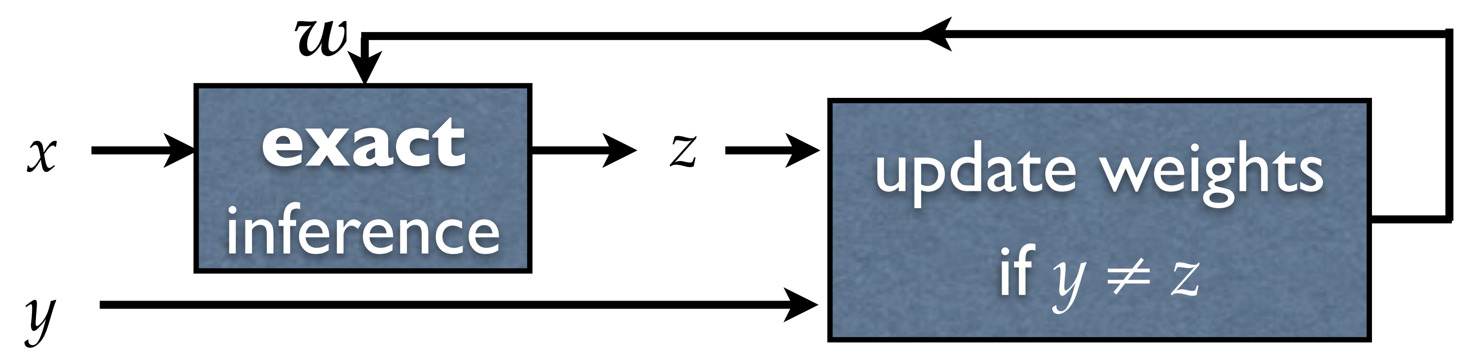




Geometry of Convergence Proof pt 1

```
repeat
    for each example (x, y) in D do
        z ← EXACT(x, w)
        if z ≠ y then
            w ← w + ΔΦ(x, y, z)
until converged
```



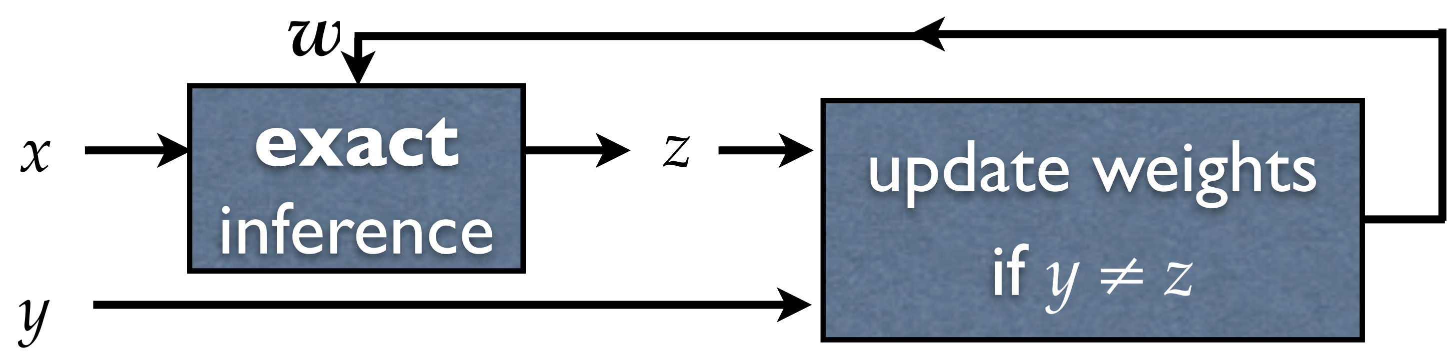


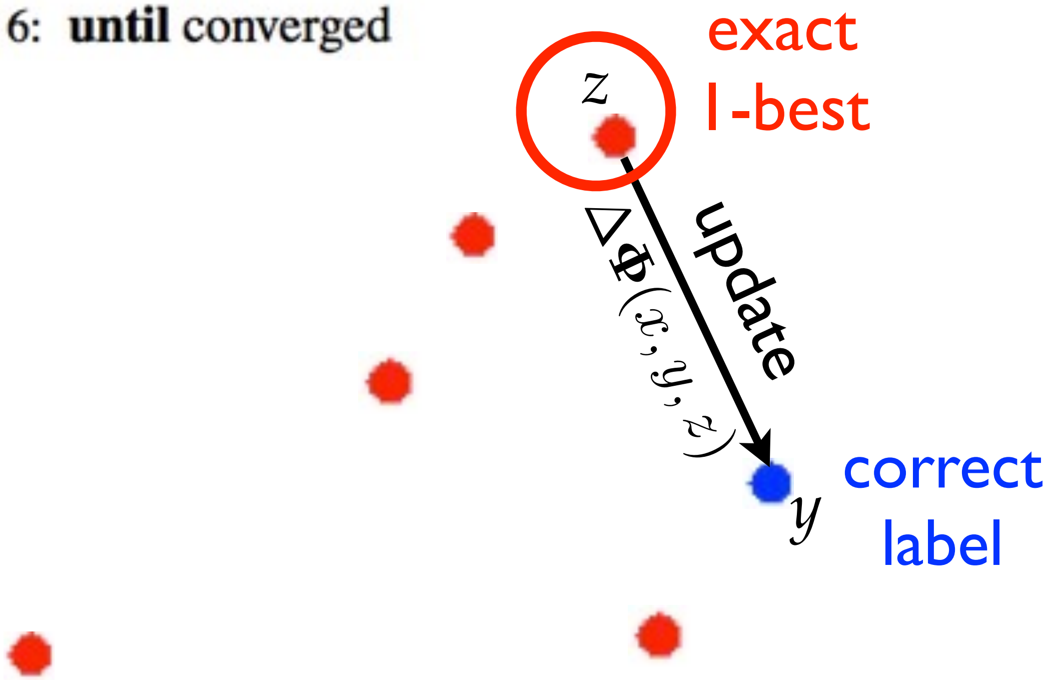
perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\boldsymbol{\Phi}(x, y, z)$$

Geometry of Convergence Proof pt 1

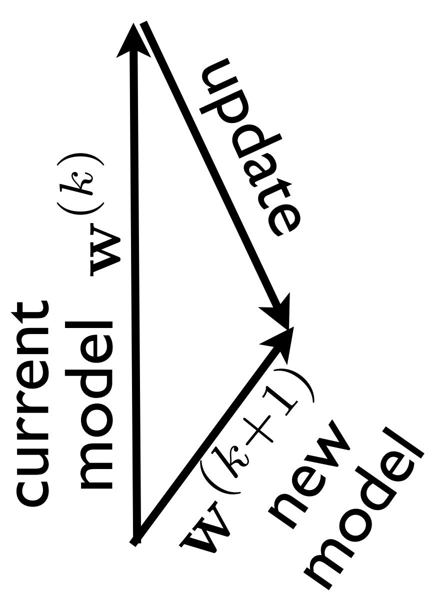
repeat
  for each example (x, y) in D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged





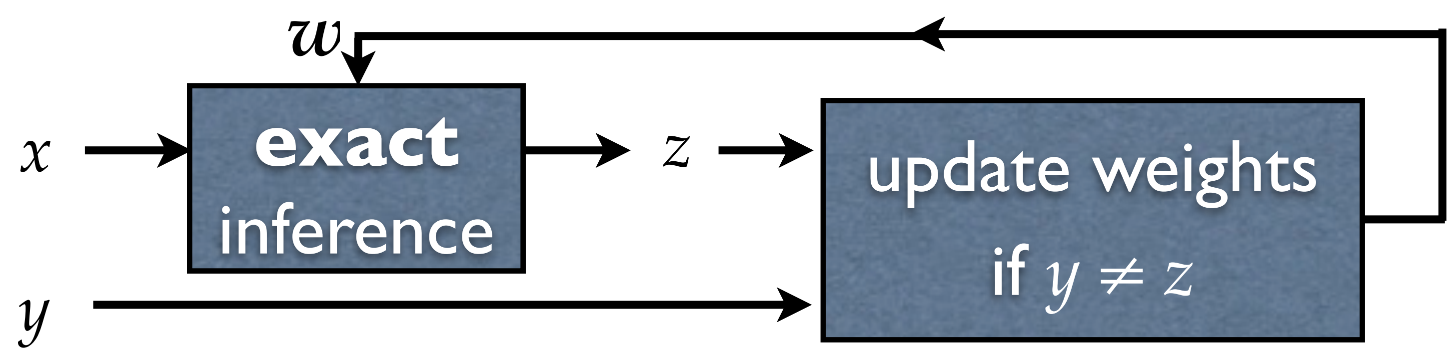
perceptron update:

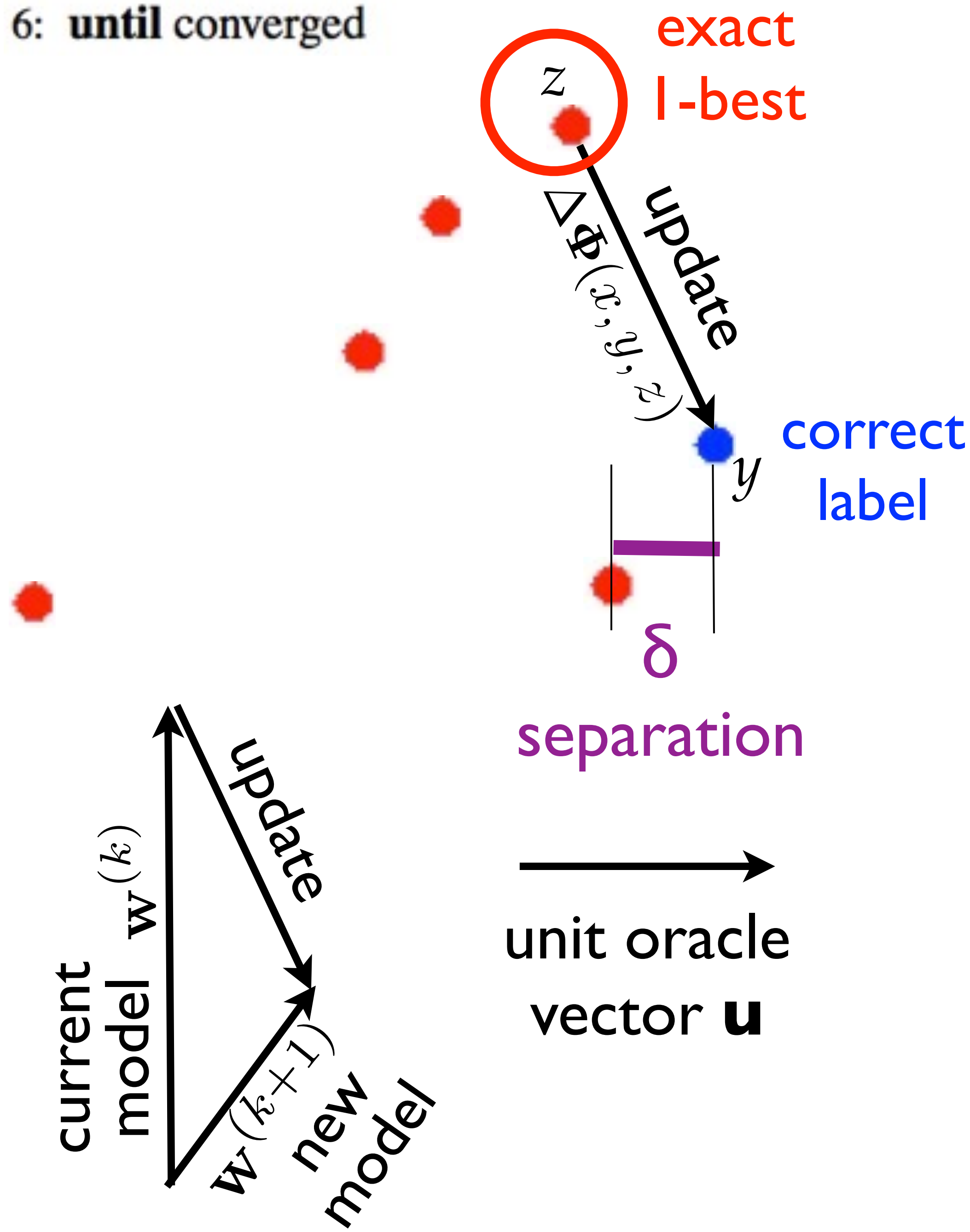
$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$



Geometry of Convergence Proof pt 1

```
repeat
  for each example (x, y) in D do
    z ← EXACT(x, w)
    if z ≠ y then
      w ← w + ΔΦ(x, y, z)
until converged
```





perceptron update:

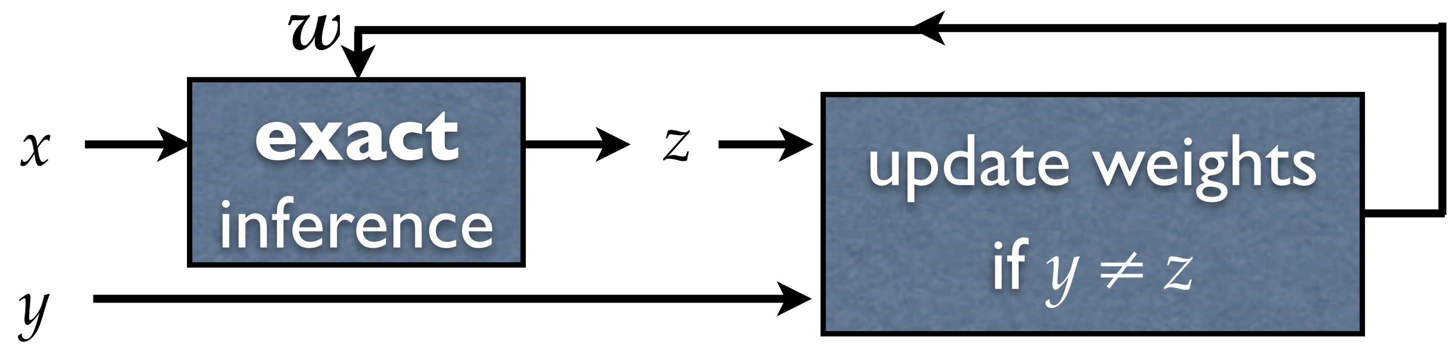
$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

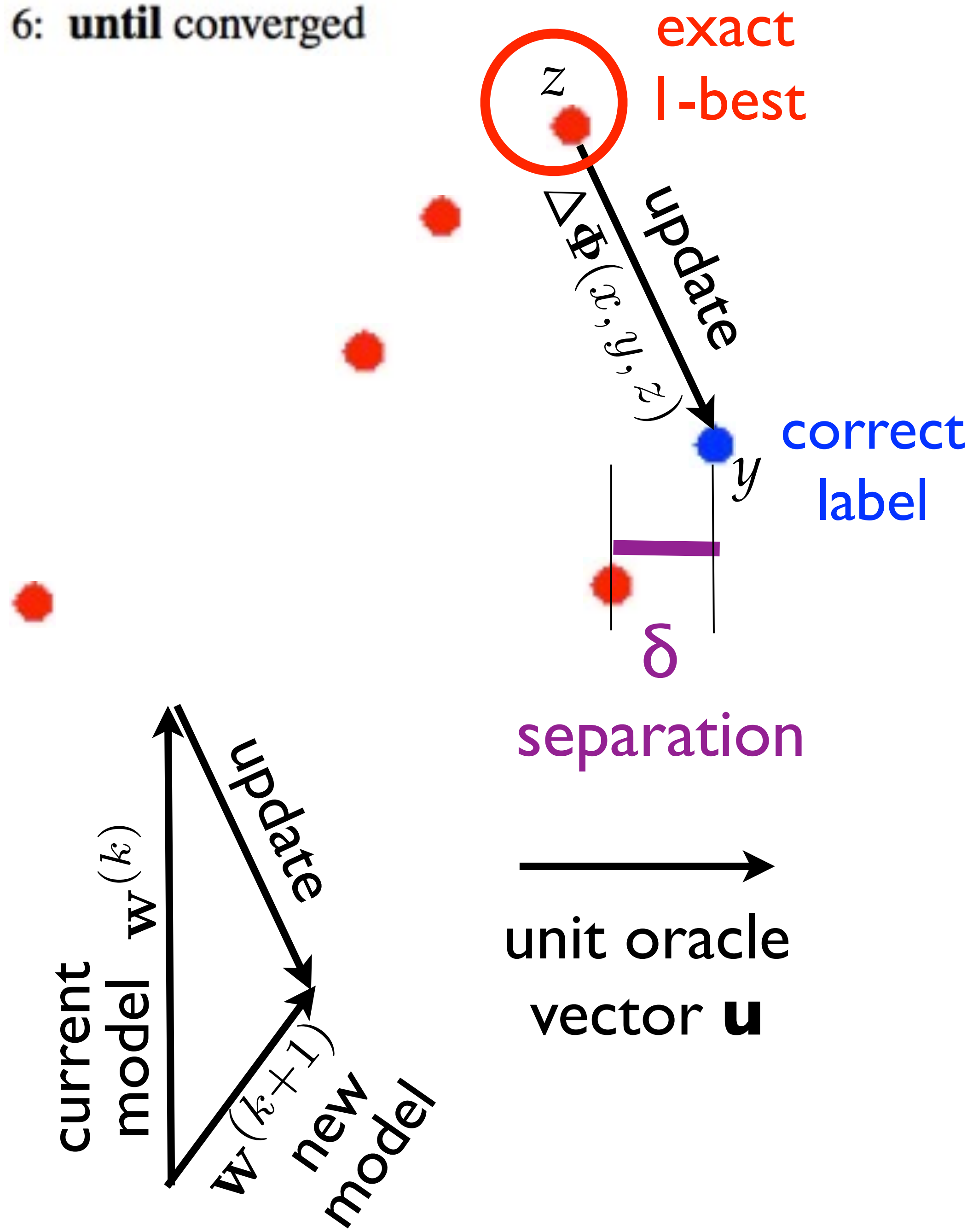
$$\mathbf{u} \cdot \mathbf{w}^{(k+1)} = \mathbf{u} \cdot \mathbf{w}^{(k)} + \boxed{\mathbf{u} \cdot \Delta\mathbf{\Phi}(x, y, z) \geq \delta}$$

margin

Geometry of Convergence Proof pt 1

```
repeat
   for each example (x, y) in D do
      z ← EXACT(x, w)
      if z ≠ y then
         w ← w + ΔΦ(x, y, z)
until converged
```





perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\Phi(x, y, z)$$

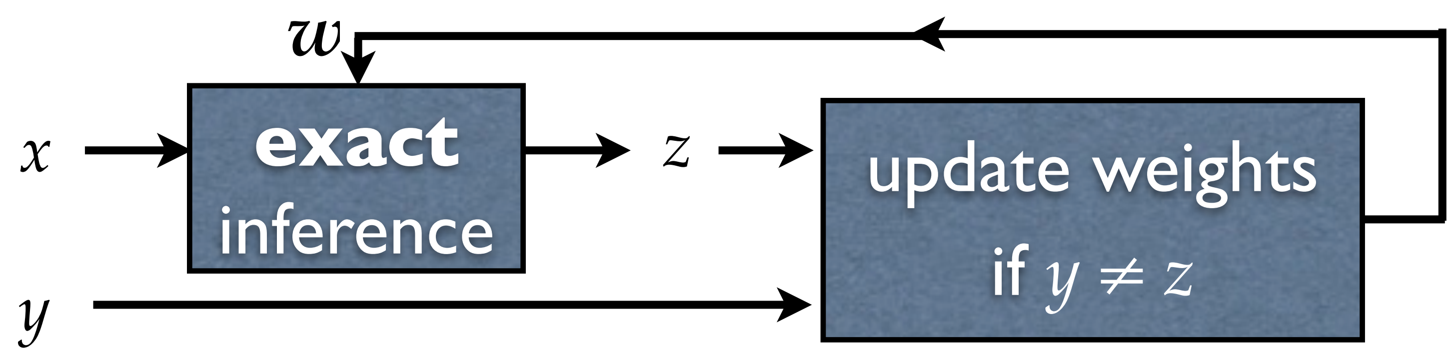
$$\mathbf{u} \cdot \mathbf{w}^{(k+1)} = \mathbf{u} \cdot \mathbf{w}^{(k)} + \boxed{\mathbf{u} \cdot \Delta\Phi(x, y, z) \geq \delta \quad \text{margin}}$$

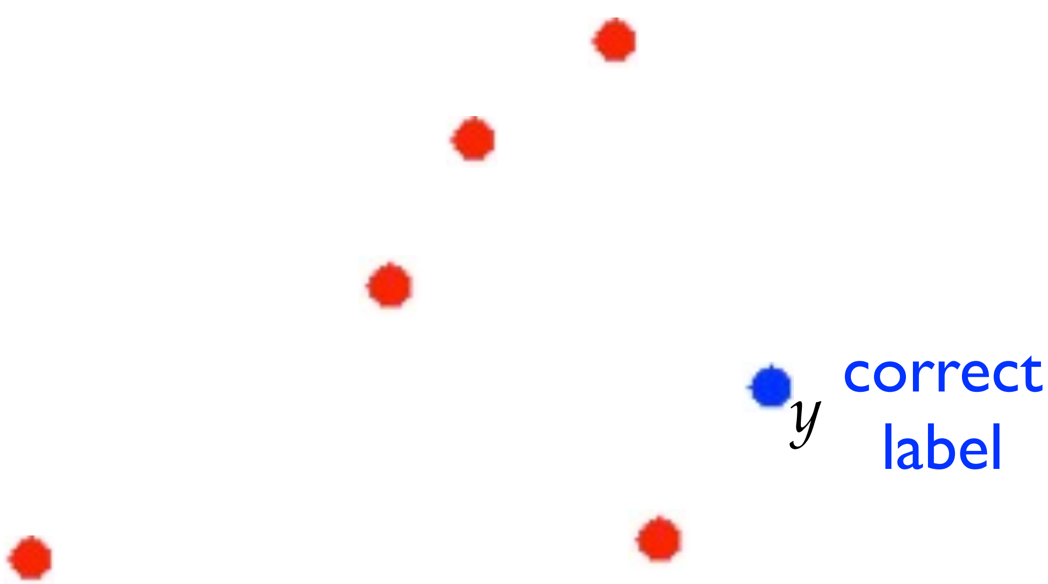
$$\mathbf{u} \cdot \mathbf{w}^{(k+1)} \geq k\delta \qquad \text{(by induction)}$$

$$\|\mathbf{w}^{k+1}\| \geq k\delta \qquad \text{(part 1: lowerbound)}$$

Geometry of Convergence Proof pt 2

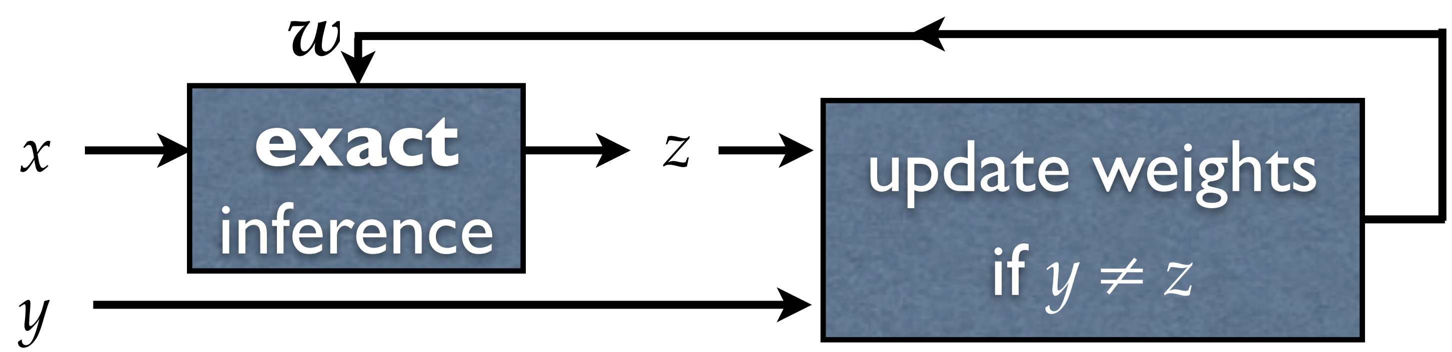
repeat
  for each example (x, y) **in** D **do**
    $z \leftarrow \text{EXACT}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\Phi(x, y, z)$
until converged

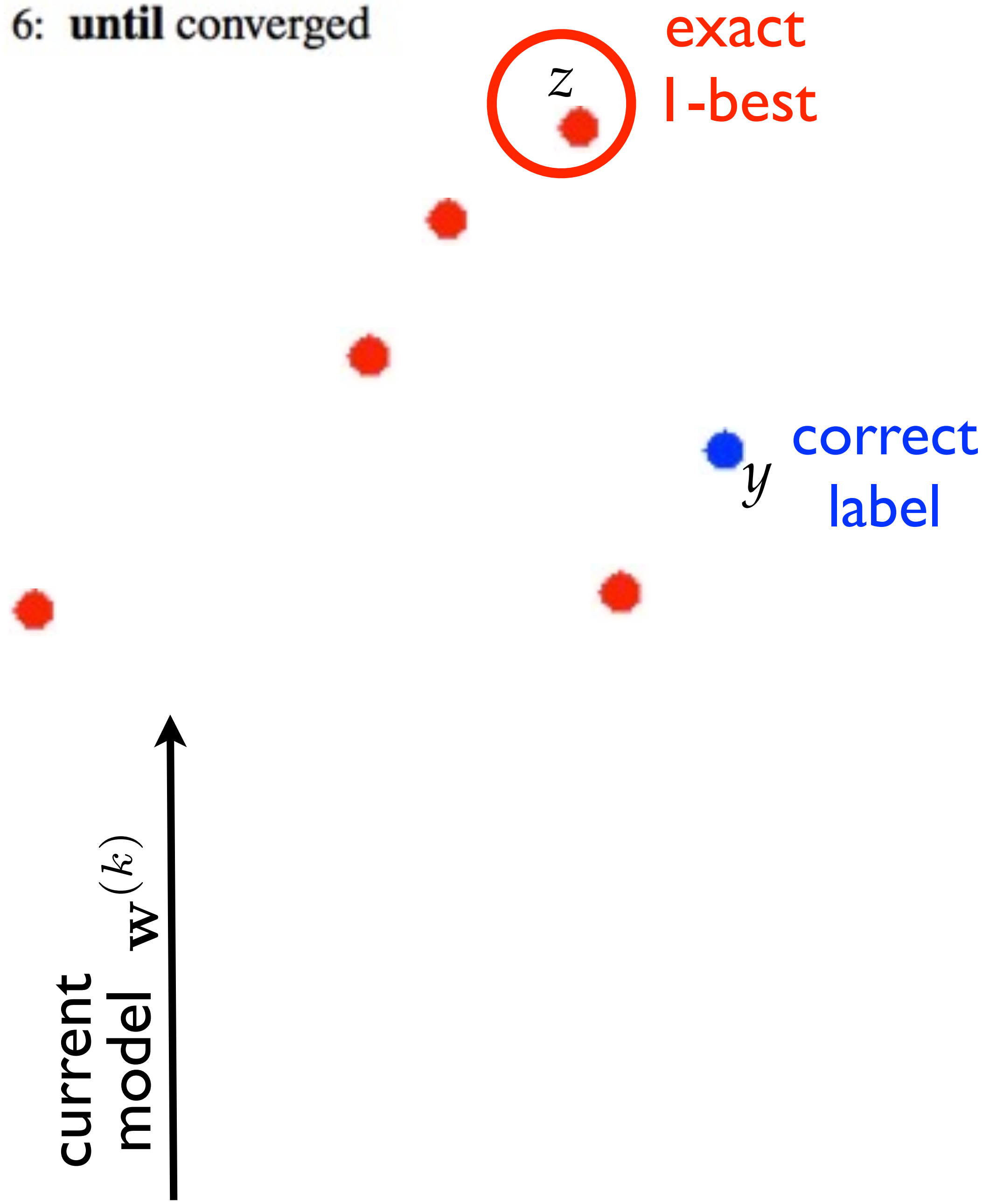




Geometry of Convergence Proof pt 2

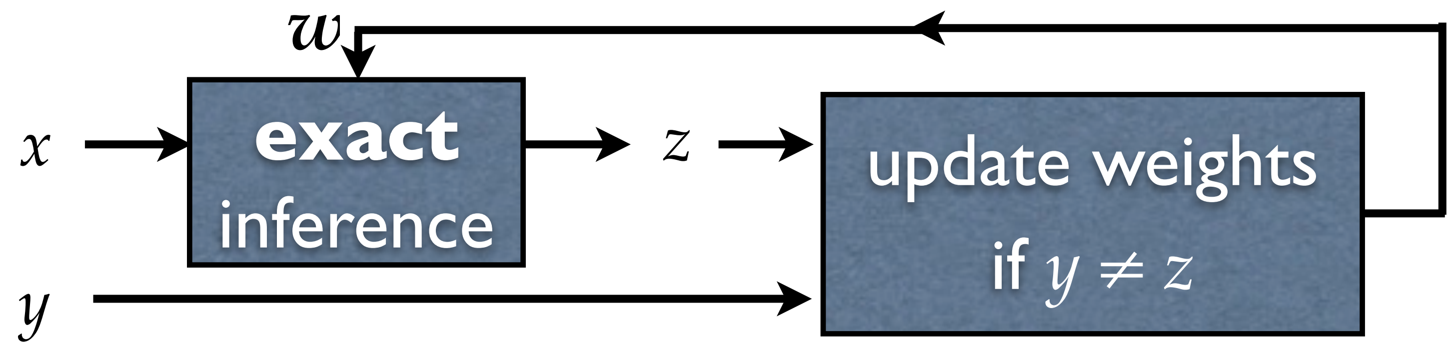
repeat
  for each example (x, y) in D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged

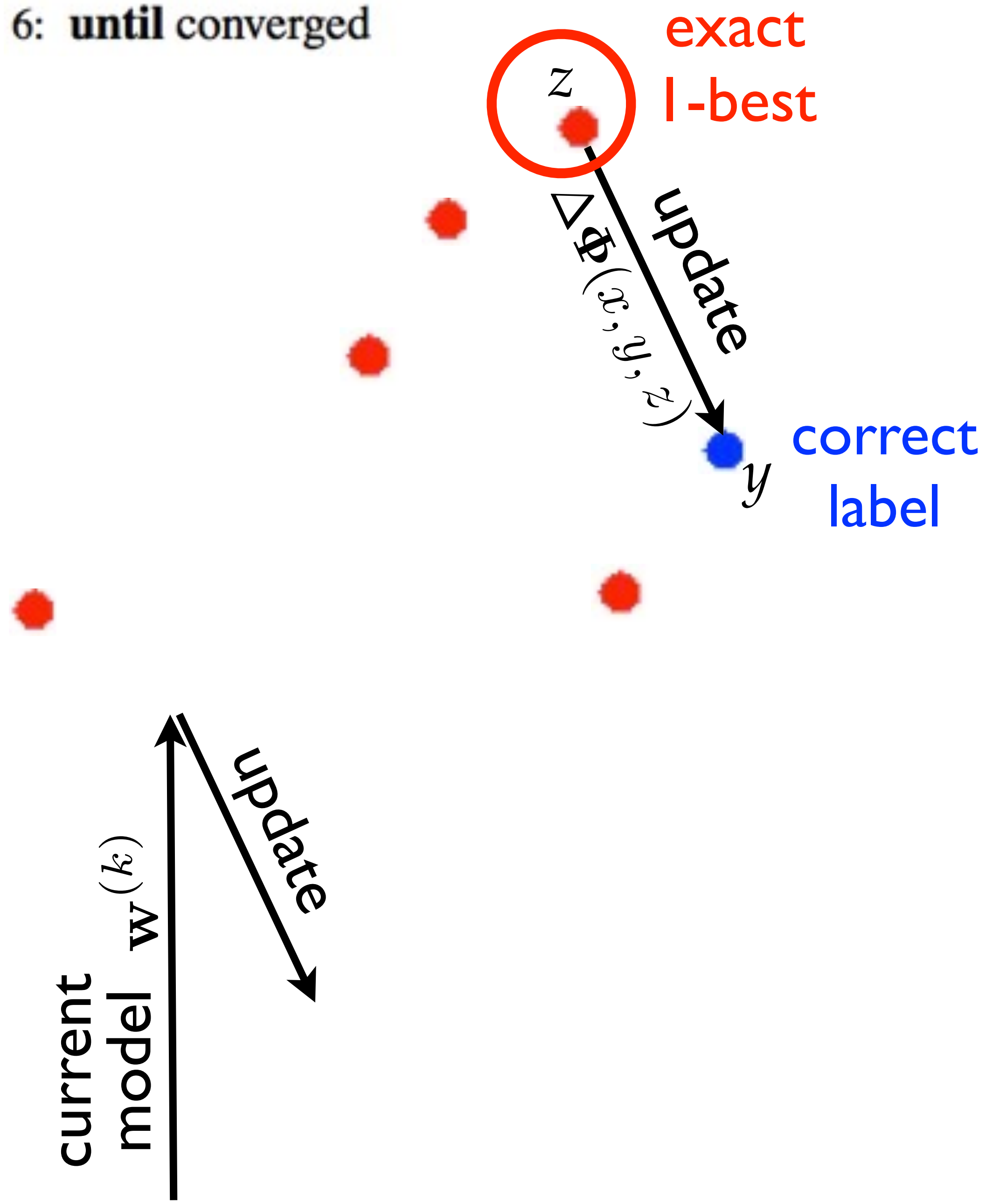




Geometry of Convergence Proof pt 2

repeat
  for each example (x, y) **in** D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged



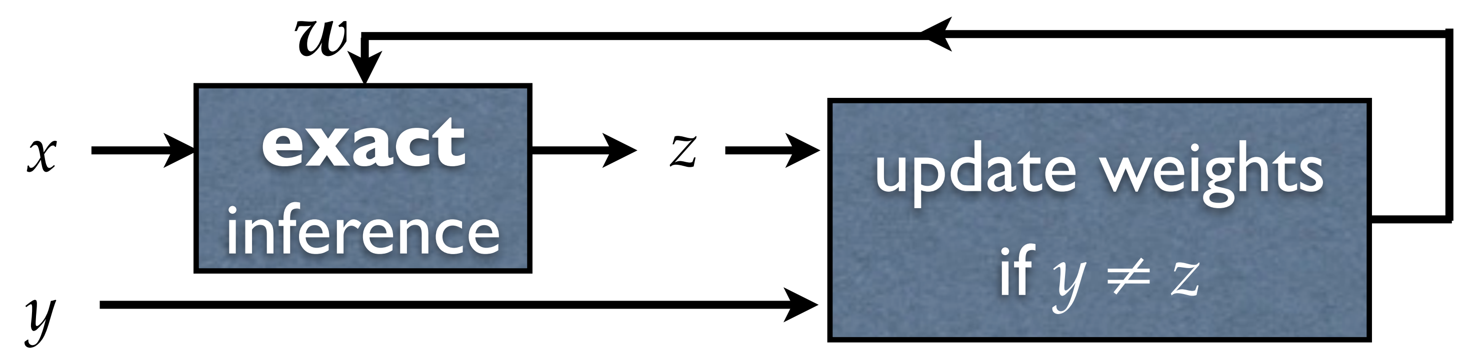


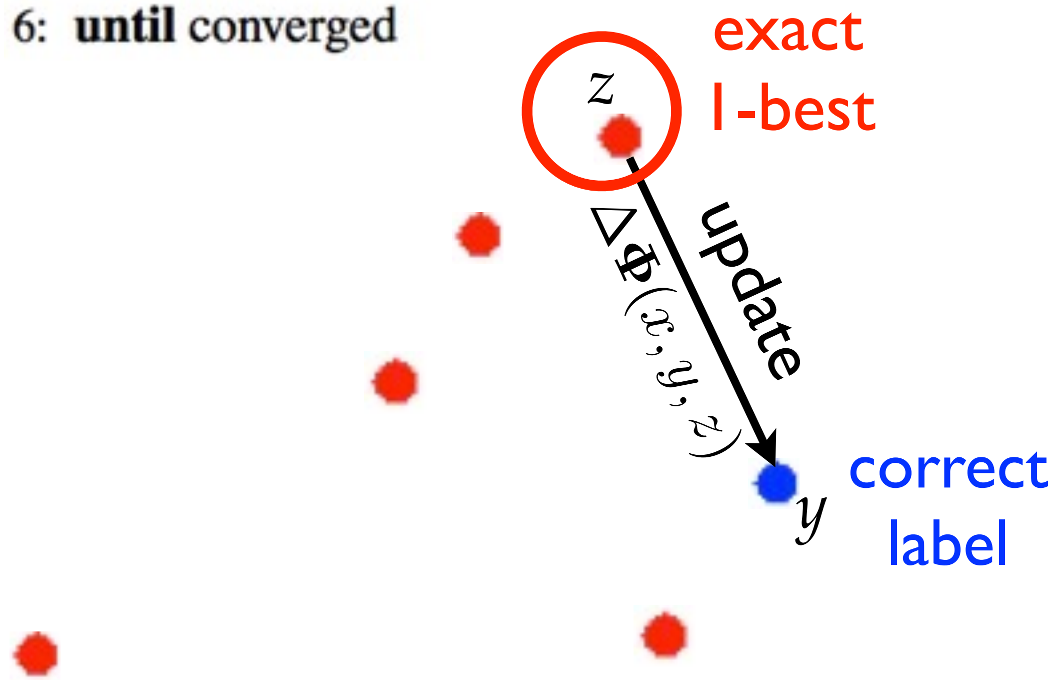
perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

Geometry of Convergence Proof pt 2

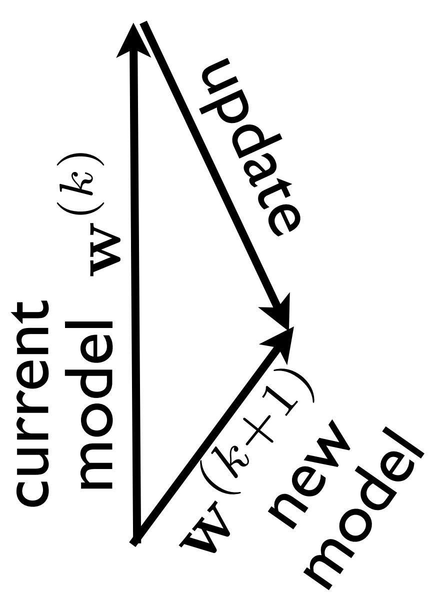
repeat
  for each example (x, y) in D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged





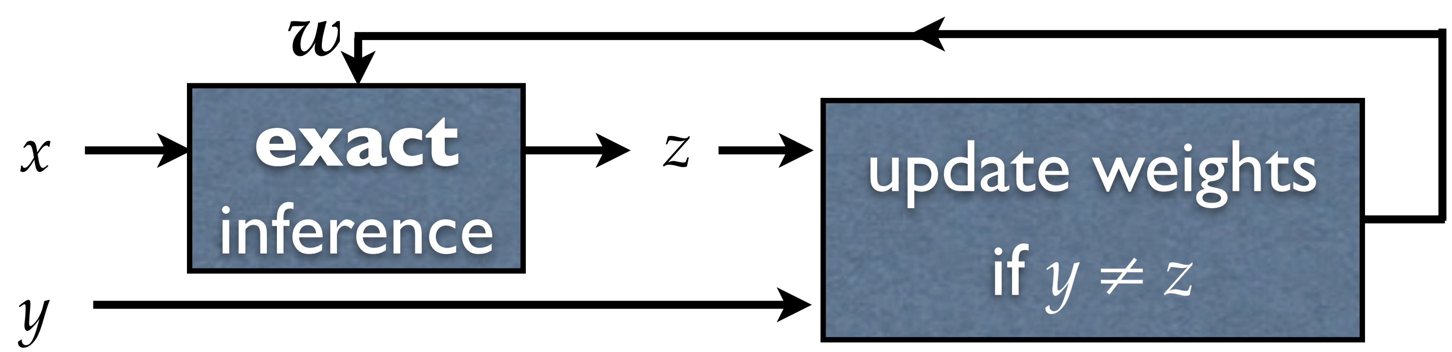
perceptron update:

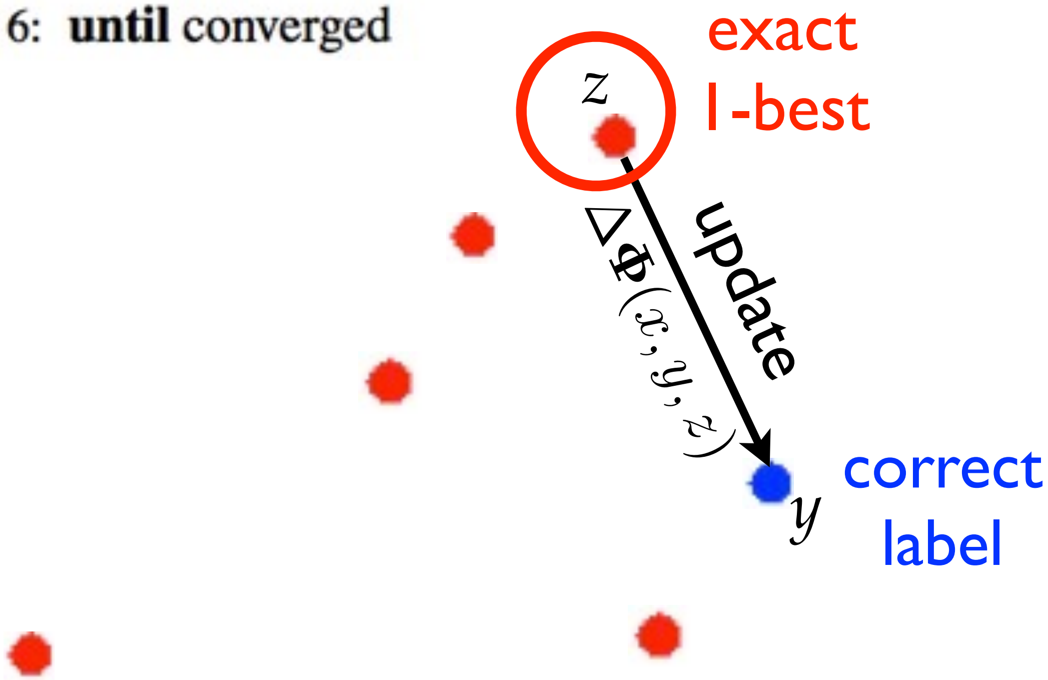
$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$



Geometry of Convergence Proof pt 2

repeat
  for each example (x, y) in D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged

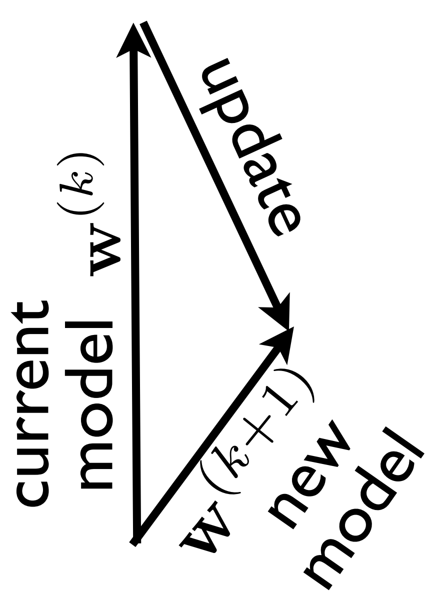




perceptron update:

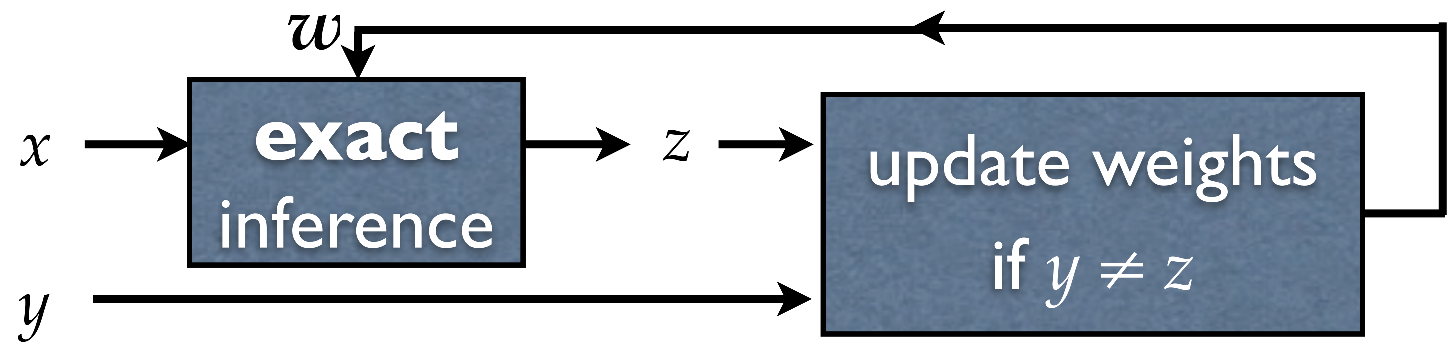
$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

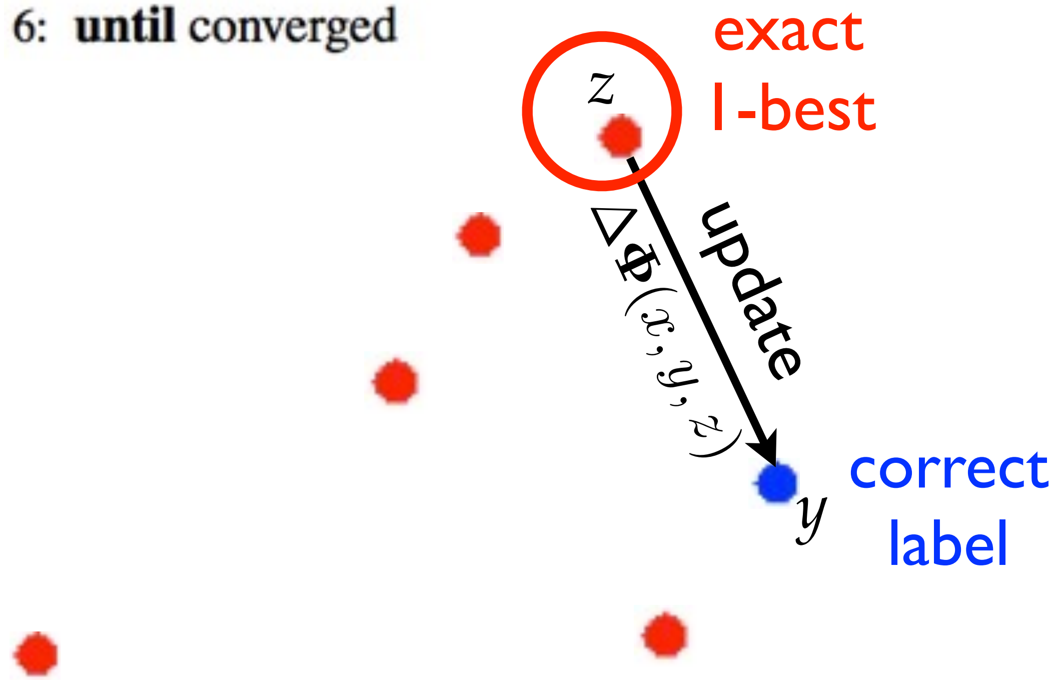
$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$



Geometry of Convergence Proof pt 2

> **repeat**
>   **for each** example (x, y) **in** D **do**
> $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
>   **if** $z \neq y$ **then**
> $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
> **until** converged



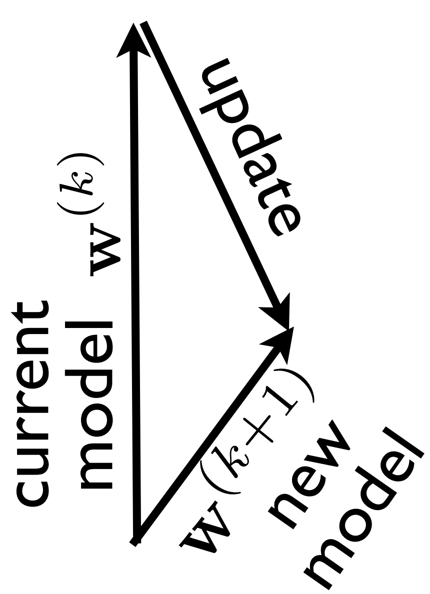


perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

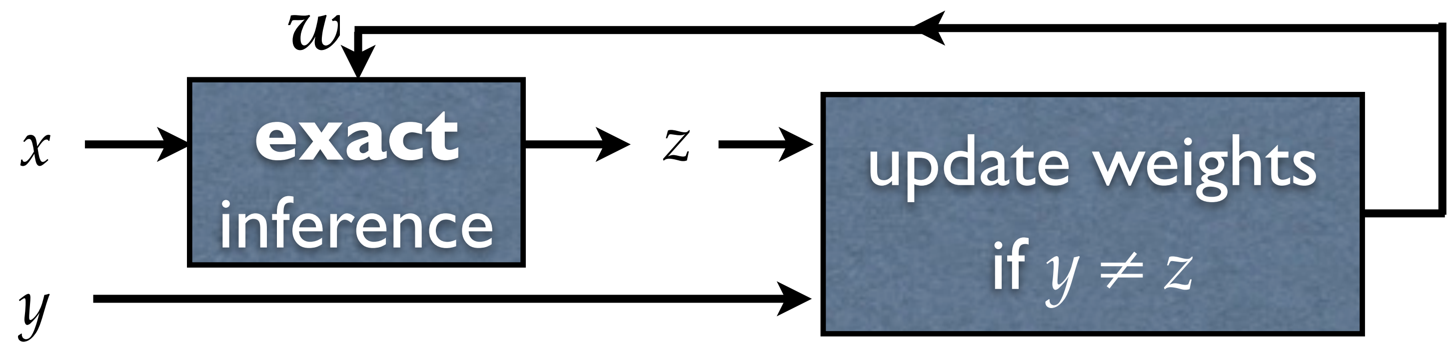
$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$

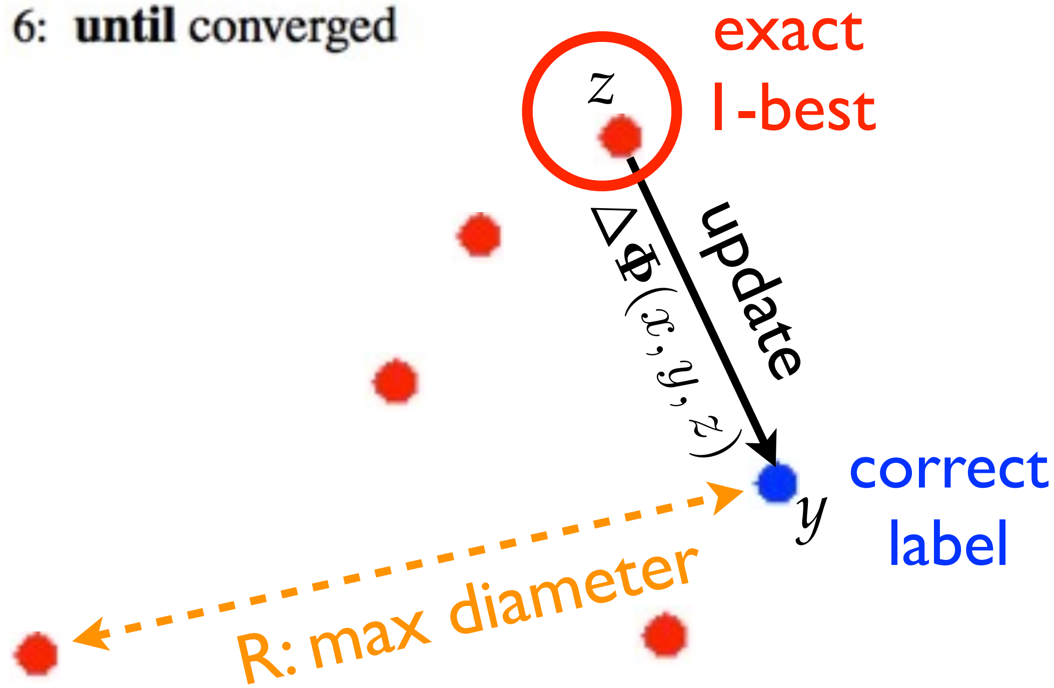
$$= \|\mathbf{w}^{(k)}\|^2 + \|\Delta\mathbf{\Phi}(x, y, z)\|^2$$

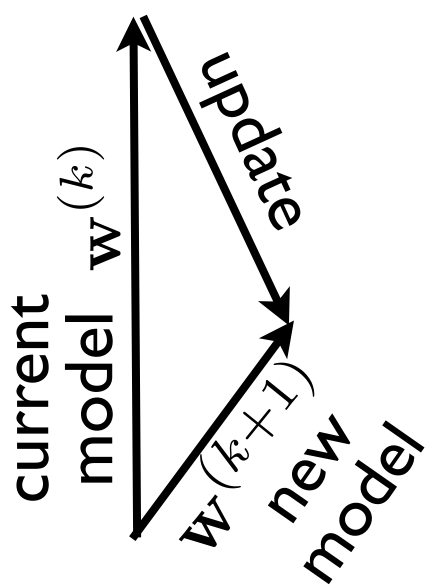


Geometry of Convergence Proof pt 2

```
repeat
    for each example (x, y) in D do
        z ← EXACT(x, w)
        if z ≠ y then
            w ← w + ΔΦ(x, y, z)
until converged
```







perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\Phi(x, y, z)$$

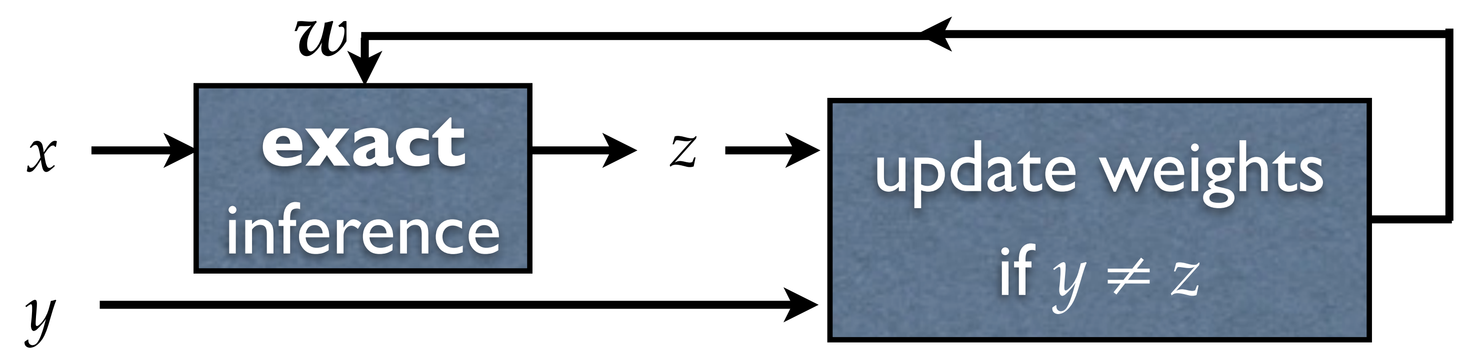
$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\Phi(x, y, z)\|^2$$

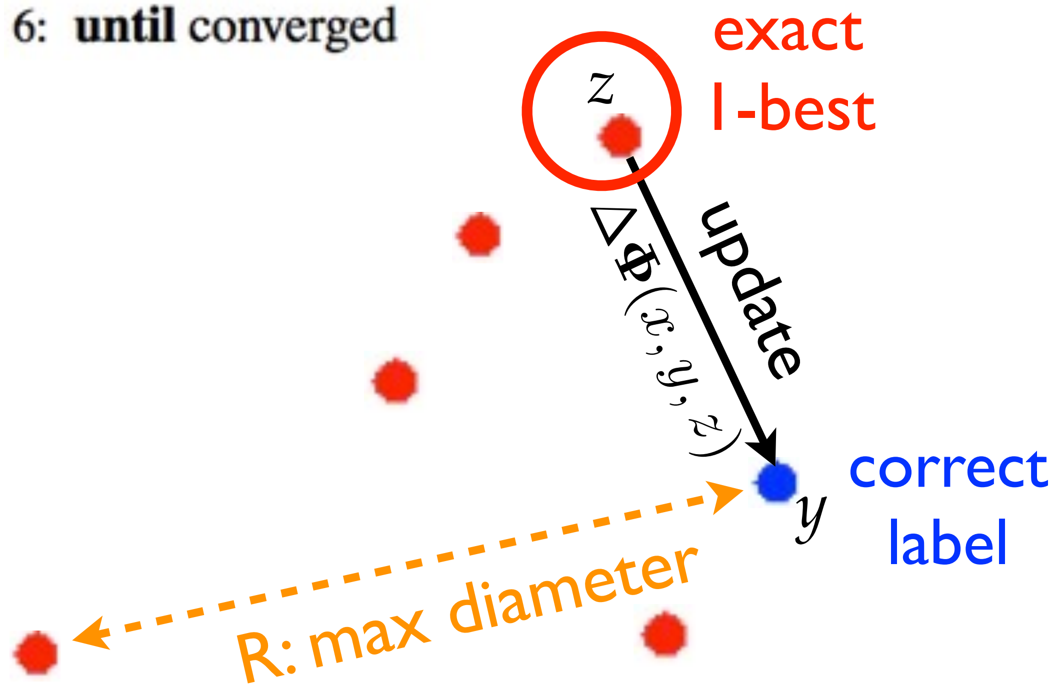
$$= \|\mathbf{w}^{(k)}\|^2 + \boxed{\underbrace{\|\Delta\Phi(x, y, z)\|^2}_{\leq R^2}}$$

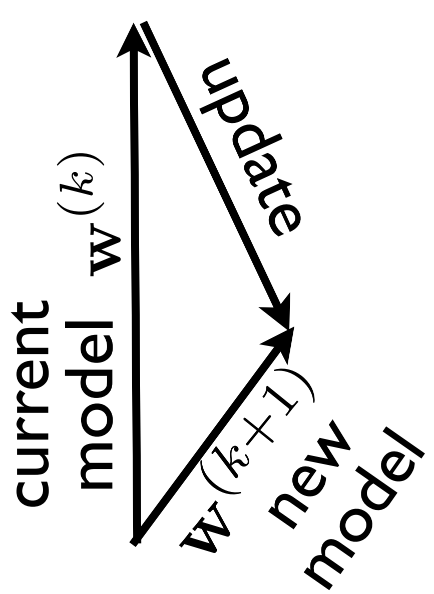
diameter

Geometry of Convergence Proof pt 2

repeat
  for each example (x, y) **in** D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged







perceptron update:

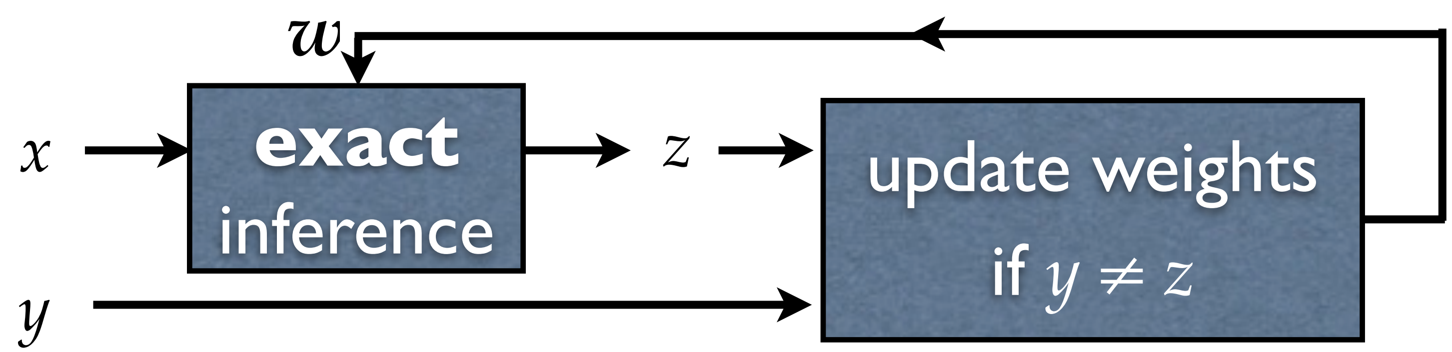
$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

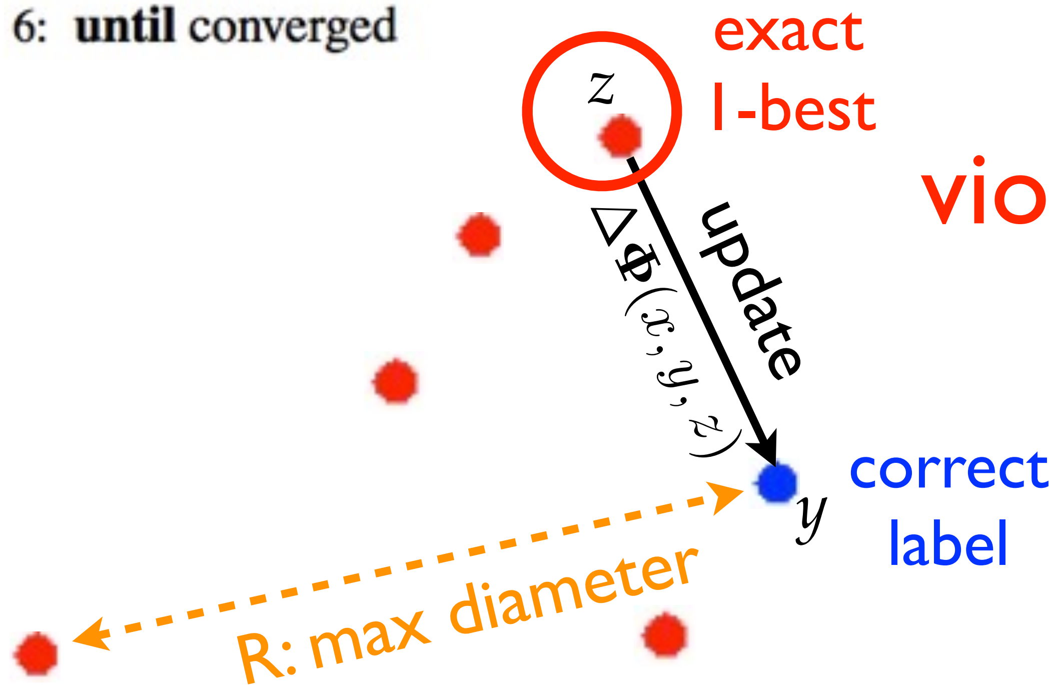
$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$

$$= \|\mathbf{w}^{(k)}\|^2 + \boxed{\underbrace{\|\Delta\mathbf{\Phi}(x, y, z)\|^2}_{\substack{\leq R^2 \\ \text{diameter}}}} + 2\,\mathbf{w}^{(k)} \cdot \Delta\mathbf{\Phi}(x, y, z)$$

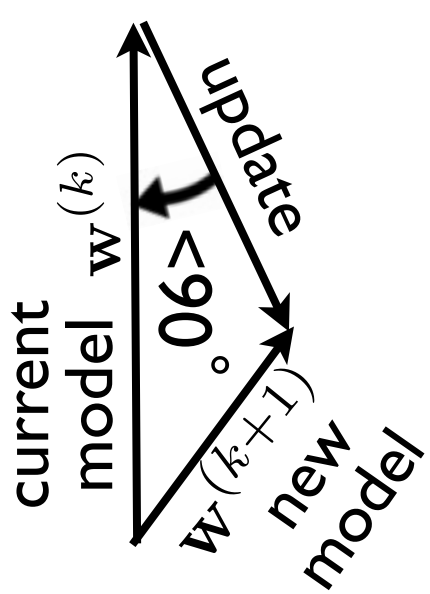
Geometry of Convergence Proof pt 2

repeat
  for each example (x, y) **in** D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged





violation: incorrect label scored higher



perceptron update:

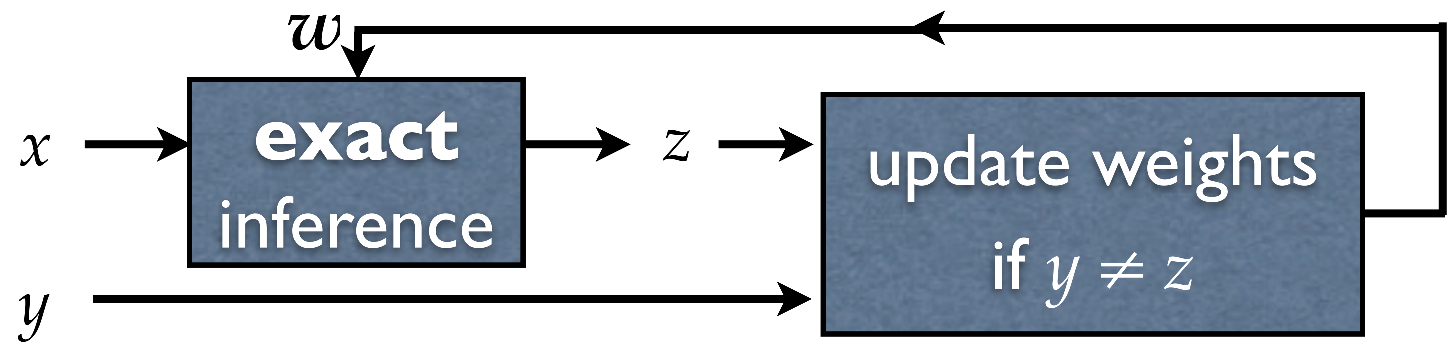
$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

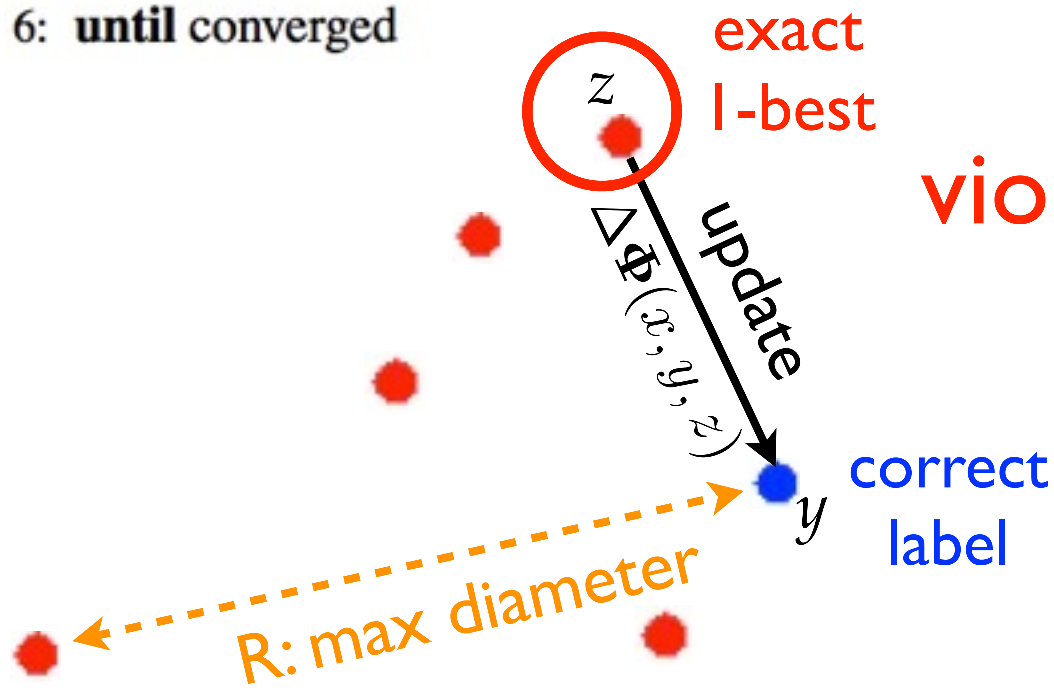
$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$

$$= \|\mathbf{w}^{(k)}\|^2 + \underbrace{\|\Delta\mathbf{\Phi}(x, y, z)\|^2}_{\leq R^2 \text{ diameter}} + \underbrace{2\, \mathbf{w}^{(k)} \cdot \Delta\mathbf{\Phi}(x, y, z)}_{\leq 0 \text{ violation}}$$

Geometry of Convergence Proof pt 2

repeat
  for each example (x, y) **in** D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged





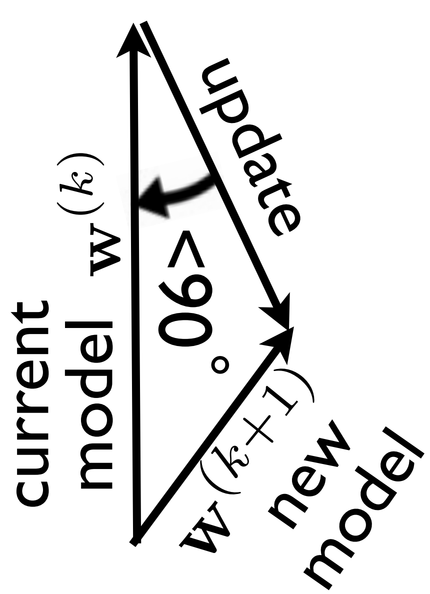
violation: incorrect label scored higher

perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$

$$= \|\mathbf{w}^{(k)}\|^2 + \underbrace{\|\Delta\mathbf{\Phi}(x, y, z)\|^2}_{\substack{\leq R^2 \\ \text{diameter}}} + \underbrace{2\,\mathbf{w}^{(k)} \cdot \Delta\mathbf{\Phi}(x, y, z)}_{\substack{\leq 0 \\ \text{violation}}}$$

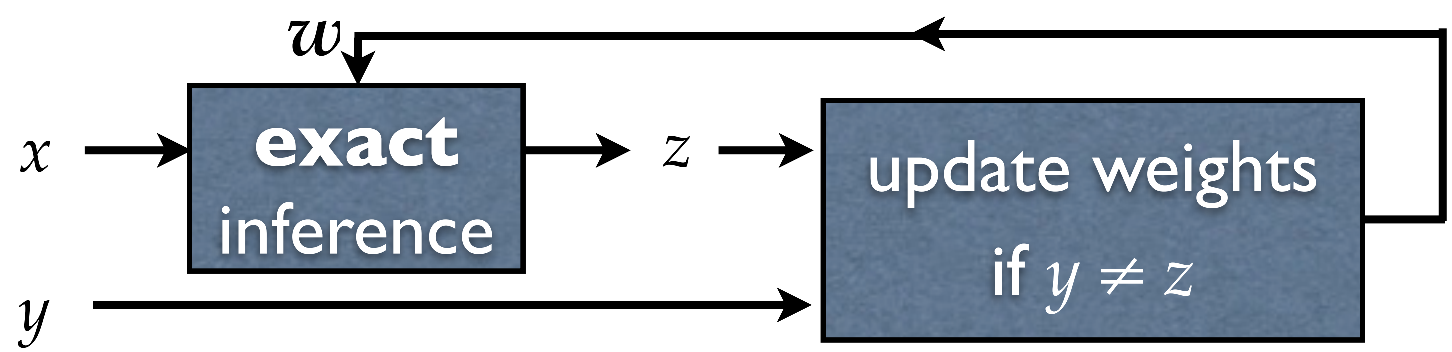


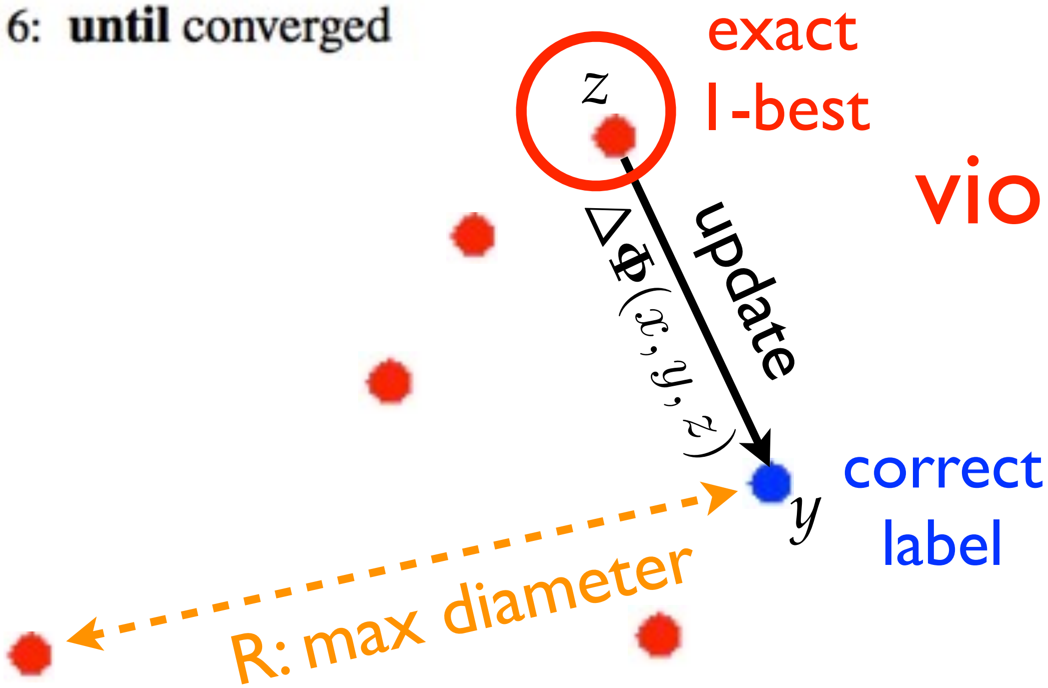
by induction: $\|\mathbf{w}^{k+1}\|^2 \leq kR^2$
$\|\mathbf{w}^{k+1}\| \leq \sqrt{k}R$

(part 2: upperbound)

Geometry of Convergence Proof pt 2

1: **repeat**
2: **for each** example (x, y) **in** D **do**
3: $z \leftarrow \text{EXACT}(x, \mathbf{w})$
4: **if** $z \neq y$ **then**
5: $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
6: **until** converged





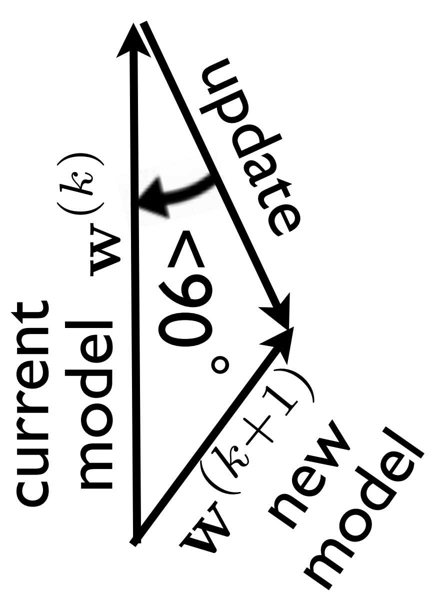
violation: incorrect label scored higher

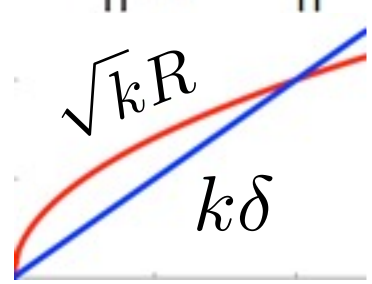
perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$

$$= \|\mathbf{w}^{(k)}\|^2 + \underbrace{\|\Delta\mathbf{\Phi}(x, y, z)\|^2}_{\leq R^2 \text{ diameter}} + 2\, \underbrace{\mathbf{w}^{(k)} \cdot \Delta\mathbf{\Phi}(x, y, z)}_{\leq 0 \text{ violation}}$$





by induction: $\|\mathbf{w}^{k+1}\|^2 \leq kR^2$ (part 2: upperbound)

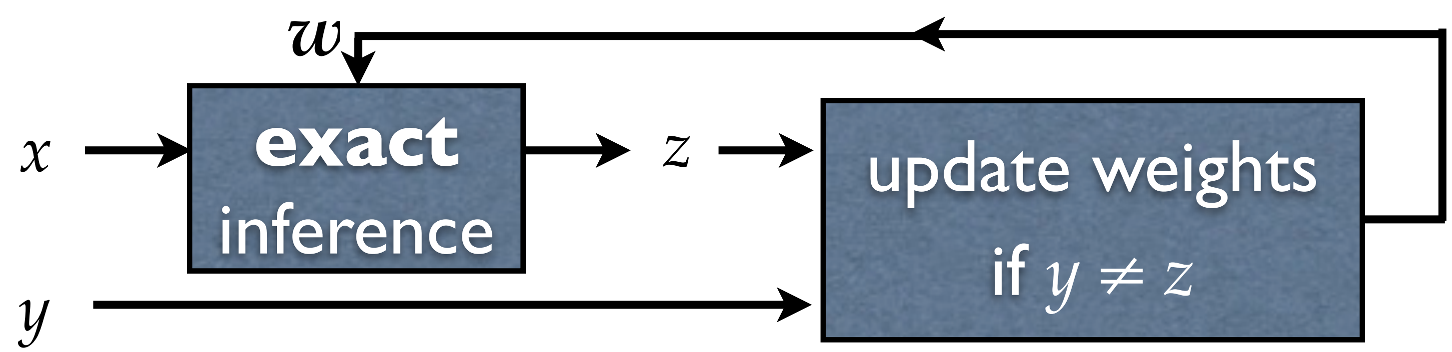
$\|\mathbf{w}^{k+1}\| \leq \sqrt{k}R$

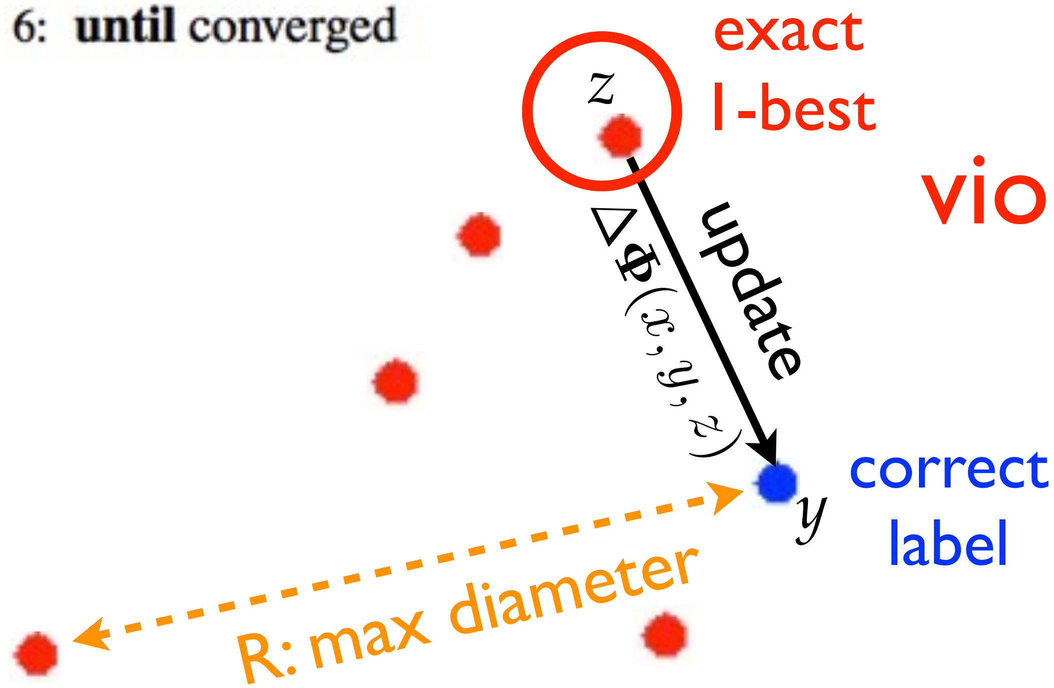
combine with: $\|\mathbf{w}^{k+1}\| \geq k\delta$ (part 1: lowerbound)

bound on # of updates: $\boxed{k \leq R^2/\delta^2}$

Geometry of Convergence Proof pt 2

repeat
  for each example (x, y) **in** D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged





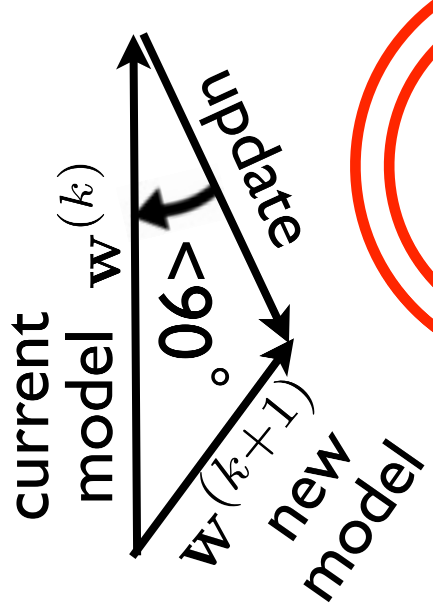
violation: incorrect label scored higher

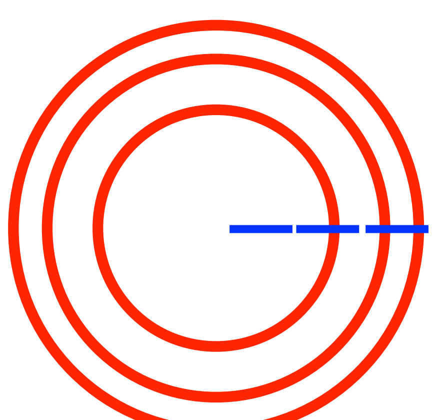
perceptron update:

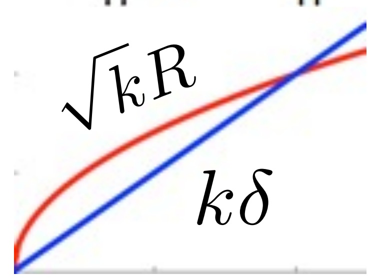
$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$

$$= \|\mathbf{w}^{(k)}\|^2 + \underbrace{\|\Delta\mathbf{\Phi}(x, y, z)\|^2}_{\leq R^2 \text{ diameter}} + 2\,\underbrace{\mathbf{w}^{(k)} \cdot \Delta\mathbf{\Phi}(x, y, z)}_{\leq 0 \text{ violation}}$$







by induction: $\|\mathbf{w}^{k+1}\|^2 \leq kR^2$ (part 2: upperbound)

$\|\mathbf{w}^{k+1}\| \leq \sqrt{k}R$

combine with: $\|\mathbf{w}^{k+1}\| \geq k\delta$ (part 1: lowerbound)

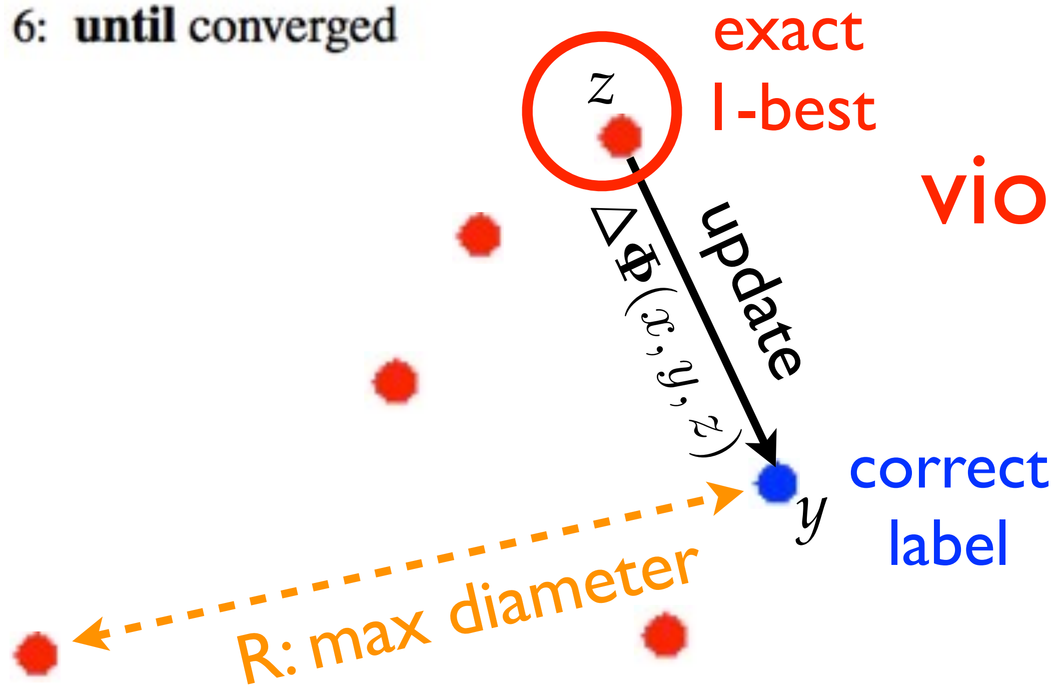
bound on # of updates: $\boxed{k \leq R^2/\delta^2}$

Geometry of Convergence Proof pt 2

repeat
  for each example (x, y) **in** D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged

summary: the proof uses 3 facts:

1. separation (margin)
2. diameter (always finite)
3. violation (guaranteed by exact search)



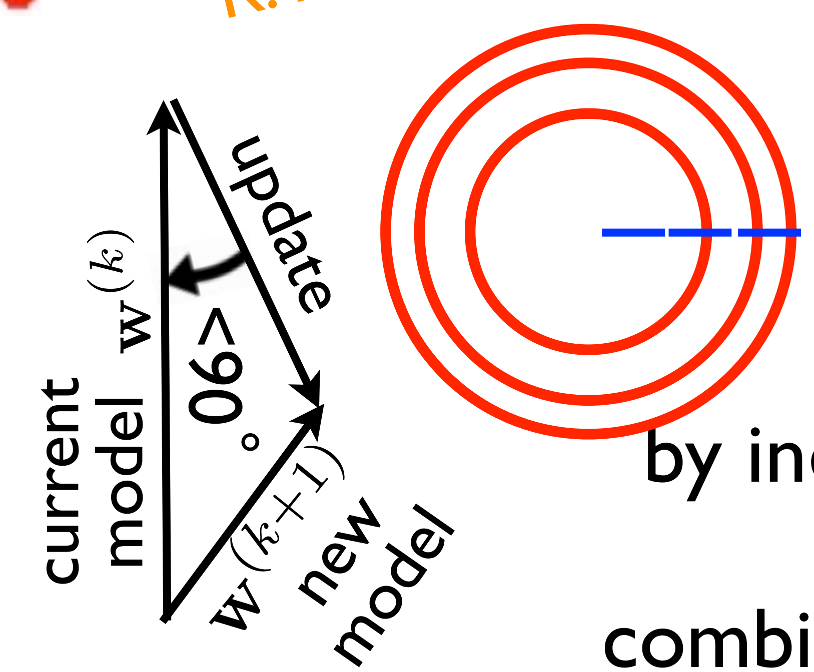
violation: incorrect label scored higher

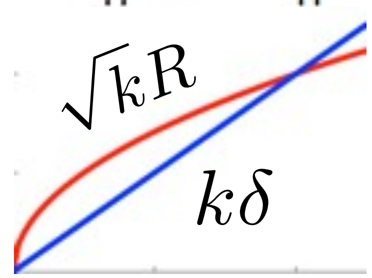
perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$

$$= \|\mathbf{w}^{(k)}\|^2 + \underbrace{\|\Delta\mathbf{\Phi}(x, y, z)\|^2}_{\leq R^2 \text{ diameter}} + 2\,\underbrace{\mathbf{w}^{(k)} \cdot \Delta\mathbf{\Phi}(x, y, z)}_{\leq 0 \text{ violation}}$$





by induction: $\|\mathbf{w}^{k+1}\|^2 \leq kR^2$ (part 2: upperbound)

$\|\mathbf{w}^{k+1}\| \leq \sqrt{k}R$

combine with: $\|\mathbf{w}^{k+1}\| \geq k\delta$ (part 1: lowerbound)

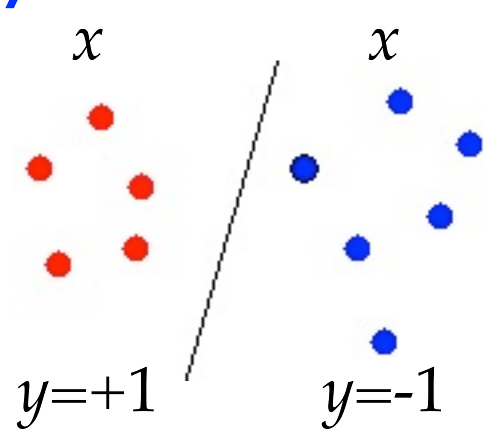
bound on # of updates: $\boxed{k \leq R^2/\delta^2}$

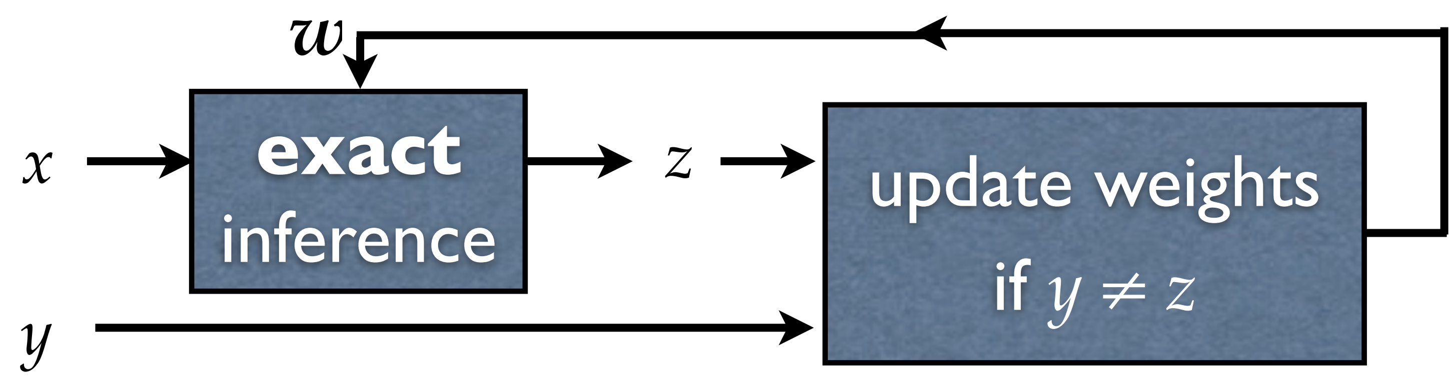
Tutorial Outline

- Overview of Structured Learning
  - Challenges in Scalability
- Structured Perceptron
  - convergence proof
- Structured Perceptron with Inexact Search
- Latent-Variable Perceptron
- Parallelizing Online Learning (Perceptron & MIRA)

Scalability Challenge 1: Inference

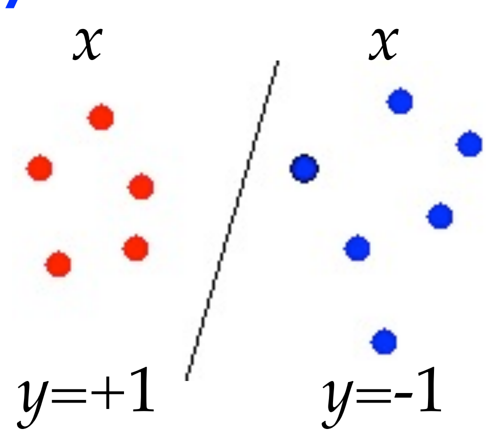
binary classification

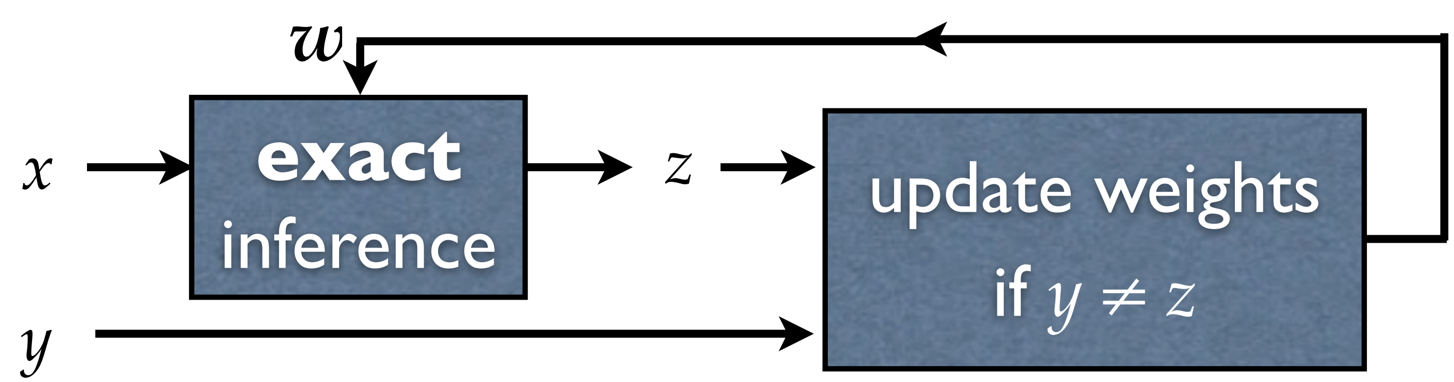




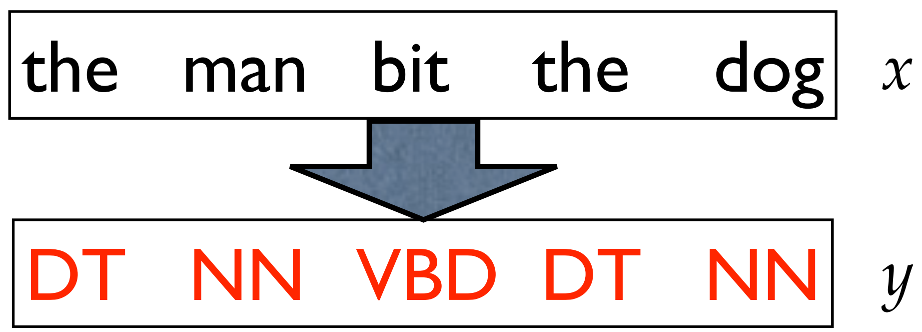
Scalability Challenge 1: Inference

binary classification



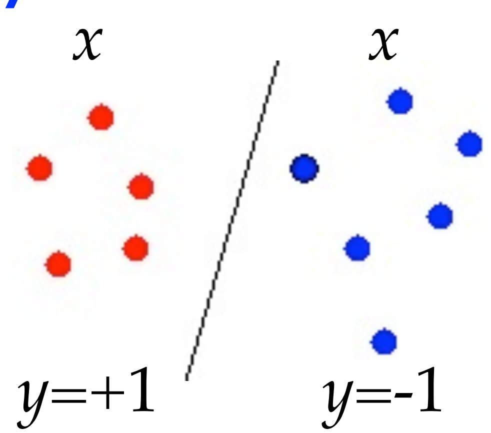


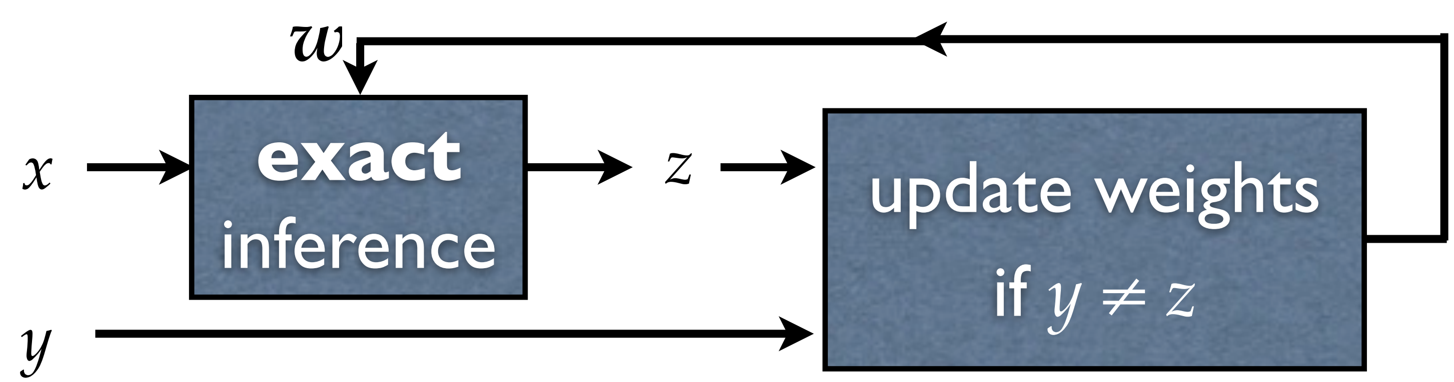
structured classification



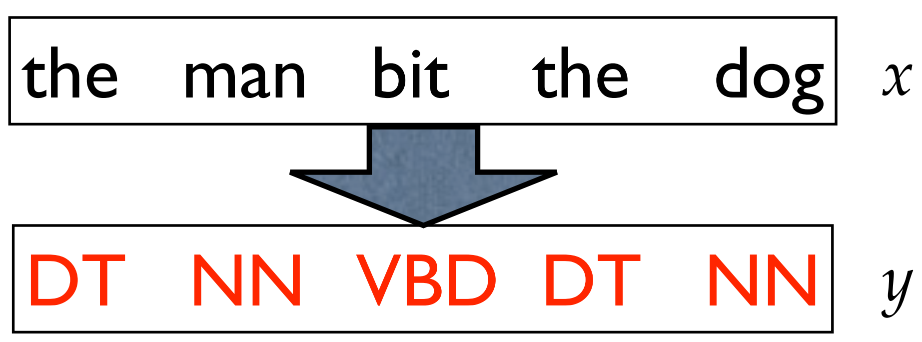
Scalability Challenge 1: Inference

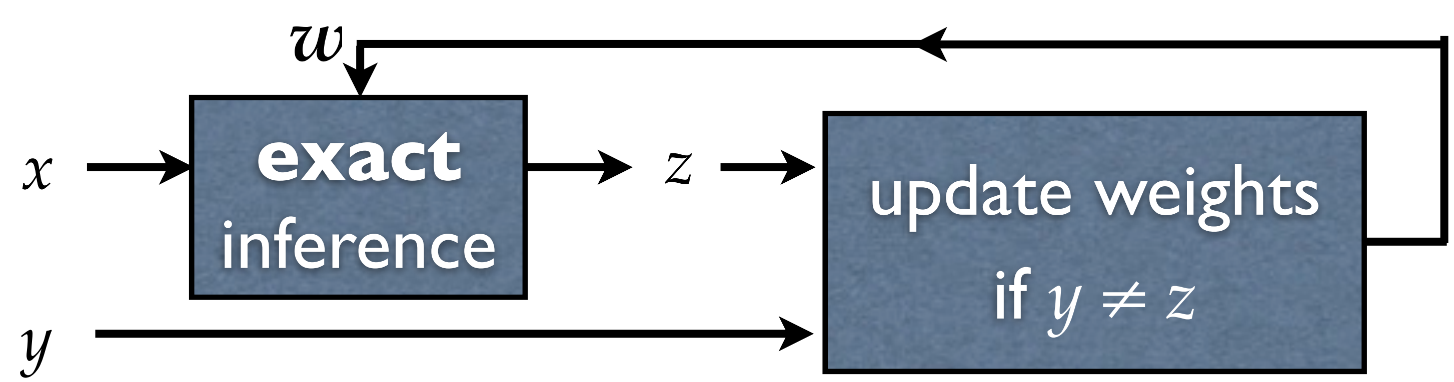
binary classification



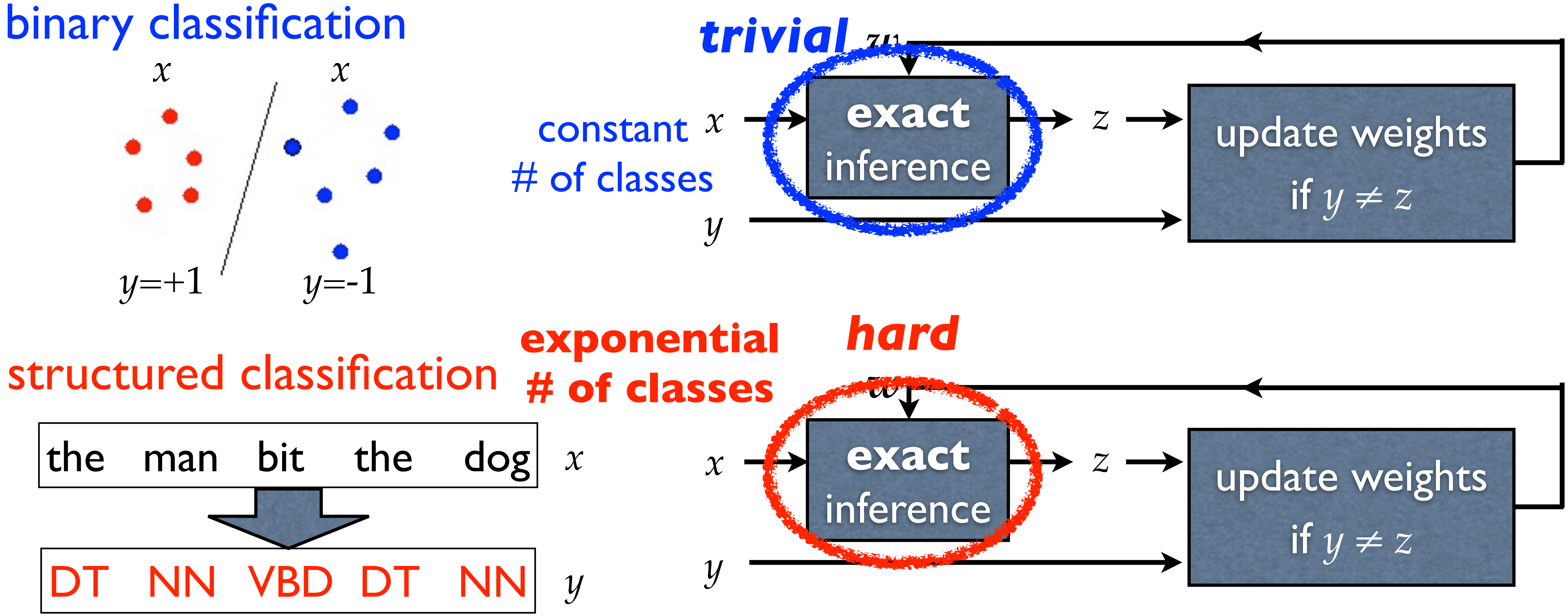


structured classification



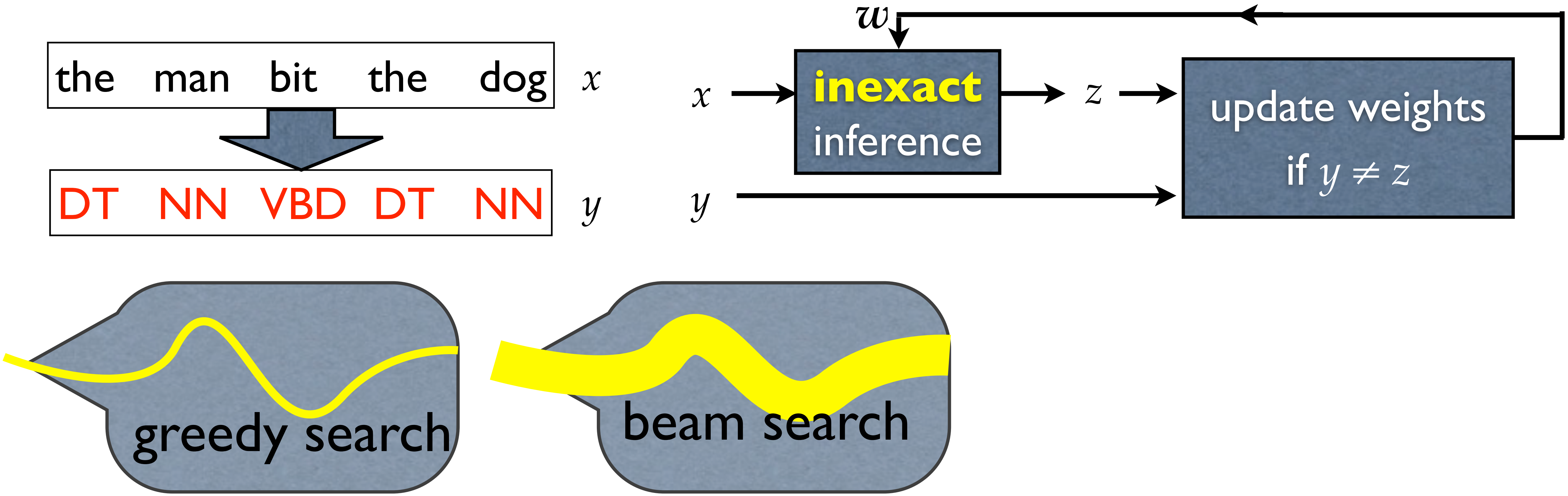


Scalability Challenge 1: Inference

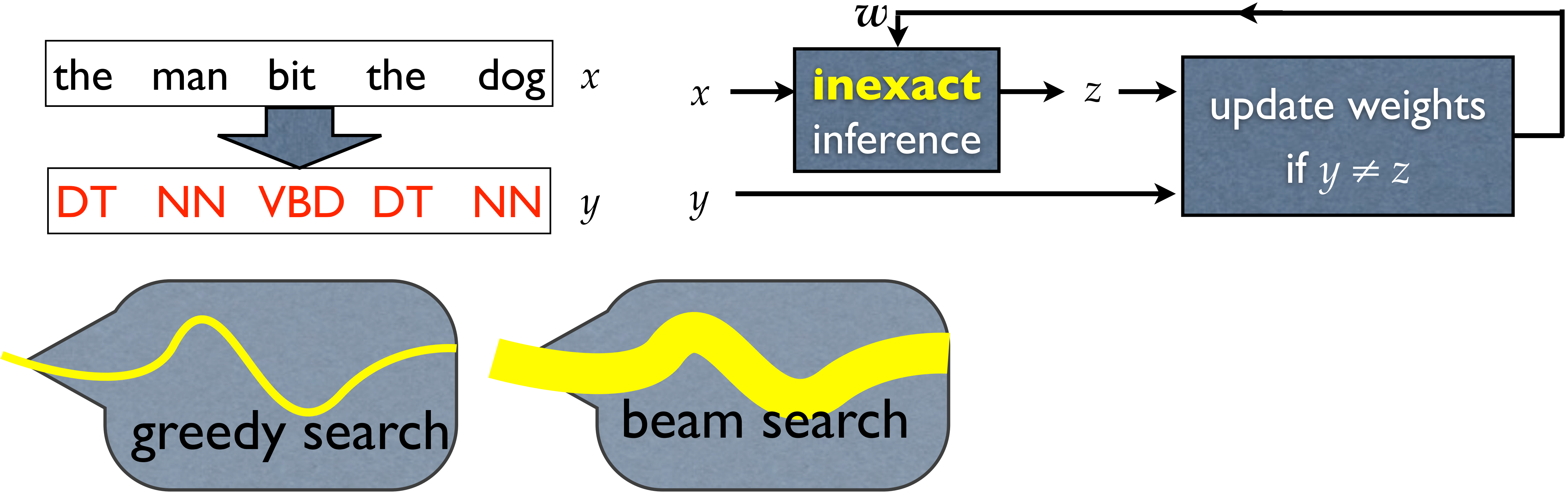


- challenge: search efficiency (exponentially many classes)
  - often use dynamic programming (DP)
  - but DP is still too slow for repeated use, e.g. parsing $O(n^3)$
  - Q: can we sacrifice search exactness for faster learning?

Perceptron w/ Inexact Inference

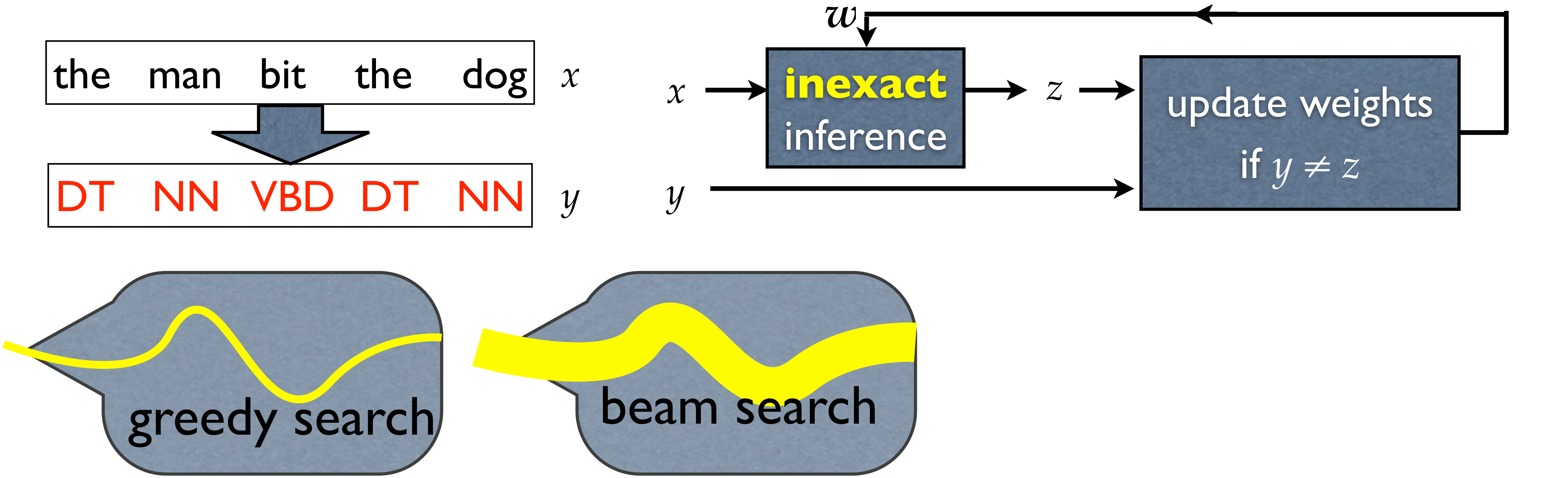


Perceptron w/ Inexact Inference



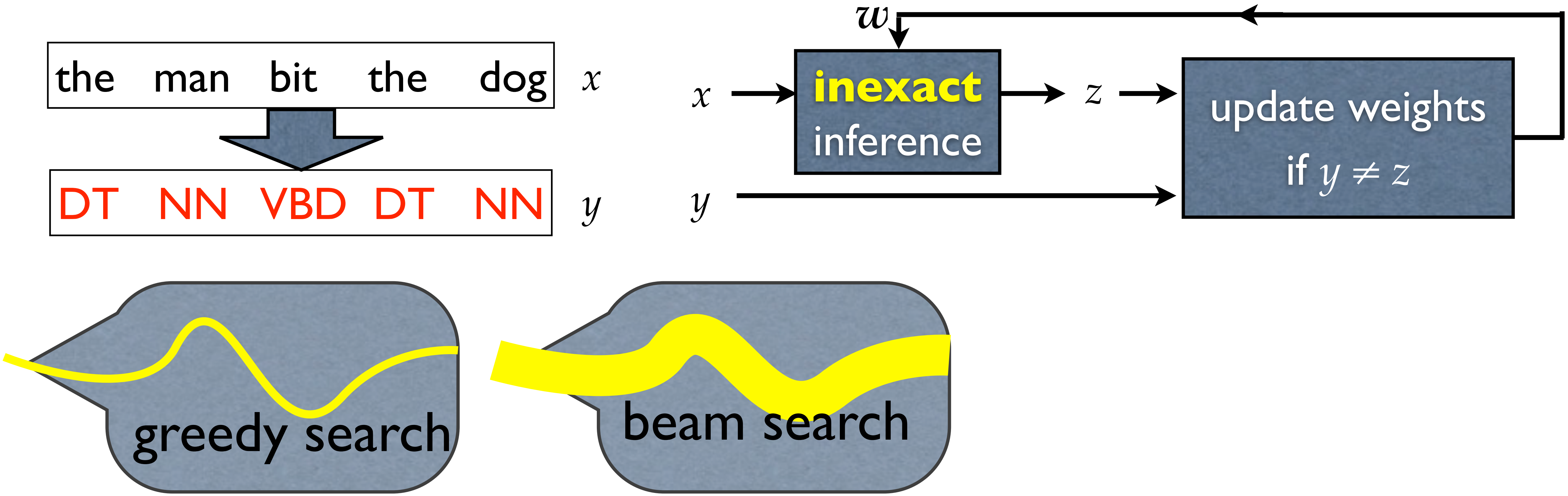
- routine use of inexact inference in NLP (e.g. beam search)

Perceptron w/ Inexact Inference



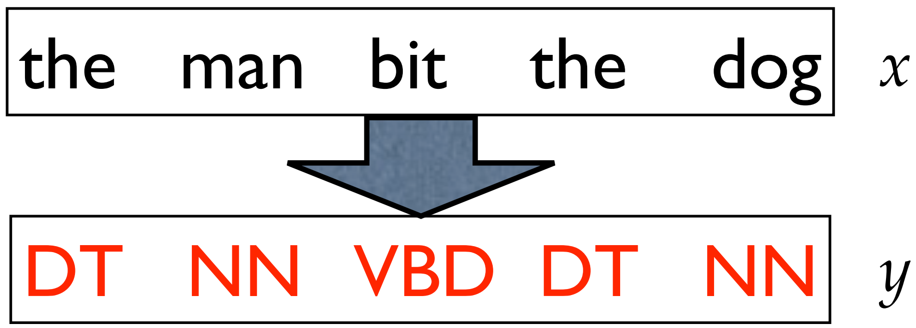
- routine use of inexact inference in NLP (e.g. beam search)
- how does structured perceptron work with inexact search?

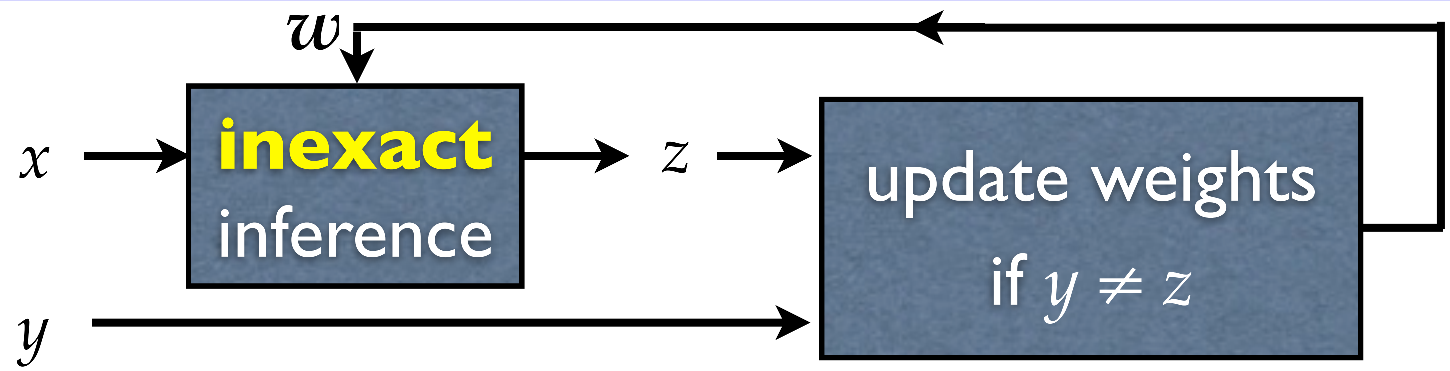
Perceptron w/ Inexact Inference

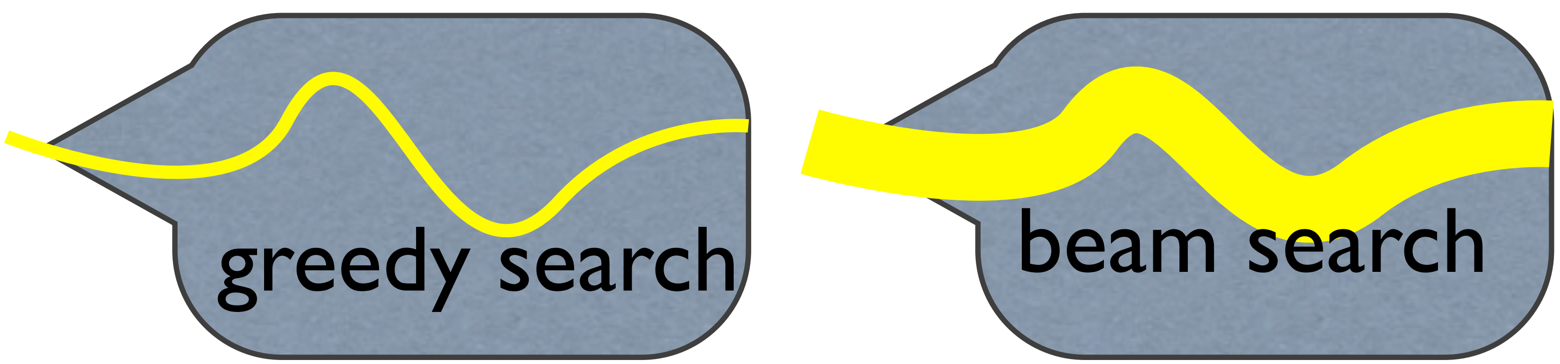


- routine use of inexact inference in NLP (e.g. beam search)
- how does structured perceptron work with inexact search?
  - so far most structured learning theory assume exact search

Perceptron w/ Inexact Inference



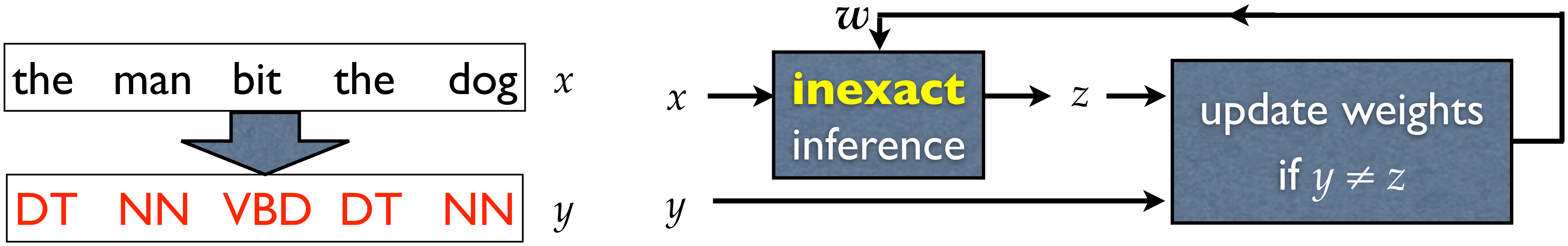


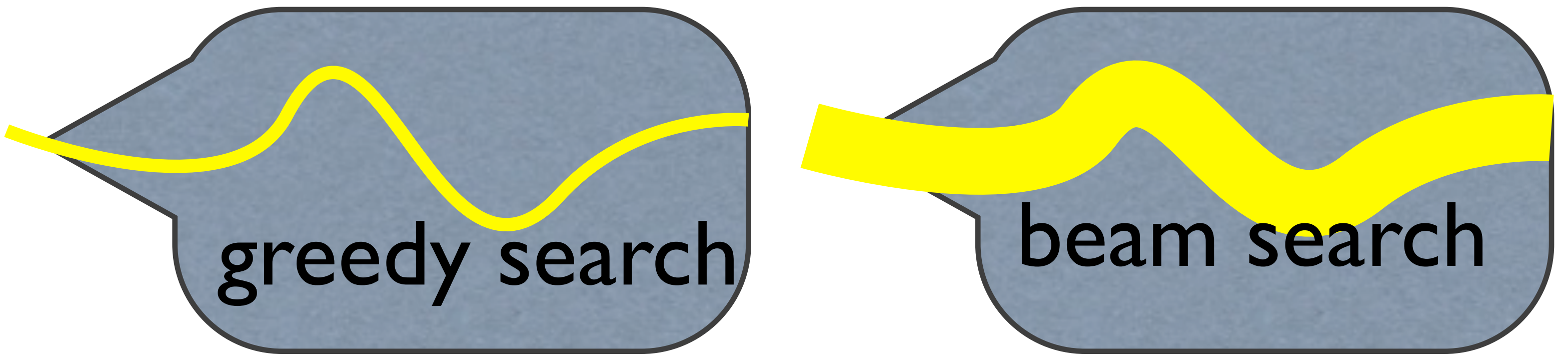


Q: does perceptron still work???

- routine use of inexact inference in NLP (e.g. beam search)
- how does structured perceptron work with inexact search?
  - so far most structured learning theory assume exact search
  - would search errors break these learning properties?

Perceptron w/ Inexact Inference

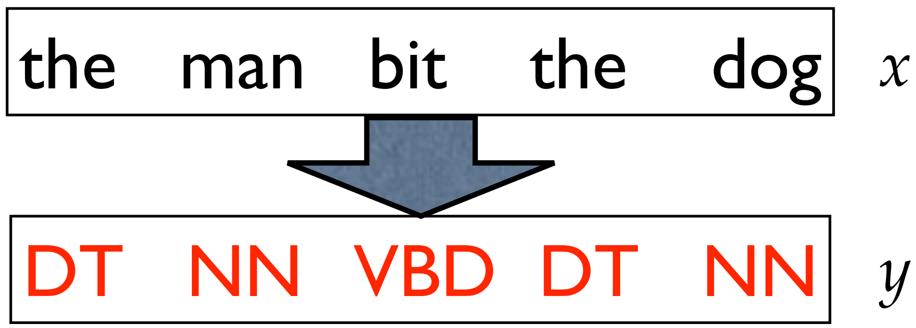


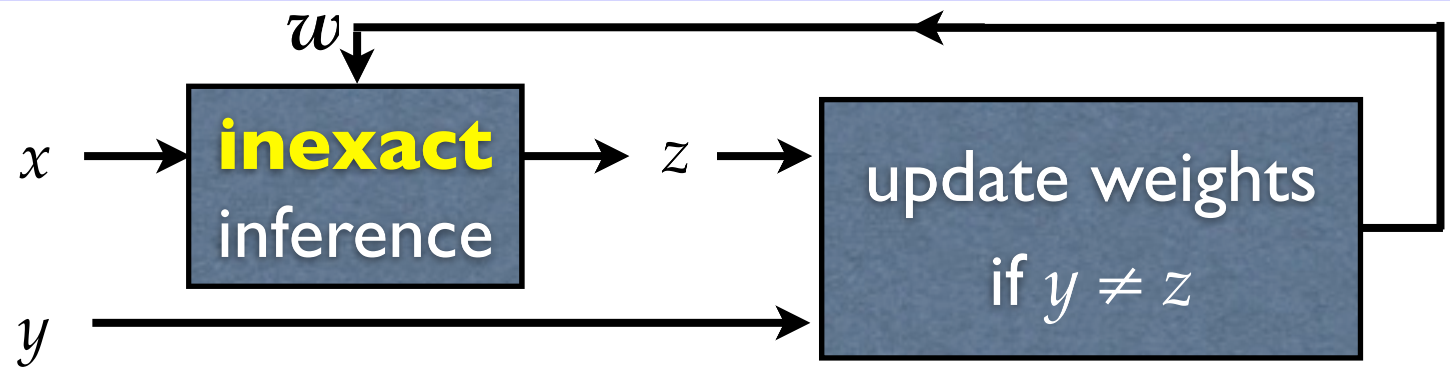


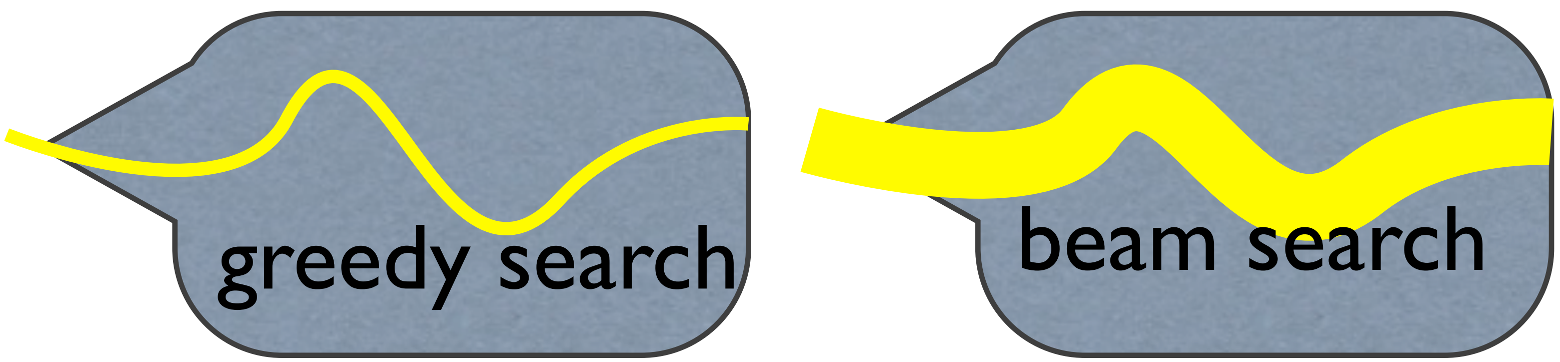
Q: does perceptron still work???

- routine use of inexact inference in NLP (e.g. beam search)
- how does structured perceptron work with inexact search?
  - so far most structured learning theory assume exact search
  - would search errors break these learning properties?
  - if so how to modify learning to accommodate inexact search?

Bad News and Good News

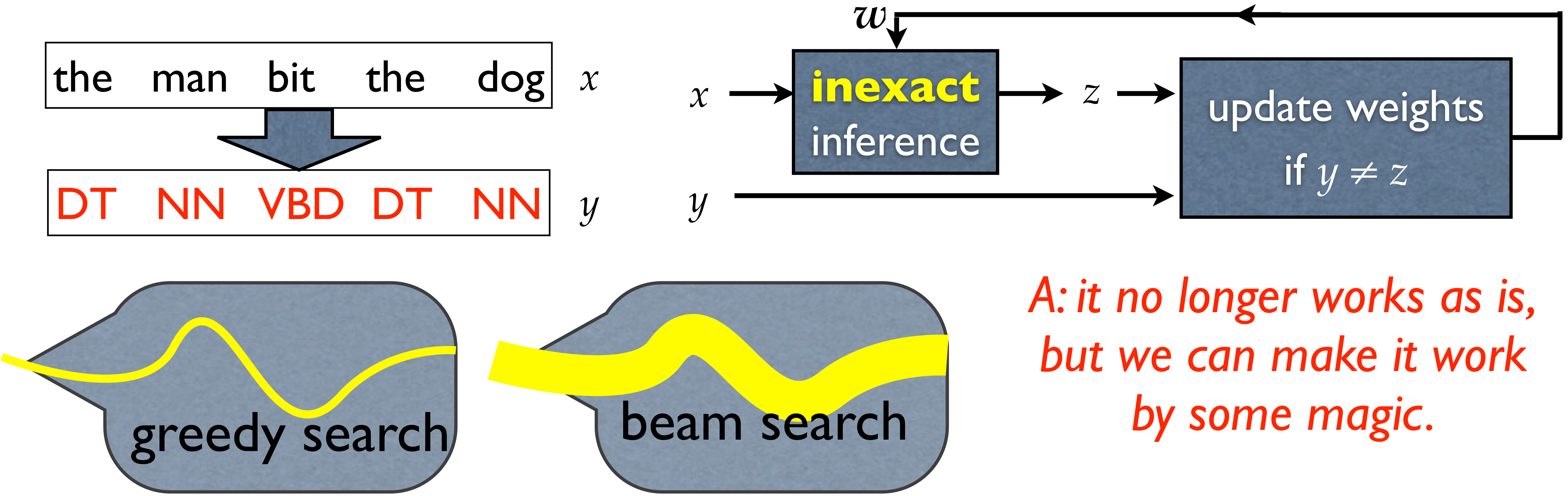






A: it no longer works as is, but we can make it work by some magic.

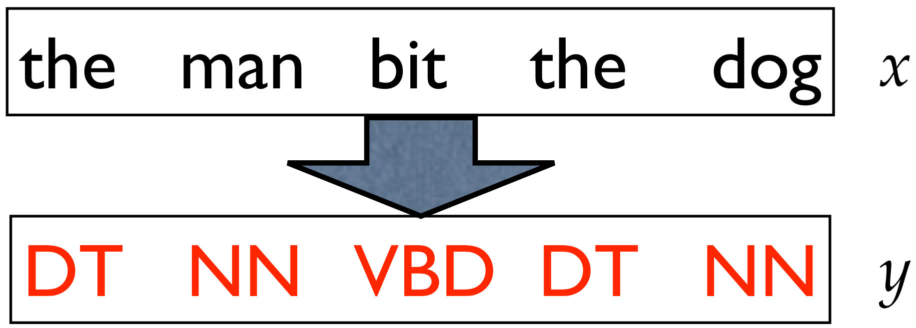
Bad News and Good News

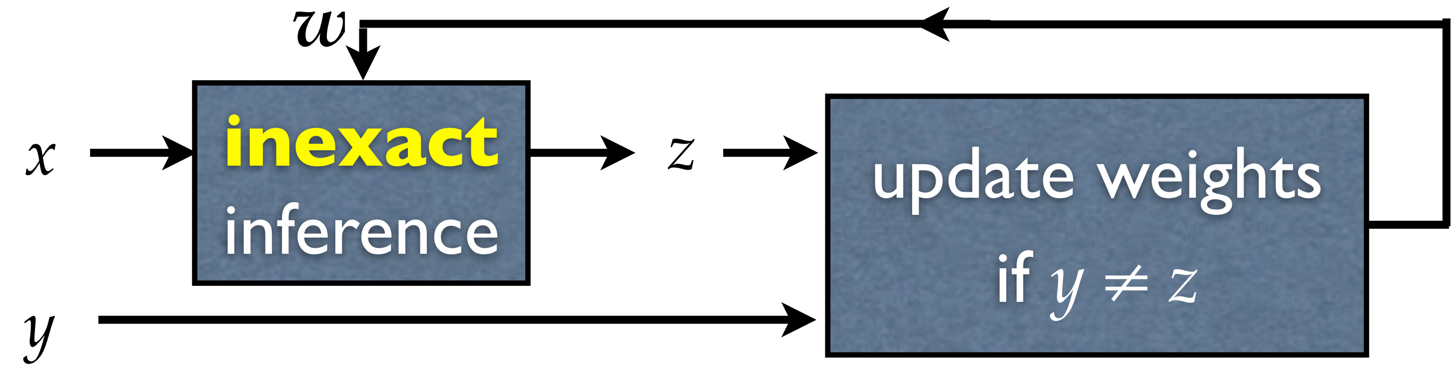


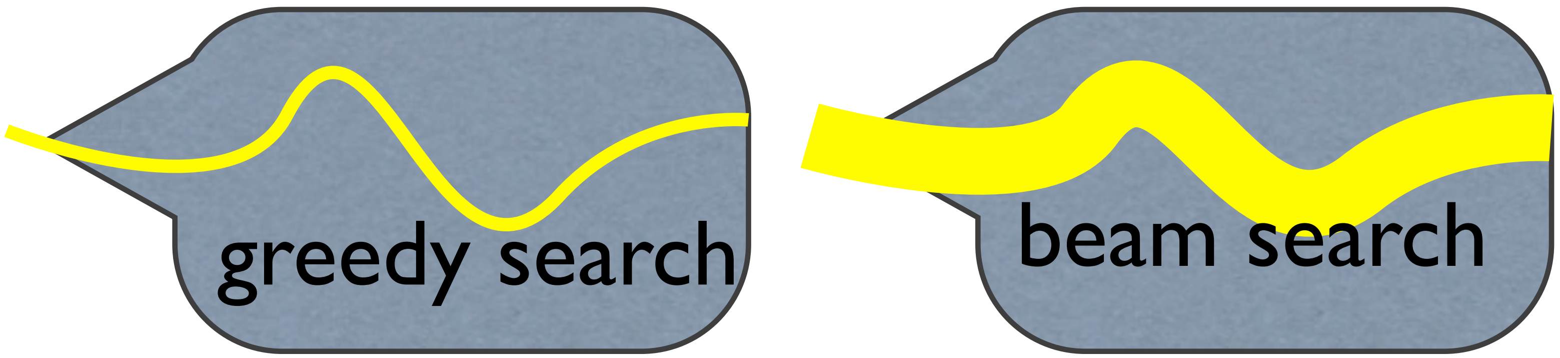
A: it no longer works as is, but we can make it work by some magic.

- bad news: no more guarantee of convergence

Bad News and Good News



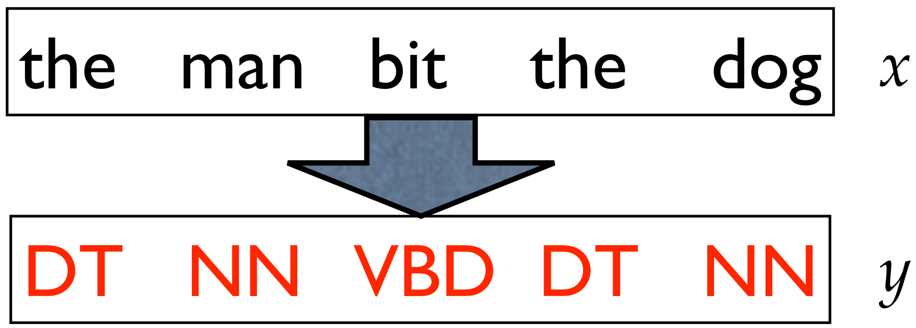


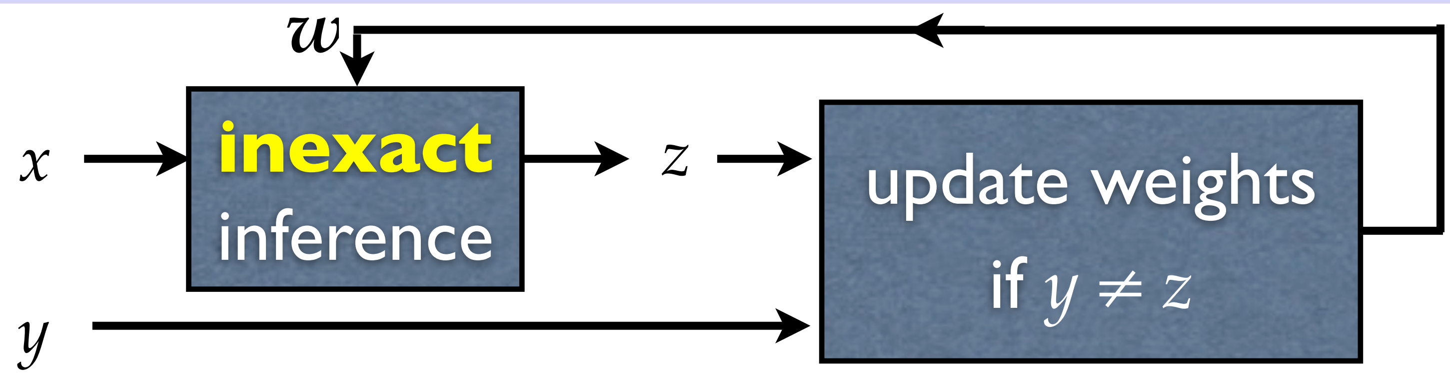


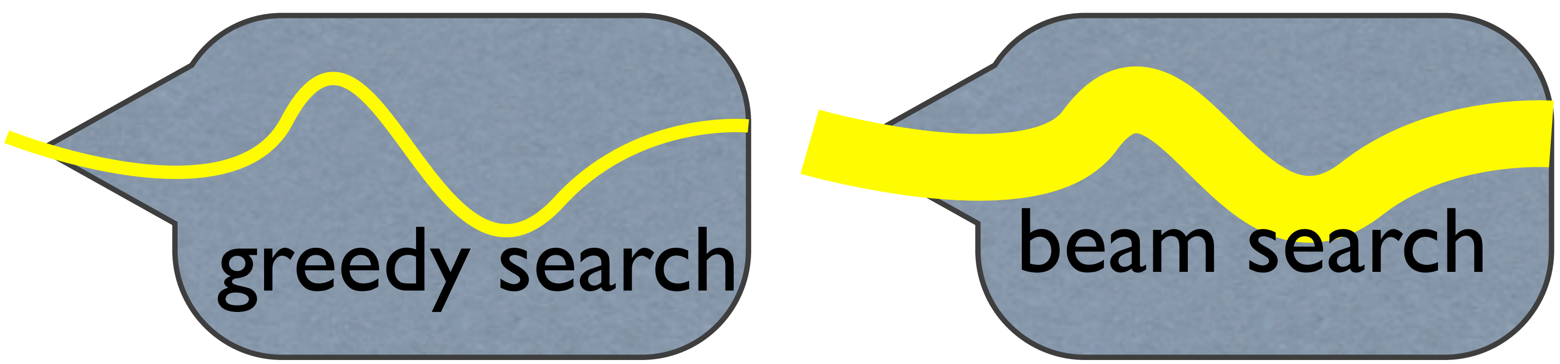
A: it no longer works as is, but we can make it work by some magic.

- bad news: no more guarantee of convergence
  - in practice perceptron degrades a lot due to search errors

Bad News and Good News



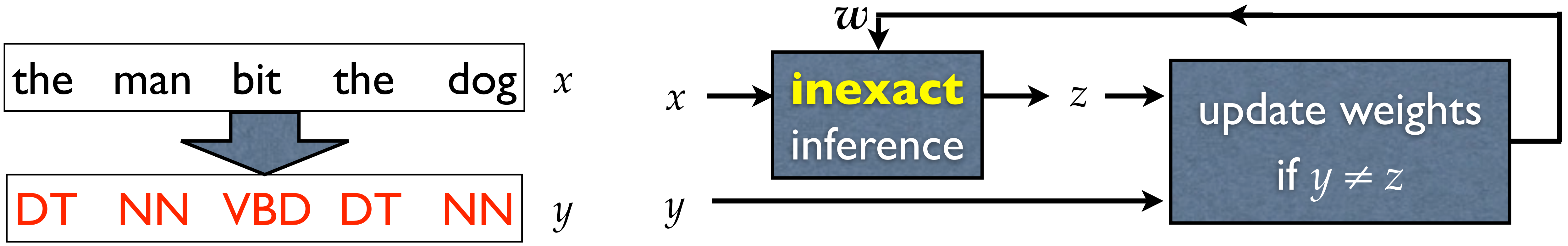


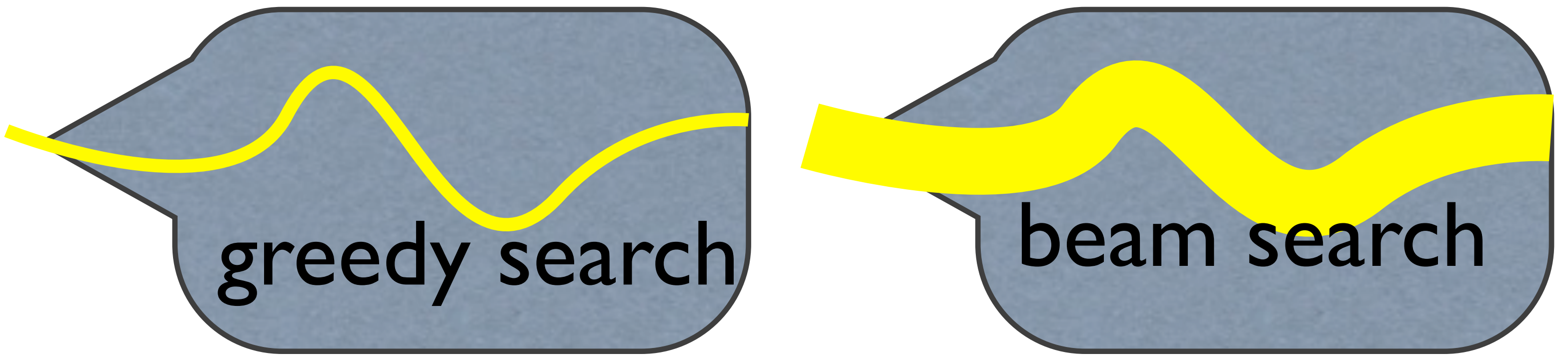


A: it no longer works as is,
but we can make it work
by some magic.

- bad news: no more guarantee of convergence
  - in practice perceptron degrades a lot due to search errors
- good news: new update methods guarantee convergence

Bad News and Good News

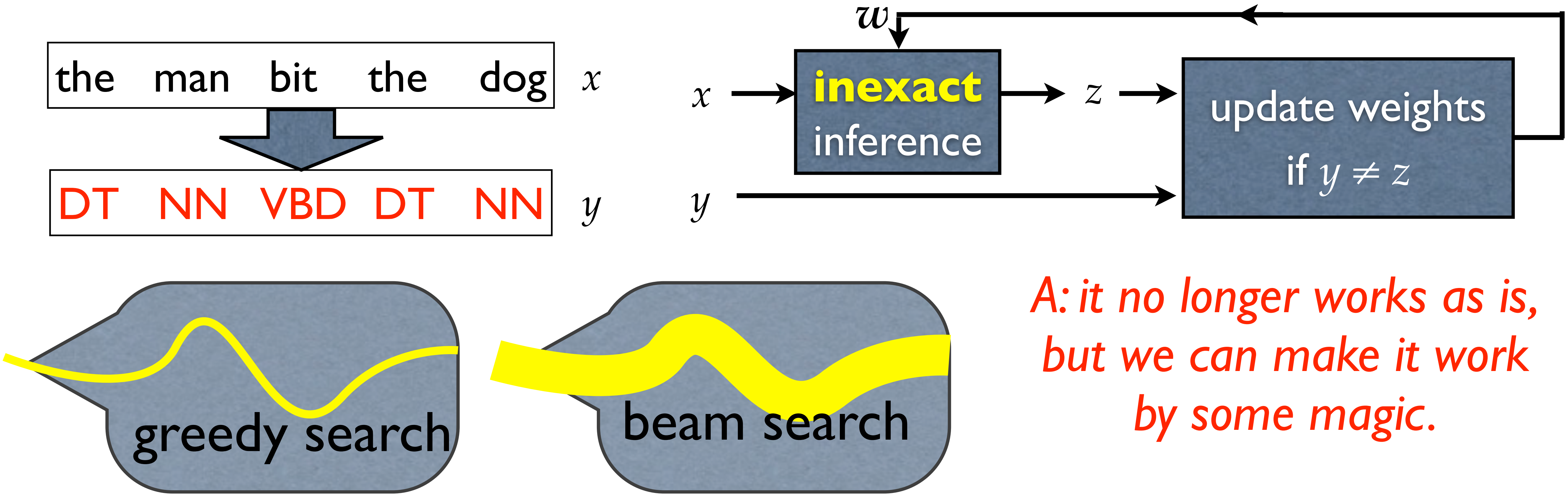




A: it no longer works as is, but we can make it work by some magic.

- bad news: no more guarantee of convergence
  - in practice perceptron degrades a lot due to search errors
- good news: new update methods guarantee convergence
  - new perceptron variants that "live with" search errors

Bad News and Good News

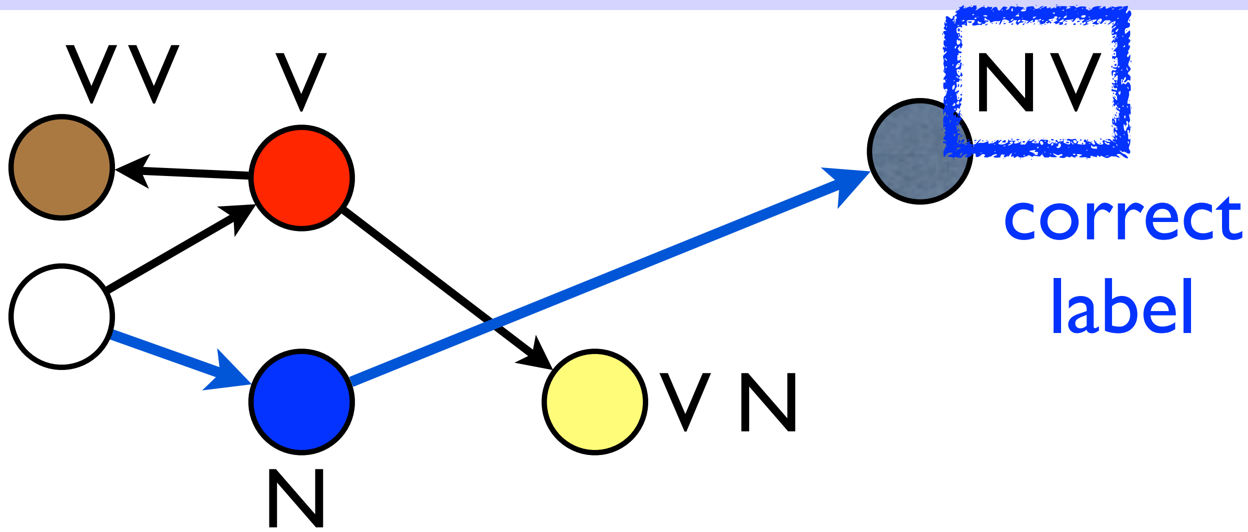


A: it no longer works as is, but we can make it work by some magic.

- bad news: no more guarantee of convergence
  - in practice perceptron degrades a lot due to search errors
- good news: new update methods guarantee convergence
  - new perceptron variants that "live with" search errors
  - in practice they work really well w/ inexact search

Convergence with Exact Search

current model



training example

time flies

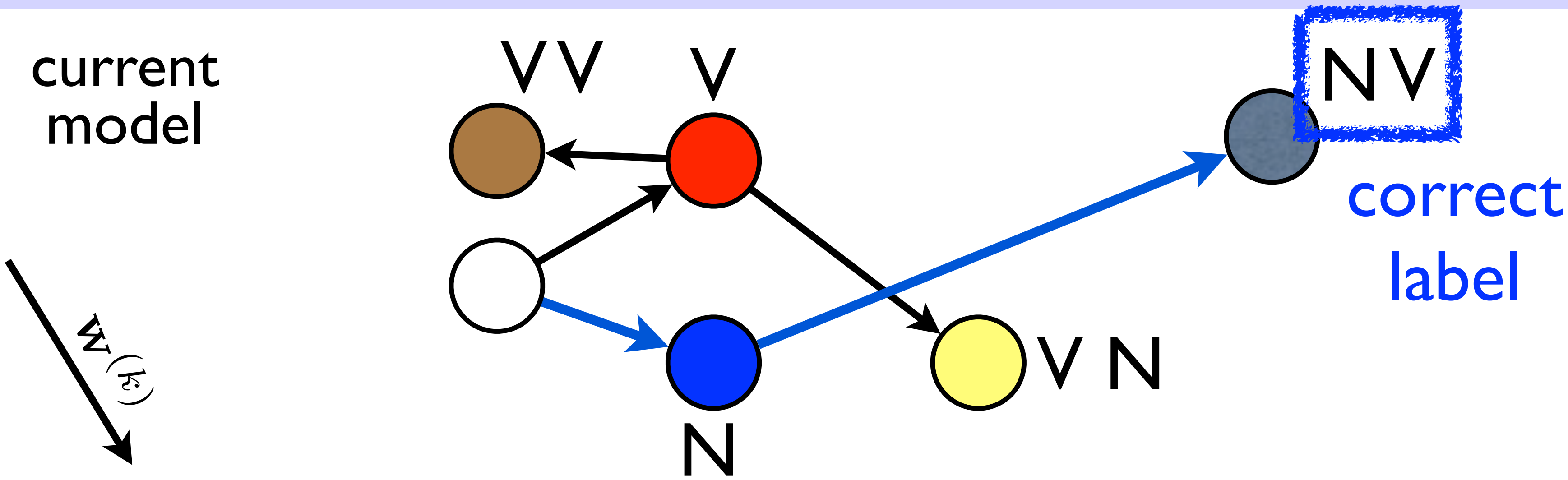
N V

output space

{N,V} x {N,V}

Convergence with Exact Search

current model



training example

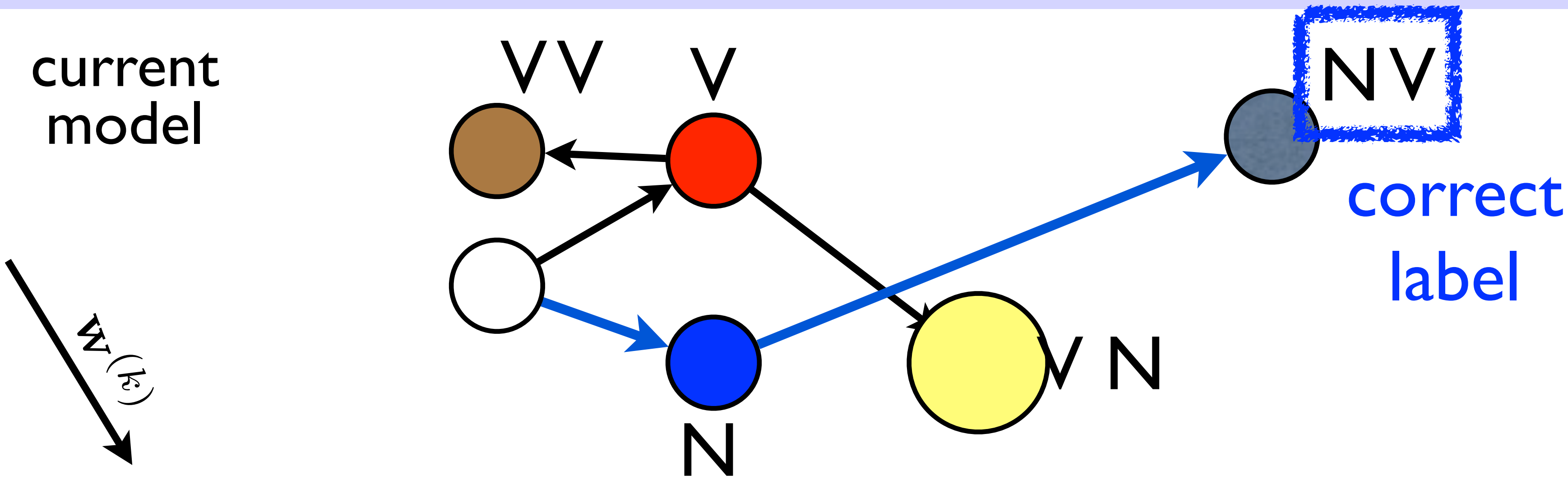
time flies

N V

output space

{N,V} x {N,V}

Convergence with Exact Search



training example

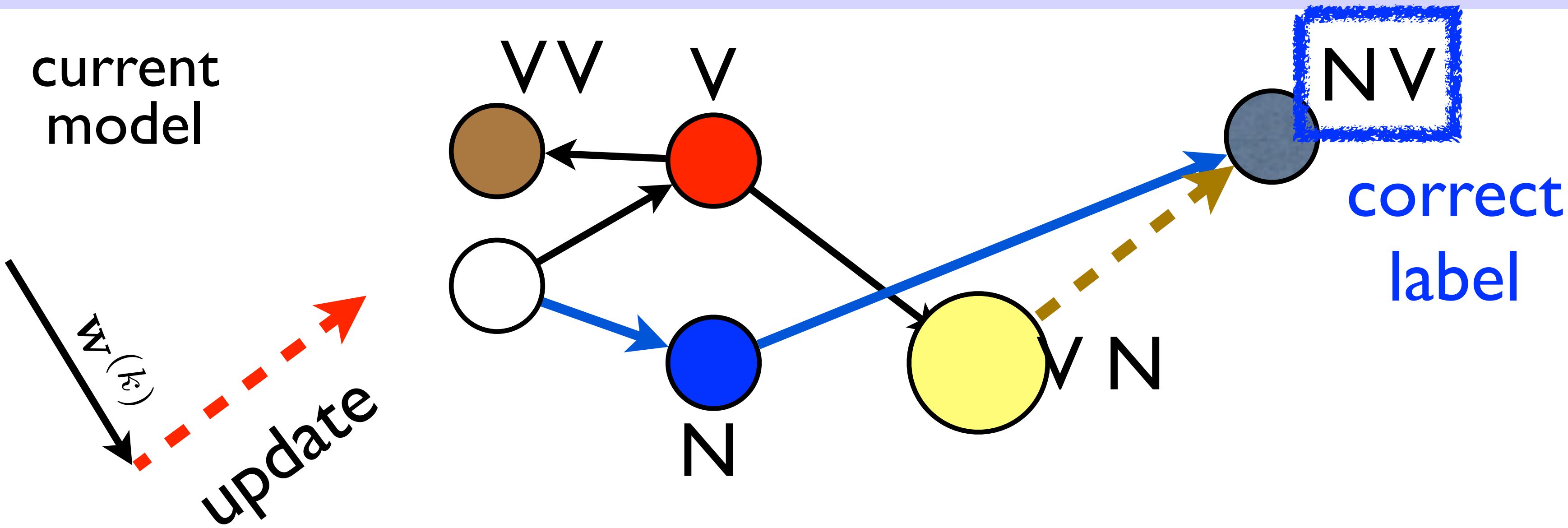
time flies

N V

output space

{N,V} x {N,V}

Convergence with Exact Search



training example

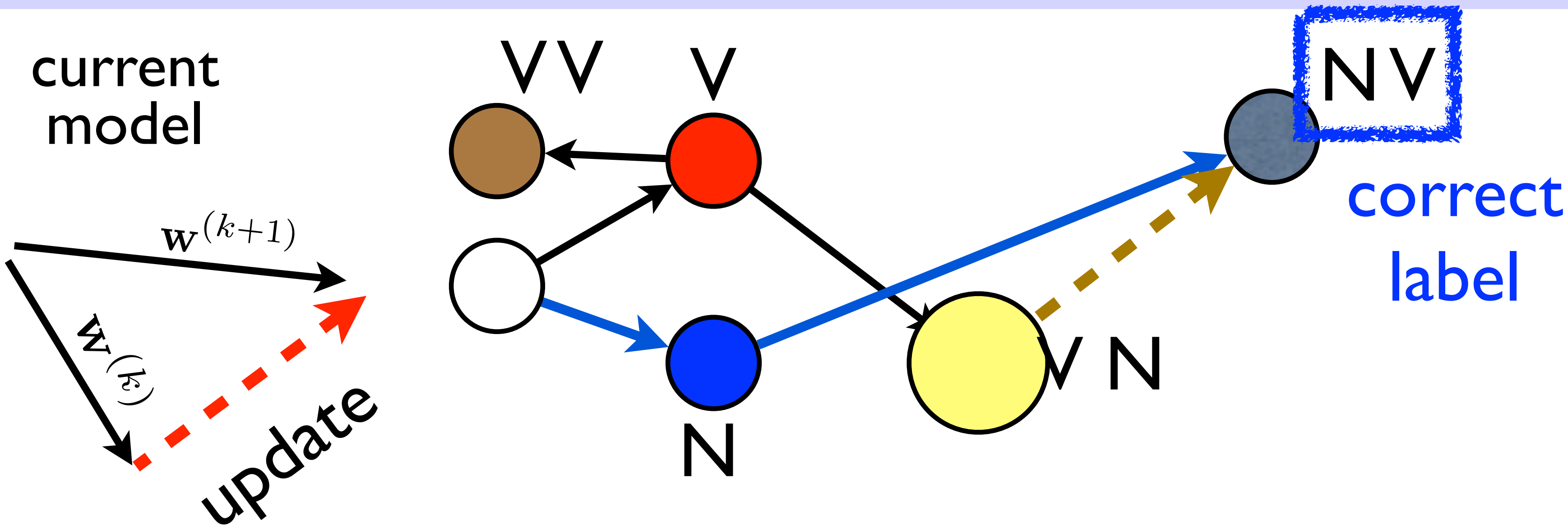
time flies

N V

output space

{N,V} x {N,V}

Convergence with Exact Search



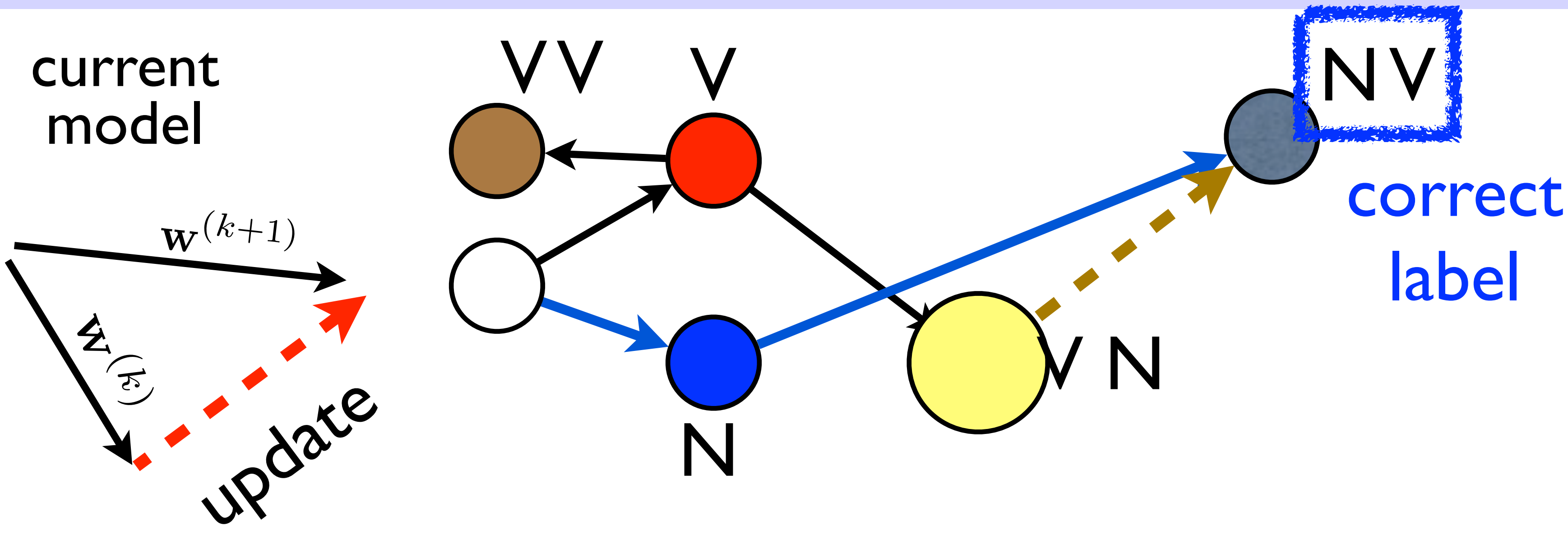
training example

time	flies
N	V

output space

{N,V} x {N,V}

Convergence with Exact Search



training example

time flies

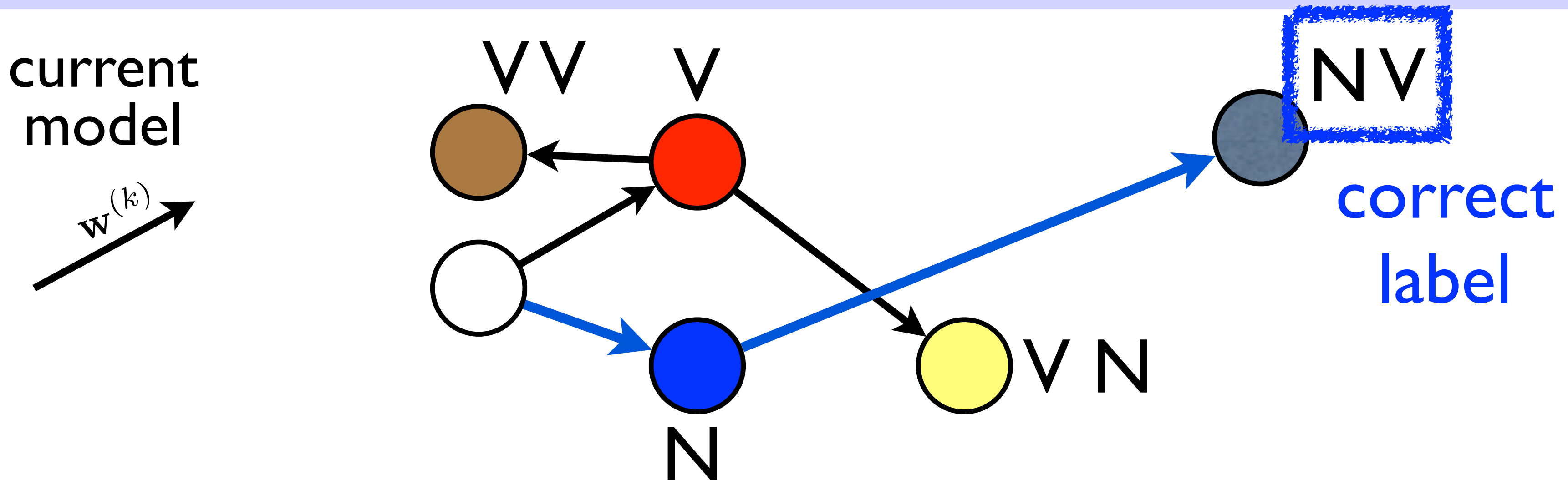
N V

output space

{N,V} x {N,V}

structured perceptron converges with exact search

No Convergence w/ Greedy Search



training example

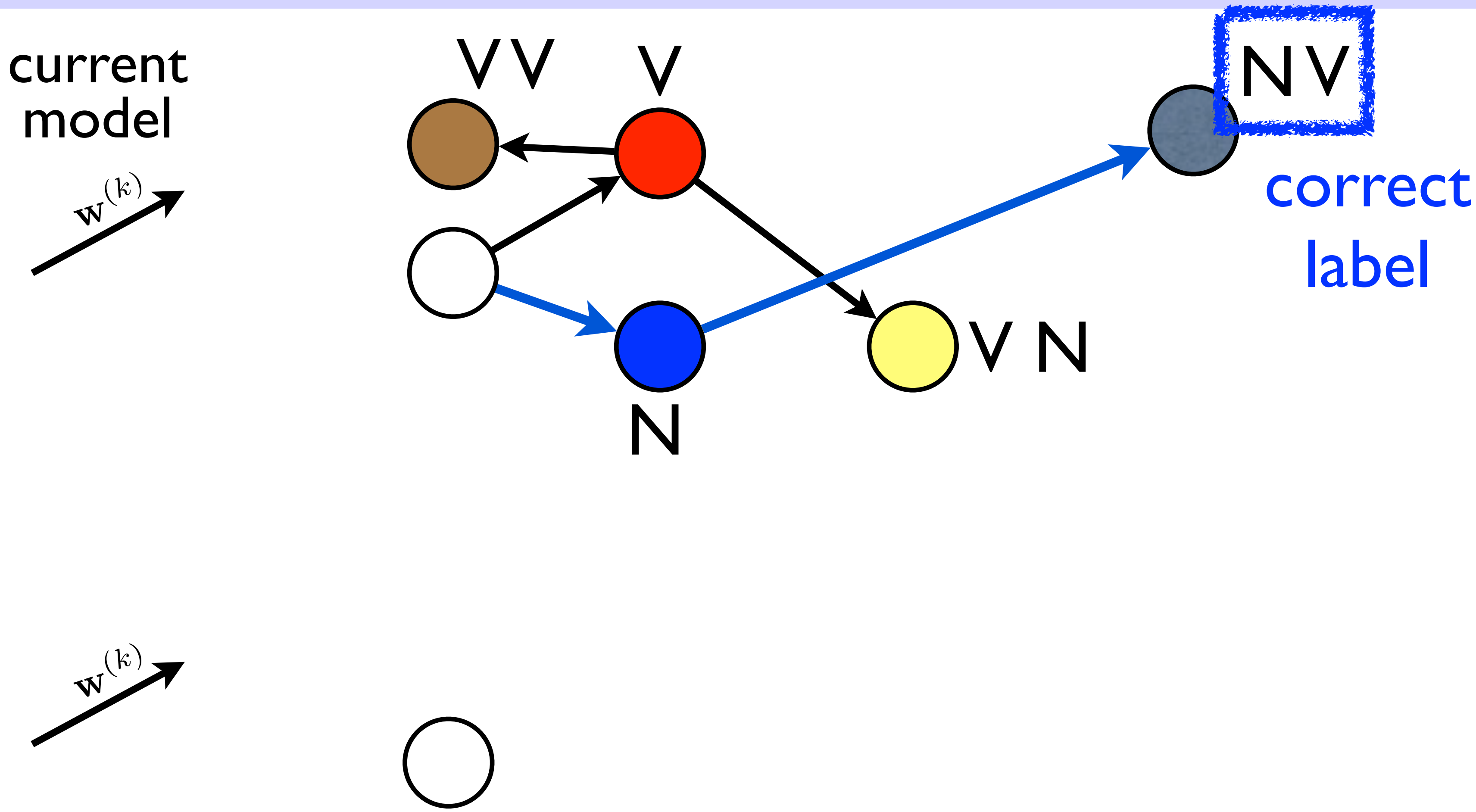
time flies

N V

output space

{N,V} x {N,V}

No Convergence w/ Greedy Search



training example

time flies

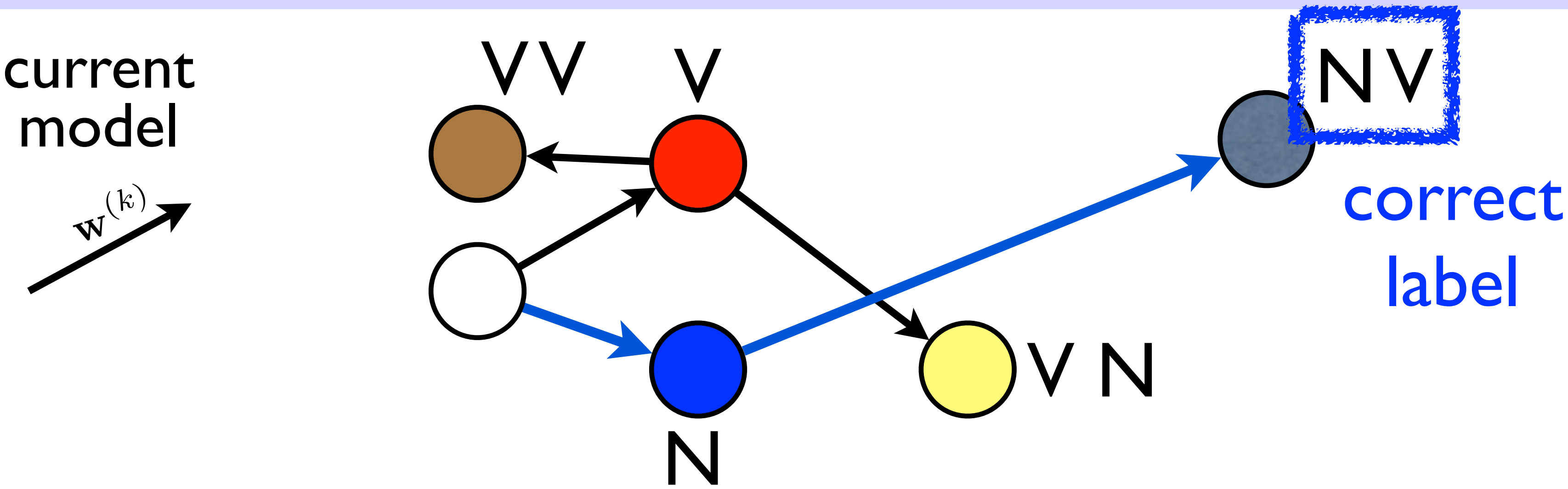
N V

output space

{N,V} x {N,V}

No Convergence w/ Greedy Search

current model



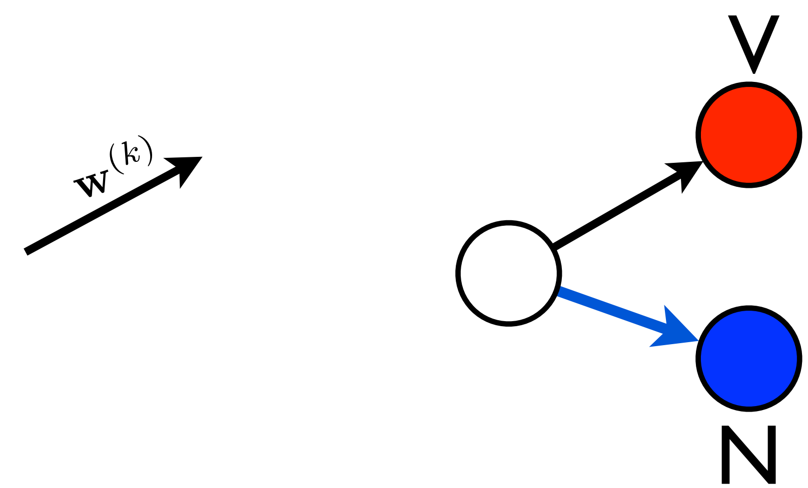
training example

time flies

N V

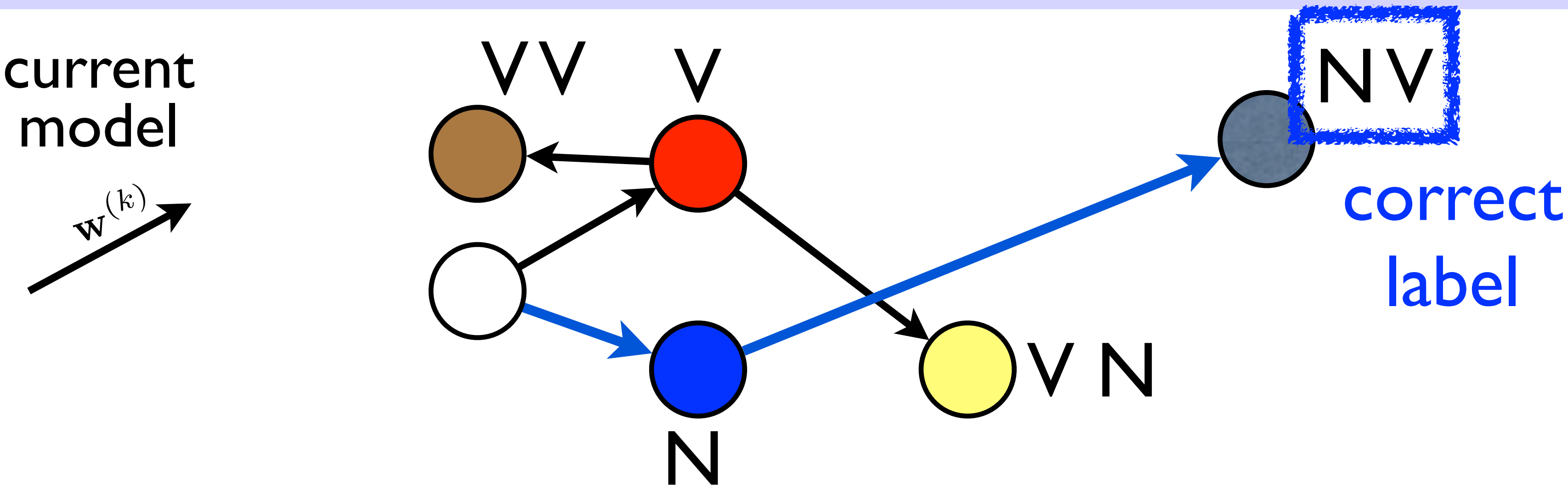
output space

{N,V} x {N,V}



No Convergence w/ Greedy Search

current model



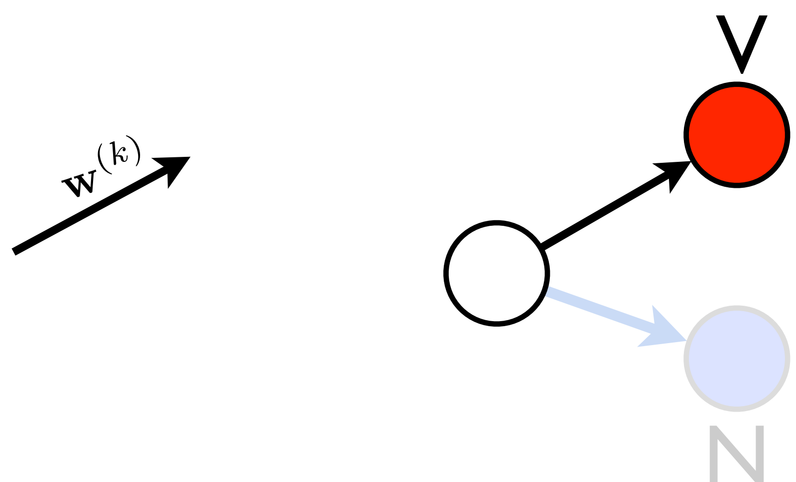
training example

time flies

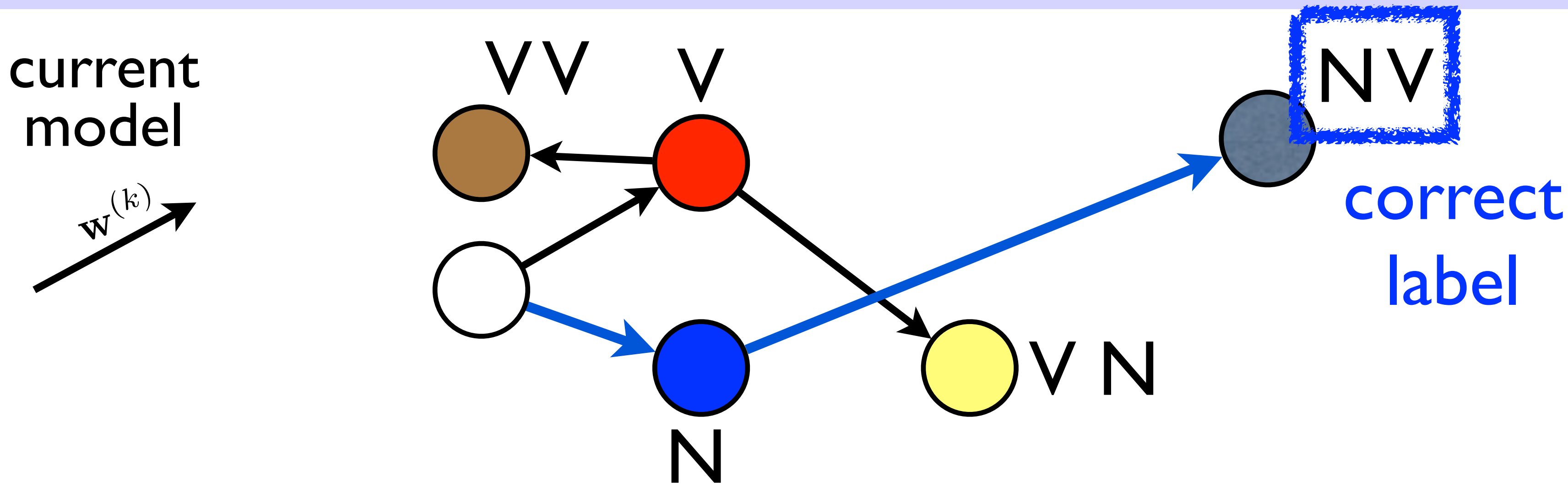
N V

output space

{N,V} x {N,V}



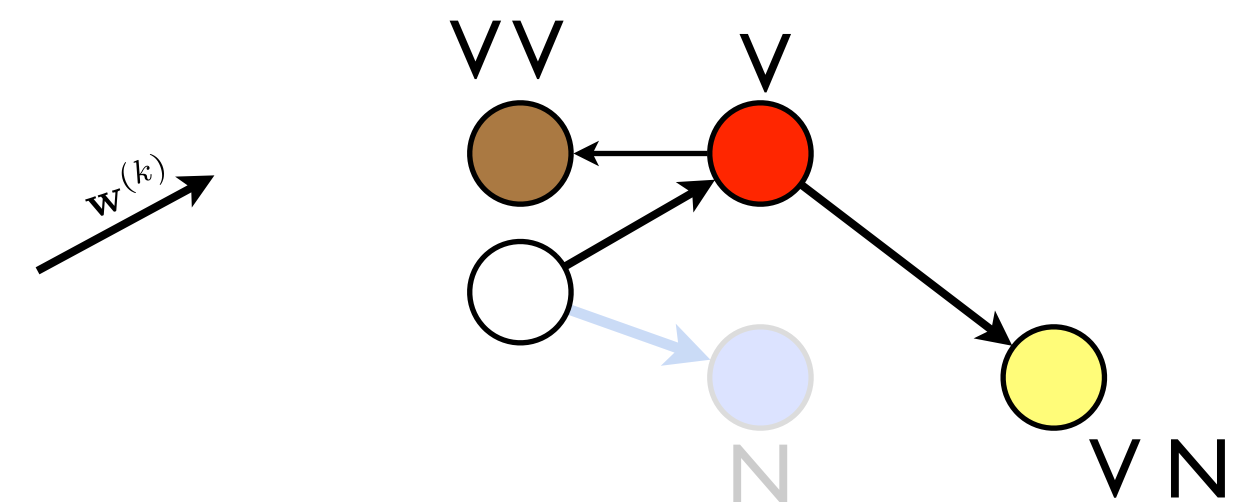
No Convergence w/ Greedy Search



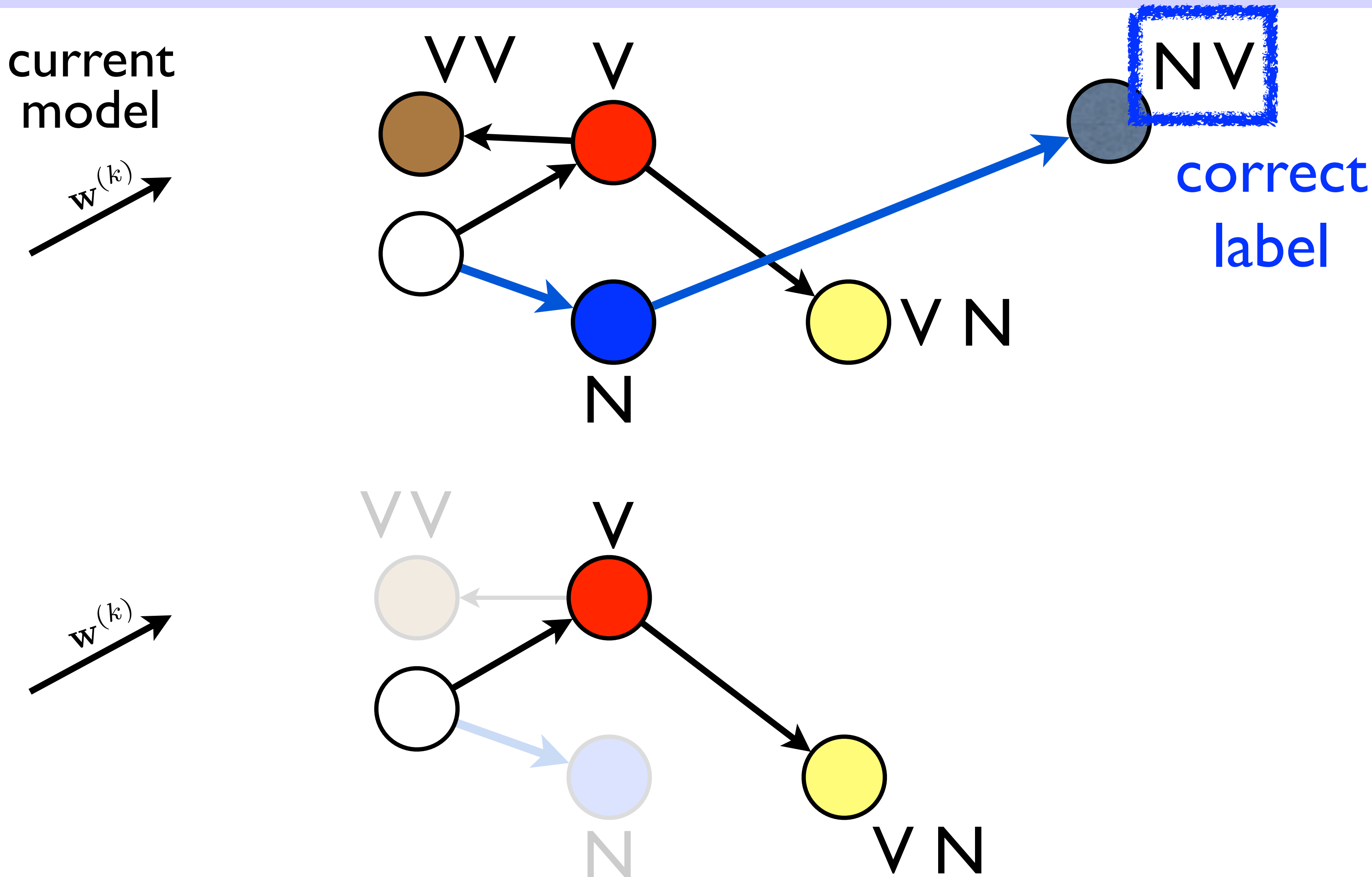
training example

time flies
N V

output space
{N,V} x {N,V}



No Convergence w/ Greedy Search

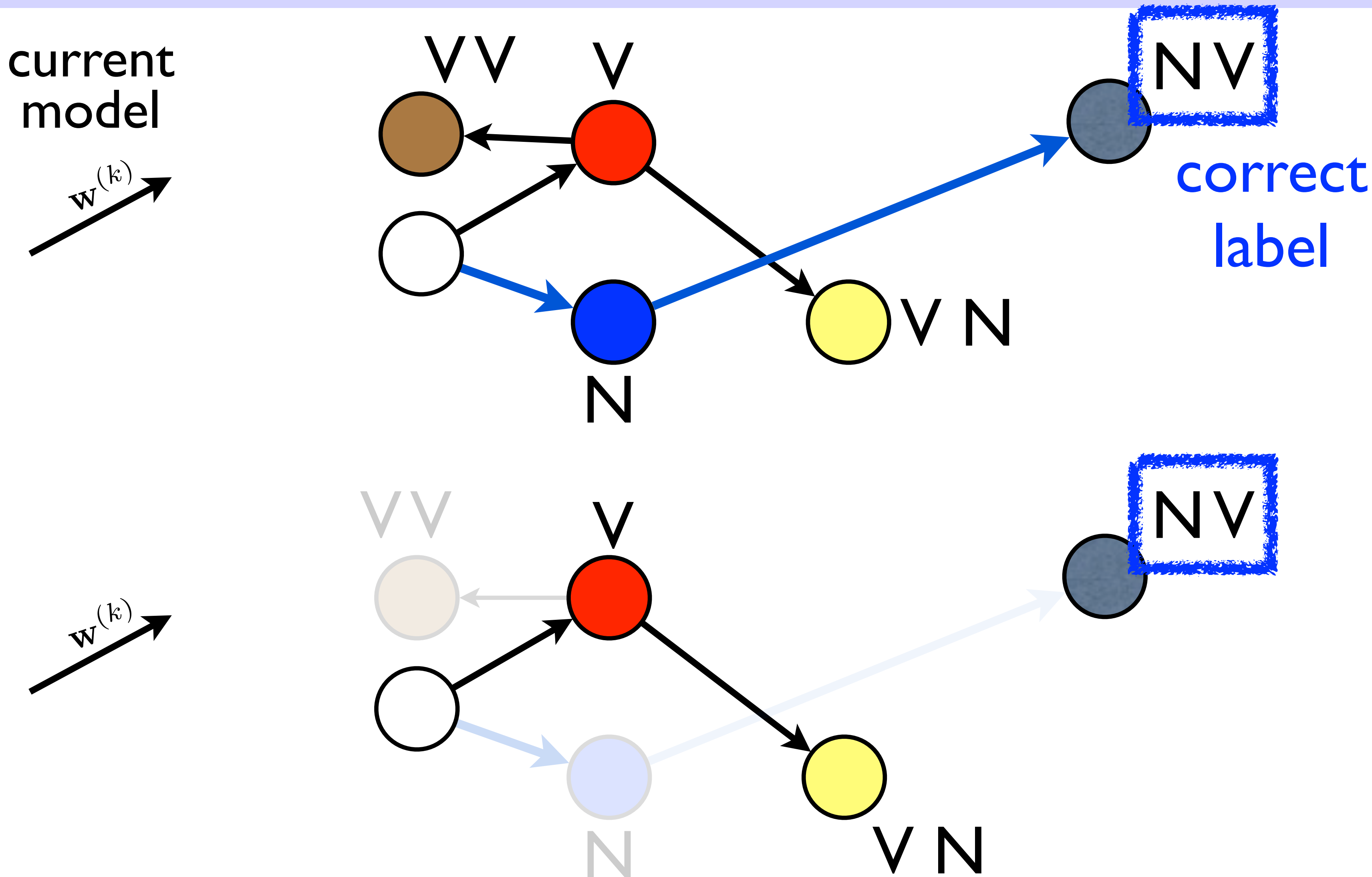


training example

time flies
N V

output space
{N,V} x {N,V}

No Convergence w/ Greedy Search



training example

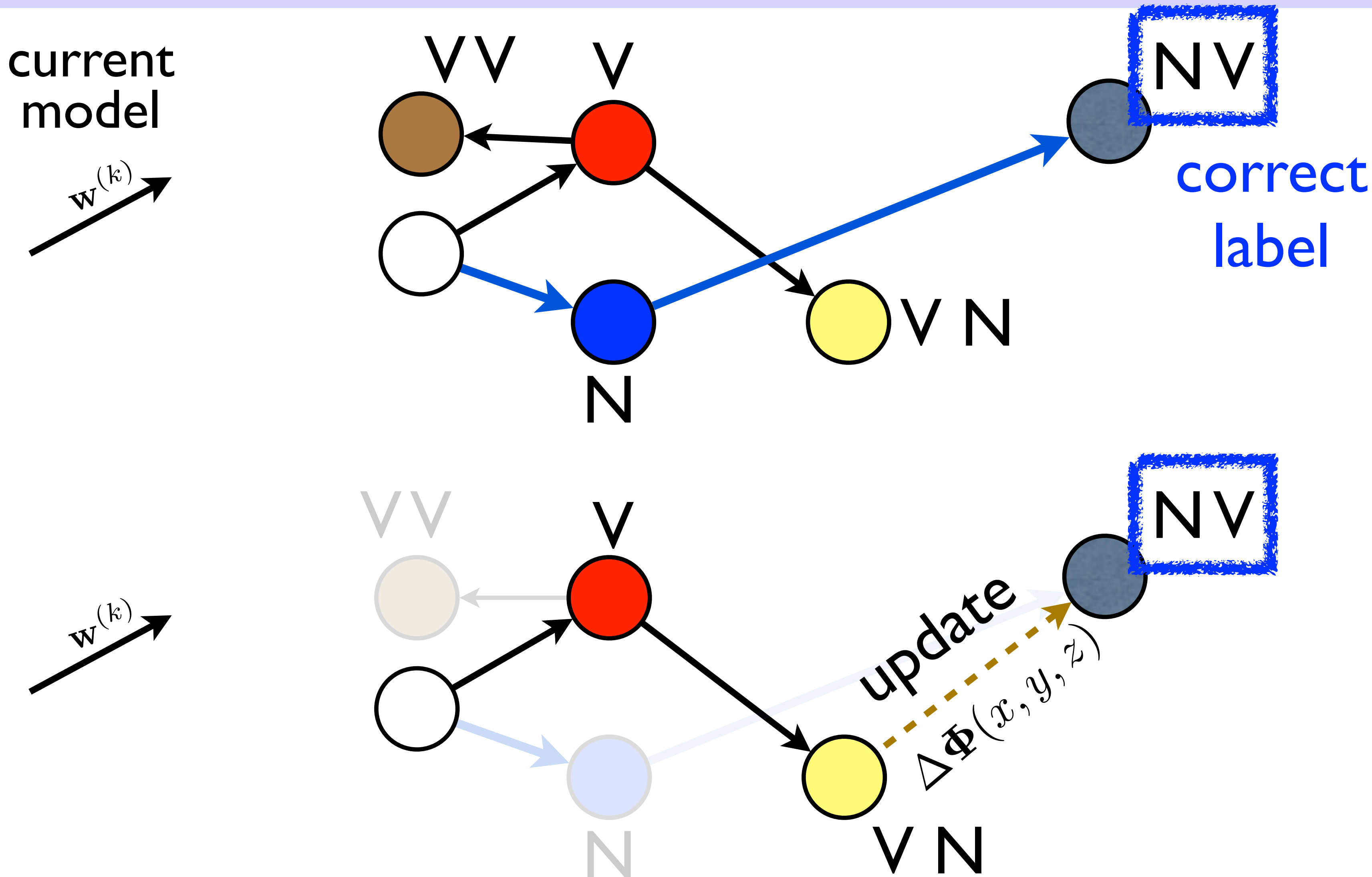
time flies

N V

output space

{N,V} x {N,V}

No Convergence w/ Greedy Search



training example

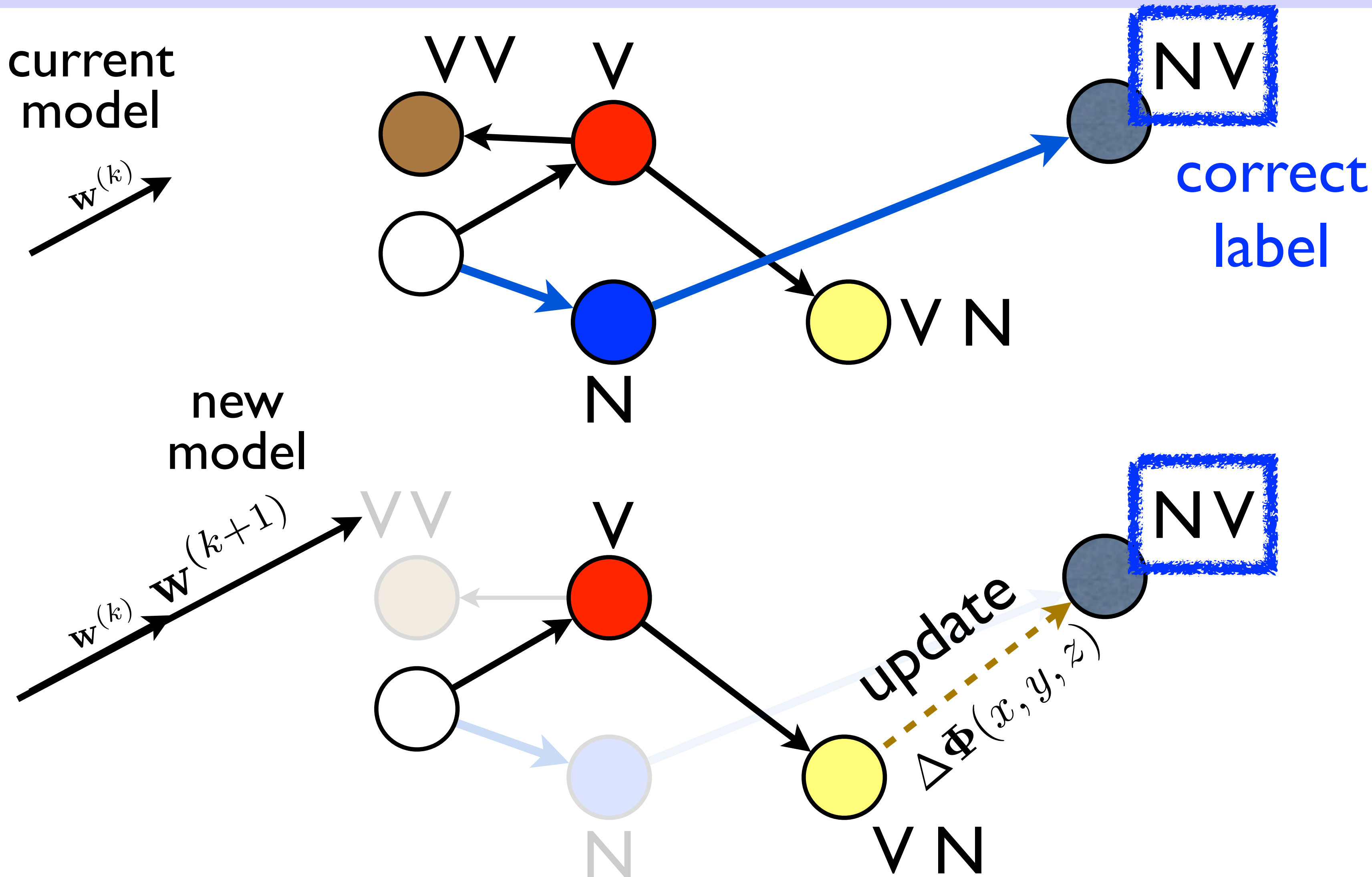
time flies

N V

output space

{N,V} x {N,V}

No Convergence w/ Greedy Search



training example

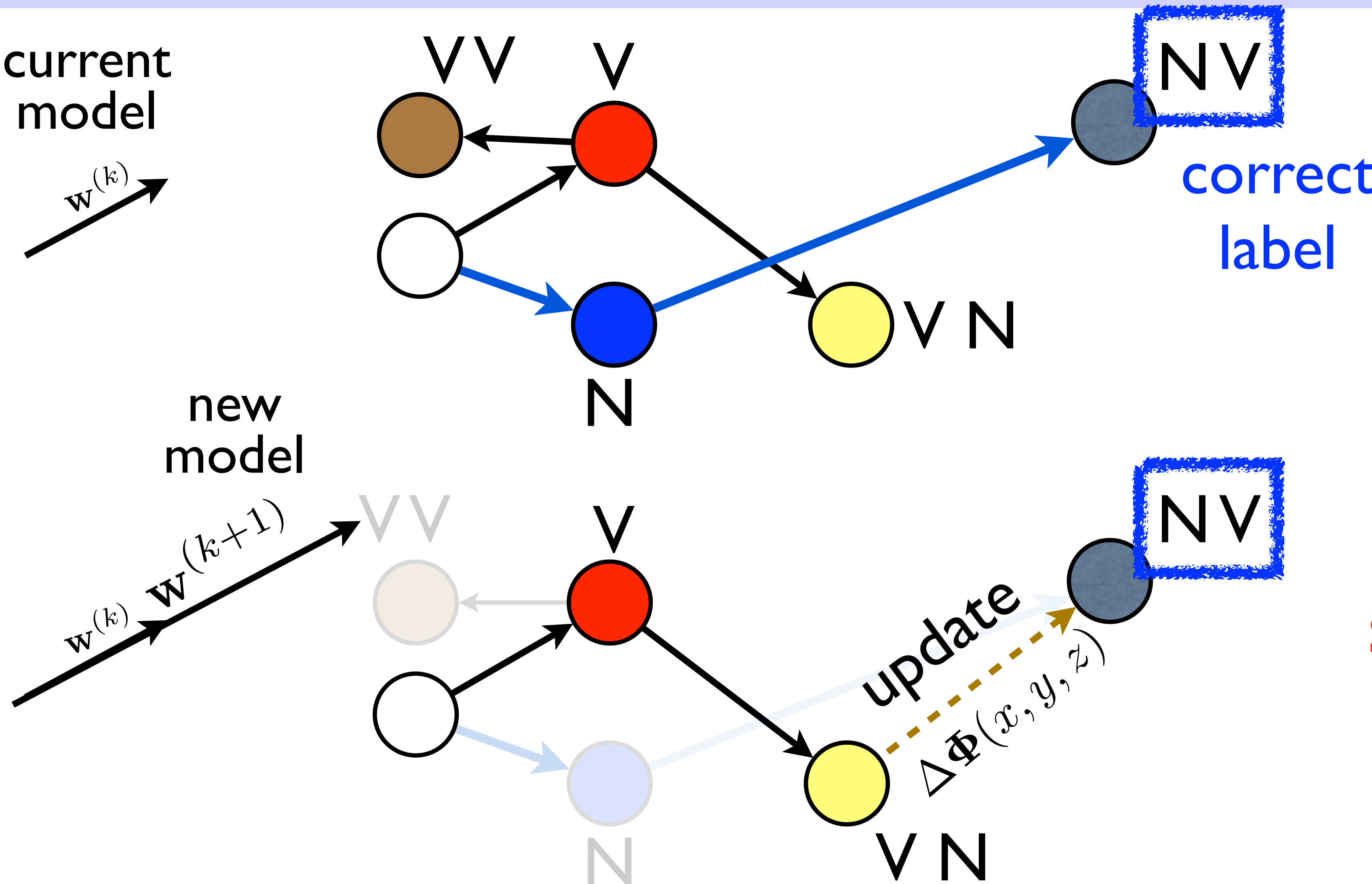
time flies

N V

output space

{N,V} x {N,V}

No Convergence w/ Greedy Search



training example

time flies

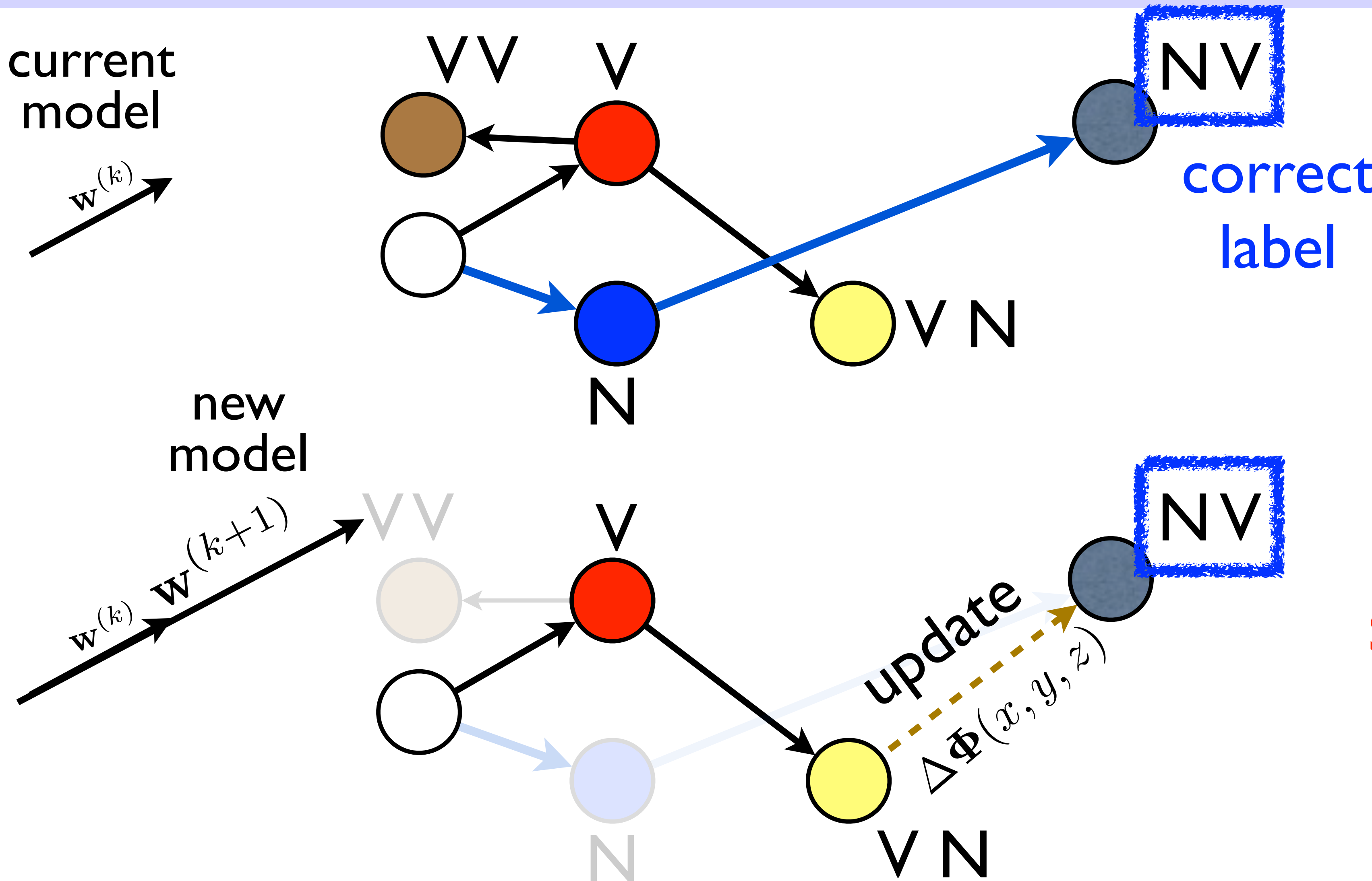
N V

output space

{N,V} x {N,V}

structured perceptron does not converge with greedy search

No Convergence w/ Greedy Search



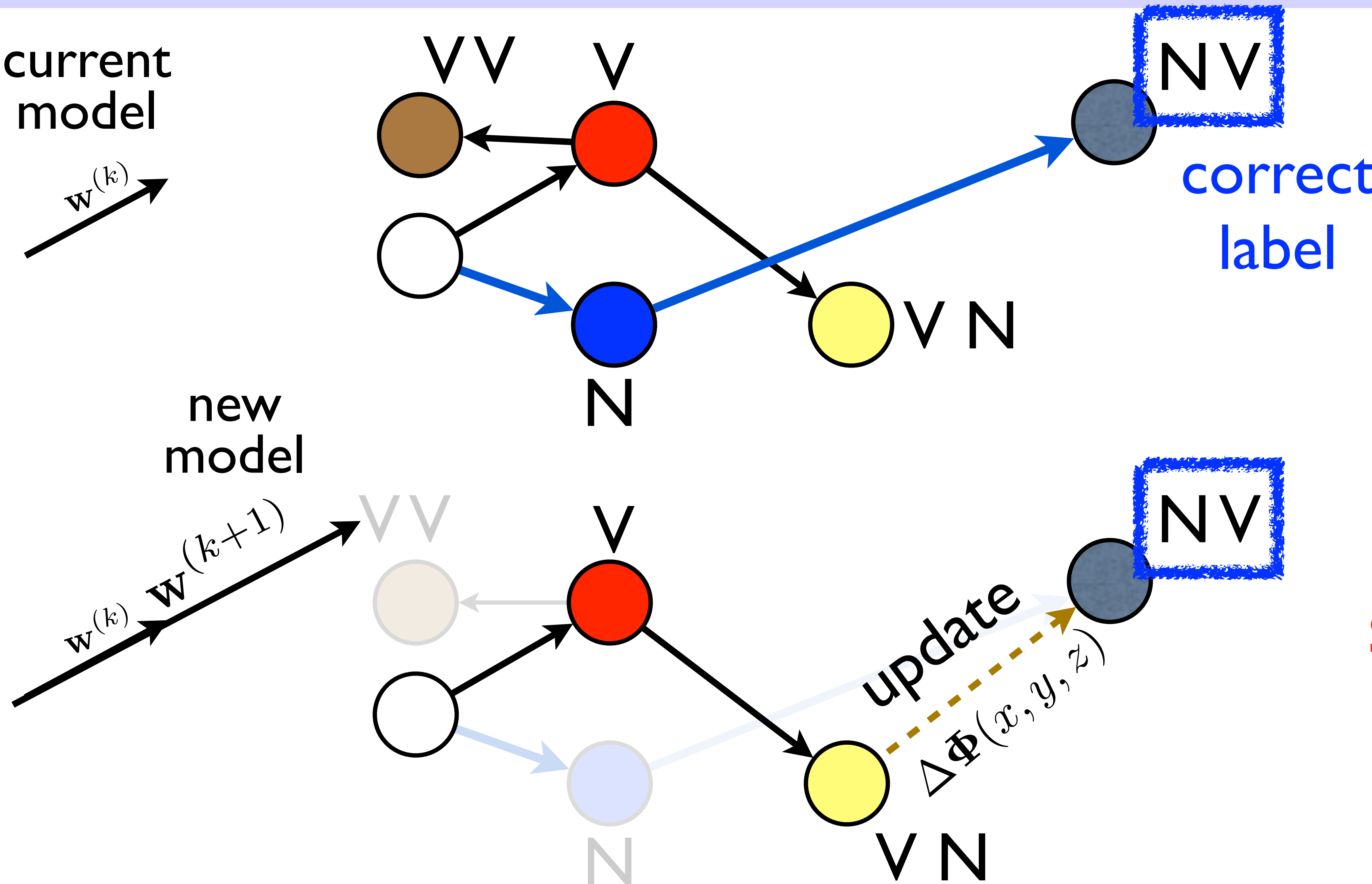
training example
time flies
N V

output space
{N,V} x {N,V}

structured perceptron
does not converge
with greedy search

Which part of the convergence proof no longer holds?

No Convergence w/ Greedy Search



training example

time flies

N V

output space

{N,V} x {N,V}

structured perceptron does not converge with greedy search

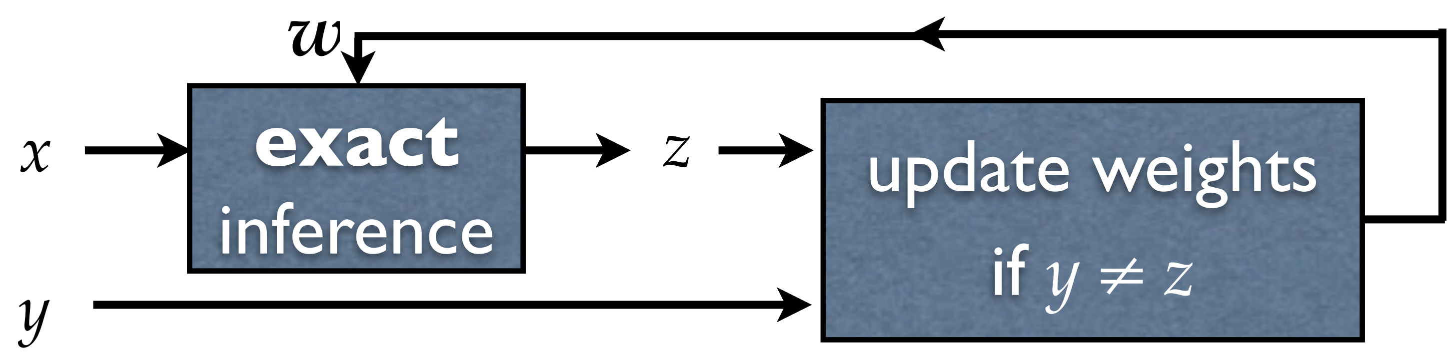
Which part of the convergence proof no longer holds?

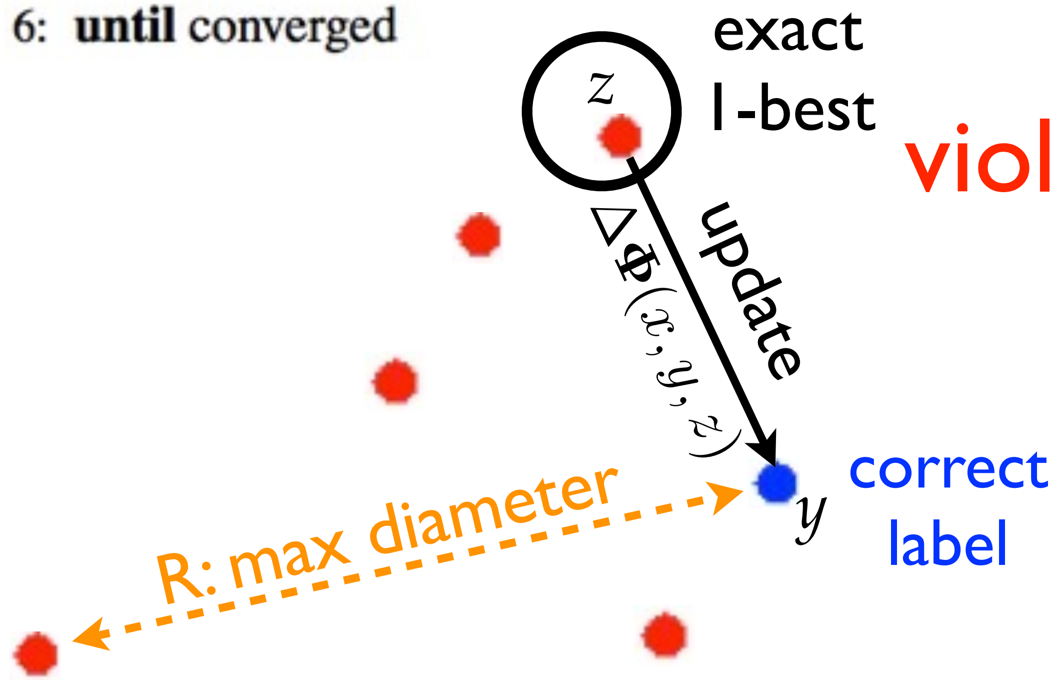
the proof only uses 3 facts:

1. separation (margin)
2. diameter (always finite)
3. violation (guaranteed by exact search)

Geometry of Convergence Proof pt 2

repeat
  for each example (x, y) **in** D **do**
    $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
    if $z \neq y$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
until converged





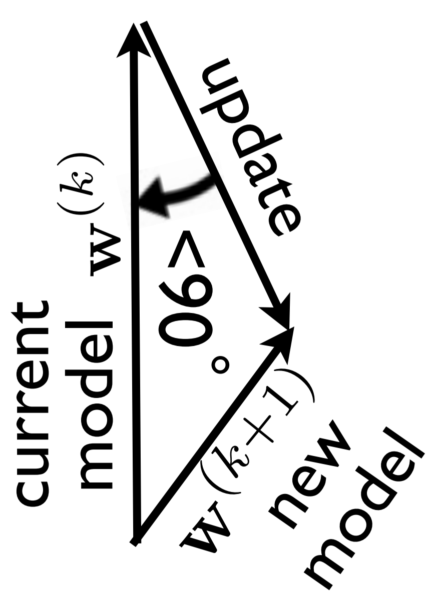
violation: incorrect label scored higher

perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

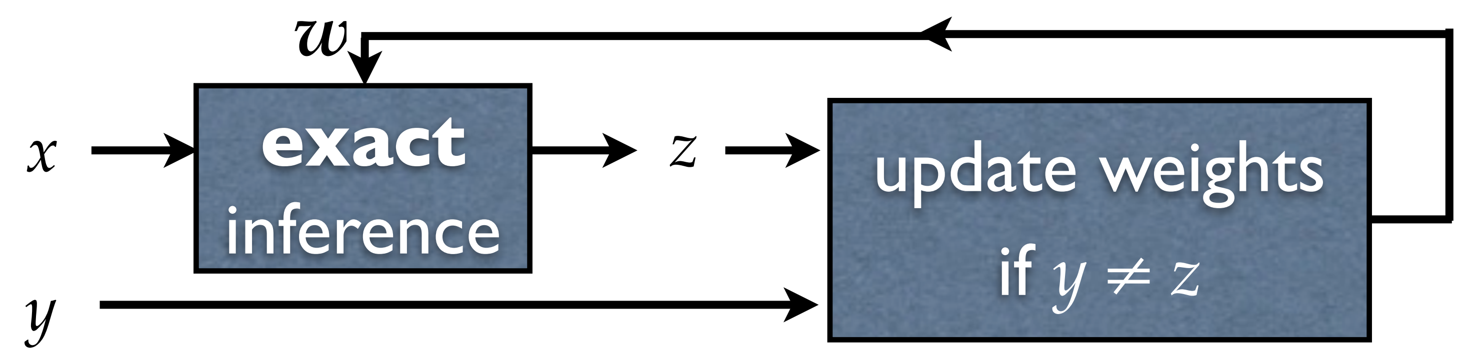
$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$

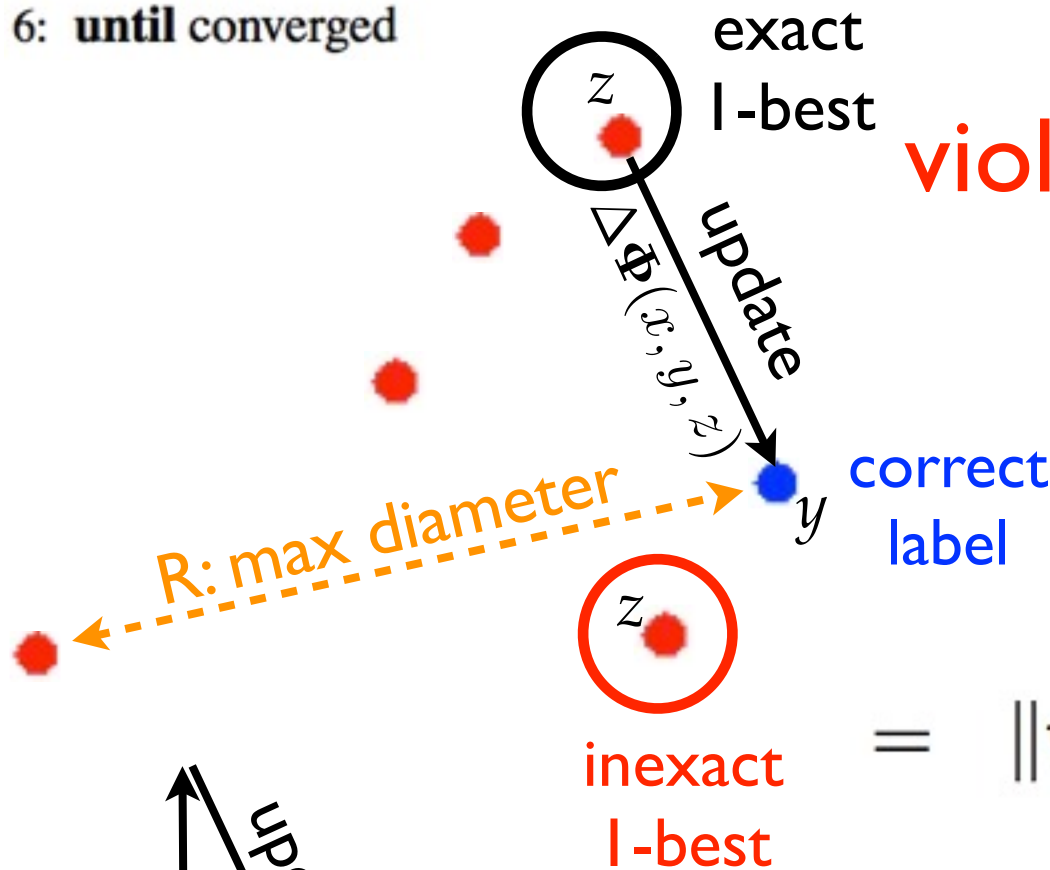
$$= \|\mathbf{w}^{(k)}\|^2 + \underbrace{\|\Delta\mathbf{\Phi}(x, y, z)\|^2}_{\leq R^2 \text{ diameter}} + 2\,\underbrace{\mathbf{w}^{(k)} \cdot \Delta\mathbf{\Phi}(x, y, z)}_{\leq 0 \text{ violation}}$$



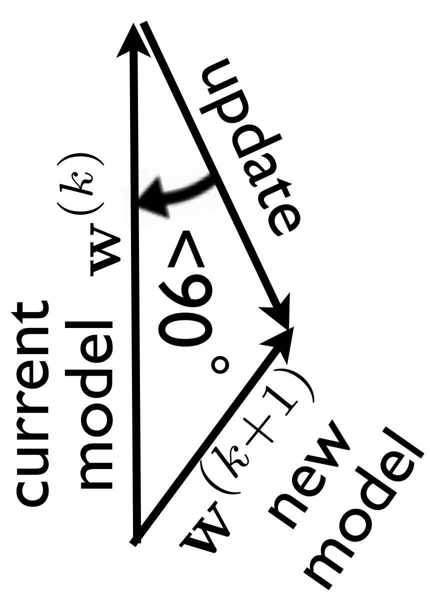
Geometry of Convergence Proof pt 2

```
repeat
    for each example (x, y) in D do
        z ← EXACT(x, w)
        if z ≠ y then
            w ← w + ΔΦ(x, y, z)
until converged
```





violation: incorrect label scored higher



perceptron update:

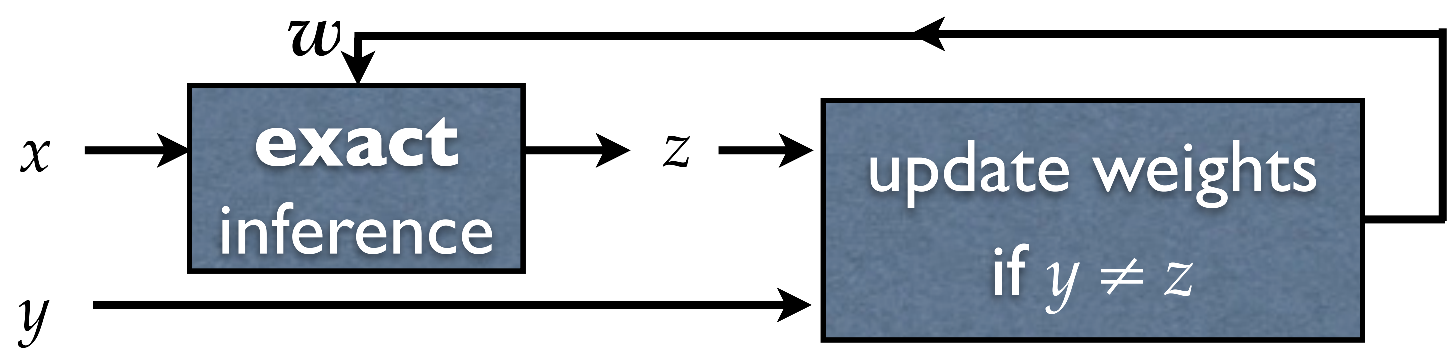
$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

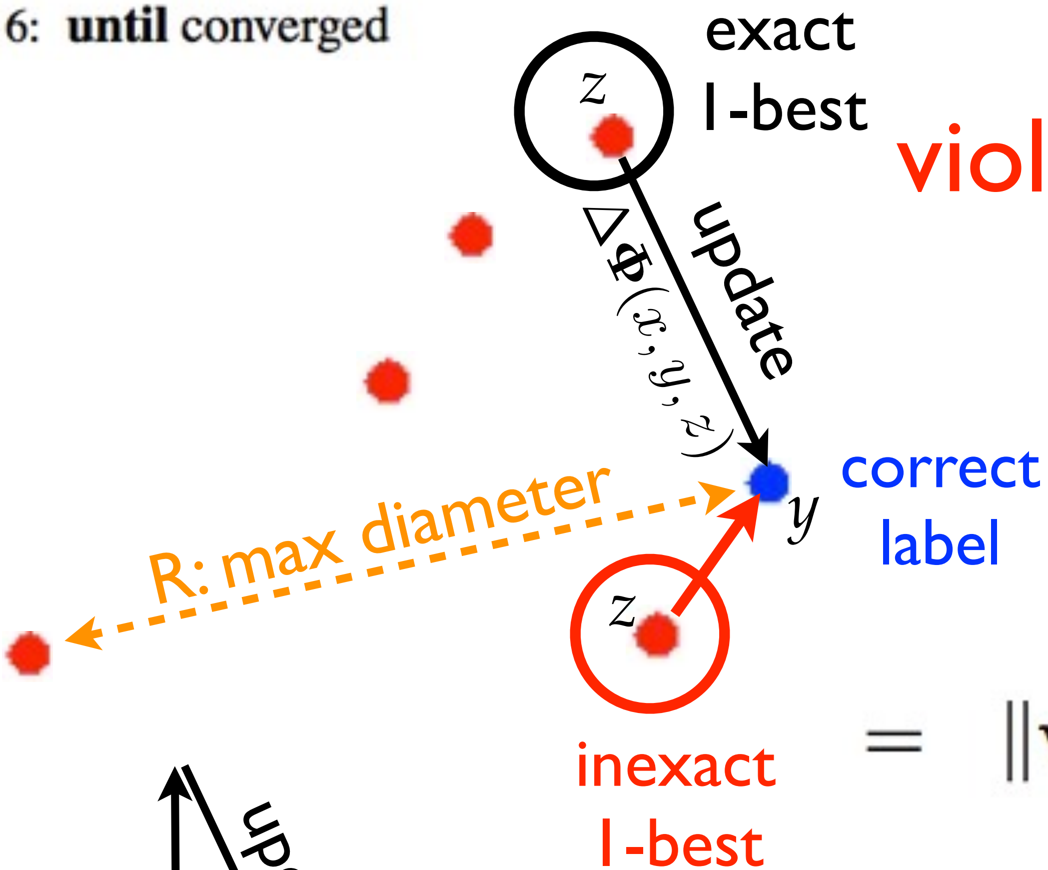
$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$

$$= \|\mathbf{w}^{(k)}\|^2 + \underbrace{\|\Delta\mathbf{\Phi}(x, y, z)\|^2}_{\le R^2 \ \text{diameter}} + \underbrace{2\, \mathbf{w}^{(k)} \cdot \Delta\mathbf{\Phi}(x, y, z)}_{\le 0 \ \text{violation}}$$

Geometry of Convergence Proof pt 2

```
repeat
    for each example (x, y) in D do
        z ← EXACT(x, w)
        if z ≠ y then
            w ← w + ΔΦ(x, y, z)
until converged
```





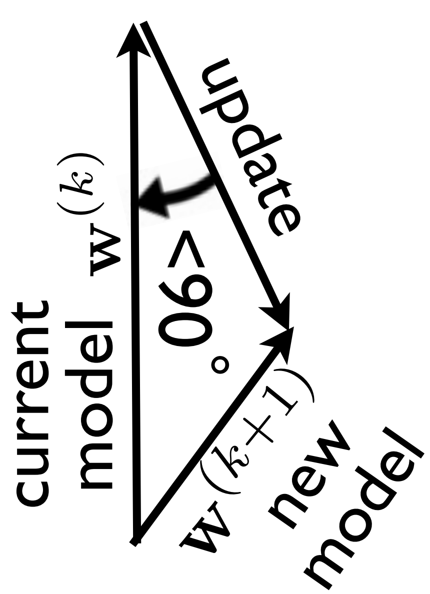
violation: incorrect label scored higher

perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\Phi(x,y,z)$$

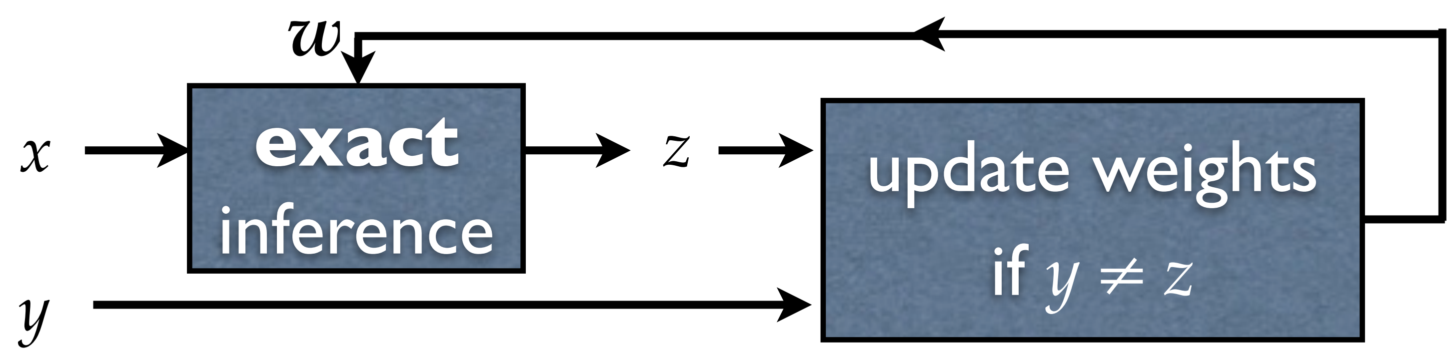
$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\Phi(x,y,z)\|^2$$

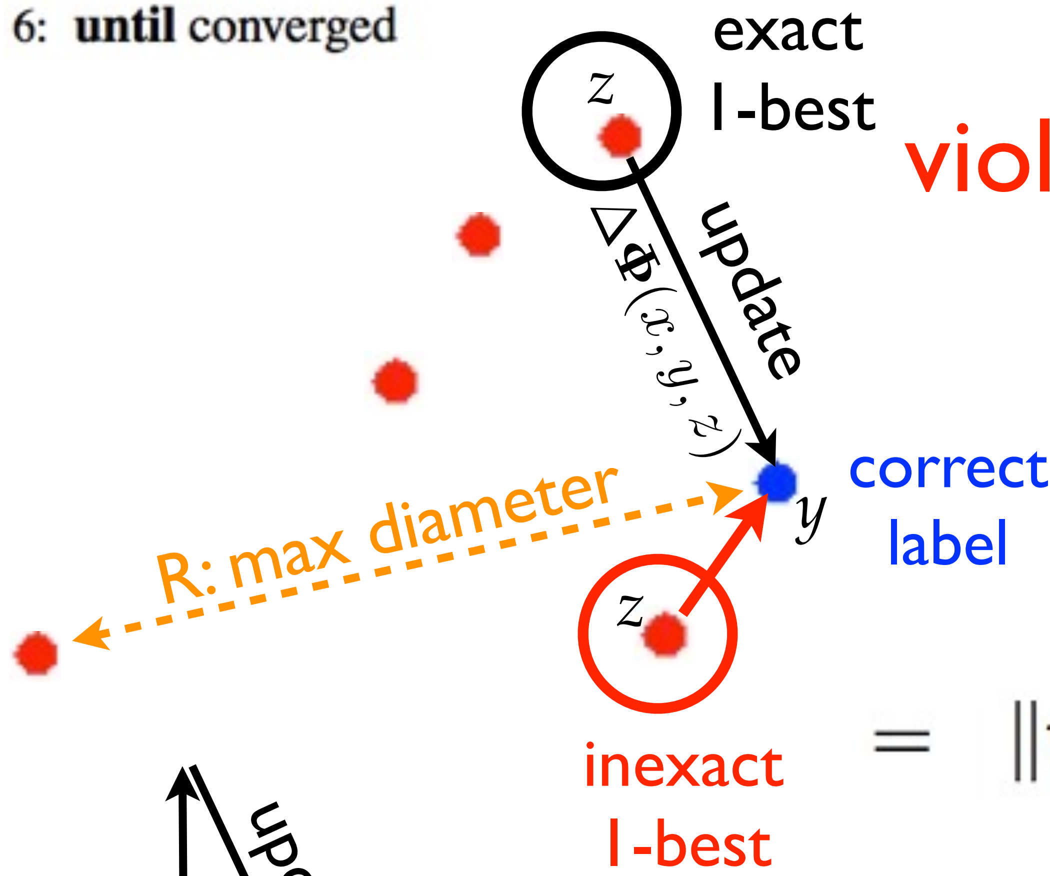
$$= \|\mathbf{w}^{(k)}\|^2 + \underbrace{\|\Delta\Phi(x,y,z)\|^2}_{\leq R^2 \text{ diameter}} + \underbrace{2\, \mathbf{w}^{(k)} \cdot \Delta\Phi(x,y,z)}_{\leq 0 \text{ violation}}$$



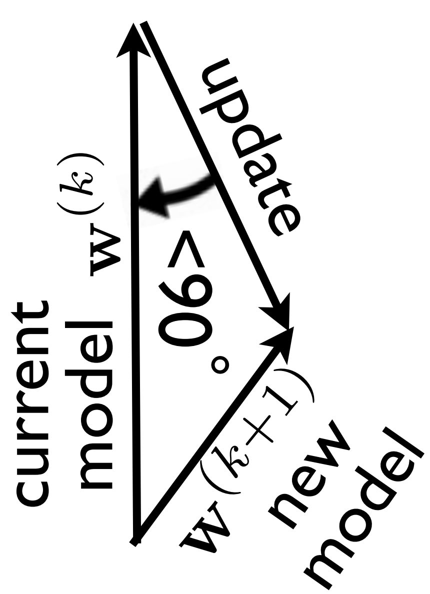
Geometry of Convergence Proof pt 2

1: **repeat**
2: **for each** example (x, y) **in** D **do**
3: $z \leftarrow \textsc{Exact}(x, \mathbf{w})$
4: **if** $z \neq y$ **then**
5: $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y, z)$
6: **until** converged





violation: incorrect label scored higher



perceptron update:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)$$

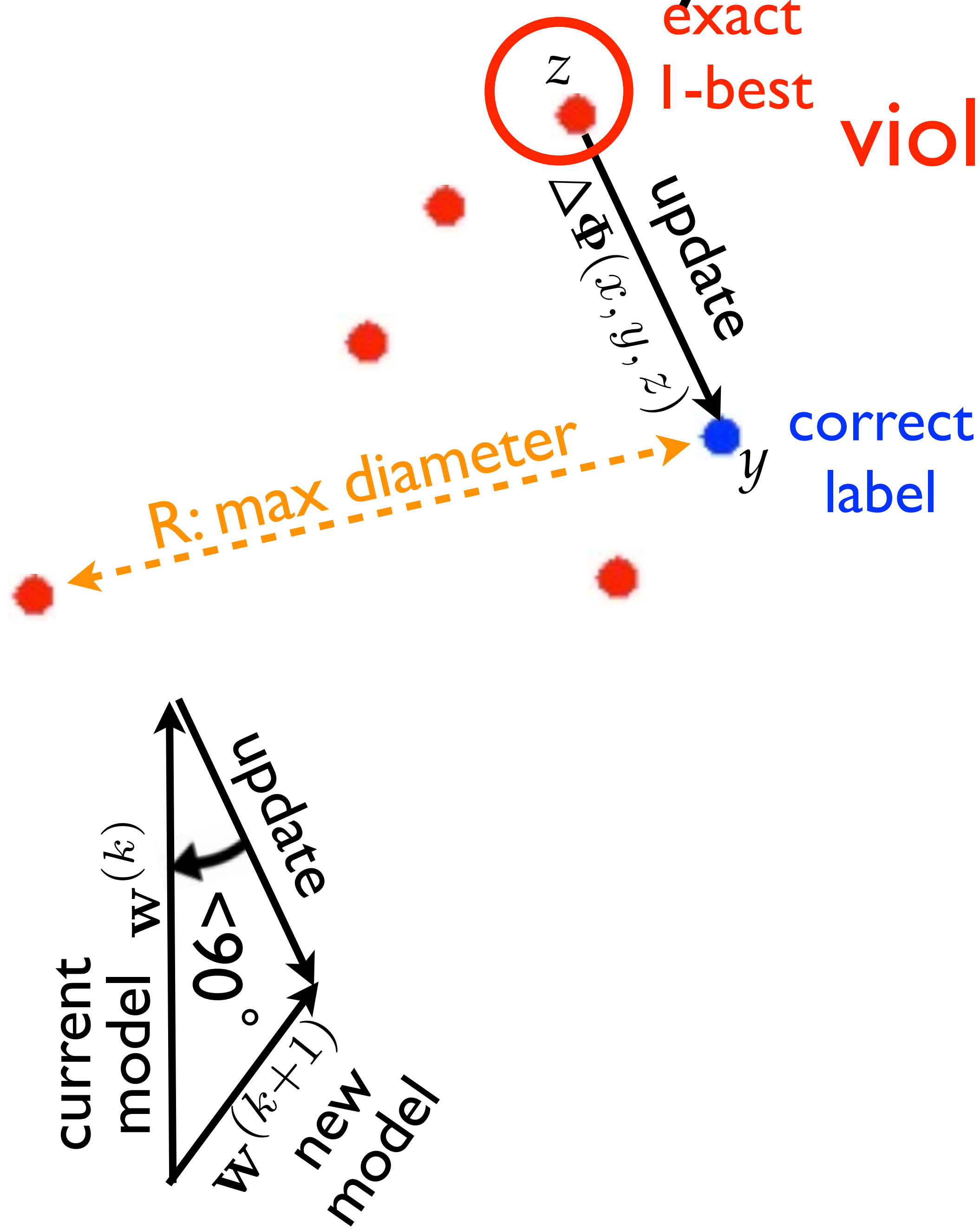
$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta\mathbf{\Phi}(x, y, z)\|^2$$

$$= \|\mathbf{w}^{(k)}\|^2 + \underbrace{\|\Delta\mathbf{\Phi}(x, y, z)\|^2}_{\le R^2 \text{ diameter}} + 2\,\underbrace{\mathbf{w}^{(k)} \cdot \Delta\mathbf{\Phi}(x, y, z)}_{\le 0 \text{ violation}}$$

inexact search doesn't guarantee violation!

Observation: Violation is all we need!

- exact search is **not** really required by the proof
  - rather, it is only used to ensure violation!



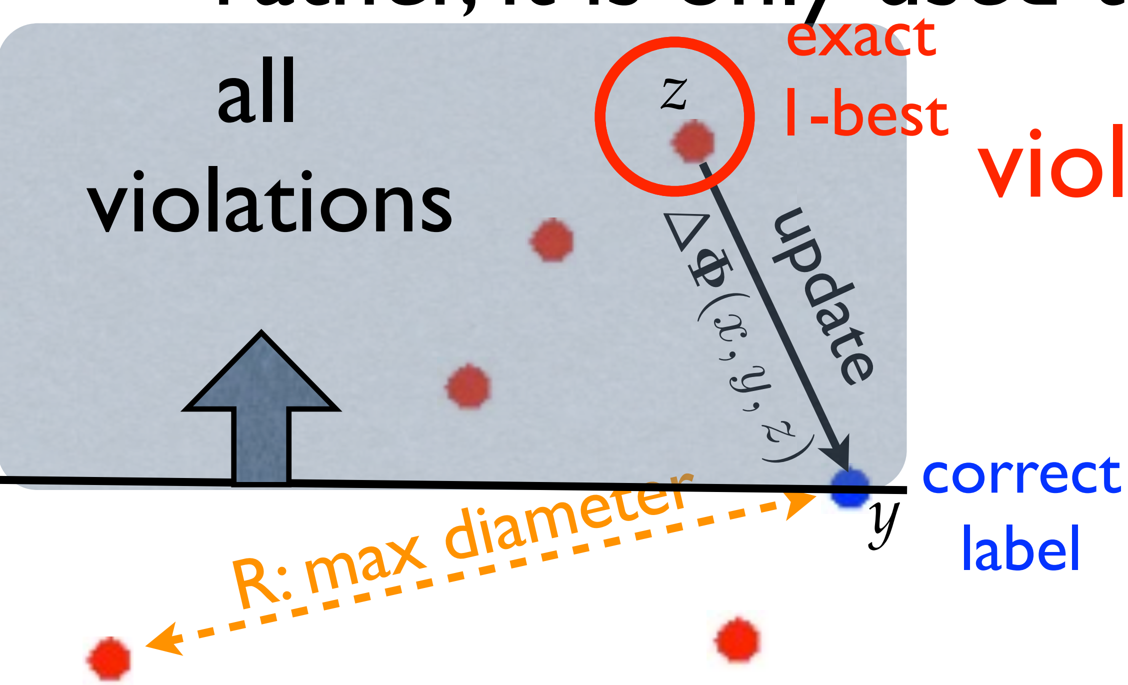
violation: incorrect label scored higher

the proof only uses 3 facts:

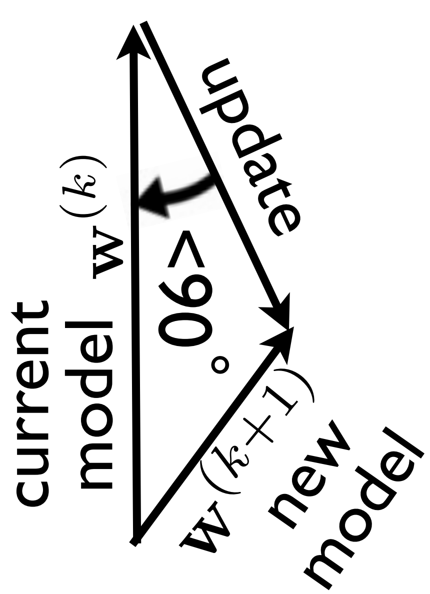
1. separation (margin)
2. diameter (always finite)
3. violation (but no need for exact)

Observation: Violation is all we need!

- exact search is **not** really required by the proof
  - rather, it is only used to ensure violation!



violation: incorrect label scored higher

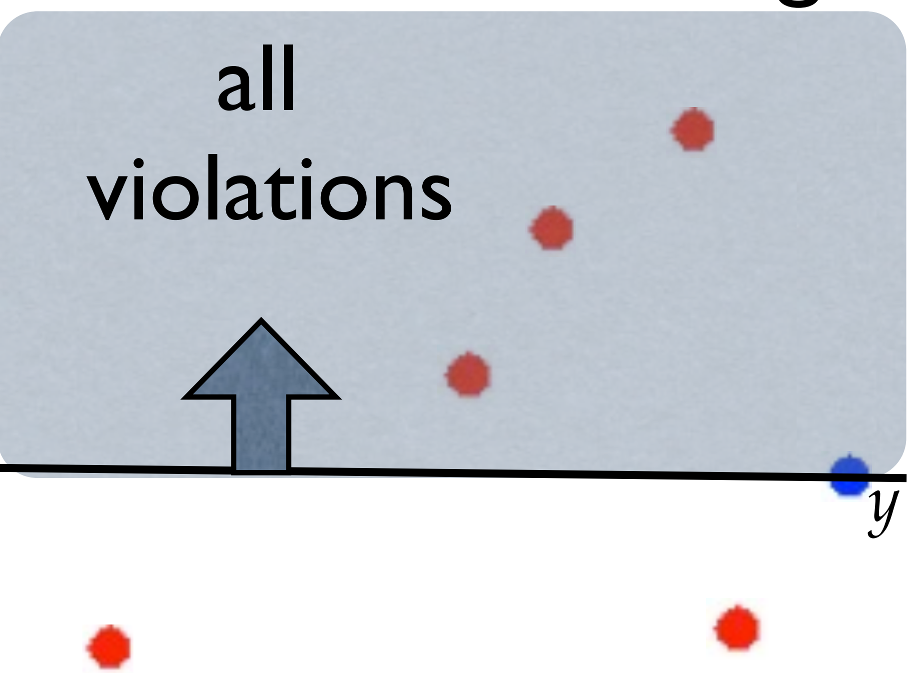


the proof only uses 3 facts:

1. separation (margin)
2. diameter (always finite)
3. violation (but no need for exact)

Violation-Fixing Perceptron

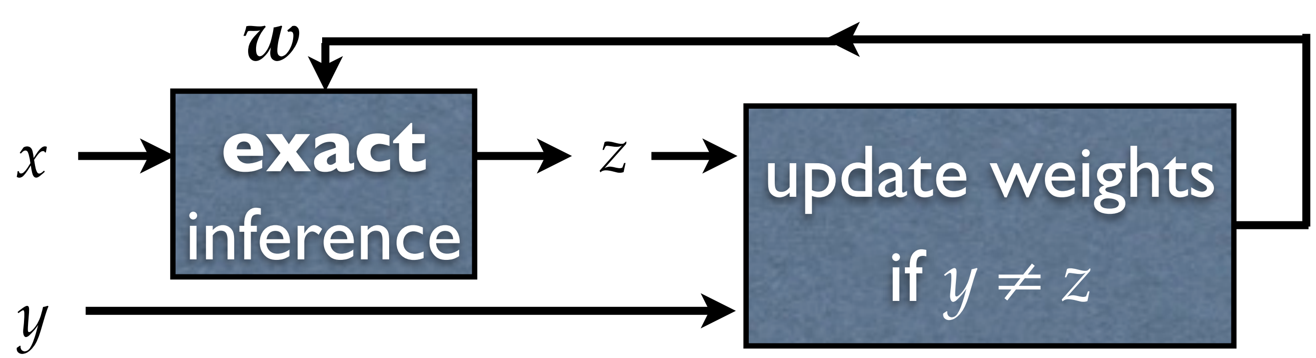
- if we guarantee violation, we don't care about exactness!
  - violation is good b/c we can at least fix a mistake



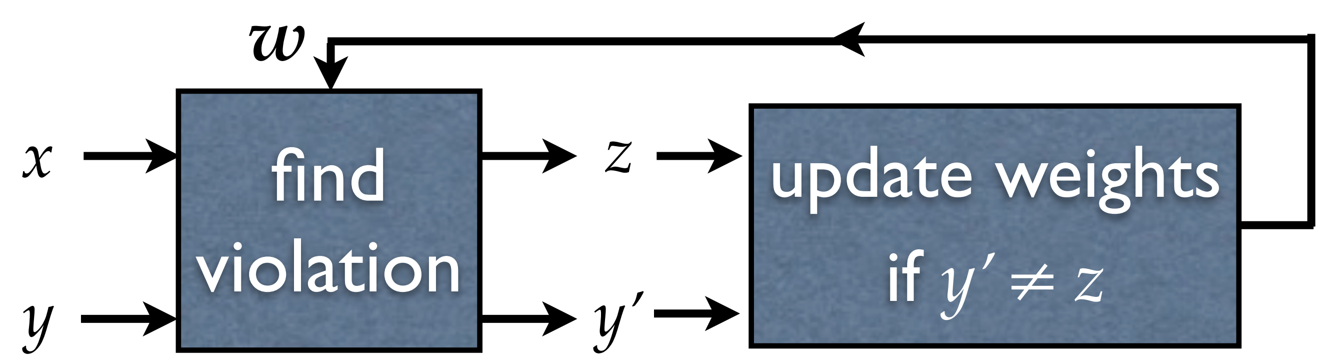
same mistake bound as before!

repeat
  for each example (x, y) **in** D **do**
    $(x, y', z) = \textsc{FindViolation}(x, y, \mathbf{w})$
    if $z \neq y$ **then** $\triangleright$ (x, y', z) is a viol
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\Phi(x, y', z)$
until converged

standard perceptron

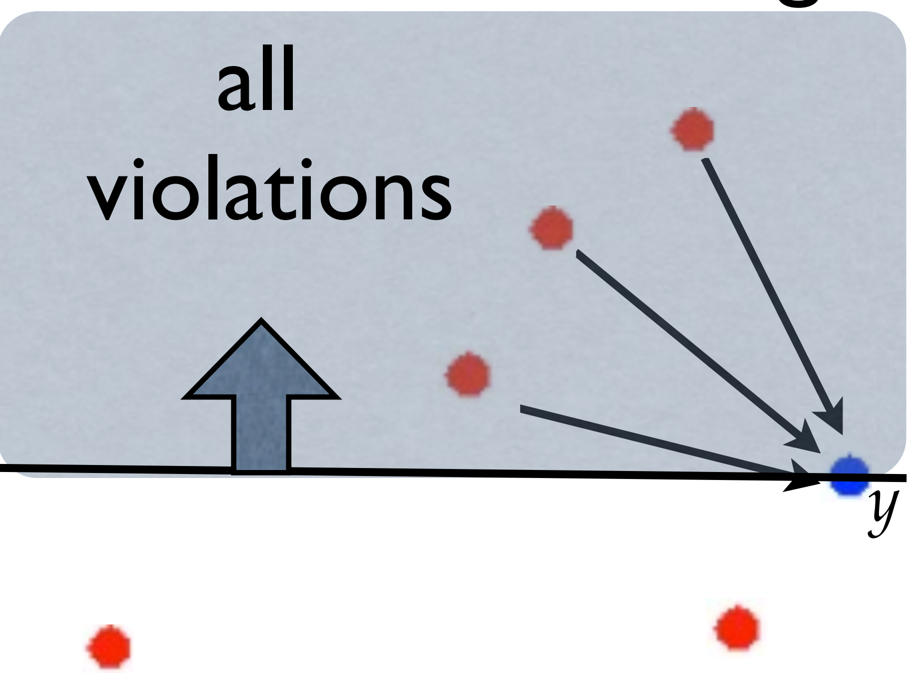


violation-fixing perceptron



Violation-Fixing Perceptron

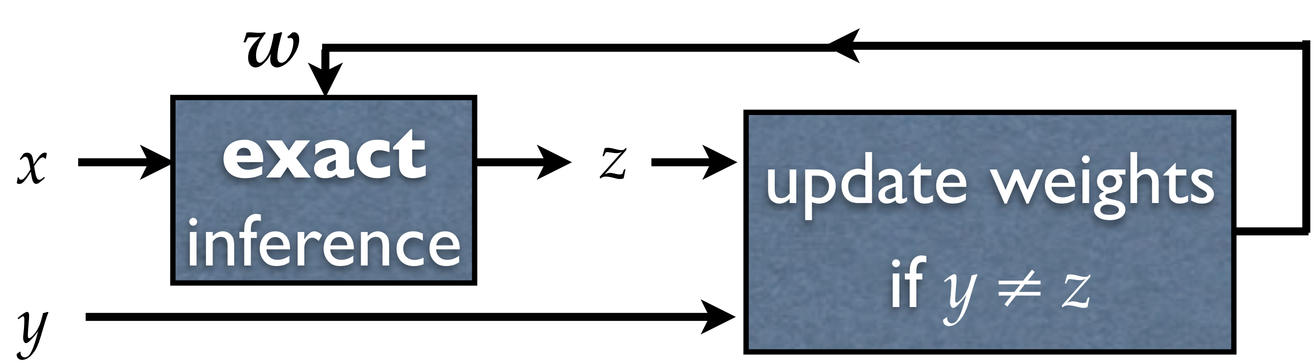
- if we guarantee violation, we don't care about exactness!
  - violation is good b/c we can at least fix a mistake



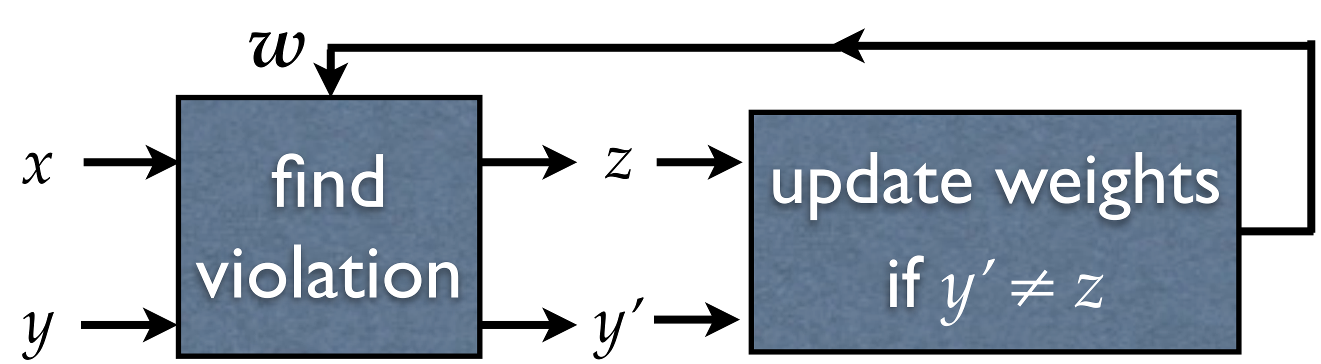
same mistake bound as before!

repeat
  for each example (x, y) **in** D **do**
    $(x, y', z) = \textsc{FindViolation}(x, y, \mathbf{w})$
    if $z \neq y$ **then** $\triangleright$ (x, y', z) is a viol
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\Phi(x, y', z)$
until converged

standard perceptron

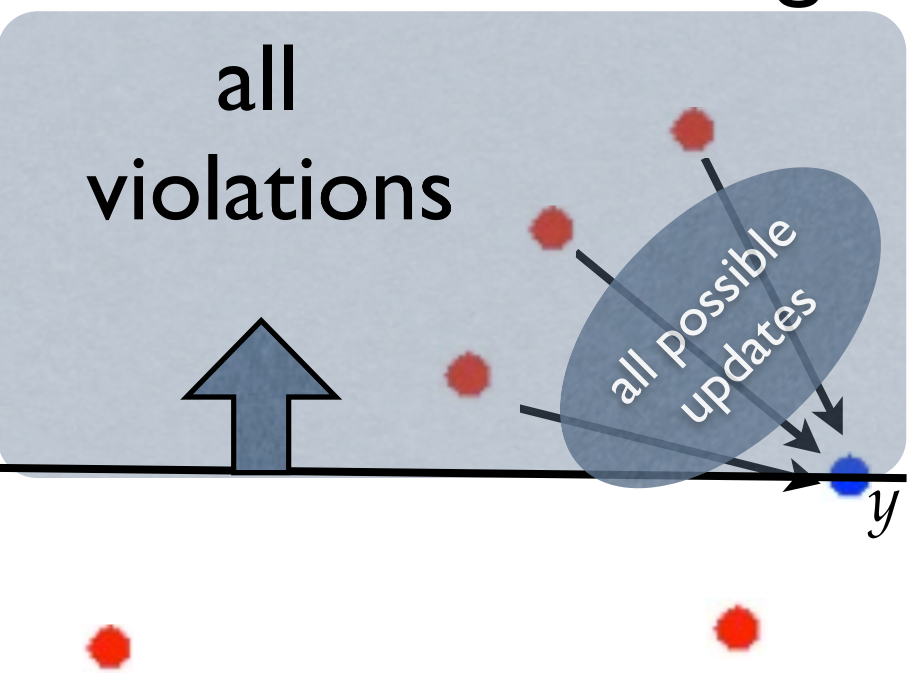


violation-fixing perceptron



Violation-Fixing Perceptron

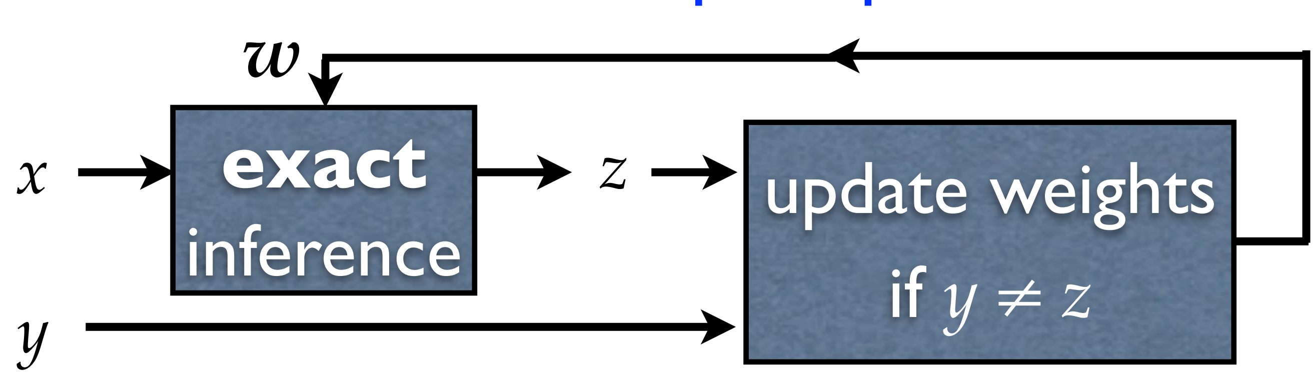
- if we guarantee violation, we don't care about exactness!
  - violation is good b/c we can at least fix a mistake



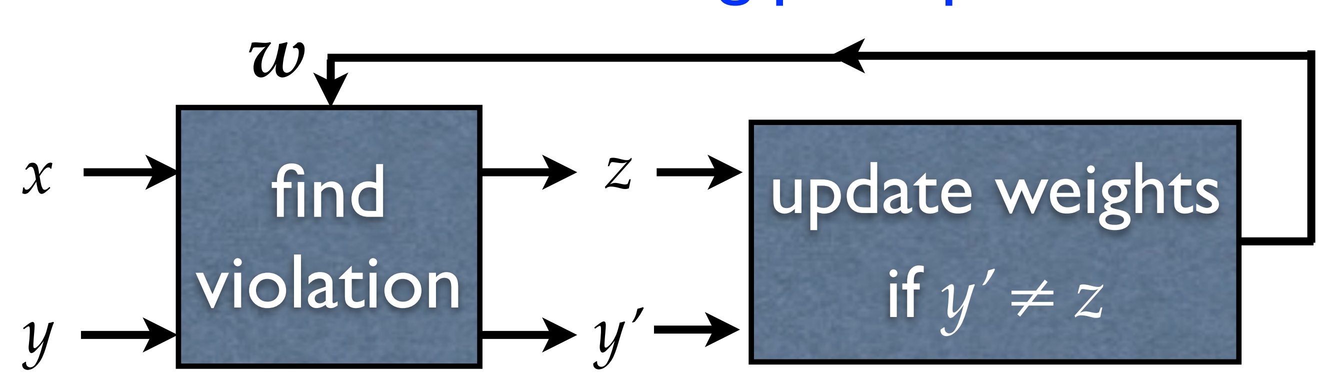
same mistake bound as before!

repeat
  for each example (x, y) **in** D **do**
    $(x, y', z) = \text{FINDVIOLATION}(x, y, \mathbf{w})$
    if $z \neq y$ **then** $\triangleright$ (x, y', z) is a viol
      $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, y', z)$
until converged

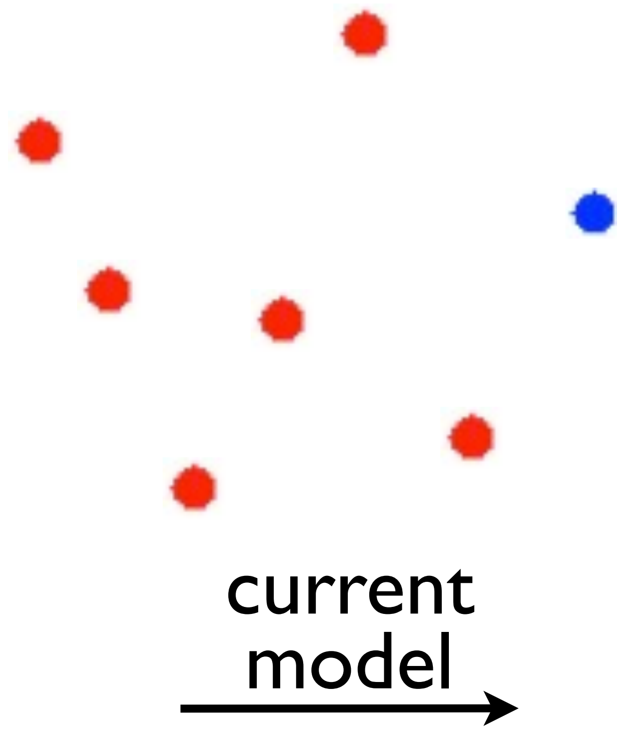
standard perceptron



violation-fixing perceptron

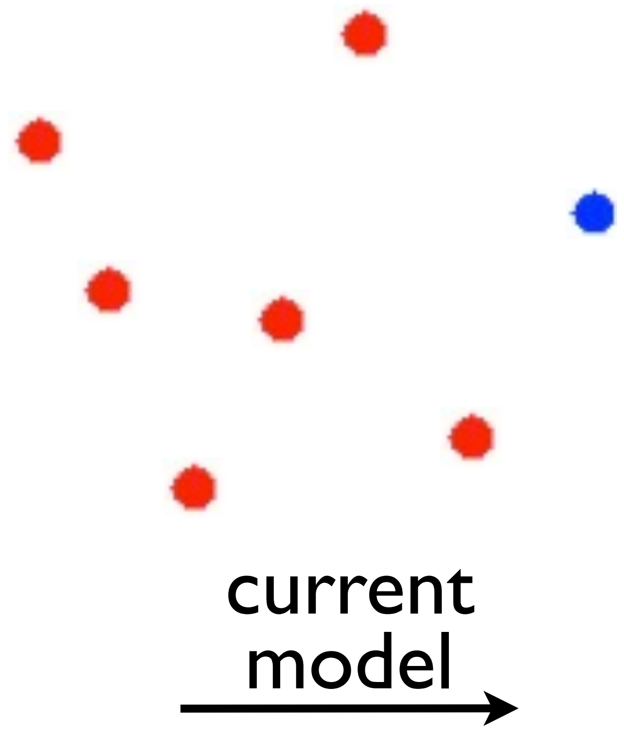


What if can't guarantee violation



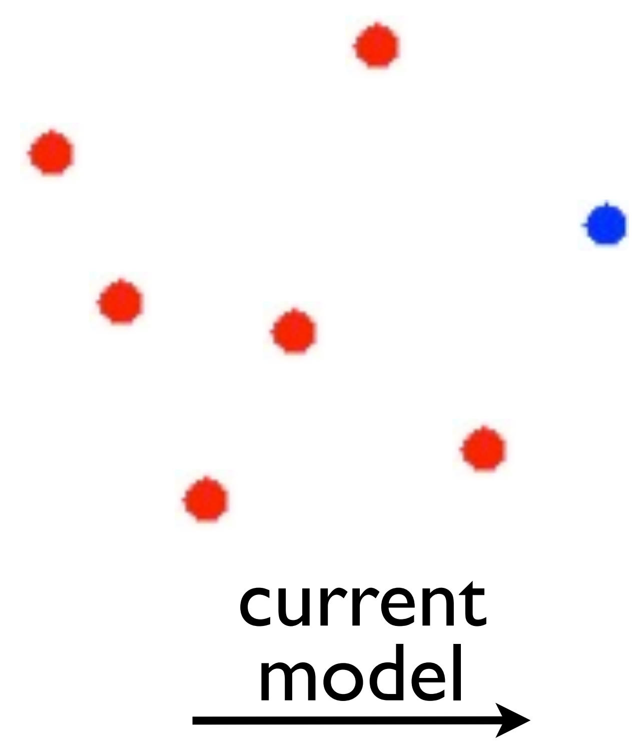
What if can't guarantee violation

- this is why perceptron doesn't work well w/ inexact search
  - because not every update is guaranteed to be a violation
  - thus the proof breaks; no convergence guarantee



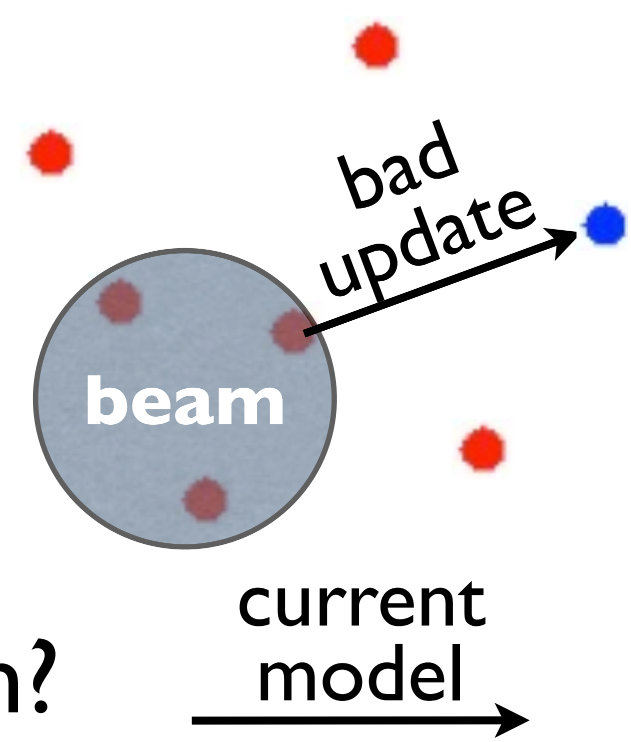
What if can't guarantee violation

- this is why perceptron doesn't work well w/ inexact search
  - because not every update is guaranteed to be a violation
  - thus the proof breaks; no convergence guarantee
- example: beam or greedy search
  - the model might prefer the correct label (if exact search)
  - but the search prunes it away
  - such a non-violation update is "bad"
    because it doesn't fix any mistake
  - the new model still misguides the search

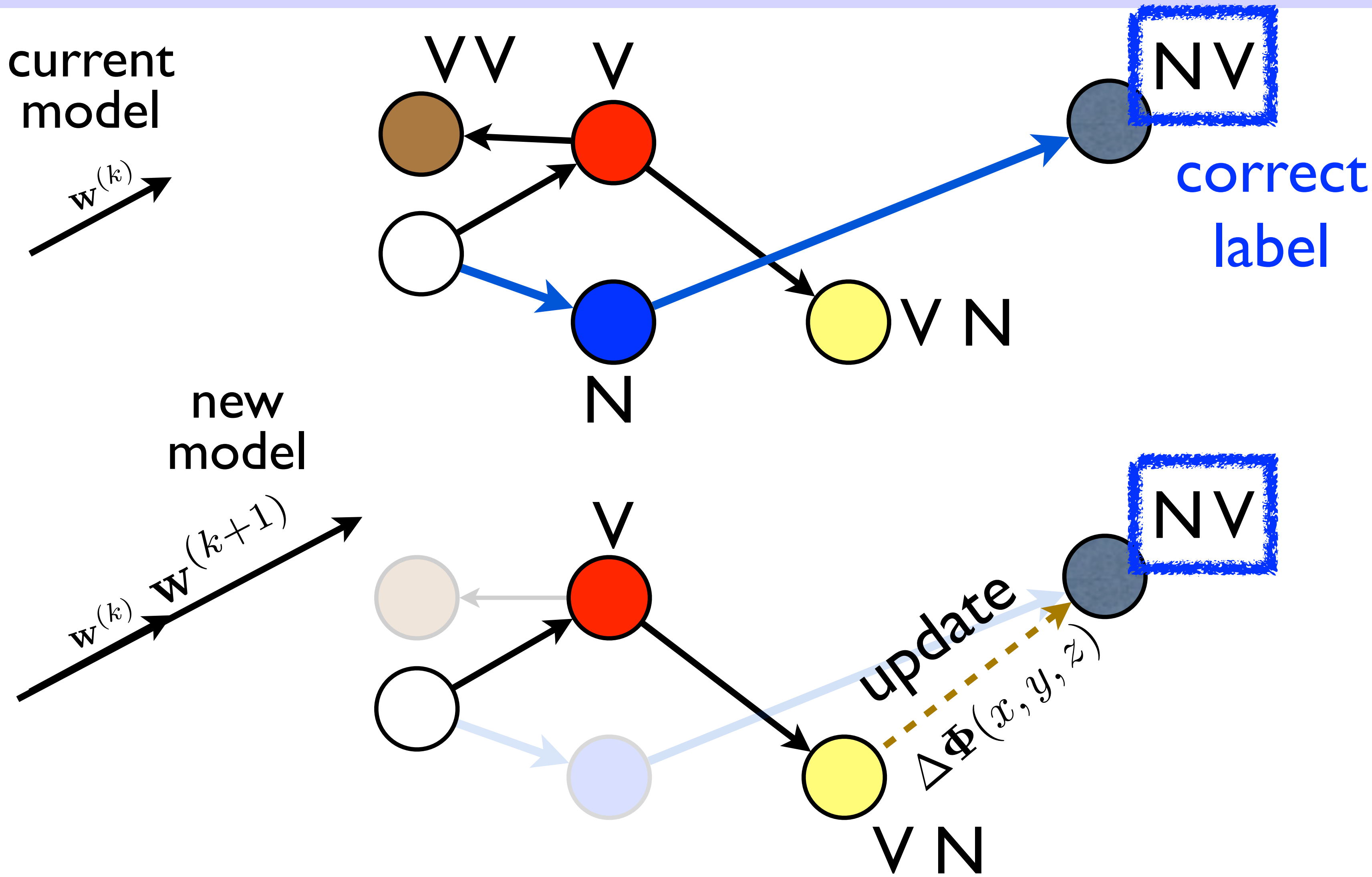


What if can't guarantee violation

- this is why perceptron doesn't work well w/ inexact search
  - because not every update is guaranteed to be a violation
  - thus the proof breaks; no convergence guarantee
- example: beam or greedy search
  - the model might prefer the correct label (if exact search)
  - but the search prunes it away
  - such a non-violation update is "bad" because it doesn't fix any mistake
  - the new model still misguides the search
- Q: how can we always guarantee violation?



Solution 1: Early update (Collins/Roark 2004)



training example

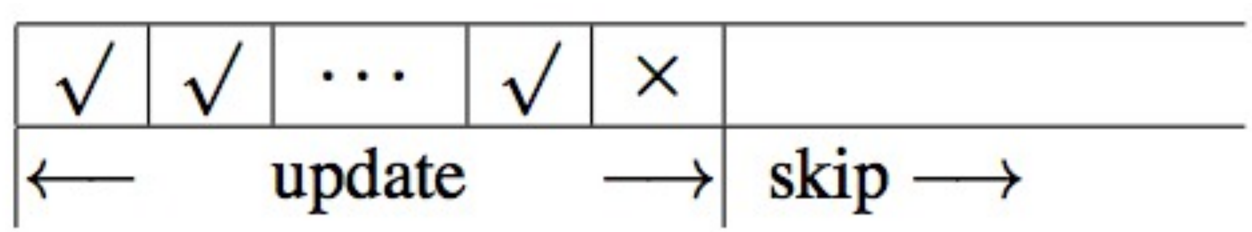
time flies

N V

output space

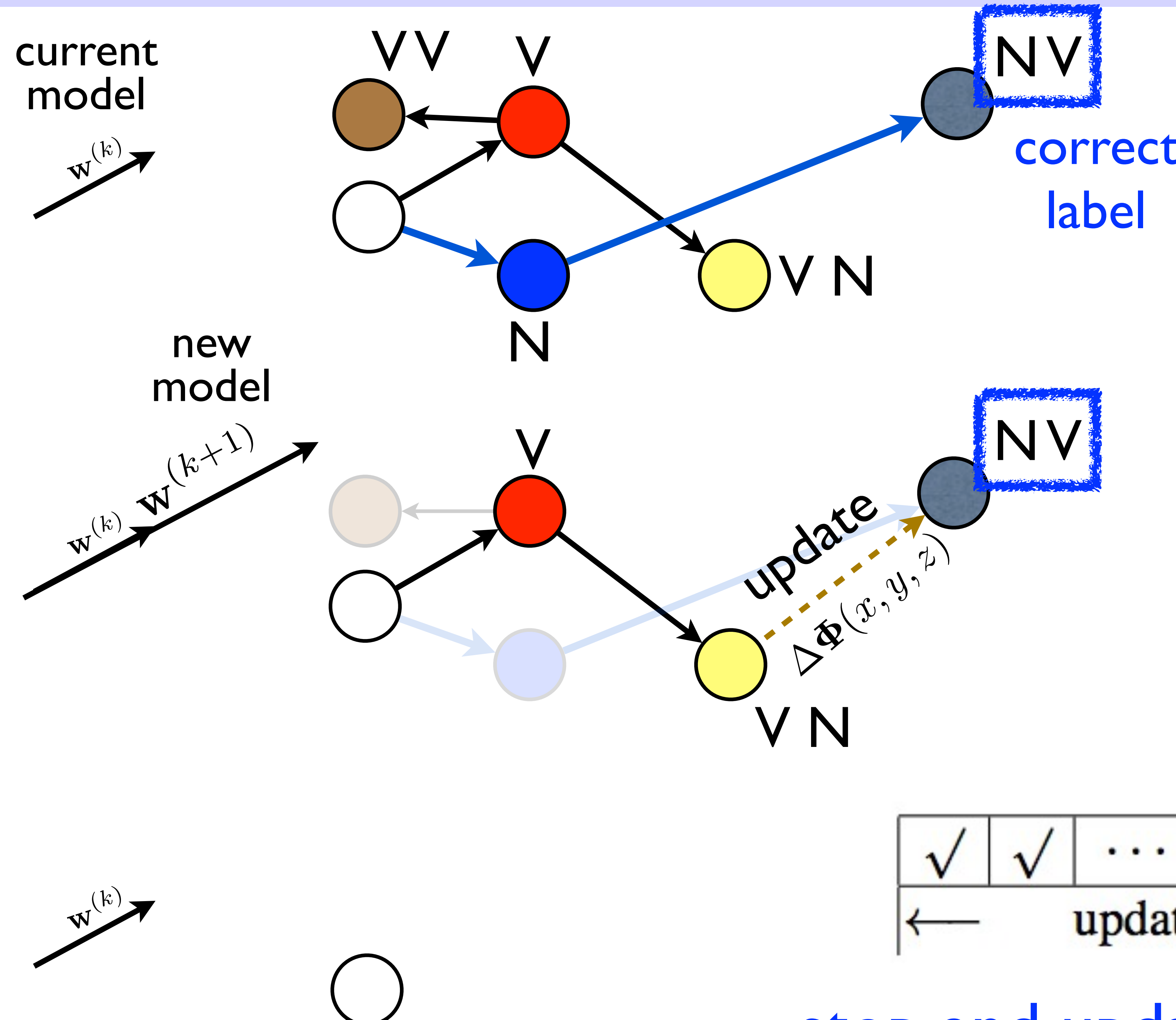
{N,V} x {N,V}

standard perceptron does not converge with greedy search



stop and update at the first mistake

Solution 1: Early update (Collins/Roark 2004)



training example

time	flies
N	V

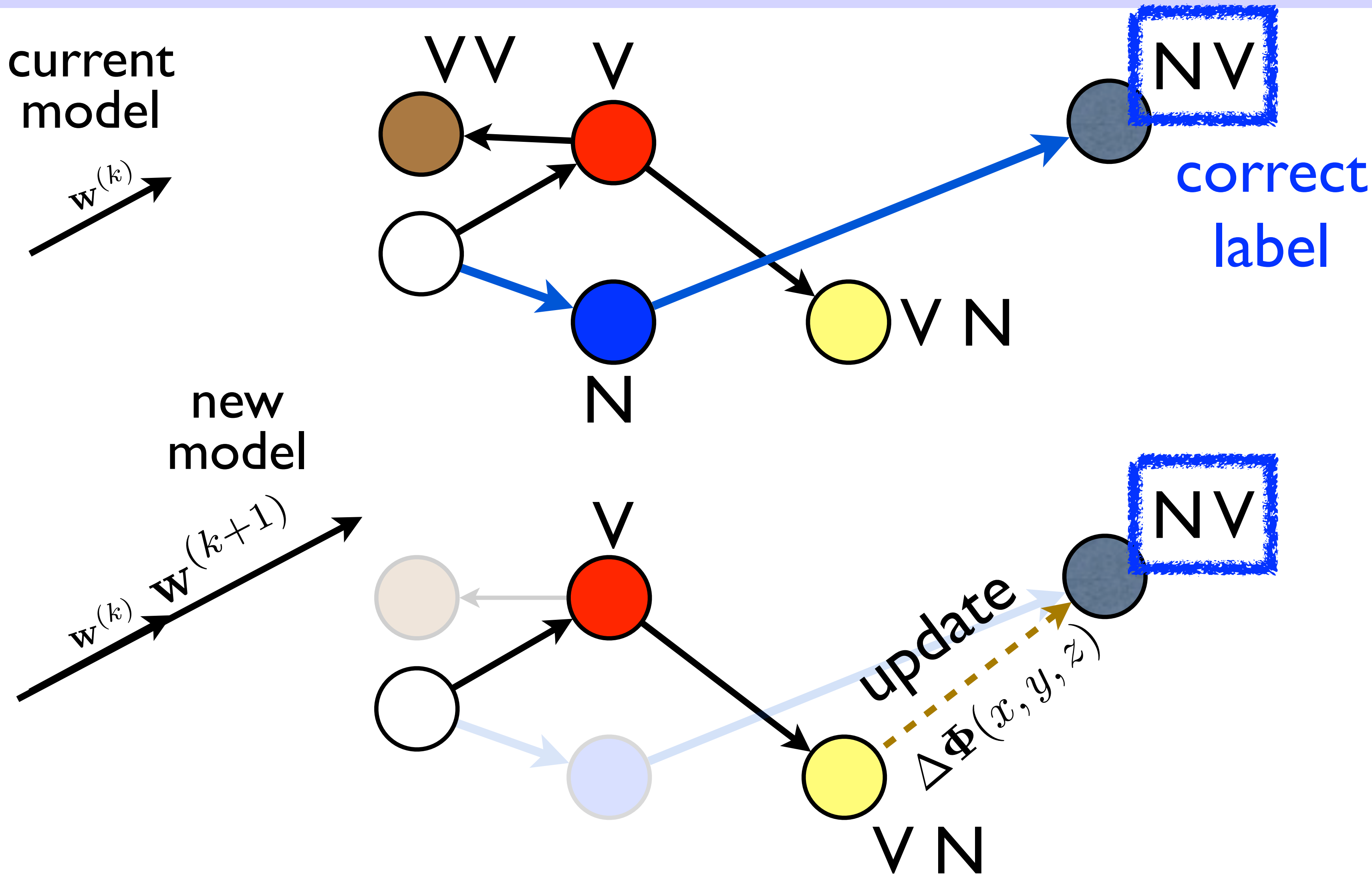
output space
{N,V} x {N,V}

standard perceptron does not converge with greedy search

$\surd$	$\surd$	$\cdots$	$\surd$	$\times$	
$\longleftarrow$ update $\longrightarrow$					skip $\longrightarrow$

stop and update at the first mistake

Solution 1: Early update (Collins/Roark 2004)



training example

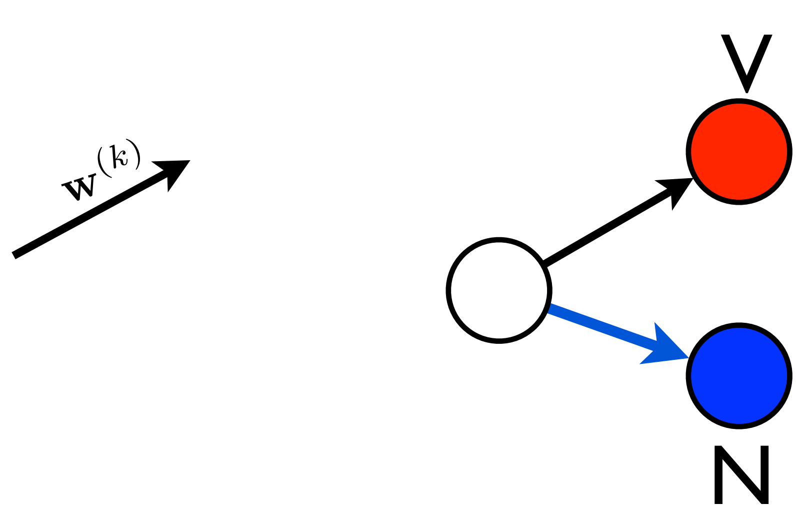
time flies

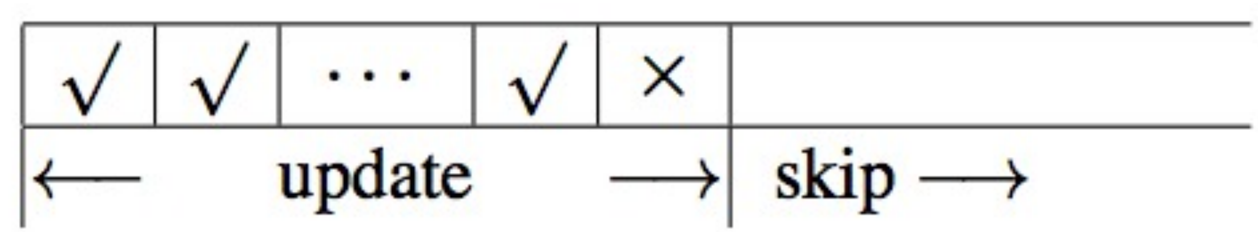
N V

output space

{N,V} x {N,V}

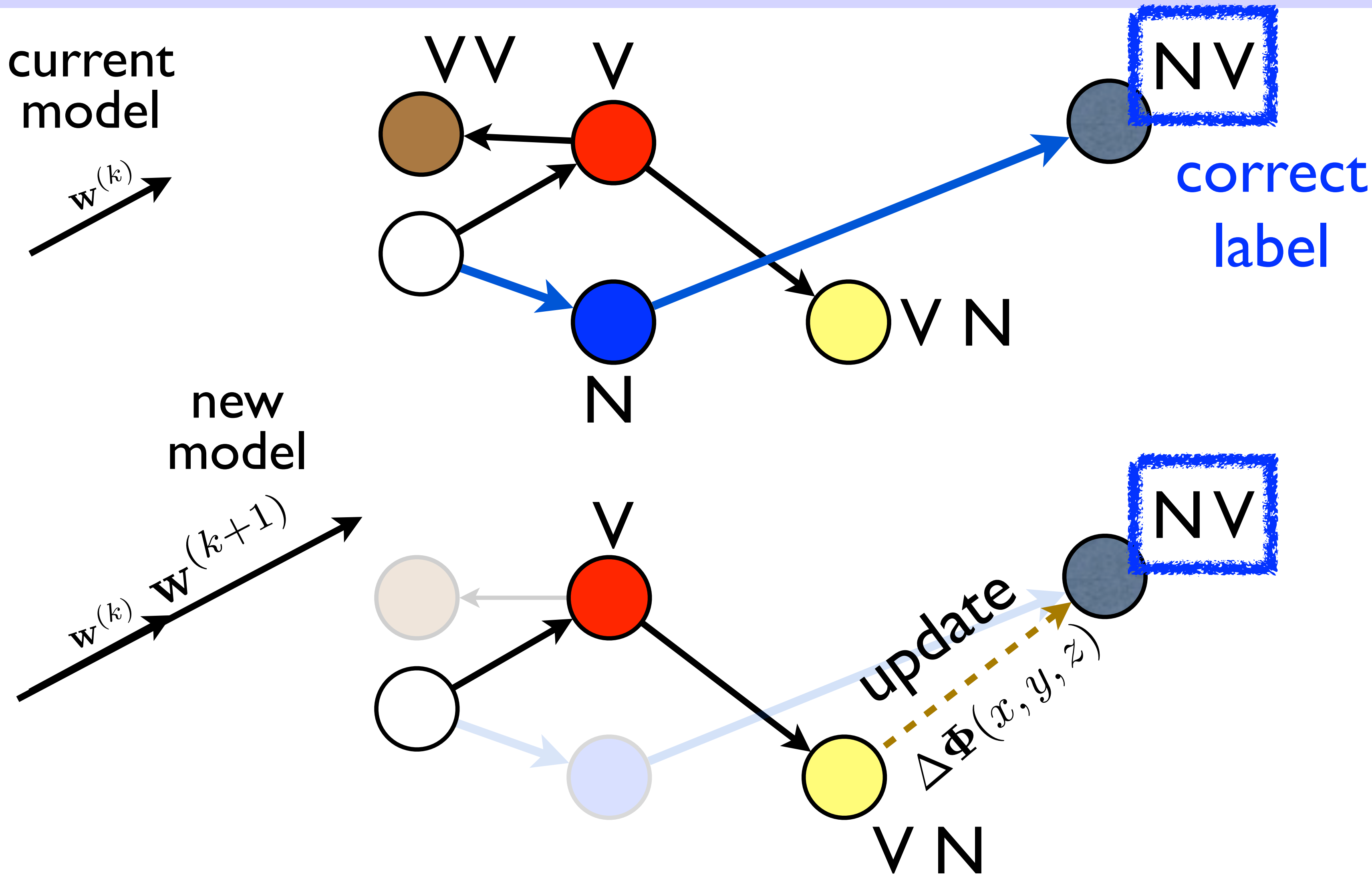
standard perceptron does not converge with greedy search





stop and update at the first mistake

Solution 1: Early update (Collins/Roark 2004)



training example

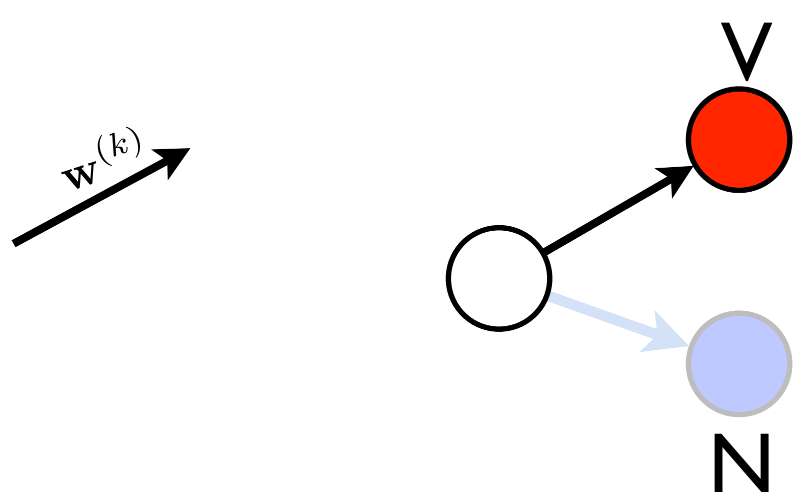
time flies

N V

output space

{N,V} x {N,V}

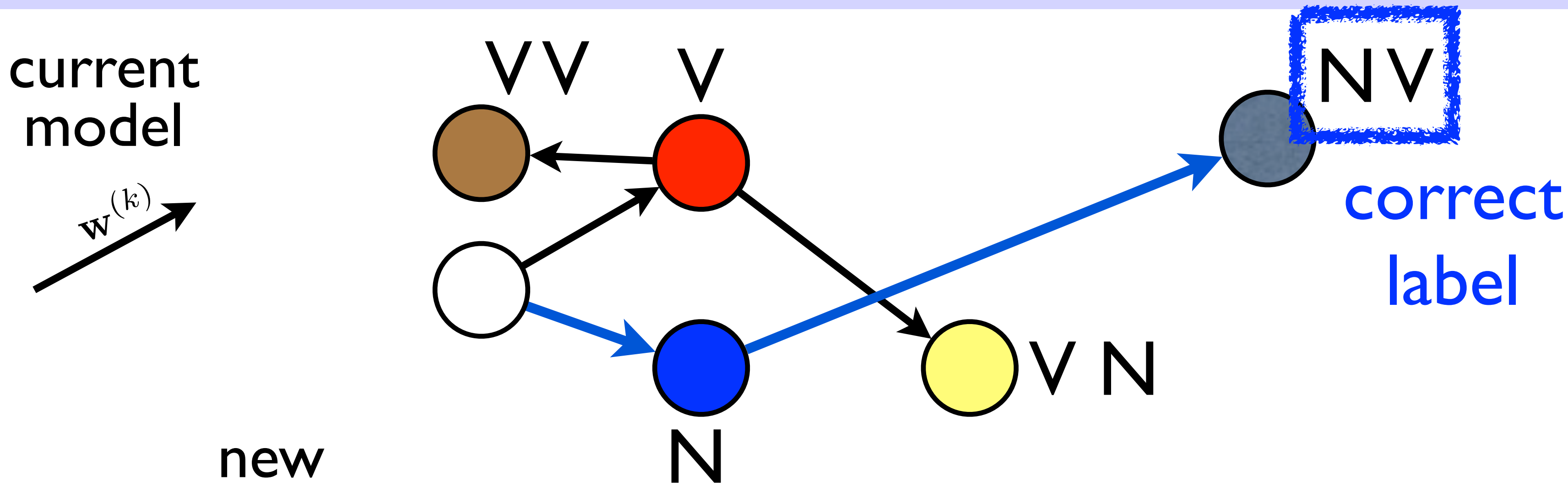
standard perceptron does not converge with greedy search



$\surd$	$\surd$	$\cdots$	$\surd$	$\times$	
$\longleftarrow$	update			$\longrightarrow$	skip $\longrightarrow$

stop and update at the first mistake

Solution 1: Early update (Collins/Roark 2004)



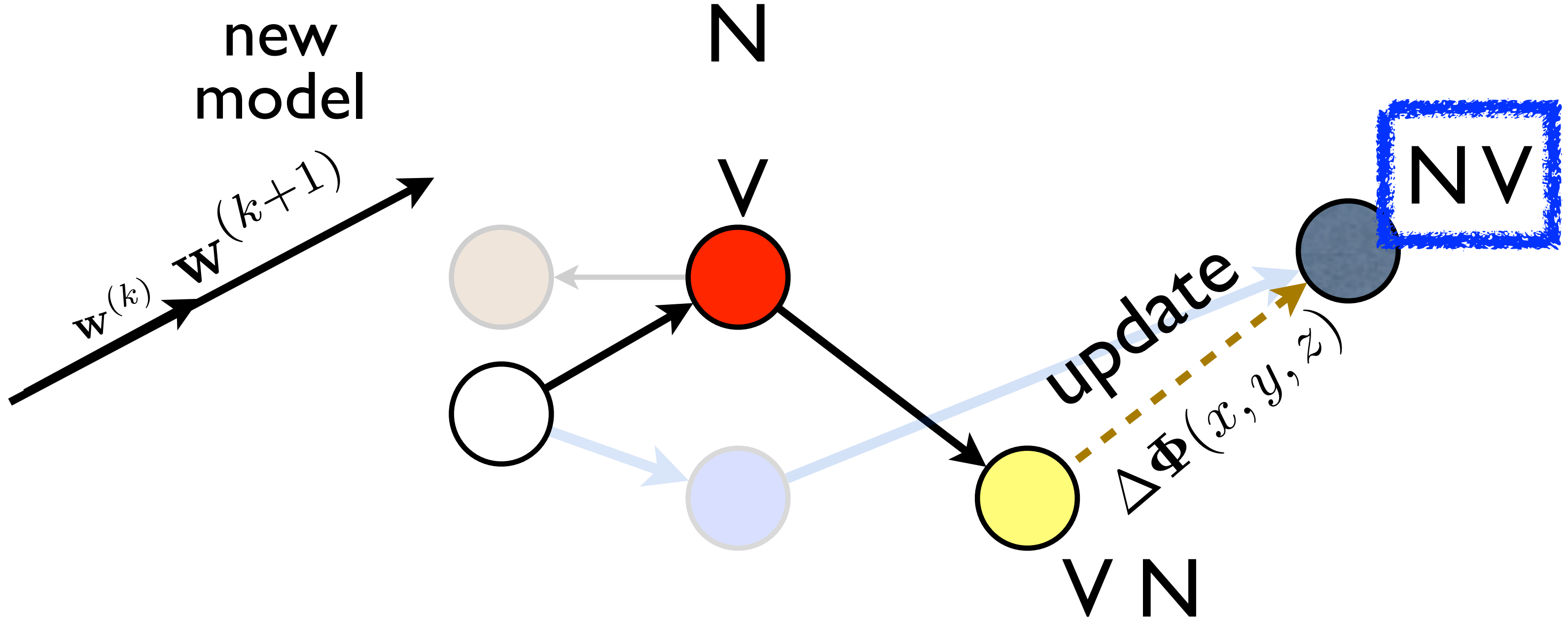
training example

time flies

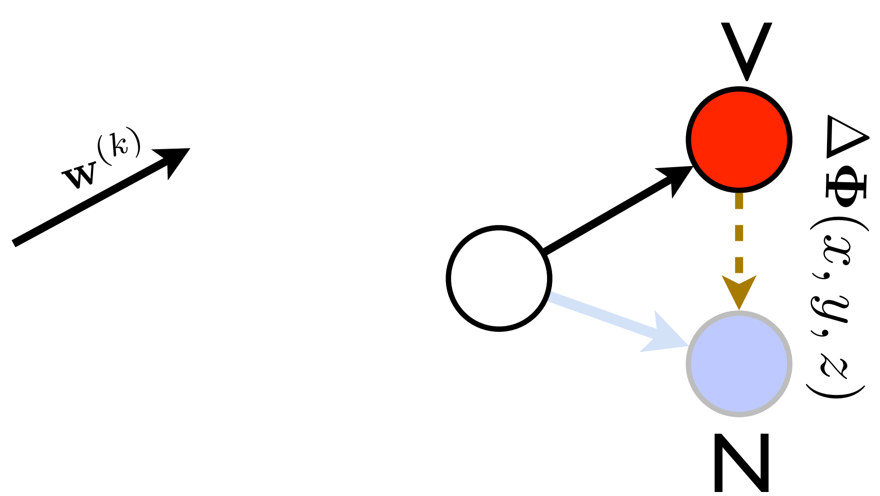
N V

output space

{N,V} x {N,V}



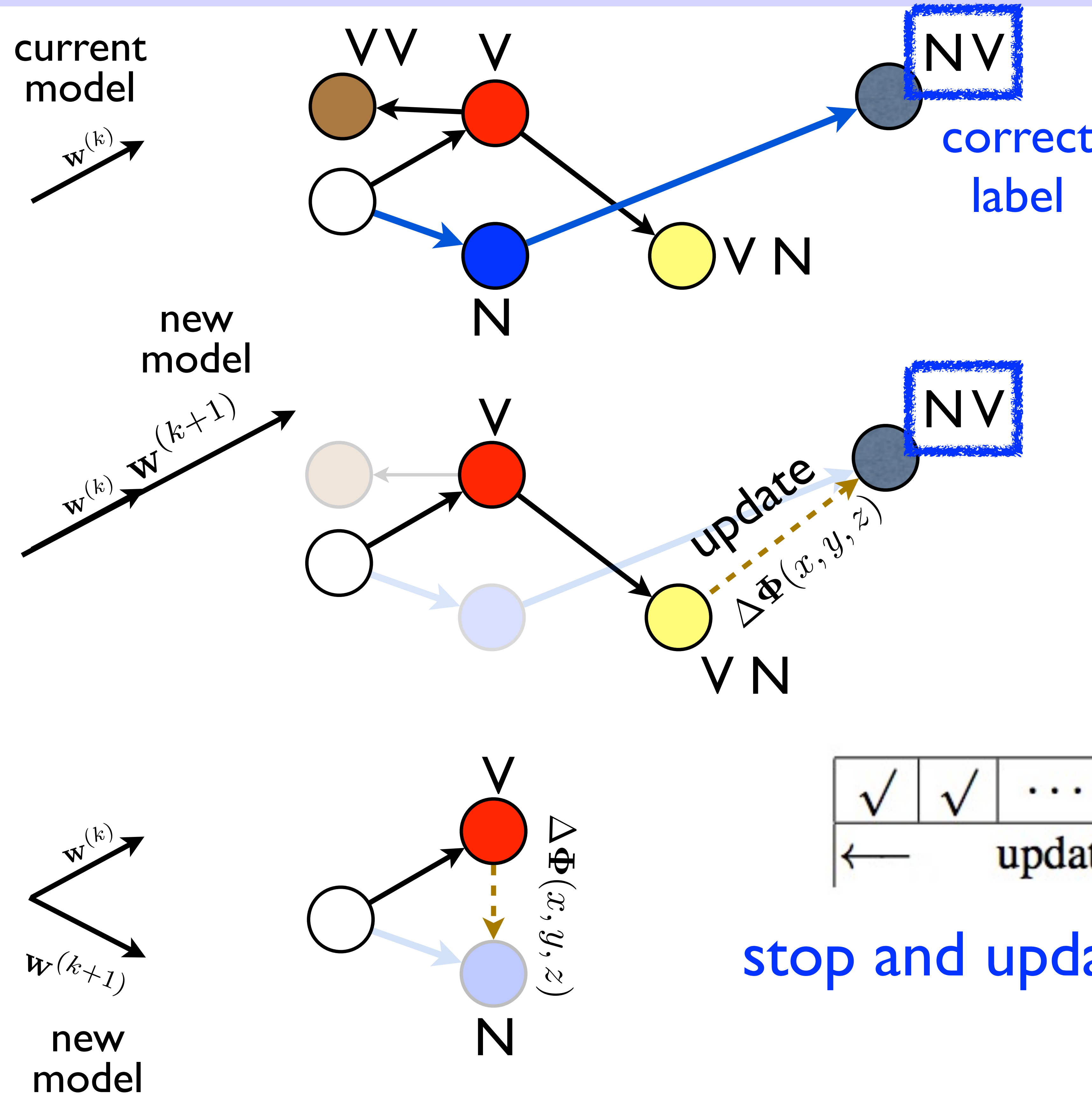
standard perceptron does not converge with greedy search



√	√	···	√	×	
← update →					skip →

stop and update at the first mistake

Solution 1: Early update (Collins/Roark 2004)



training example

time flies

N V

output space

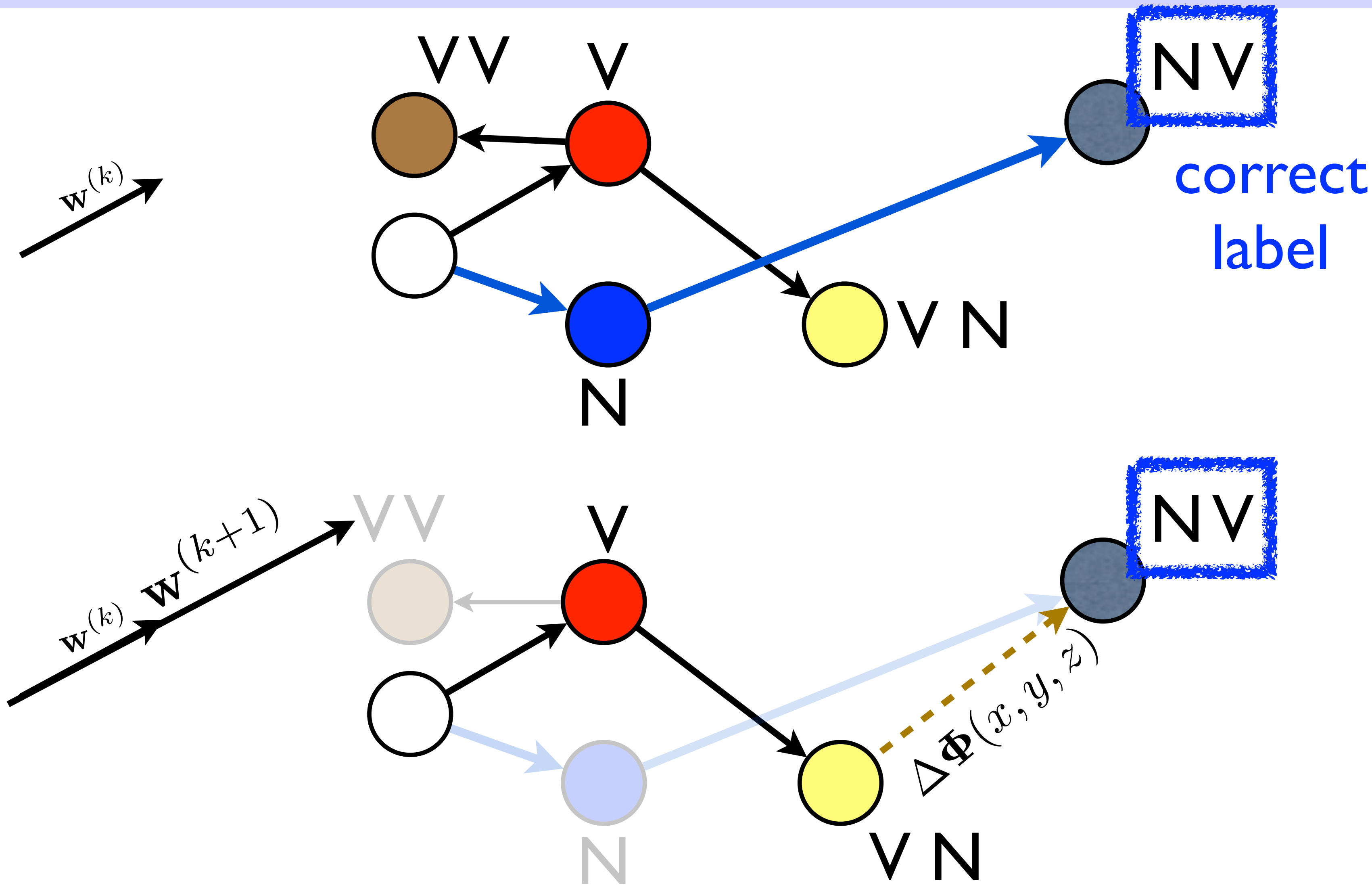
{N,V} x {N,V}

standard perceptron does not converge with greedy search

√	√	···	√	×	
←	update			→	skip →

stop and update at the first mistake

Early Update: Guarantees Violation



training example

time flies

N V

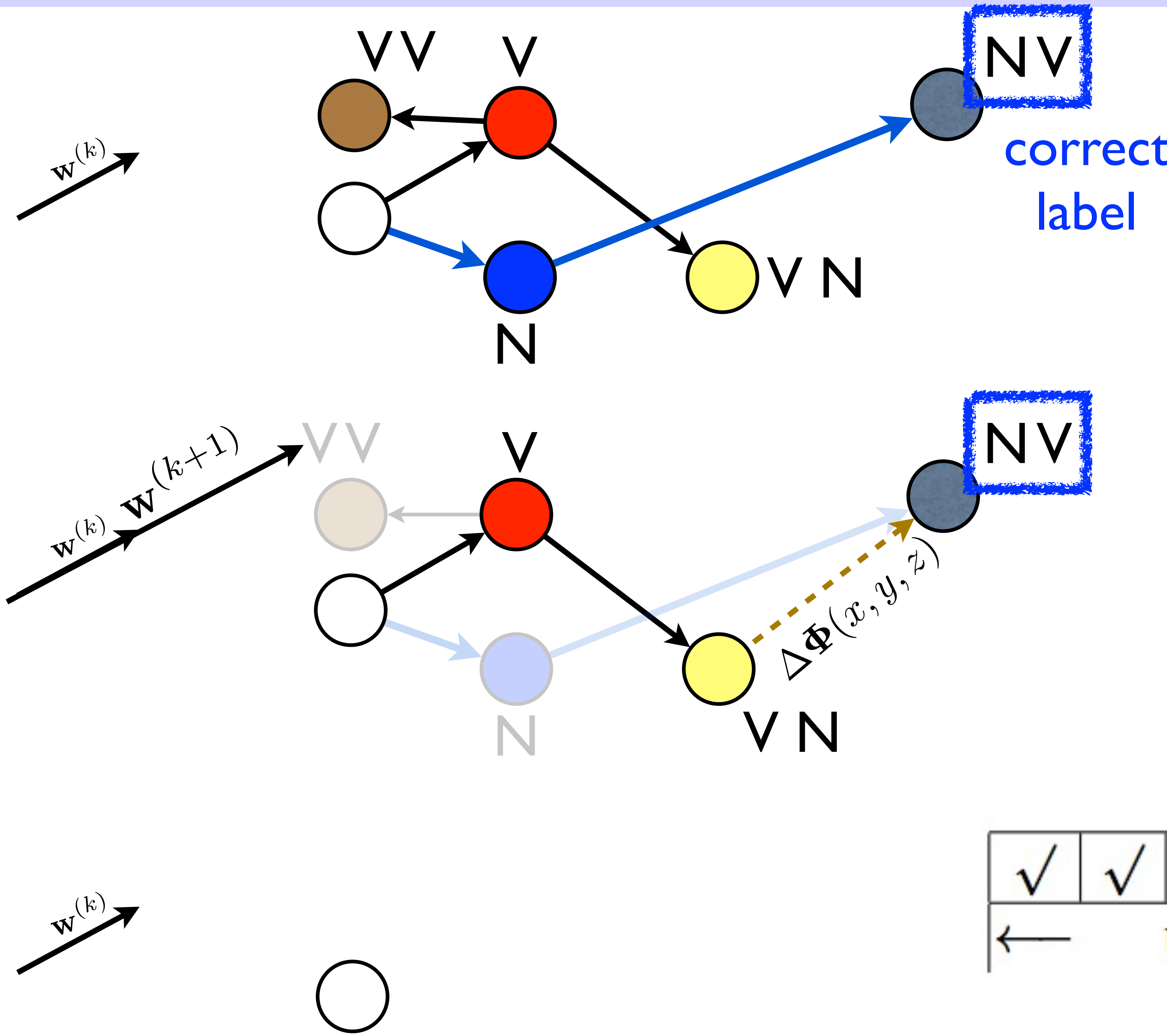
output space

{N,V} x {N,V}

standard update doesn't converge b/c it doesn't guarantee violation

√	√	···	√	×	
⟵		update		⟶	skip ⟶

Early Update: Guarantees Violation



training example

time flies

N V

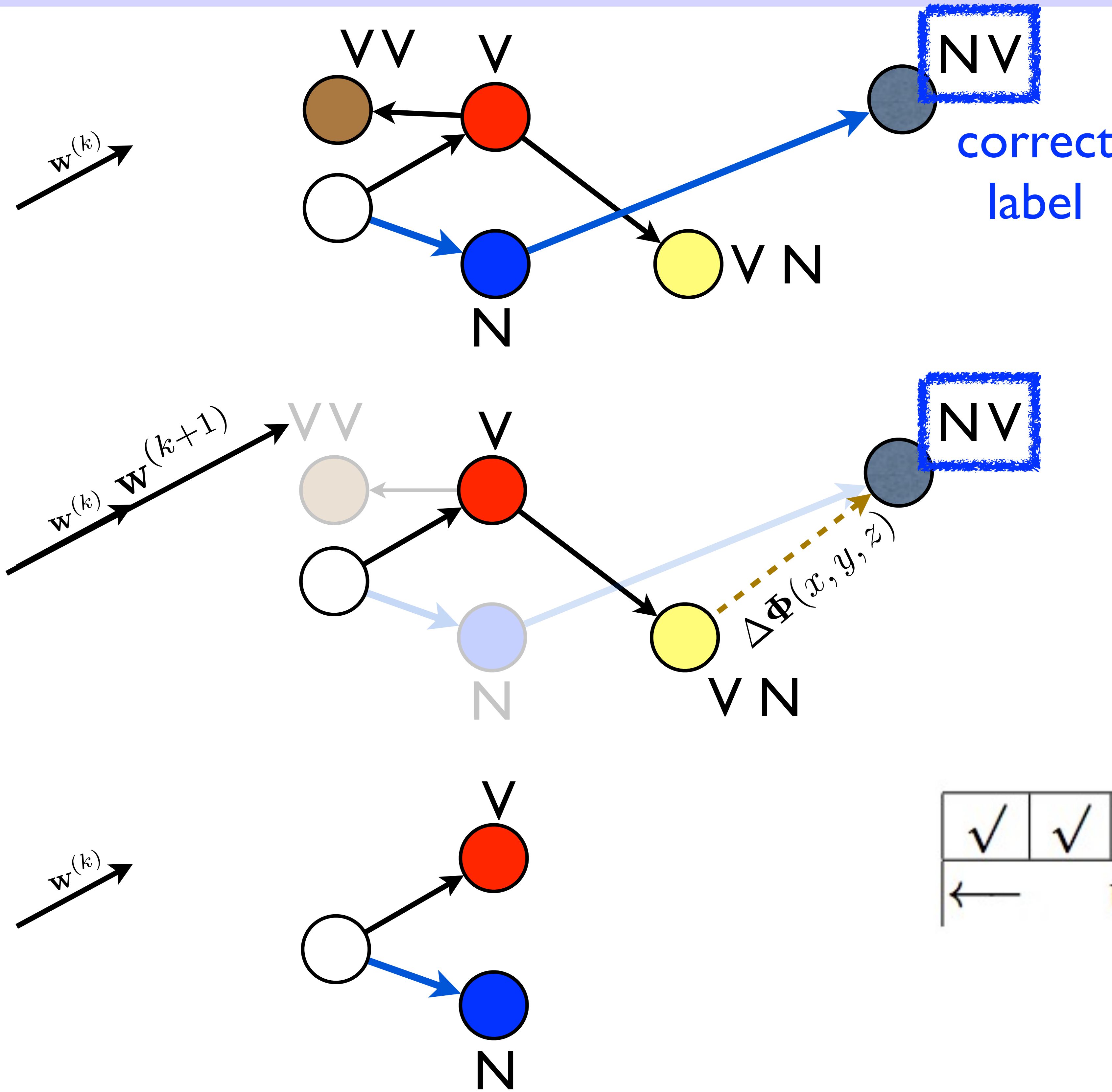
output space

{N,V} x {N,V}

standard update doesn't converge b/c it doesn't guarantee violation

√	√	···	√	×	
← update →					skip →

Early Update: Guarantees Violation



training example

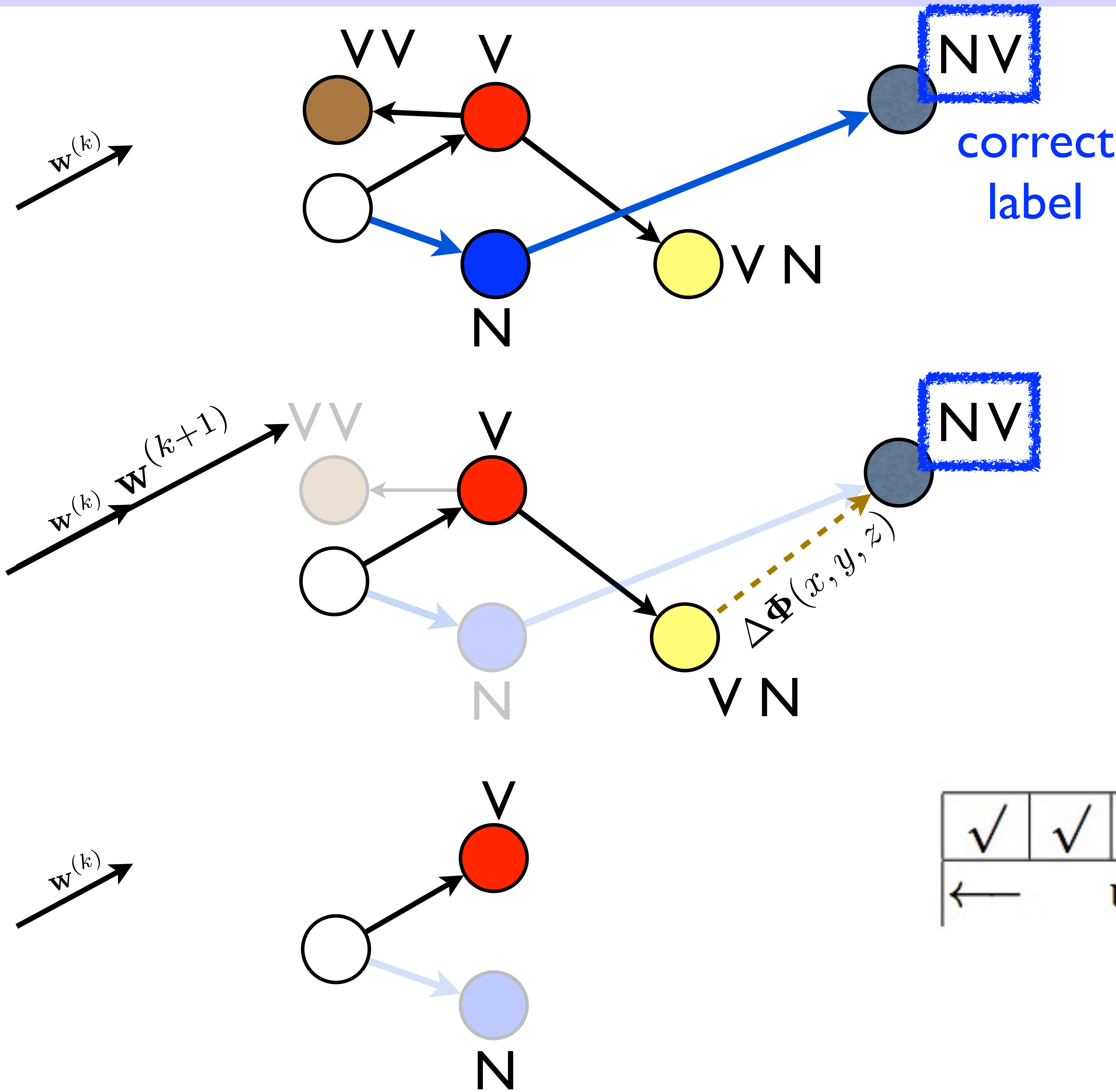
time	flies
N	V

output space
{N,V} x {N,V}

standard update doesn't converge b/c it doesn't guarantee violation

√	√	···	√	×	
⟵	update		⟶		skip ⟶

Early Update: Guarantees Violation



training example

time flies

N V

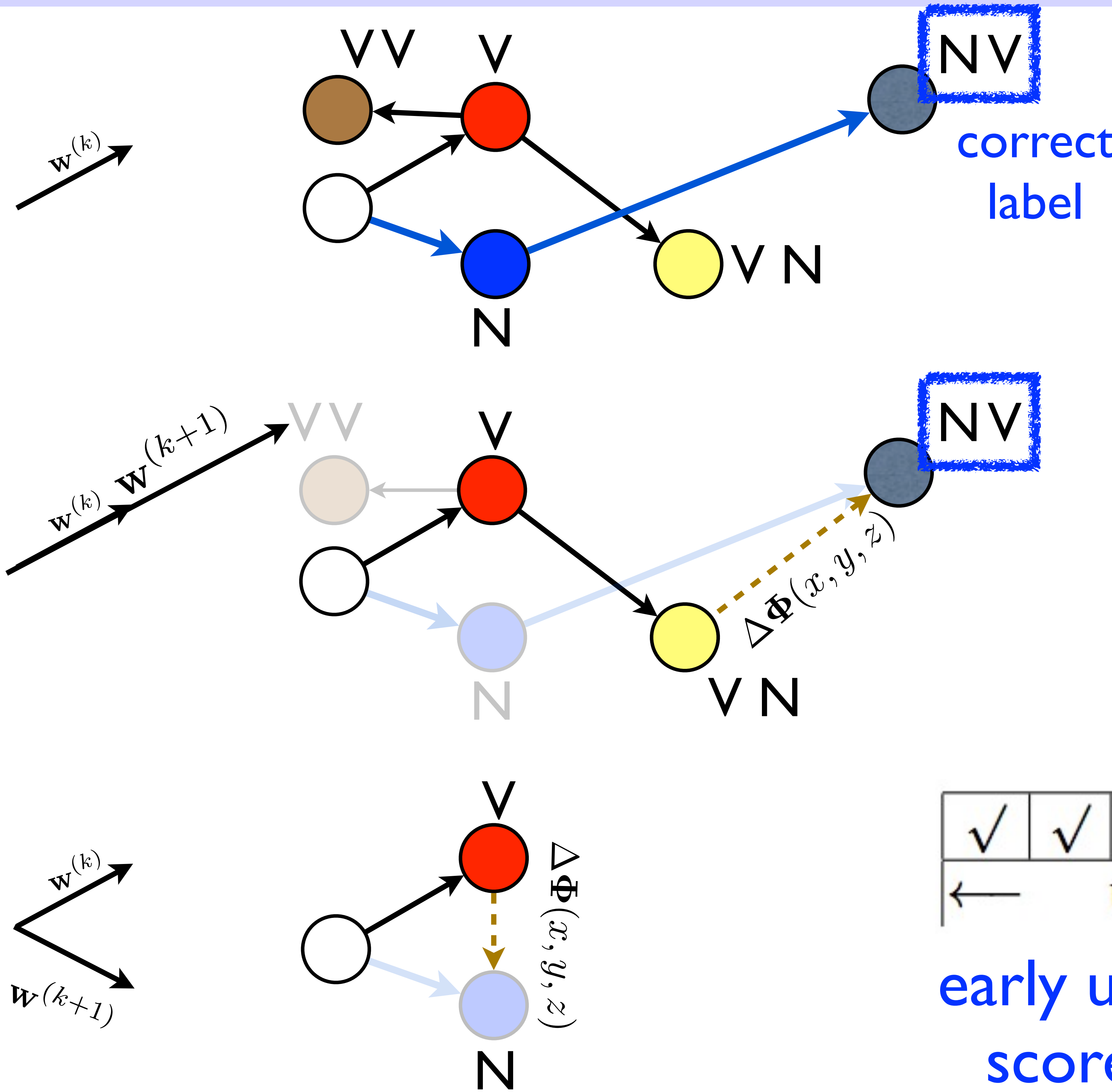
output space

{N,V} x {N,V}

standard update doesn't converge b/c it doesn't guarantee violation

$\surd$	$\surd$	$\cdots$	$\surd$	$\times$	
$\longleftarrow$	update			$\longrightarrow$	skip $\longrightarrow$

Early Update: Guarantees Violation



training example

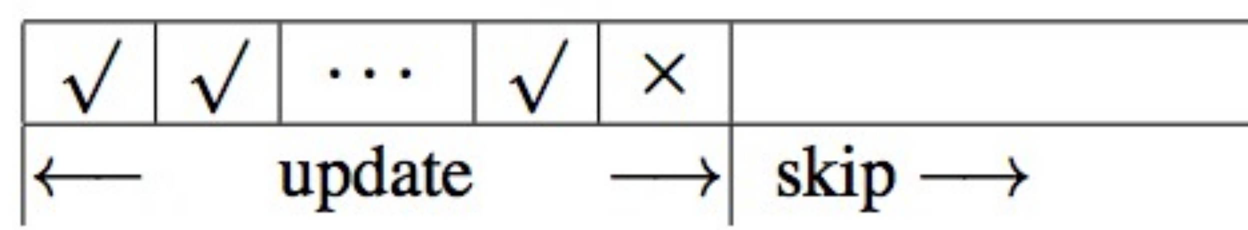
time flies

N V

output space

{N,V} x {N,V}

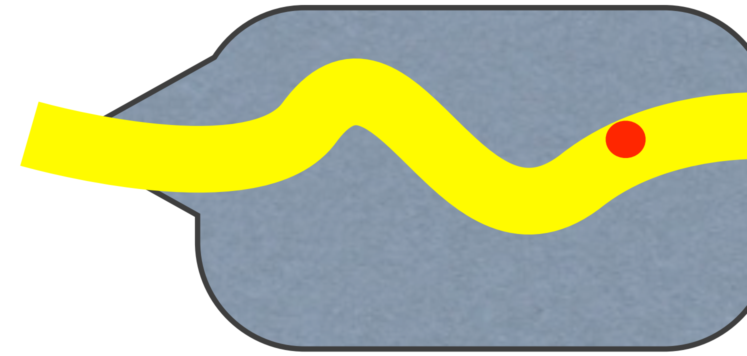
standard update doesn't converge b/c it doesn't guarantee violation



early update: incorrect prefix scores higher: a violation!

Early Update: from Greedy to Beam

- beam search is a generalization of greedy (where b=1)
  - at each stage we keep top b hypothesis
  - widely used: tagging, parsing, translation...

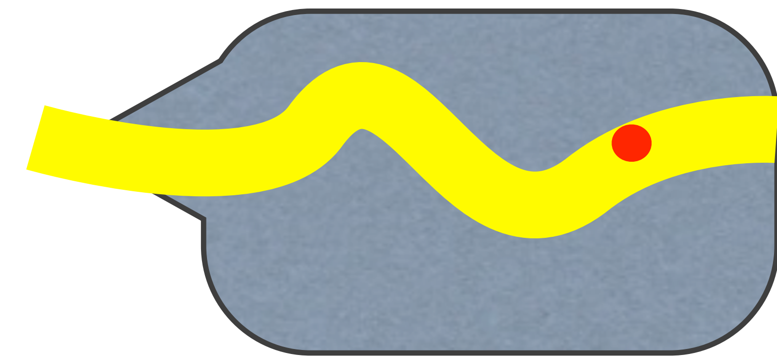


- early update -- when correct label first falls off the beam
  - up to this point the incorrect prefix should score higher
- standard update (full update) -- no guarantee!

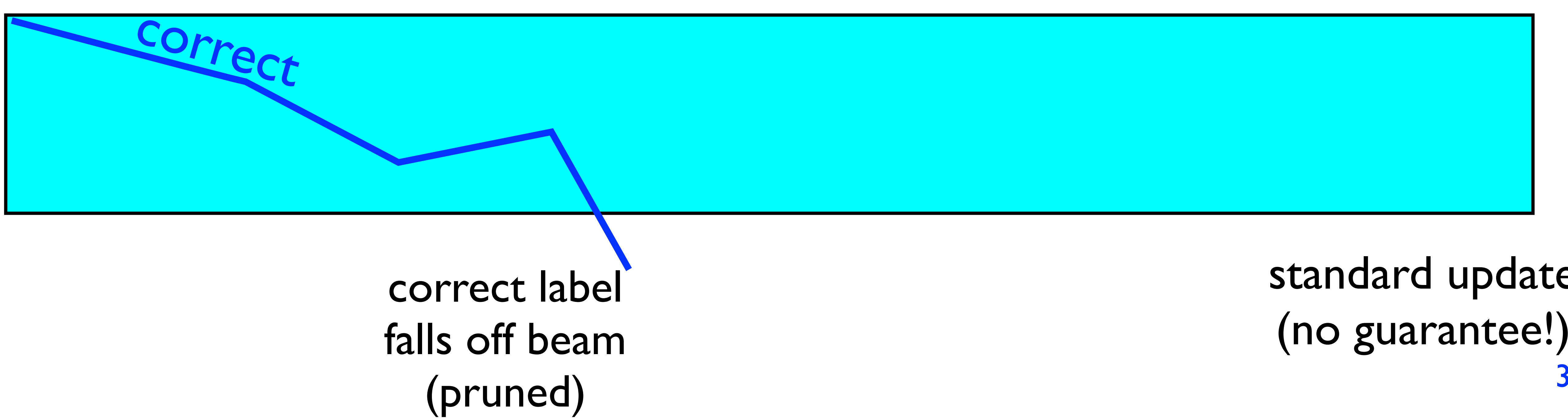
standard update
(no guarantee!)

Early Update: from Greedy to Beam

- beam search is a generalization of greedy (where b=1)
  - at each stage we keep top b hypothesis
  - widely used: tagging, parsing, translation...

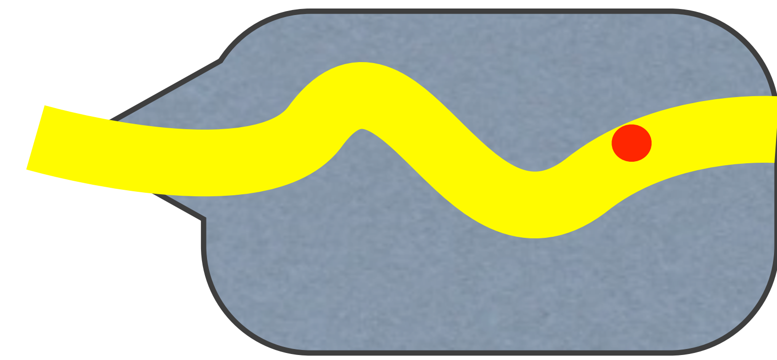


- early update -- when correct label first falls off the beam
  - up to this point the incorrect prefix should score higher
- standard update (full update) -- no guarantee!

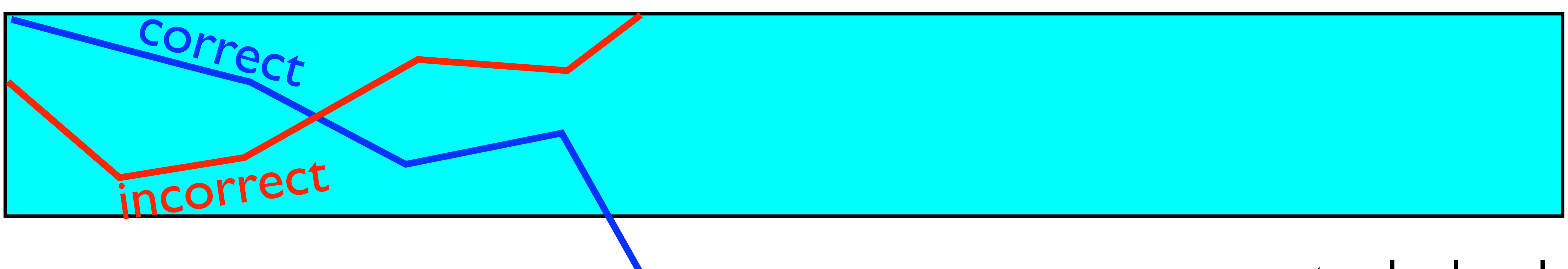


Early Update: from Greedy to Beam

- beam search is a generalization of greedy (where b=1)
  - at each stage we keep top b hypothesis
  - widely used: tagging, parsing, translation...

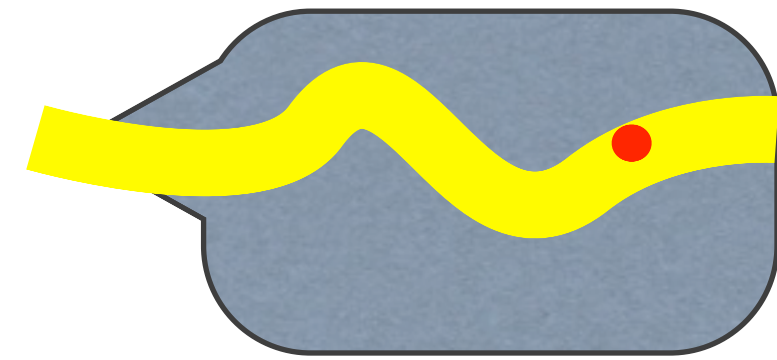


- early update -- when correct label first falls off the beam
  - up to this point the incorrect prefix should score higher
- standard update (full update) -- no guarantee!

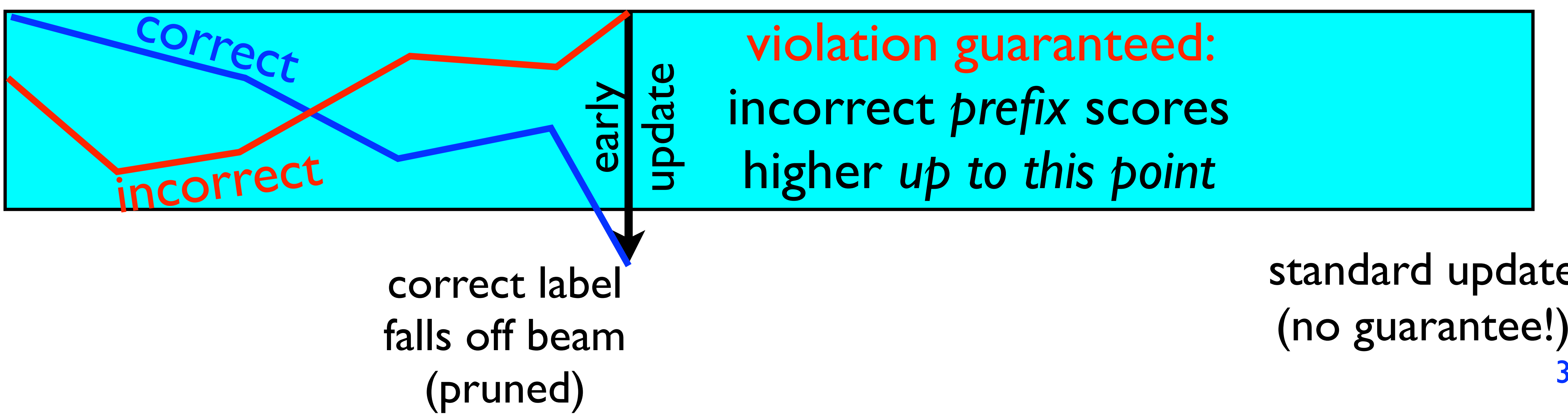


Early Update: from Greedy to Beam

- beam search is a generalization of greedy (where b=1)
  - at each stage we keep top b hypothesis
  - widely used: tagging, parsing, translation...

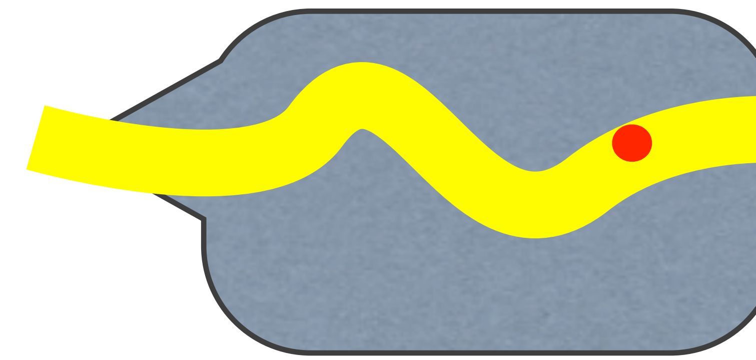


- early update -- when correct label first falls off the beam
  - up to this point the incorrect prefix should score higher
- standard update (full update) -- no guarantee!

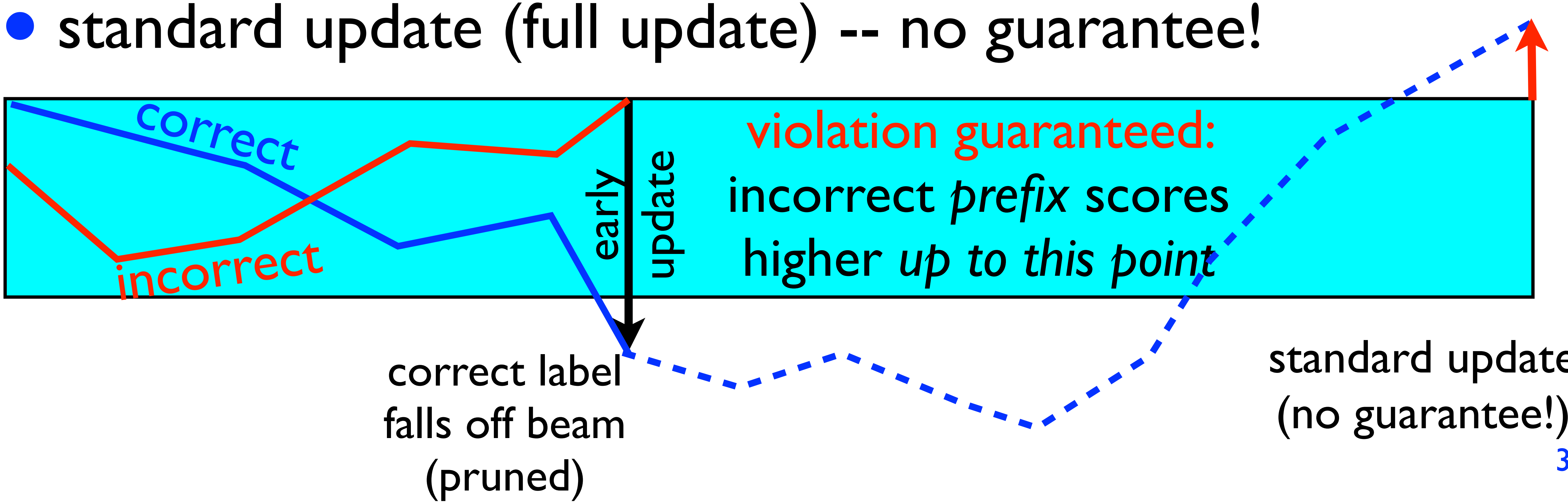


Early Update: from Greedy to Beam

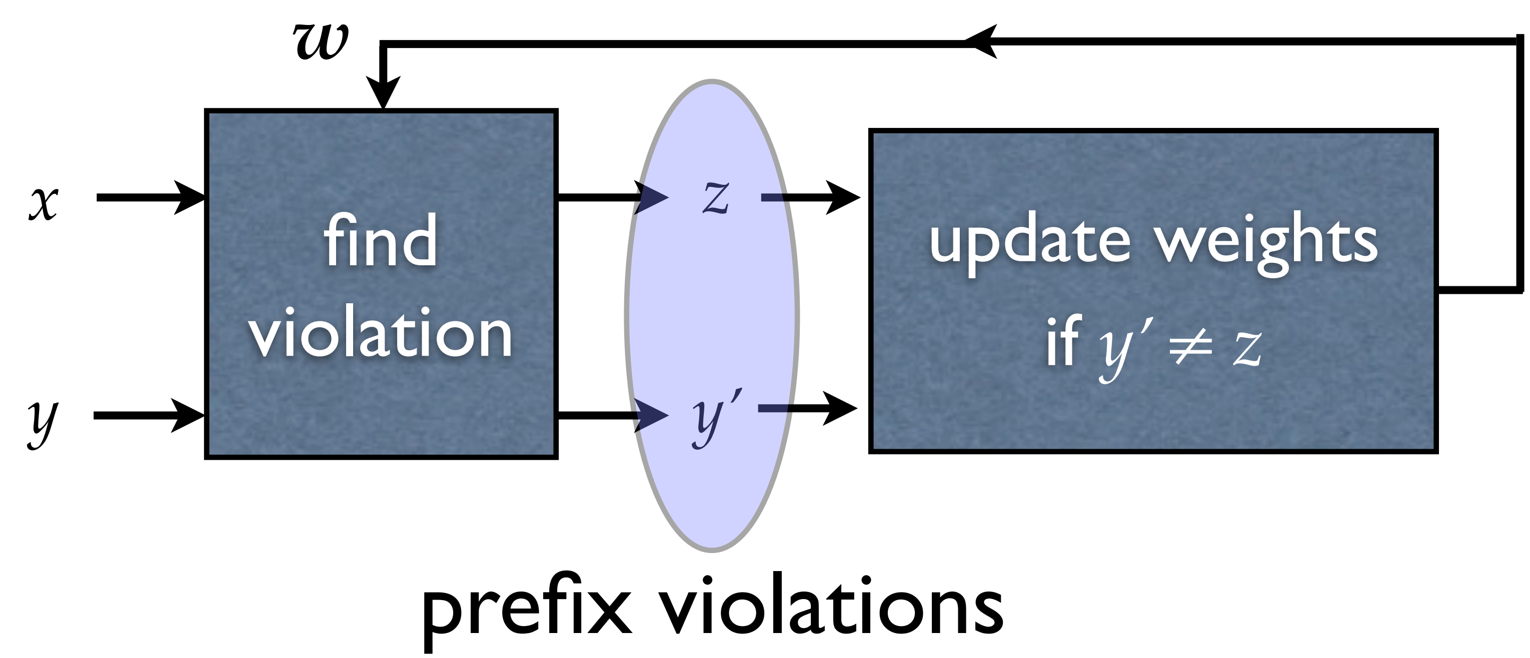
- beam search is a generalization of greedy (where b=1)
  - at each stage we keep top b hypothesis
  - widely used: tagging, parsing, translation...

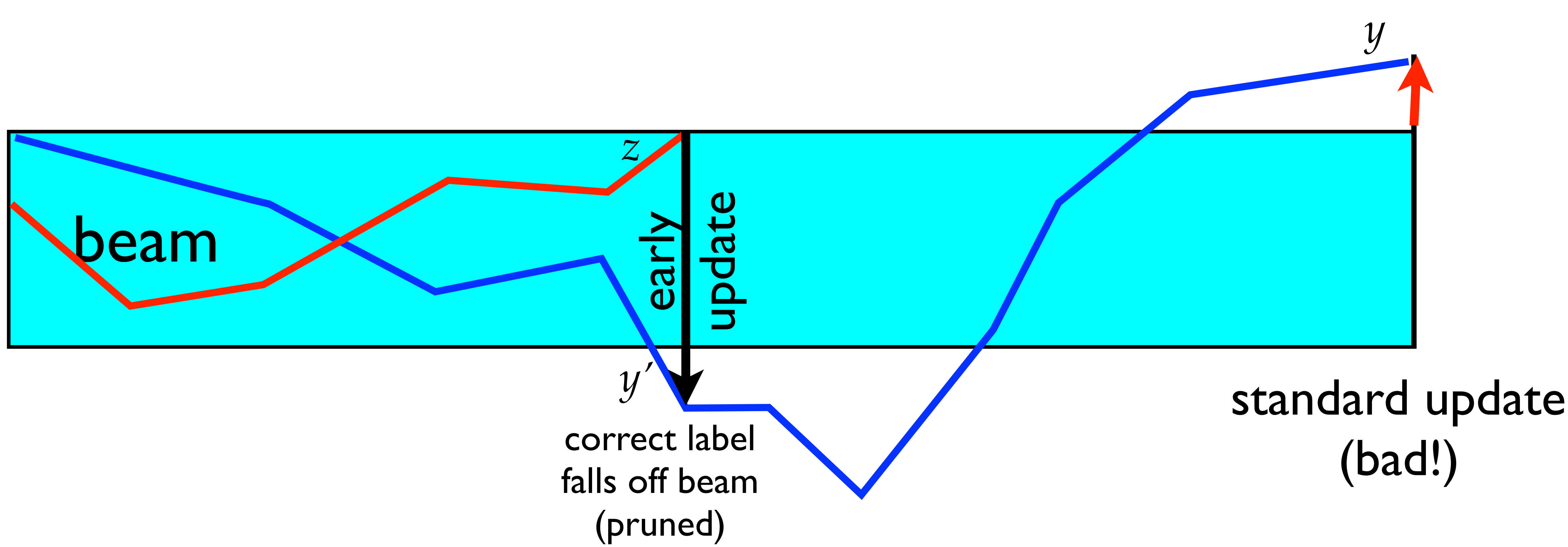


- early update -- when correct label first falls off the beam
  - up to this point the incorrect prefix should score higher
- standard update (full update) -- no guarantee!



Early Update as Violation-Fixing

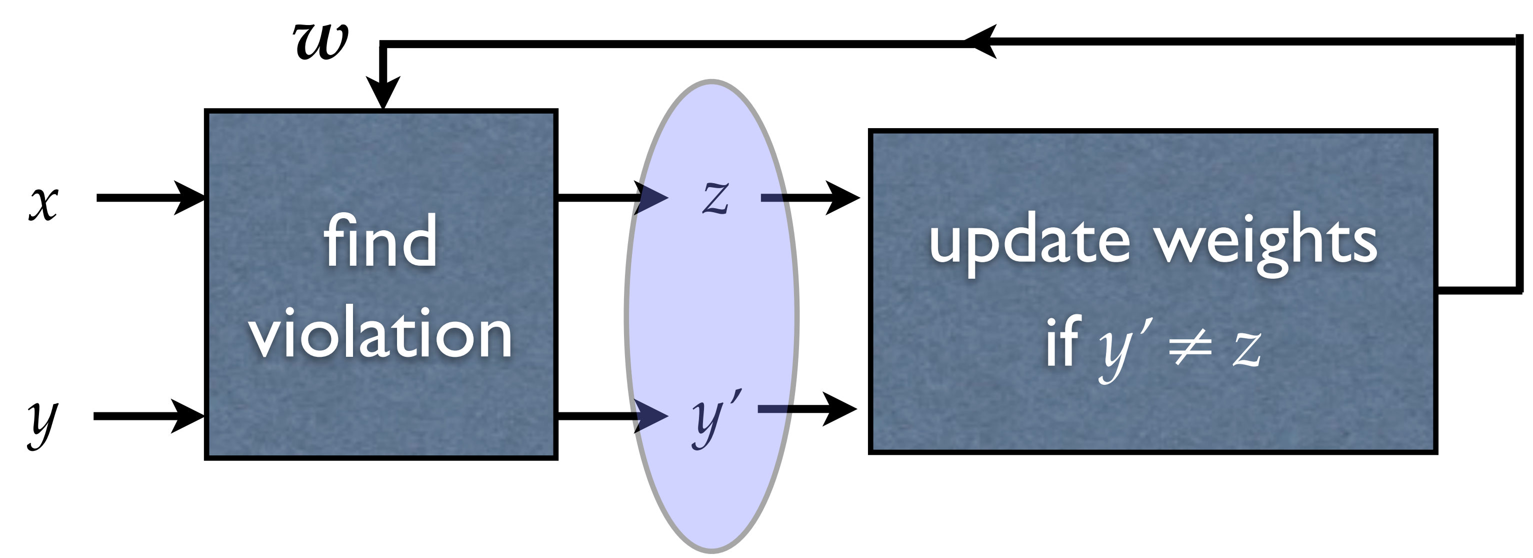




Early Update as Violation-Fixing

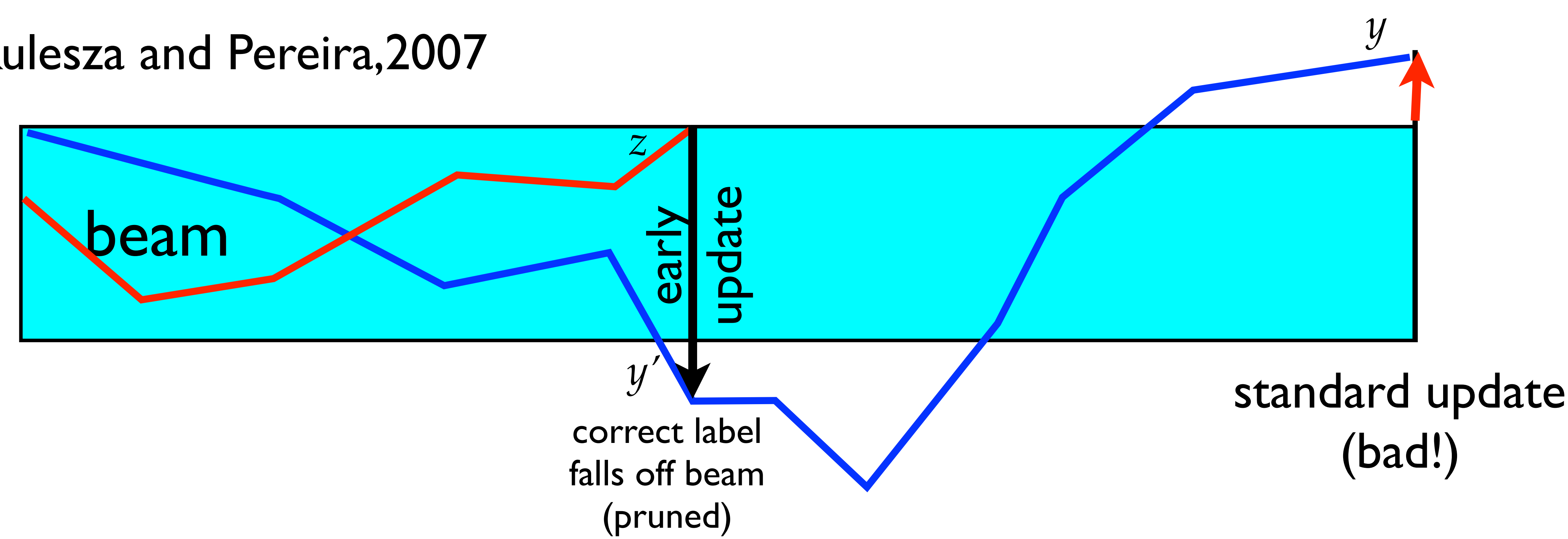
also new definition of
"beam separability":

a correct prefix should
score higher than
any incorrect prefix
of the same length
(maybe too strong)

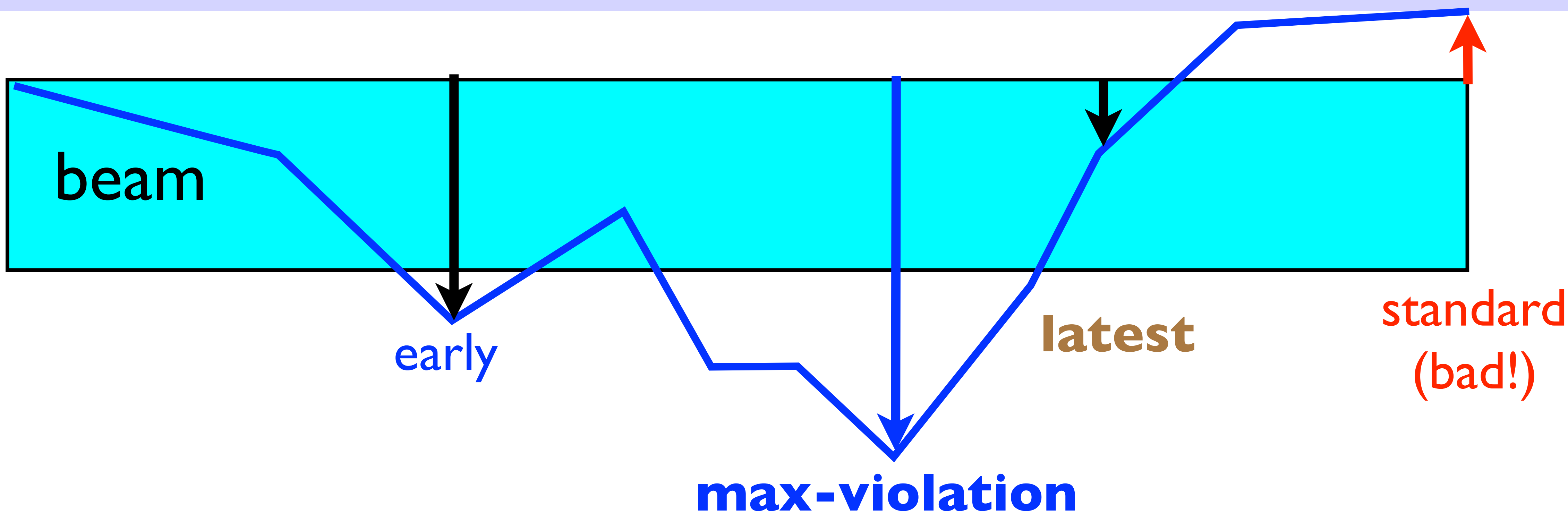


prefix violations

cf. Kulesza and Pereira,2007

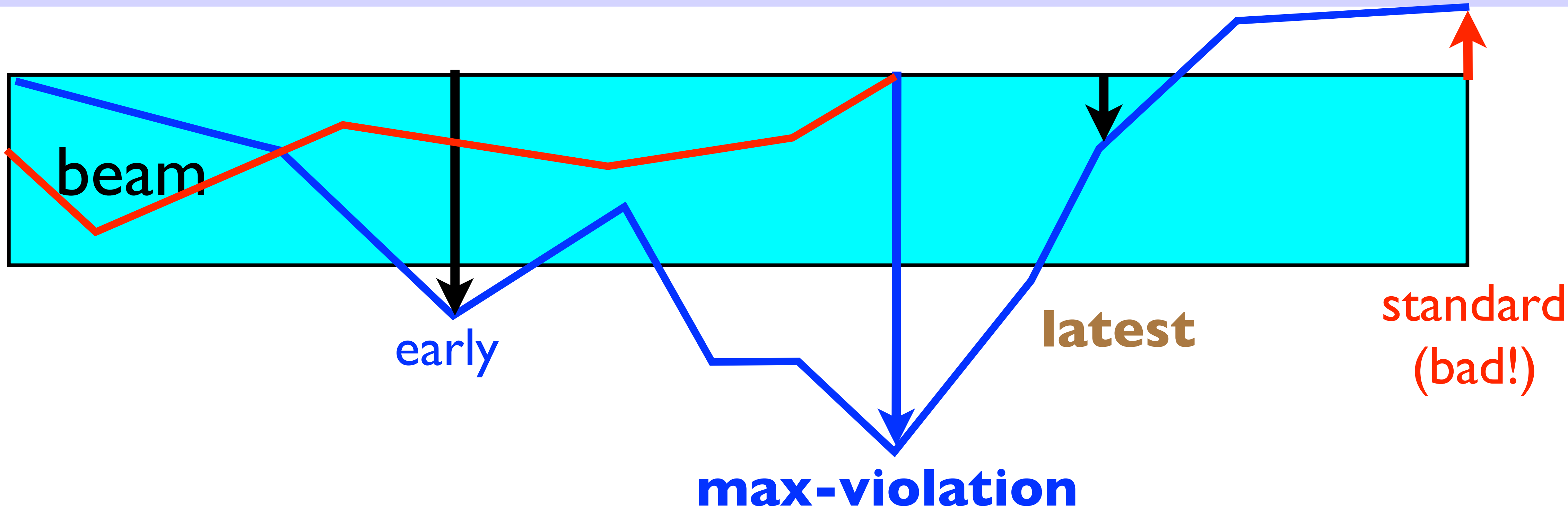


Solution 2: Max-Violation (Huang et al 2012)



- we now established a theory for early update (Collins/Roark)
- but it learns too slowly due to partial updates
- max-violation: use the prefix where violation is maximum
  - "worst-mistake" in the search space
- all these update methods are violation-fixing perceptrons

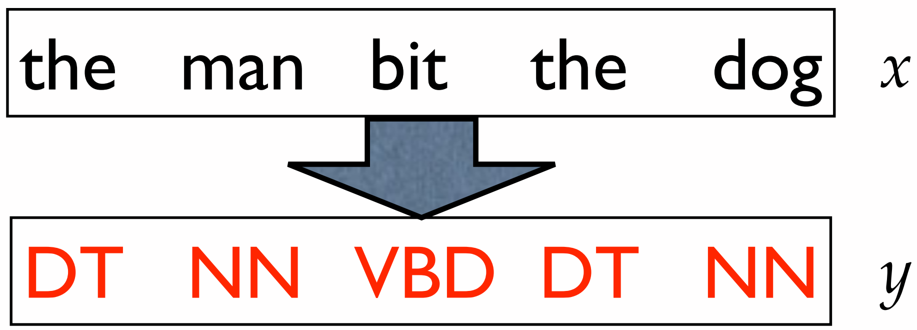
Solution 2: Max-Violation (Huang et al 2012)



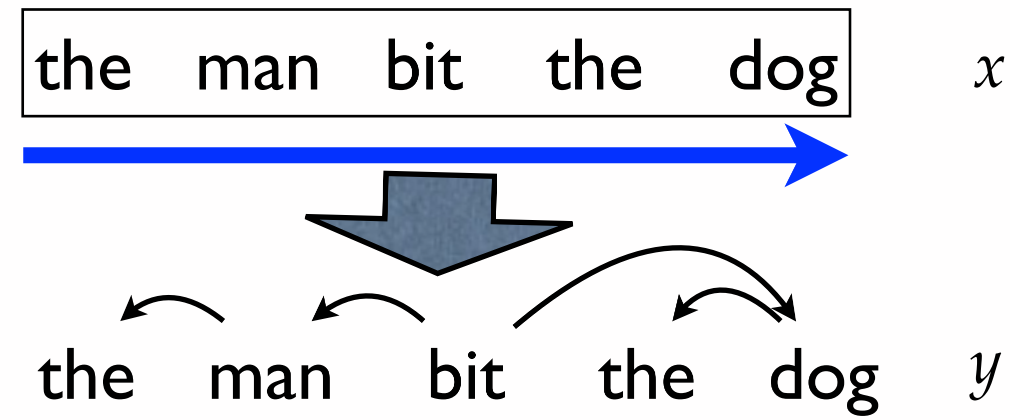
- we now established a theory for early update (Collins/Roark)
- but it learns too slowly due to partial updates
- max-violation: use the prefix where violation is maximum
  - "worst-mistake" in the search space
- all these update methods are violation-fixing perceptrons

Four Experiments

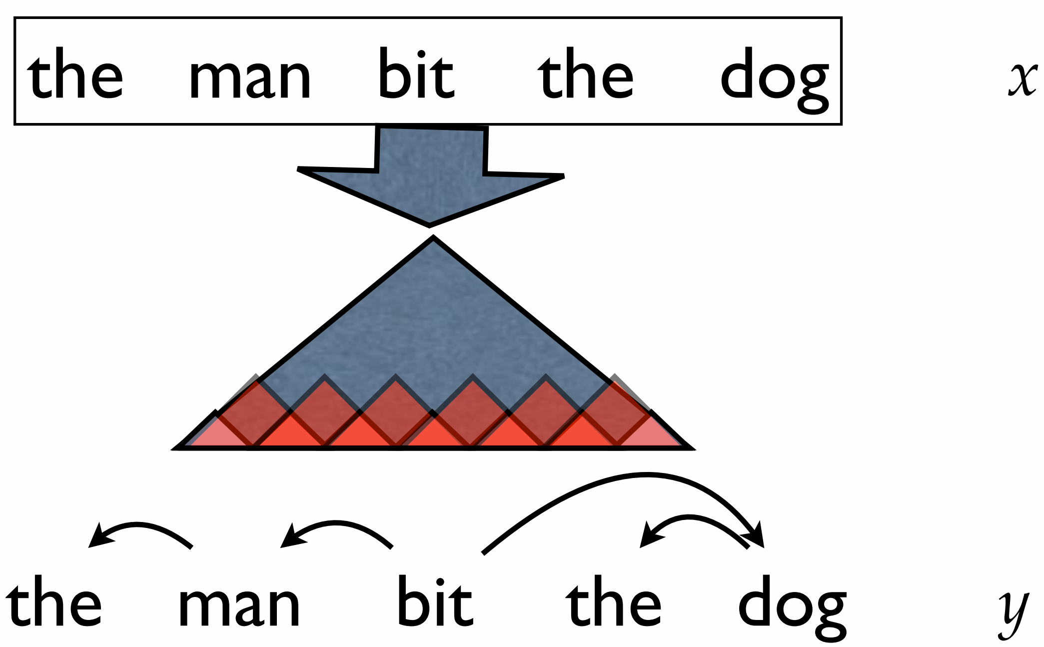
part-of-speech tagging



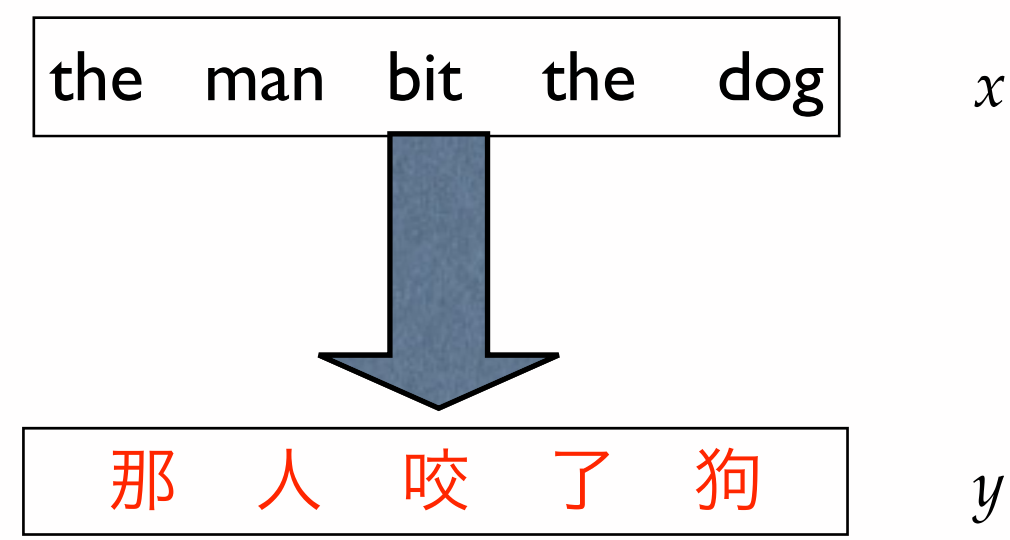
incremental parsing



bottom-up parsing w/ cube pruning

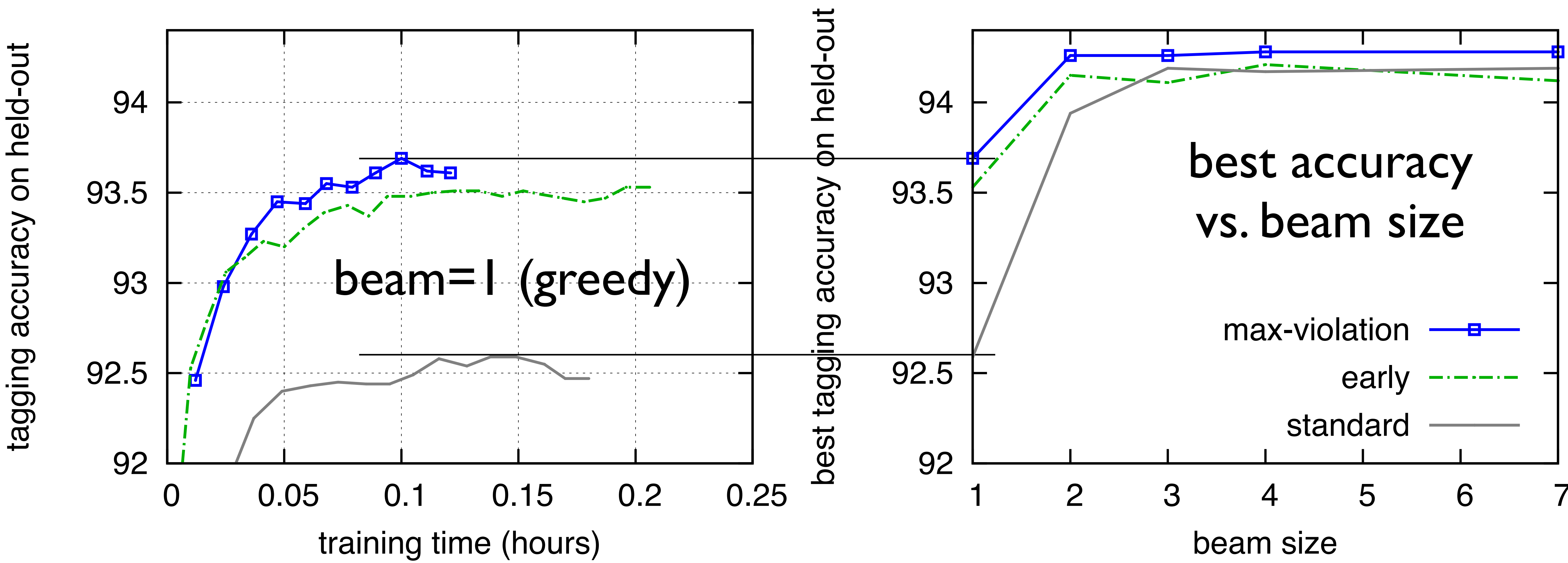


machine translation



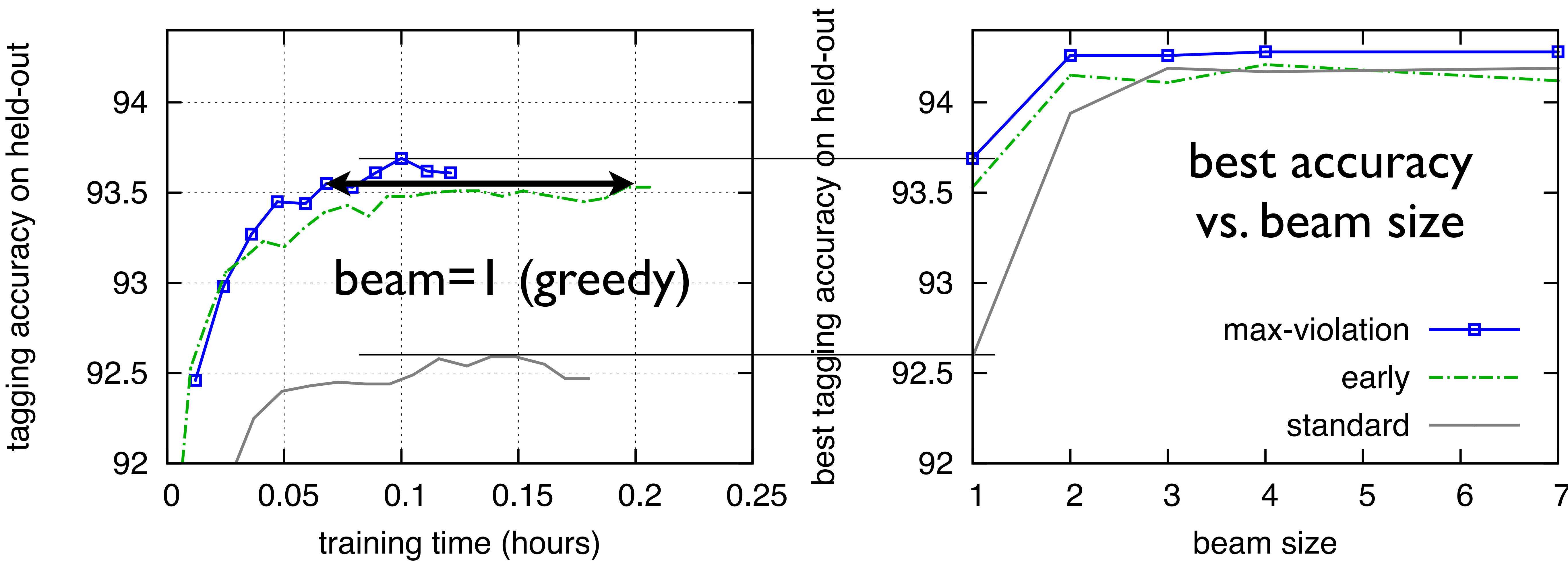
Max-Violation > Early >> Standard

- exp 1 on part-of-speech tagging w/ beam search (on CTB5)
- early and max-violation >> standard update at smallest beams
  - this advantage shrinks as beam size increases
  - max-violation converges faster than early (and slightly better)



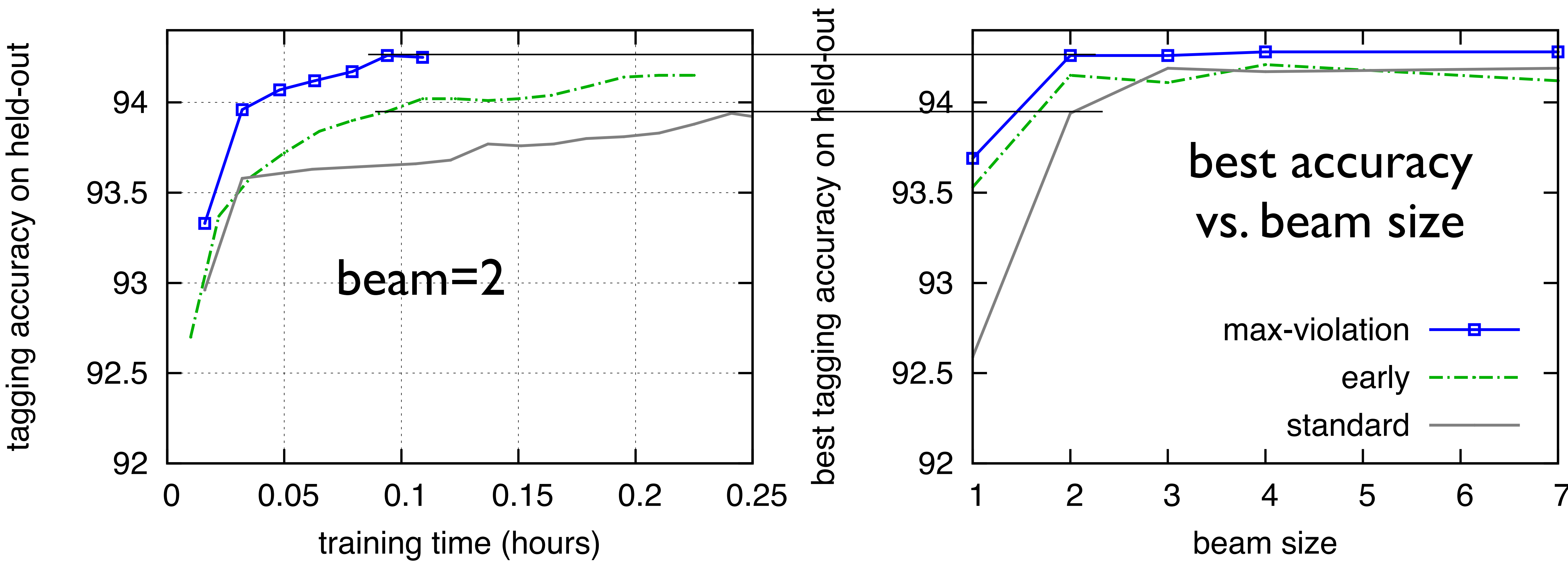
Max-Violation > Early >> Standard

- exp 1 on part-of-speech tagging w/ beam search (on CTB5)
- early and max-violation >> standard update at smallest beams
  - this advantage shrinks as beam size increases
  - max-violation converges faster than early (and slightly better)



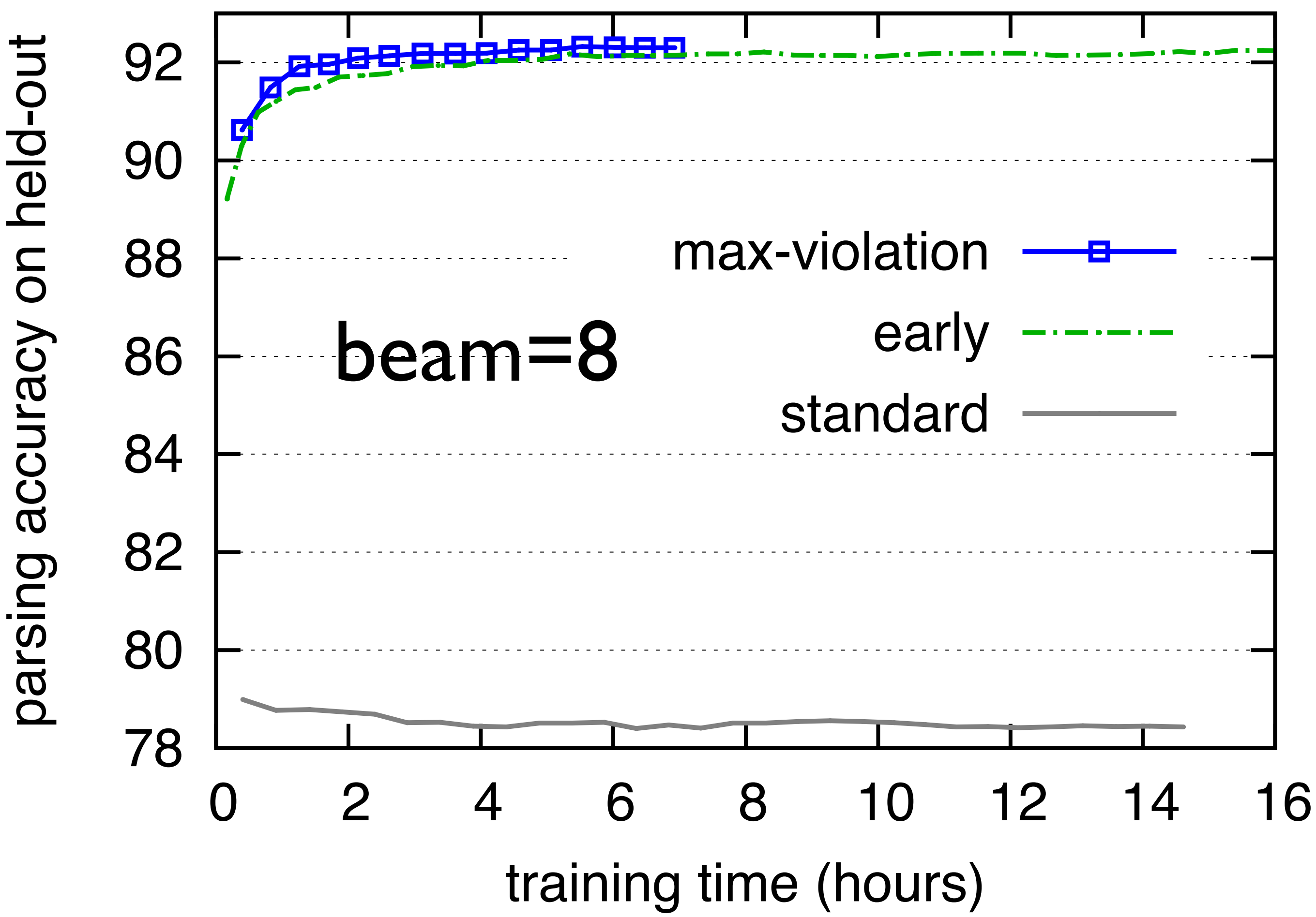
Max-Violation > Early >> Standard

- exp 1 on part-of-speech tagging w/ beam search (on CTB5)
- early and max-violation >> standard update at smallest beams
  - this advantage shrinks as beam size increases
  - max-violation converges faster than early (and slightly better)



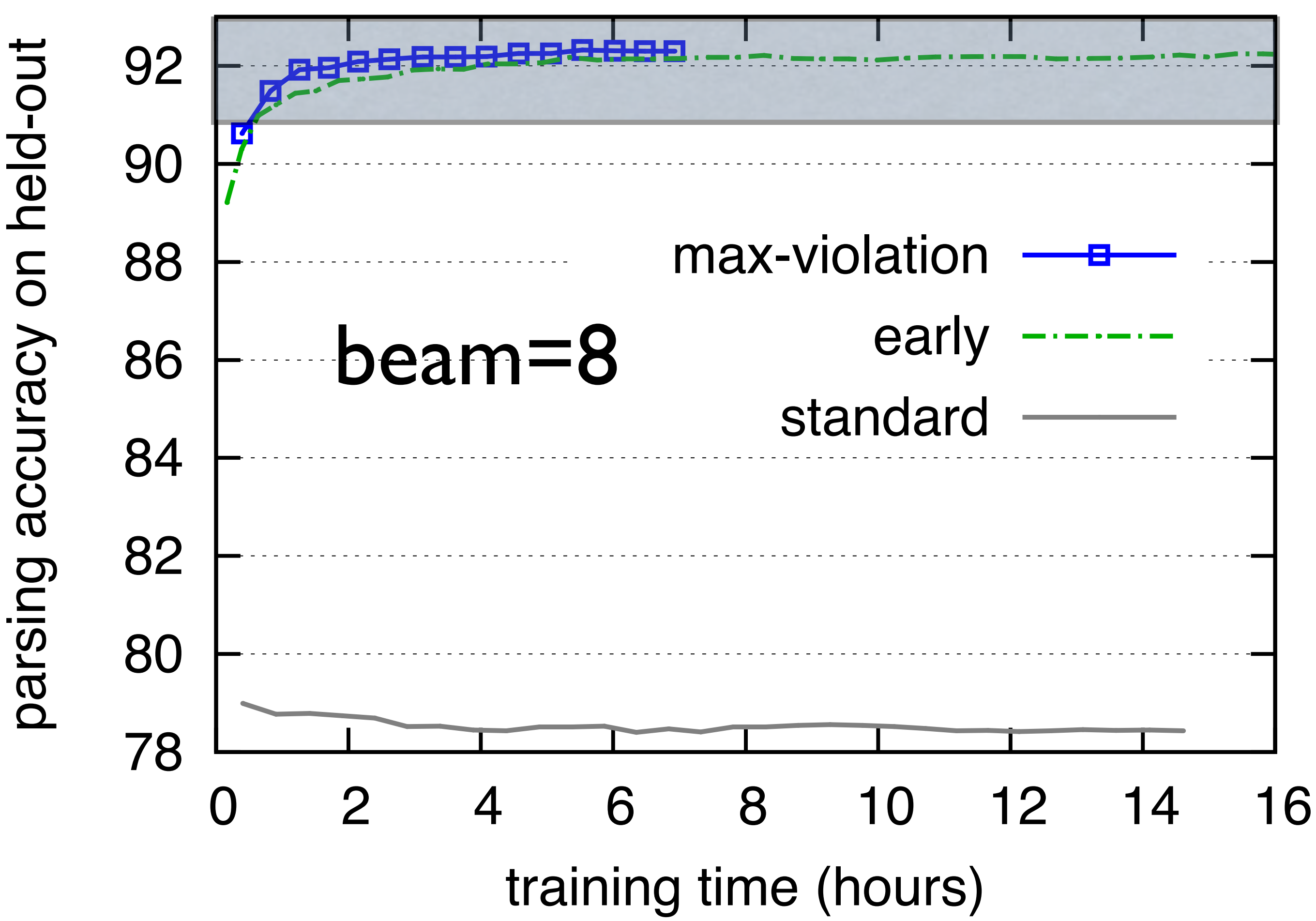
Max-Violation > Early >> Standard

- exp 2 on my incremental dependency parser (Huang & Sagae 10)
- standard update is horrible due to search errors
- early update: 38 iterations, 15.4 hours (92.24)
- max-violation: 10 iterations, 4.6 hours (92.25)



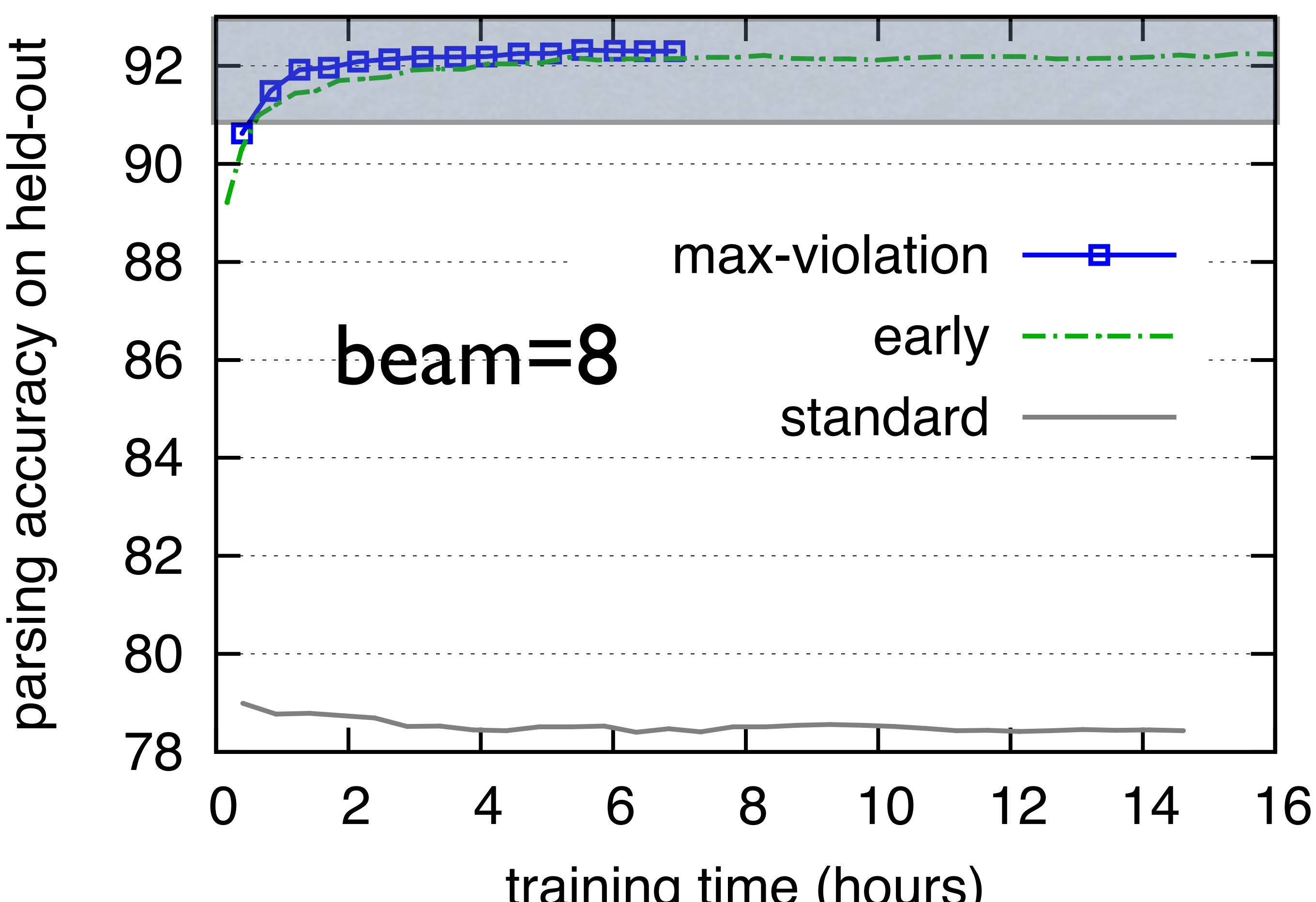
Max-Violation > Early >> Standard

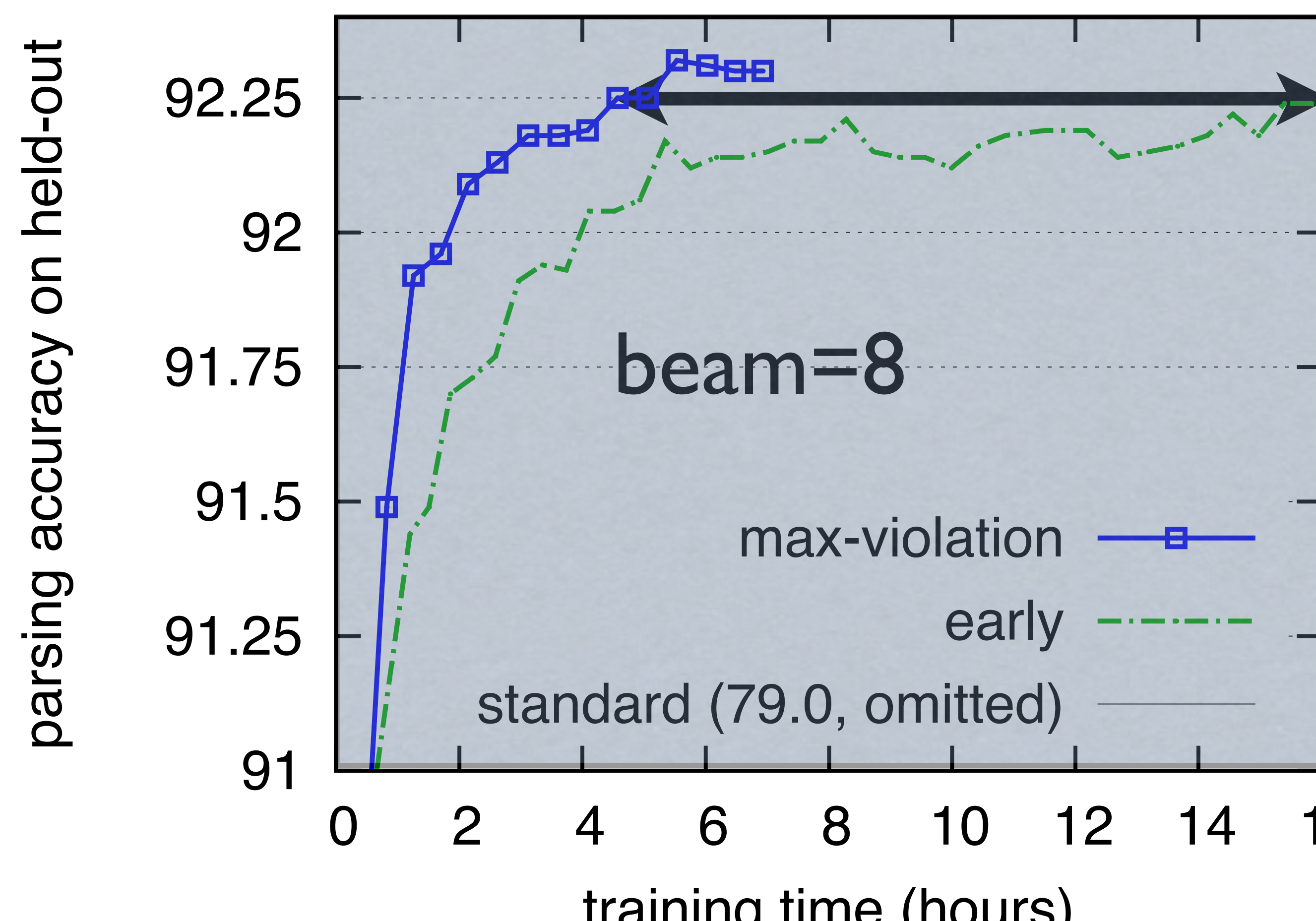
- exp 2 on my incremental dependency parser (Huang & Sagae 10)
- standard update is horrible due to search errors
- early update: 38 iterations, 15.4 hours (92.24)
- max-violation: 10 iterations, 4.6 hours (92.25)



Max-Violation > Early >> Standard

- exp 2 on my incremental dependency parser (Huang & Sagae 10)
- standard update is horrible due to search errors
- early update: 38 iterations, 15.4 hours (92.24)
- max-violation: 10 iterations, 4.6 hours (92.25)

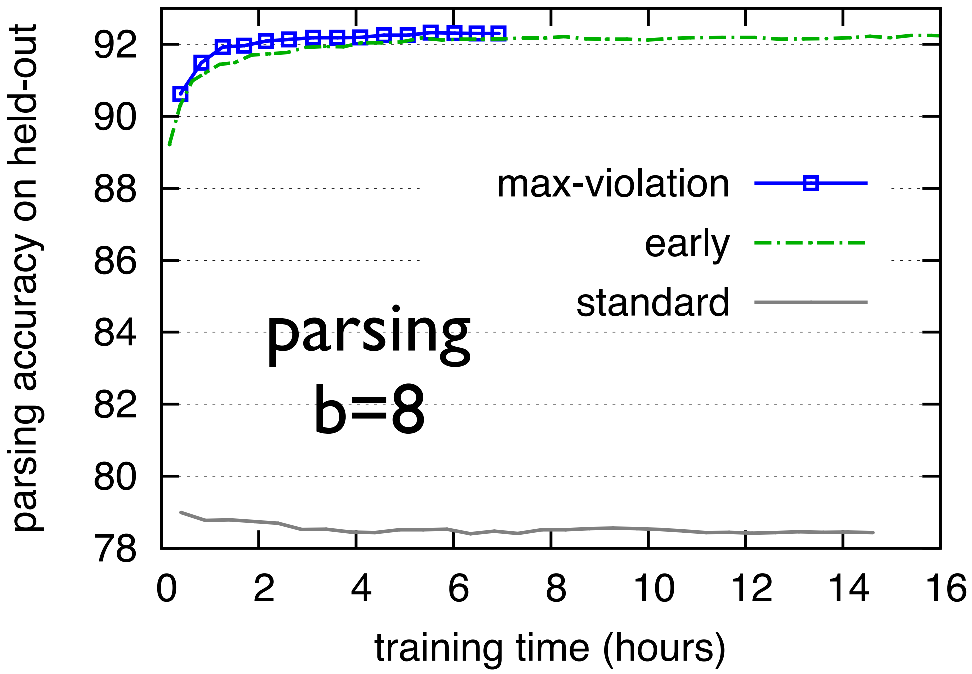


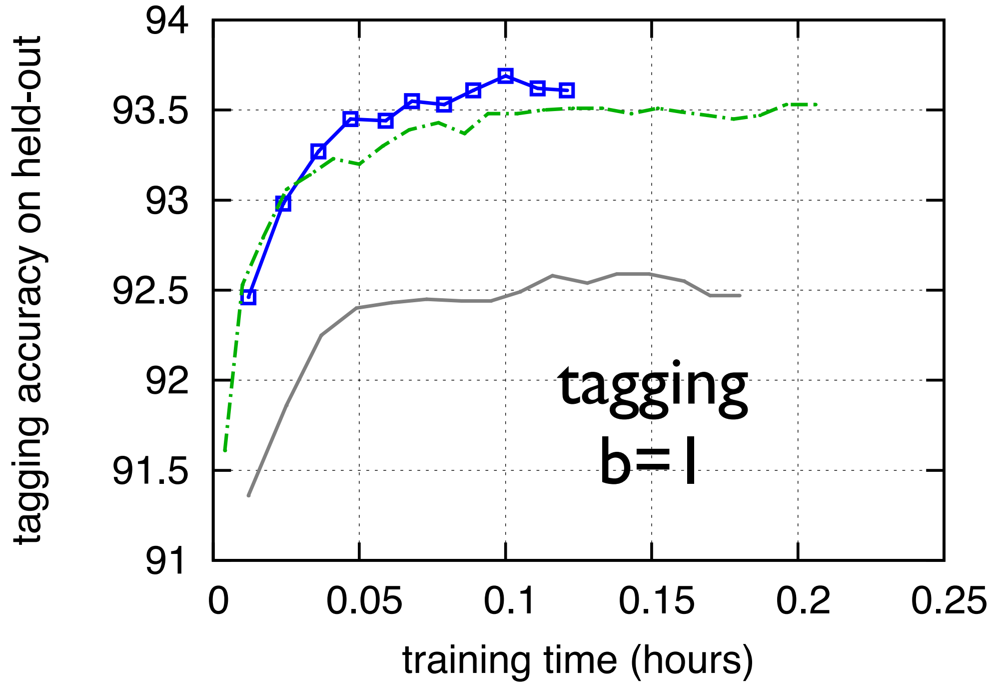


max-violation is 3.3x faster than early update

Why standard update so bad for parsing

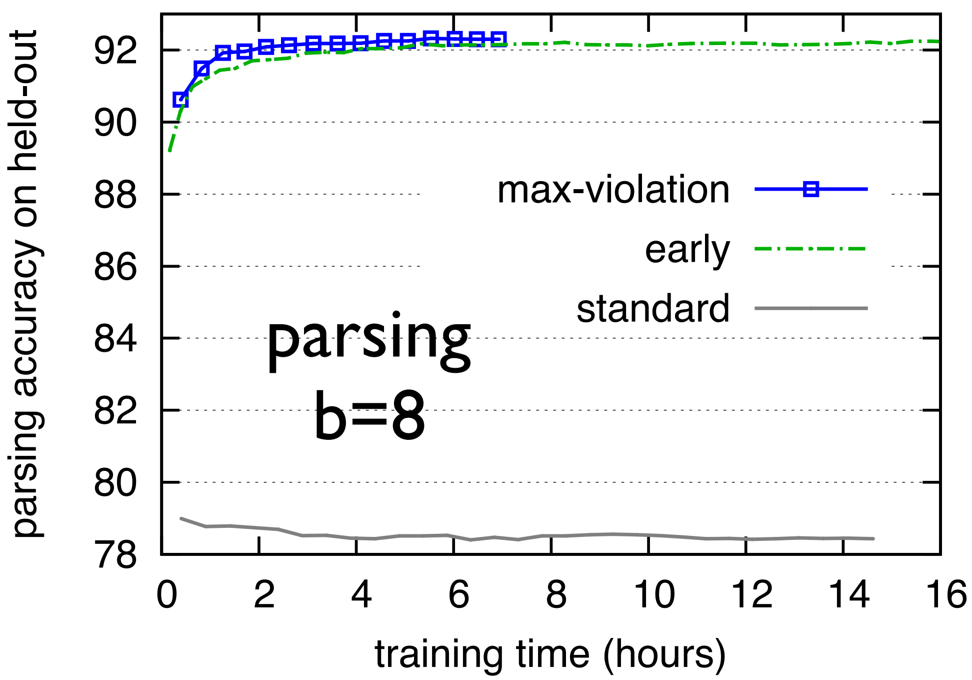
- standard update works horribly with severe search error
    - due to large number of *invalid* updates (non-violation)

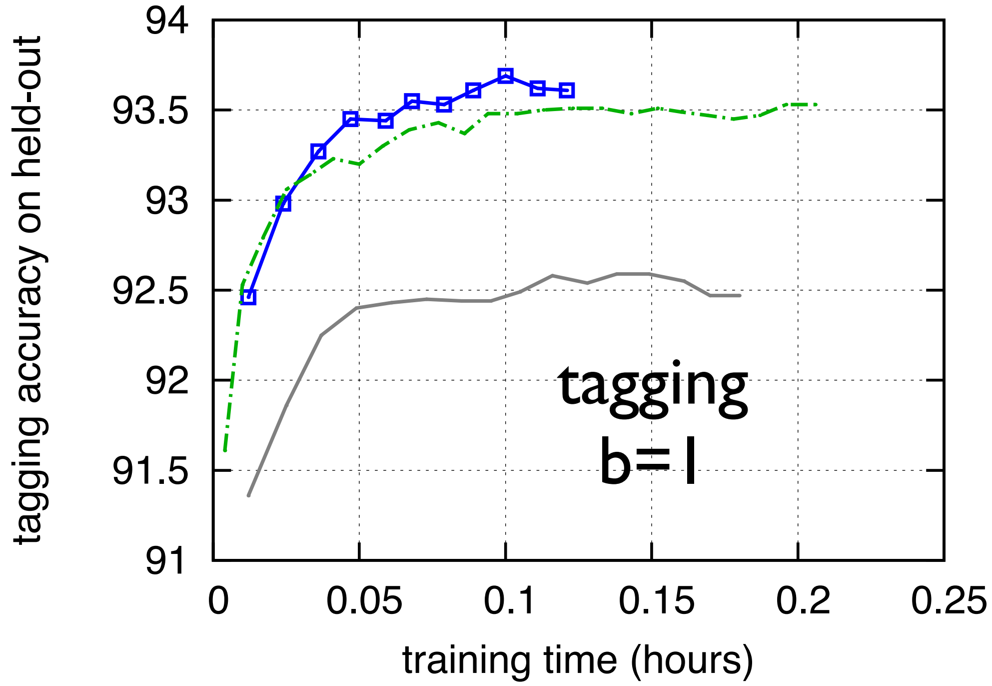


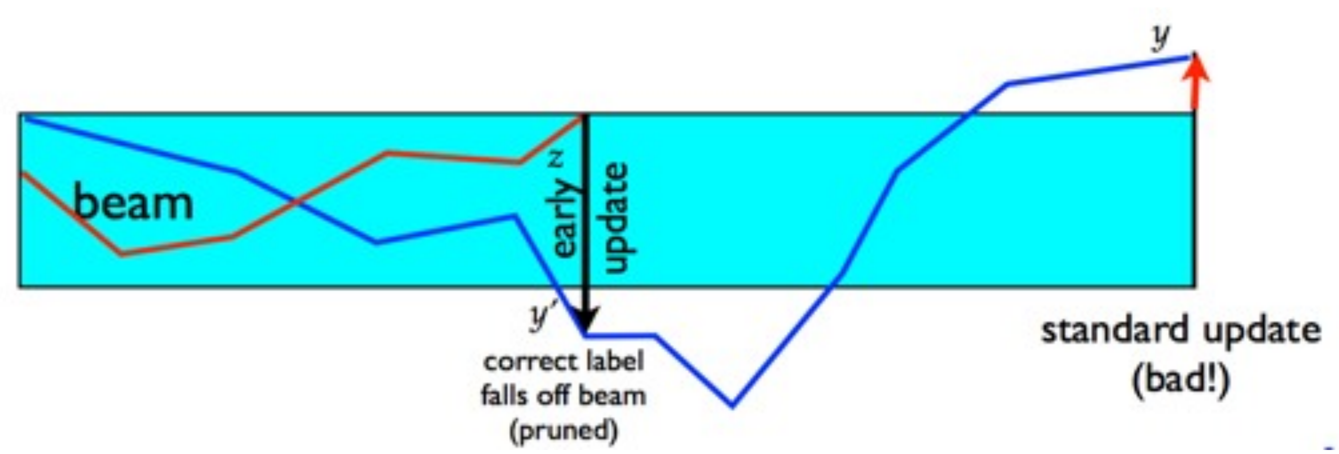


Why standard update so bad for parsing

- standard update works horribly with severe search error
  - due to large number of *invalid* updates (non-violation)

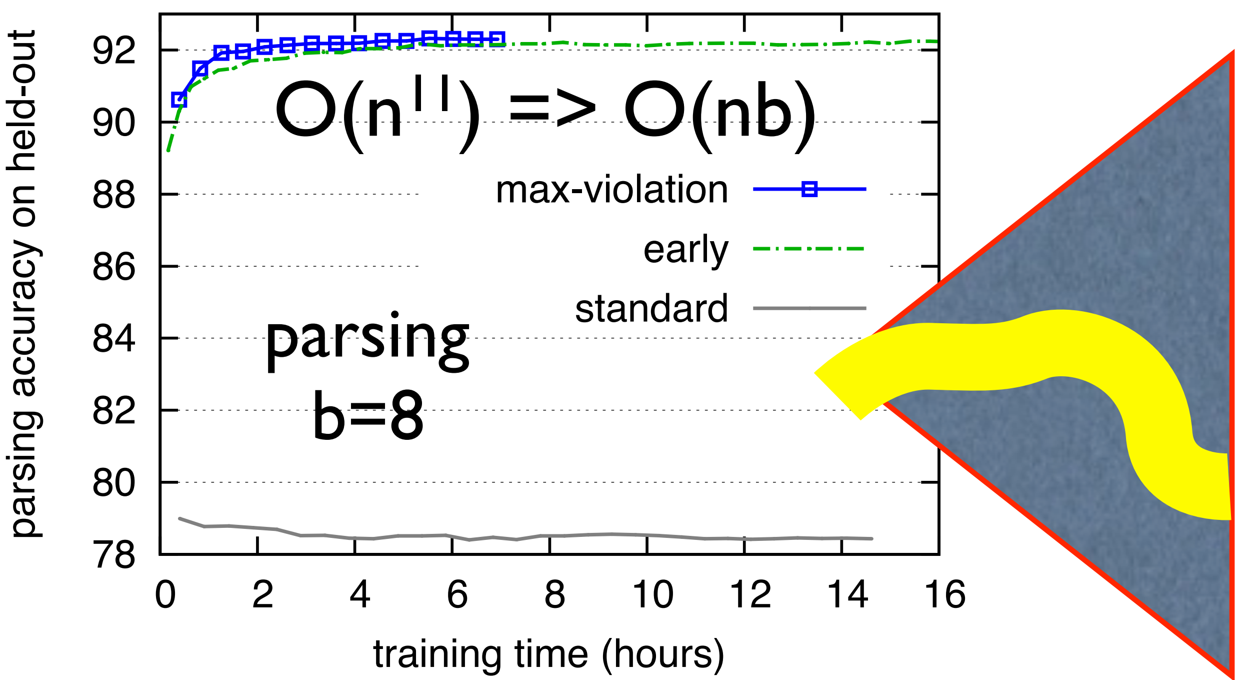


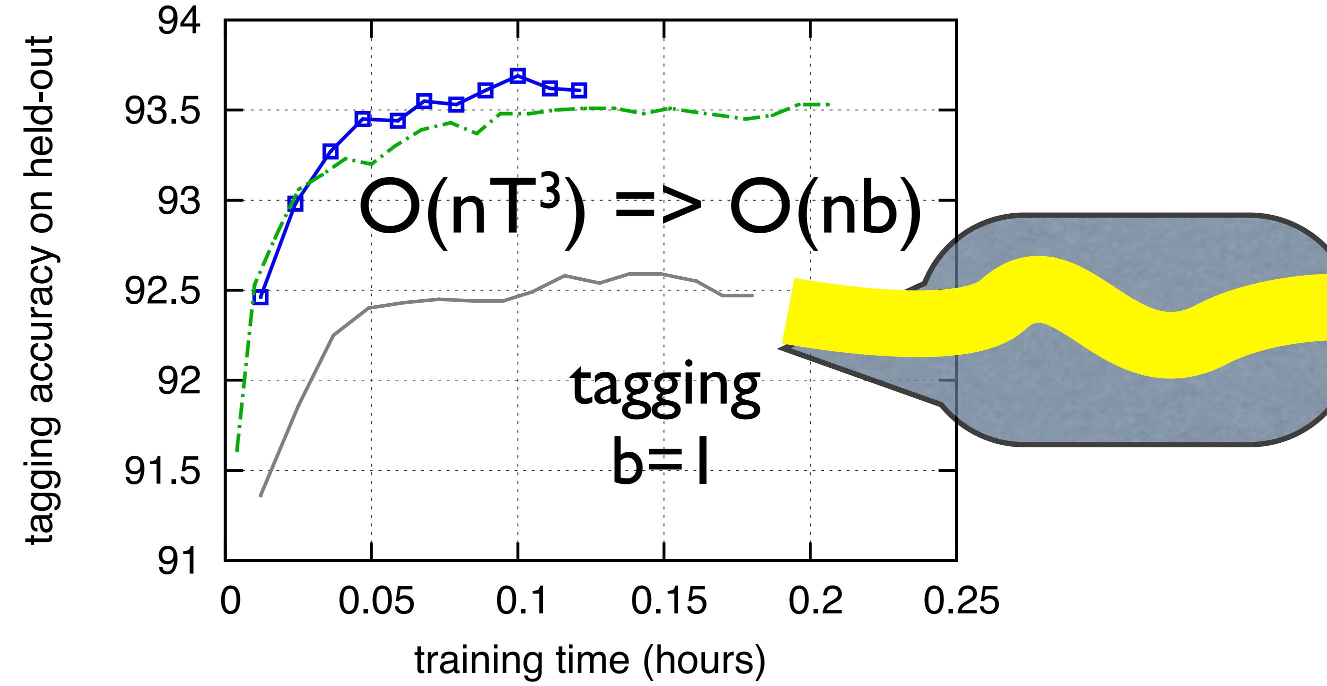


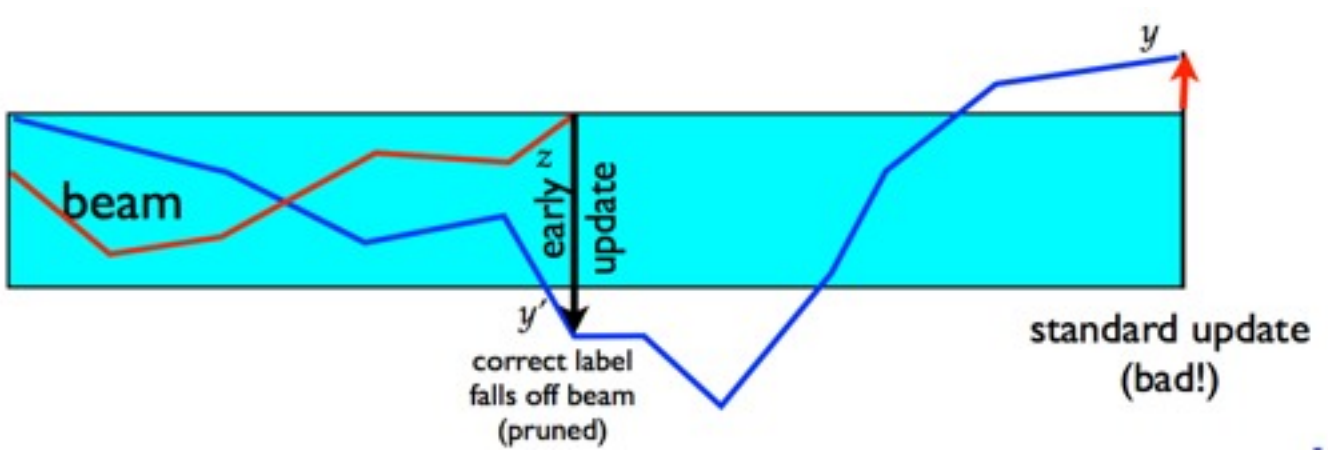


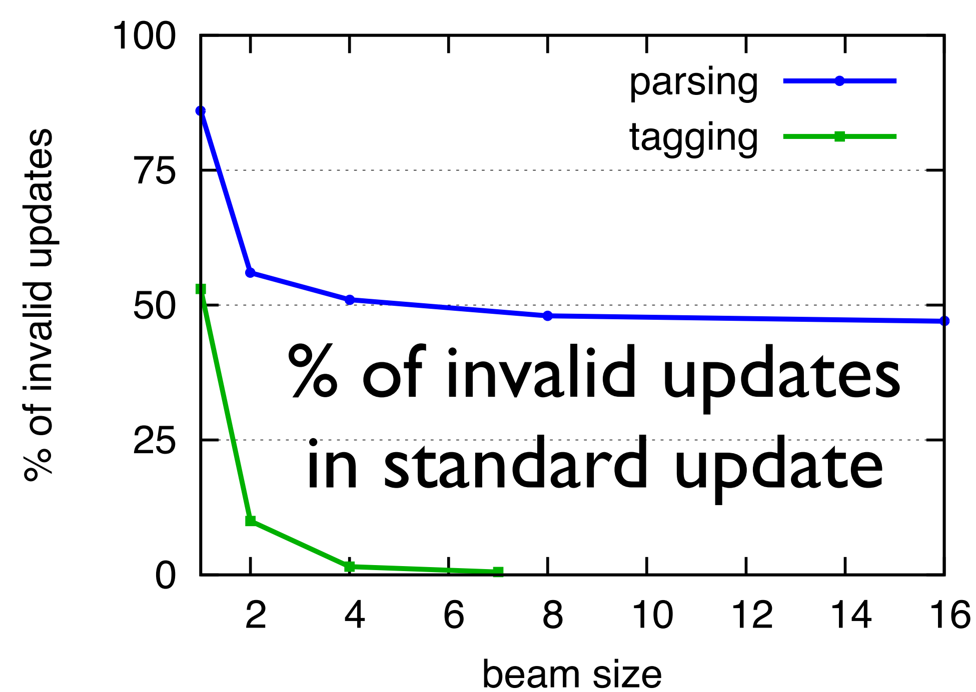
Why standard update so bad for parsing

- standard update works horribly with severe search error
  - due to large number of *invalid* updates (non-violation)



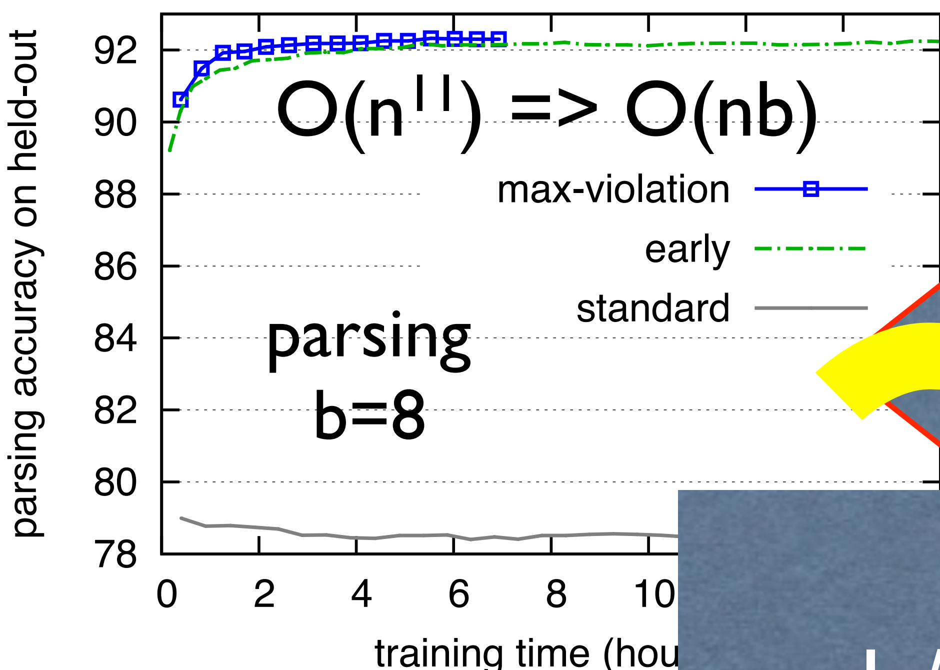


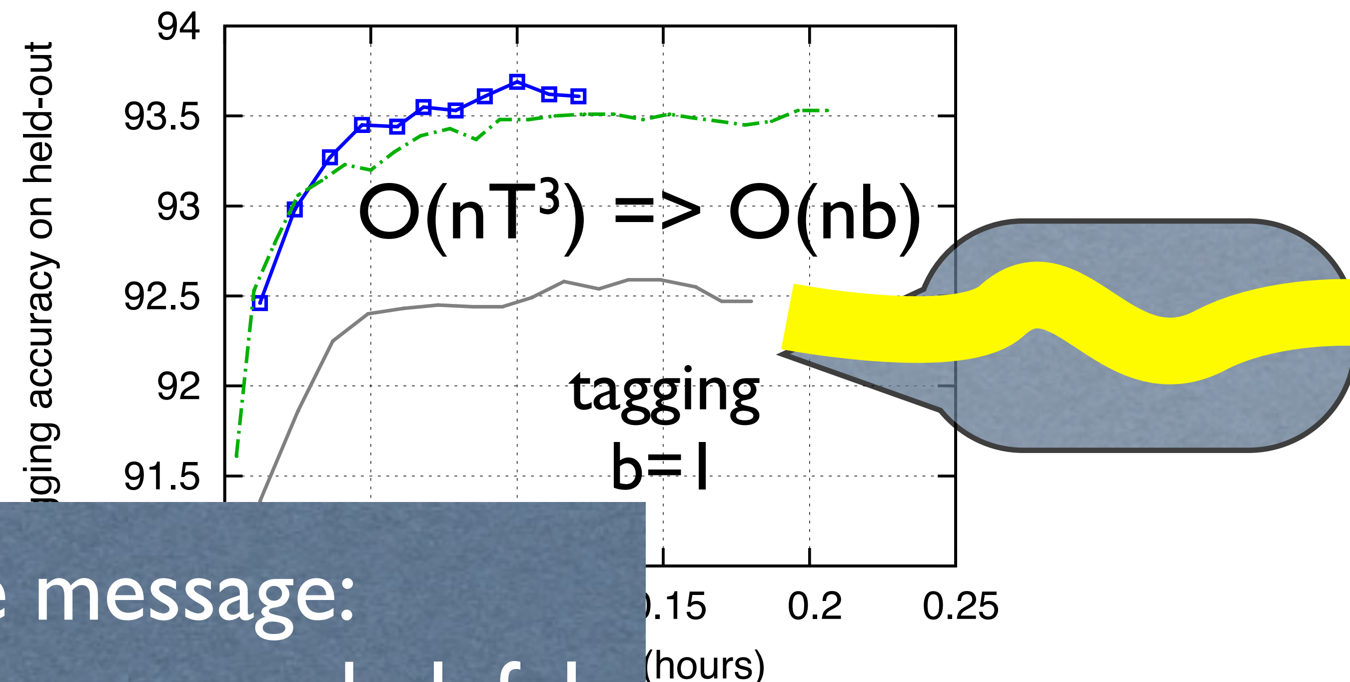




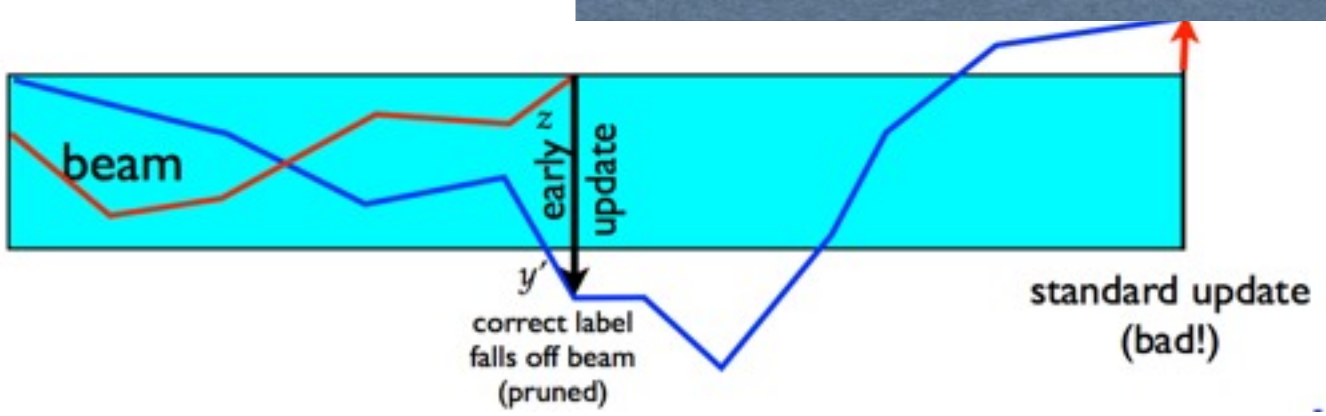
Why standard update so bad for parsing

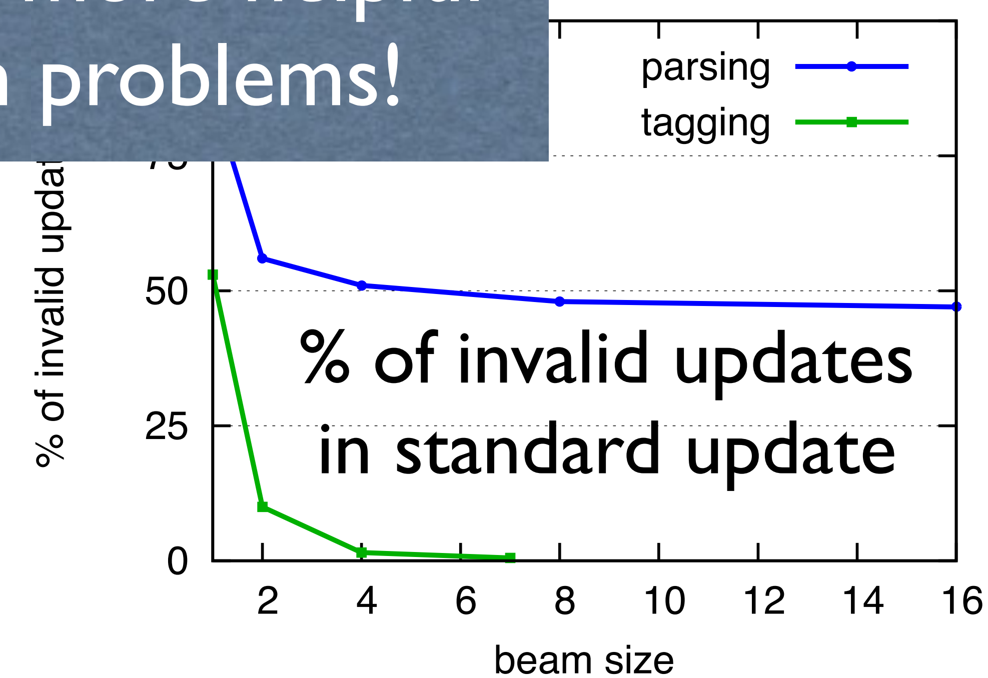
- standard update works horribly with severe search error
    - due to large number of *invalid* updates (non-violation)





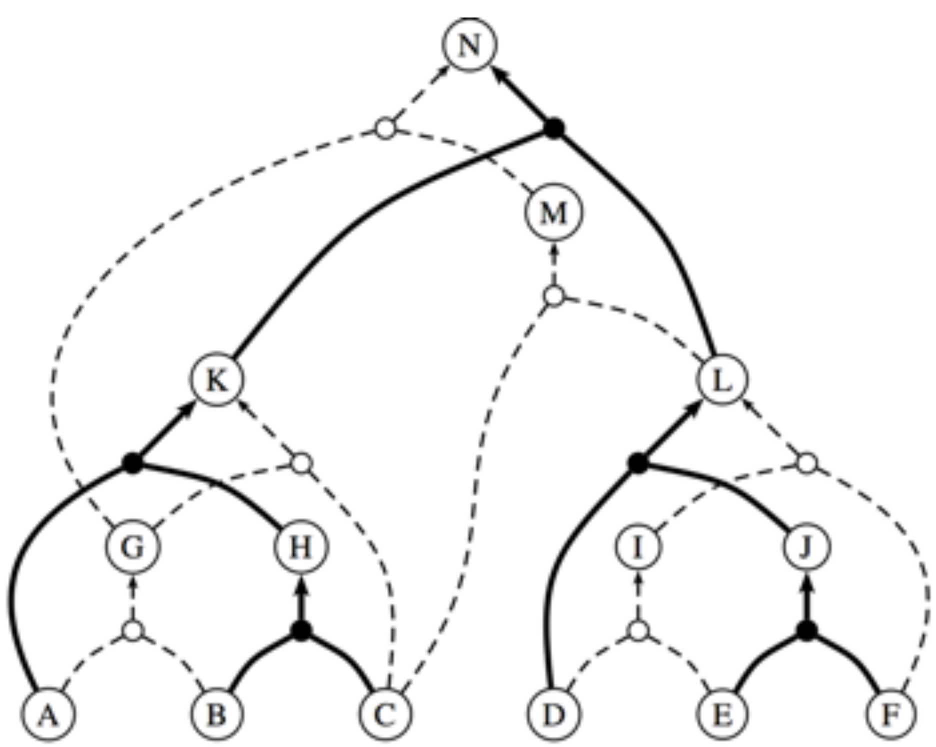
take-home message:
early/max-violation more helpful
for harder search problems!

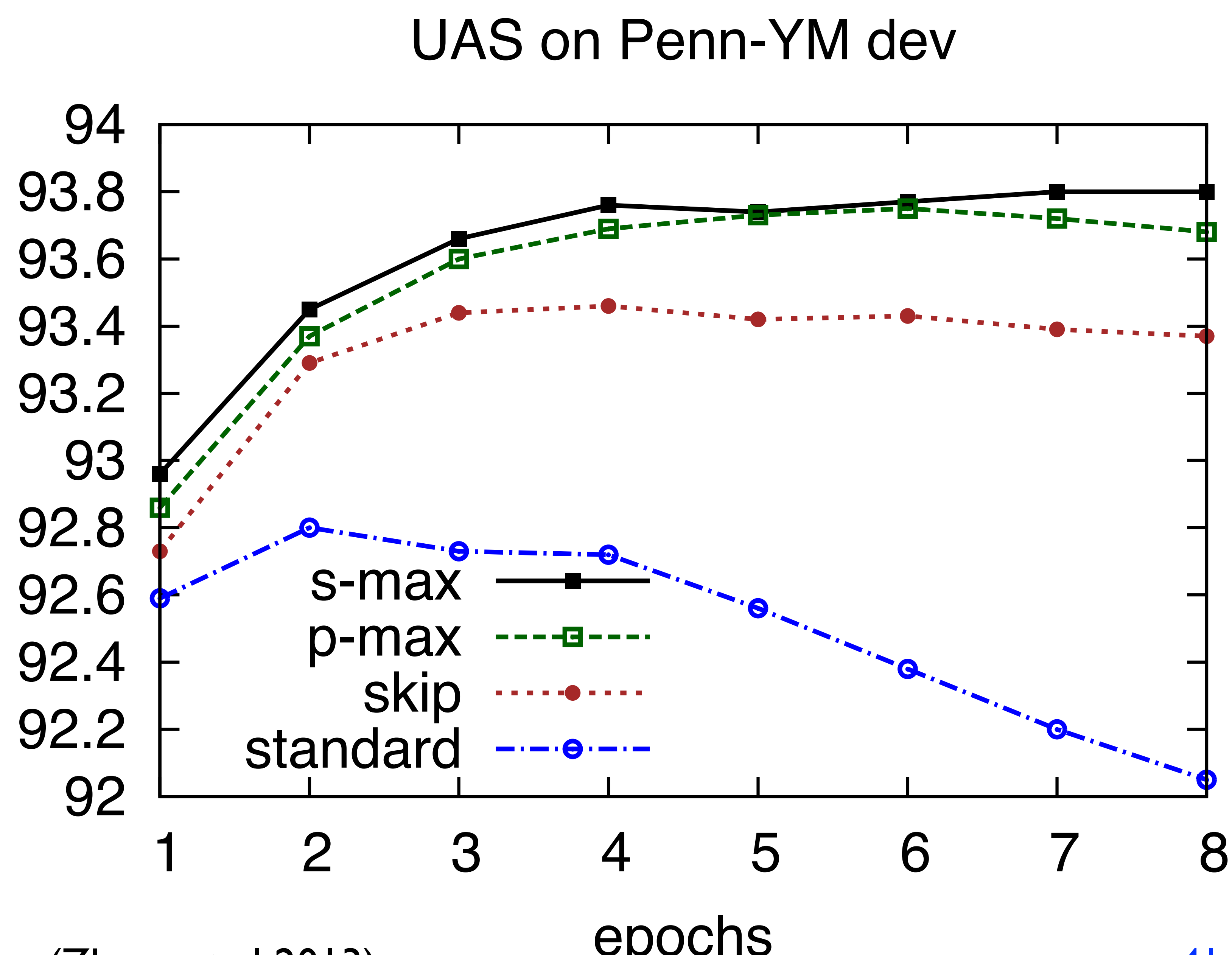




Exp 3: Bottom-up Parsing

- CKY parsing with cube pruning for higher-order features
- we extended our framework from graphs to hypergraphs

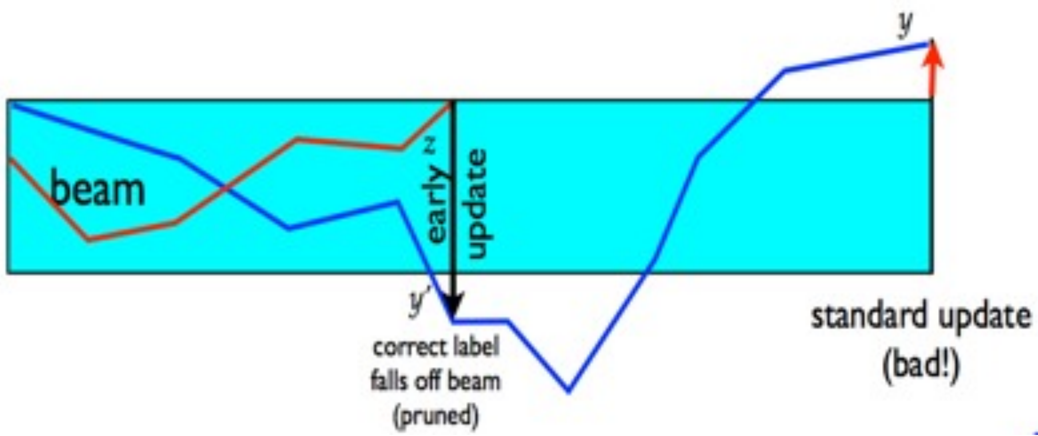


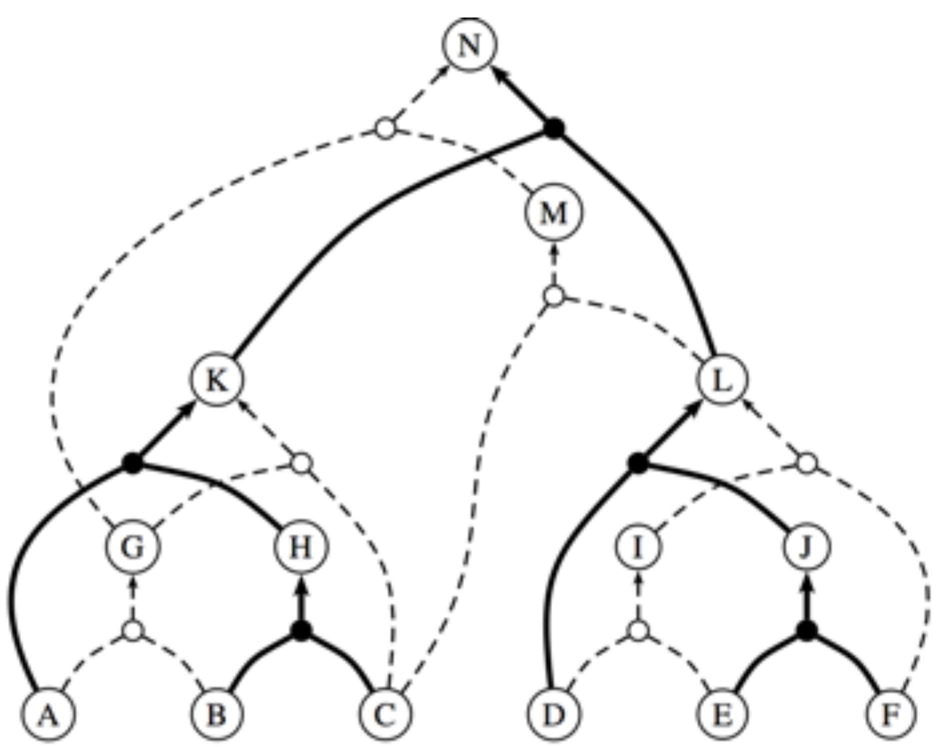


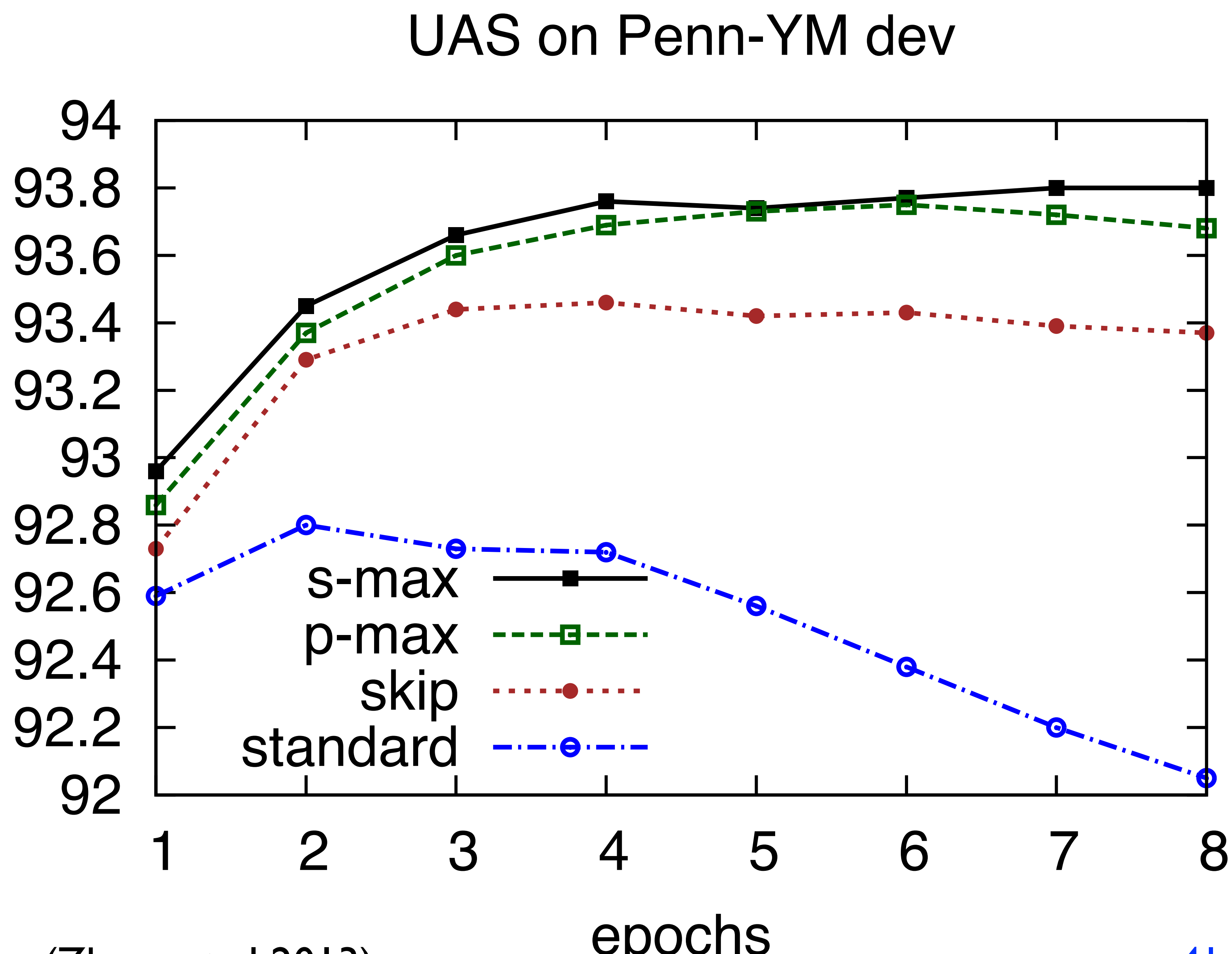
(Zhang et al 2013)

Exp 3: Bottom-up Parsing

- CKY parsing with cube pruning for higher-order features
- we extended our framework from graphs to hypergraphs



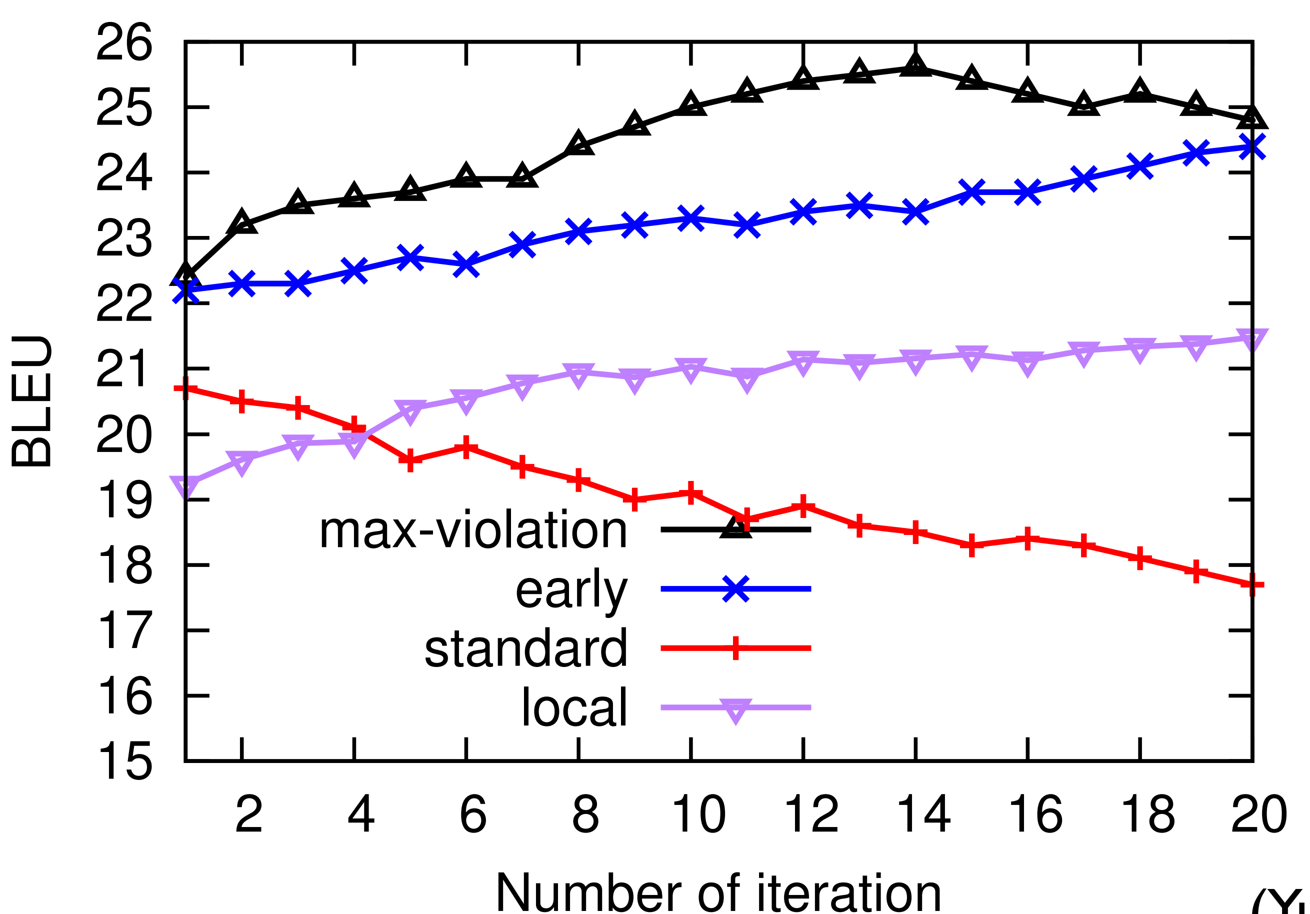




(Zhang et al 2013)

Exp 4: Machine Translation

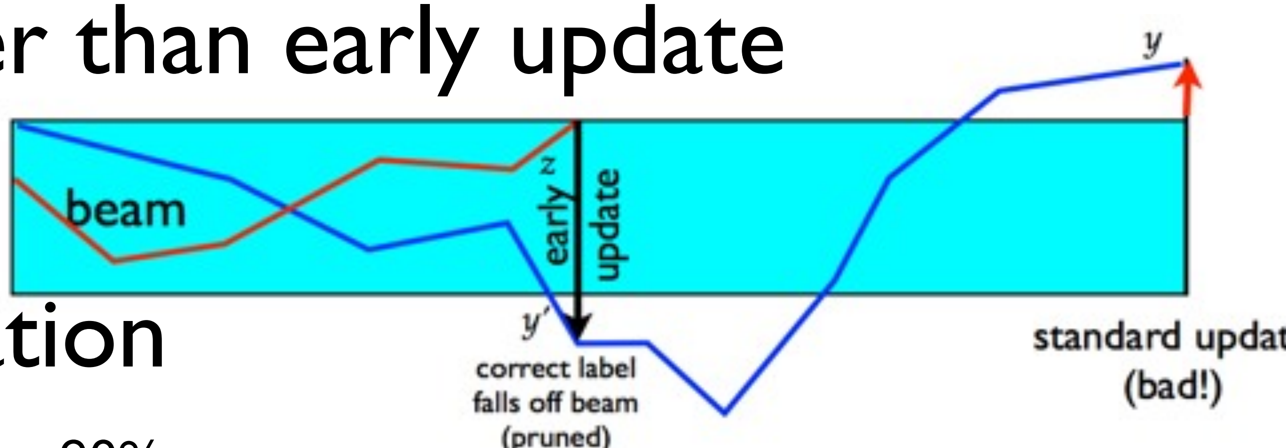
- standard perceptron works poorly for machine translation
  - b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update
- first truly successful effort in large-scale training for translation

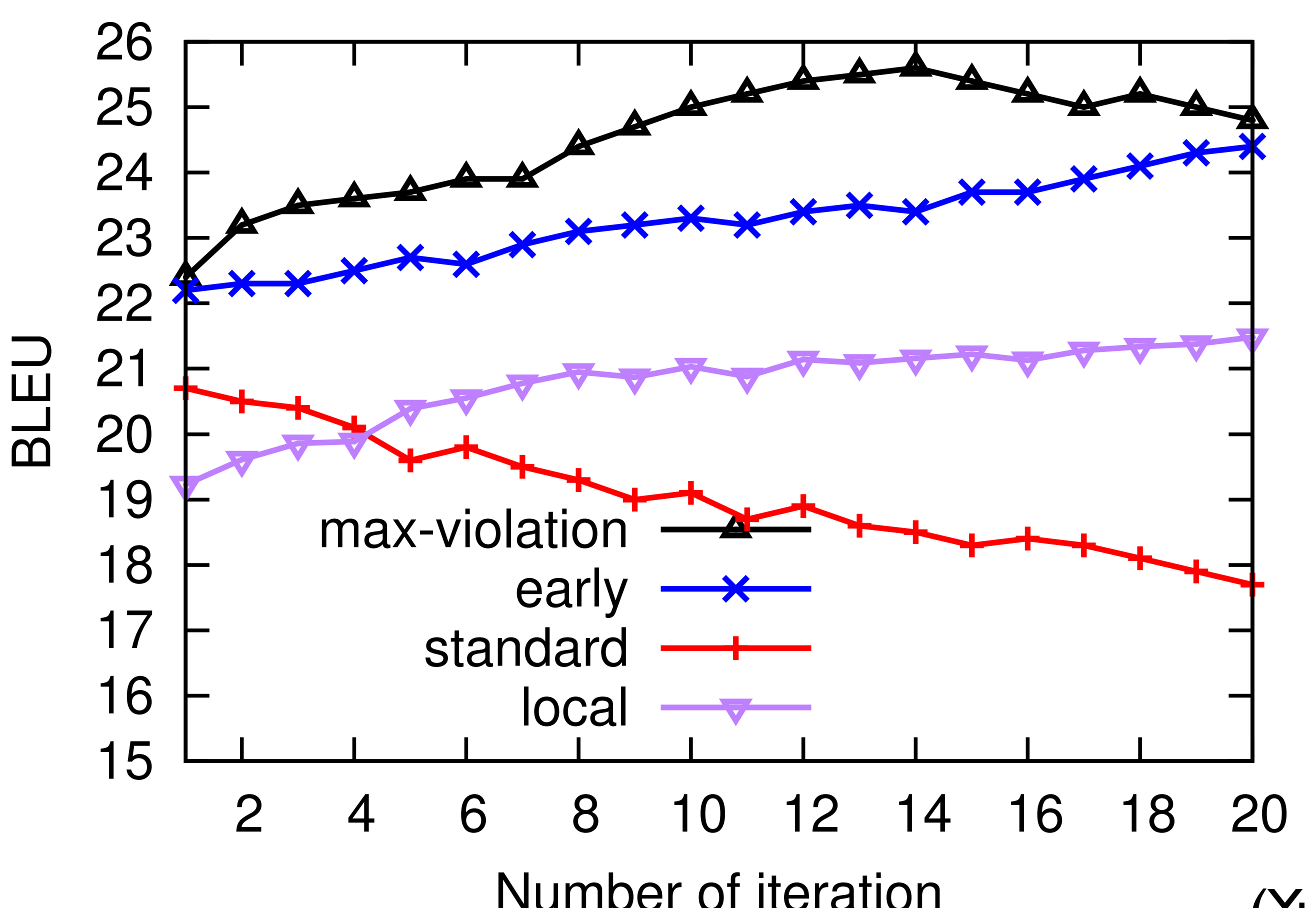


(Yu et al 2013)

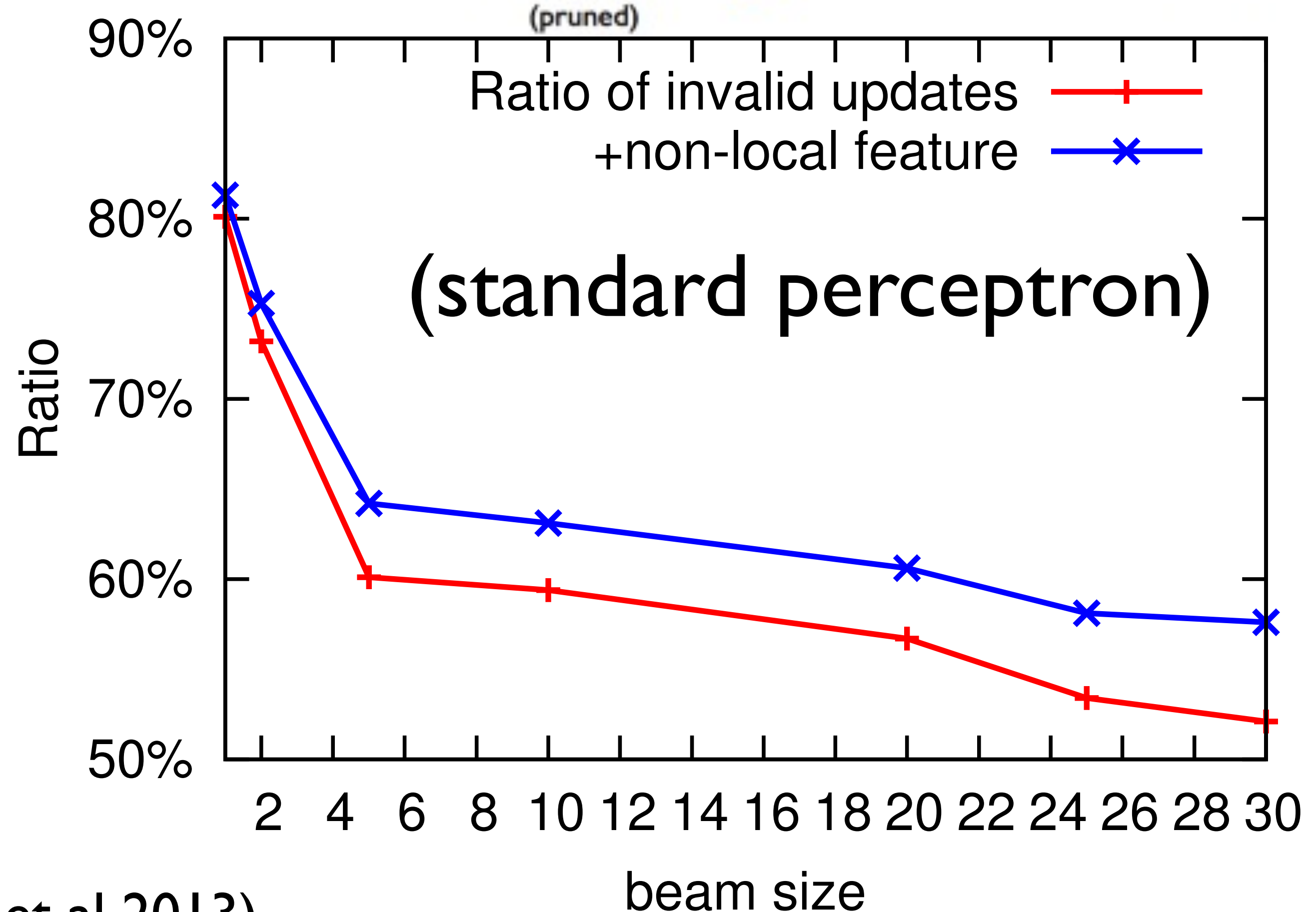
Exp 4: Machine Translation

- standard perceptron works poorly for machine translation
  - b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update
- first truly successful effort in large-scale training for translation



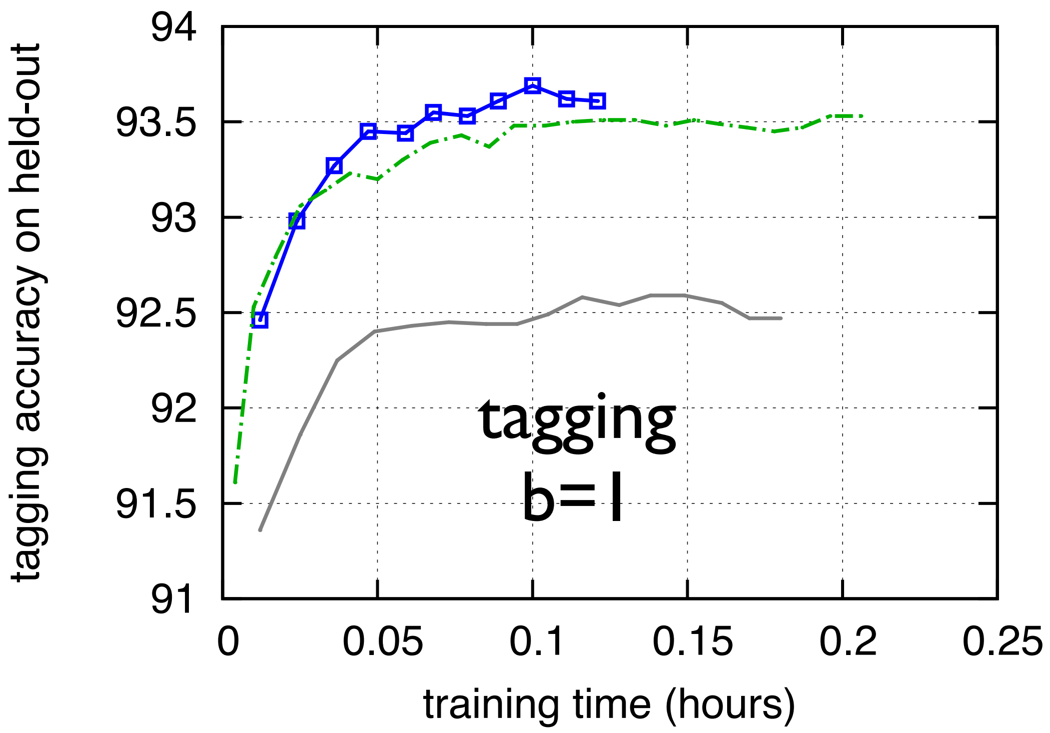


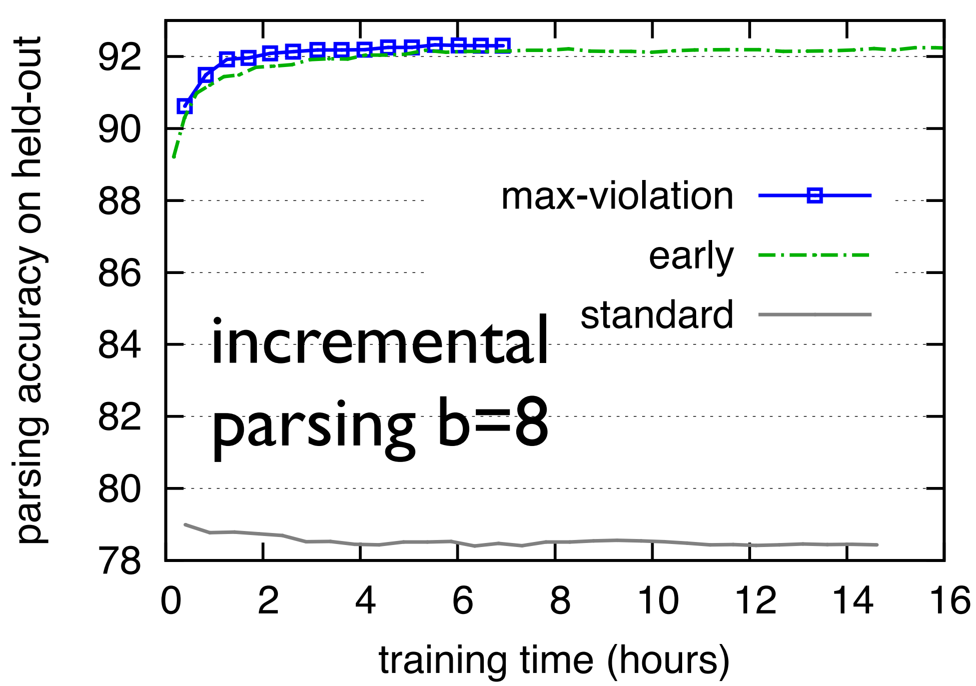
(Yu et al 2013)

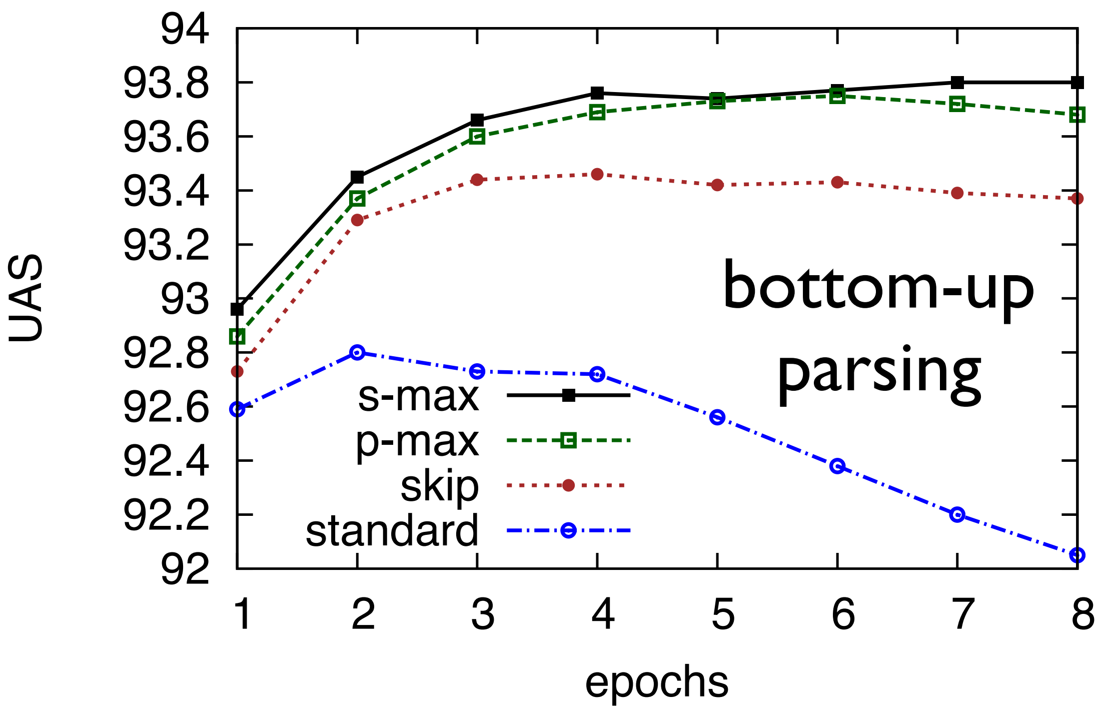


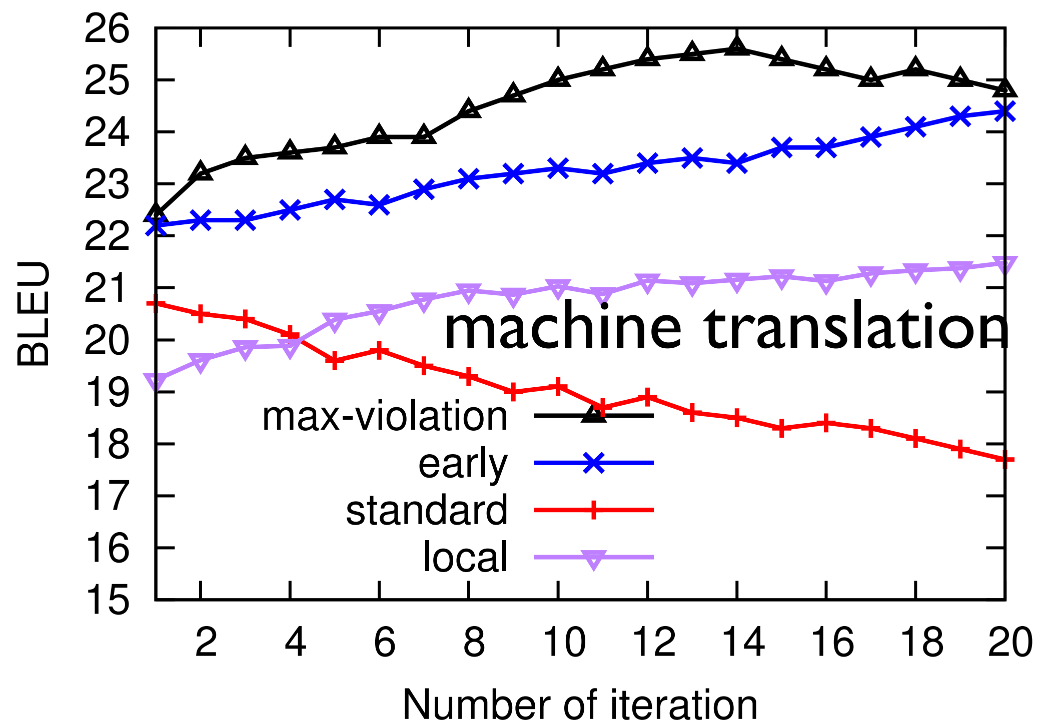
Comparison of Four Exps

- the harder your search, the more advantageous









Related Work and Discussions

Related Work and Discussions

- our “violation-fixing” framework include as special cases
  - early-update (Collins and Roark, 2004)
  - LaSO (Daume and Marcu, 2005)
  - not sure about Searn (Daume et al, 2009)

Related Work and Discussions

- our "violation-fixing" framework include as special cases
  - early-update (Collins and Roark, 2004)
  - LaSO (Daume and Marcu, 2005)
  - not sure about Searn (Daume et al, 2009)
- "beam-separability" or "greedy-separability" related to:
  - "algorithmic-separability" of (Kulesza and Pereira, 2007)
  - but these conditions are too strong to hold in practice

Related Work and Discussions

- our "violation-fixing" framework include as special cases
  - early-update (Collins and Roark, 2004)
  - LaSO (Daume and Marcu, 2005)
  - not sure about Searn (Daume et al, 2009)
- "beam-separability" or "greedy-separability" related to:
  - "algorithmic-separability" of (Kulesza and Pereira, 2007)
  - but these conditions are too strong to hold in practice
- under-generating (beam) vs. over-generating (LP-relax.)
  - Kulesza & Pereira and Martins et al (2011): LP-relaxation
  - Finley and Joachims (2008): both under and over for SVM

Conclusions So Far

- structured perceptron is simple, scalable, and powerful
  - (almost) same convergence proof from multiclass perceptron
- but it doesn't work very well with inexact search
- solution: violation-fixing perceptron framework
  - convergence under new defs of *separability*
  - learn to "live with" search errors
  - in particular, "max-violation" works great
    - converges fast, and results in high accuracy
  - they are more helpful to harder search problems!

Tutorial Outline

- Overview of Structured Learning
  - Challenges in Scalability
- Structured Perceptron
  - convergence proof
- Structured Perceptron with Inexact Search

- Latent-Variable Perceptron
- Parallelizing Online Learning (Perceptron & MIRA)

Learning with Latent Variables

- aka "weakly-supervised" or "partially-observed" learning
- learning from "natural annotations"; more scalable
- examples: translation, transliteration, semantic parsing...

布什 与 沙龙 会谈 x

latent
derivation

Bush talked with Sharon y

parallel text

(Liang et al 2006;
Yu et al 2013;
Xiao and Xiong 2013)

Learning with Latent Variables

- aka "weakly-supervised" or "partially-observed" learning
- learning from "natural annotations"; more scalable
- examples: translation, transliteration, semantic parsing...

布什 与 沙龙 会谈 x

latent derivation

Bush talked with Sharon y

parallel text

(Liang et al 2006;
Yu et al 2013;
Xiao and Xiong 2013)

computer

latent derivation

コンピューター

ko n py u : ta :

transliteration

(Knight & Graehl, 1998;
Kondrak et al 2007, etc.)

Learning with Latent Variables

- aka "weakly-supervised" or "partially-observed" learning
- learning from "natural annotations"; more scalable
- examples: translation, transliteration, semantic parsing...

布什 与 沙龙 会谈 x

latent derivation

Bush talked with Sharon y

parallel text

(Liang et al 2006;
Yu et al 2013;
Xiao and Xiong 2013)

computer

latent derivation

コンピューター

ko n py u : ta :

transliteration

(Knight & Graehl, 1998;
Kondrak et al 2007, etc.)

What is the largest state?

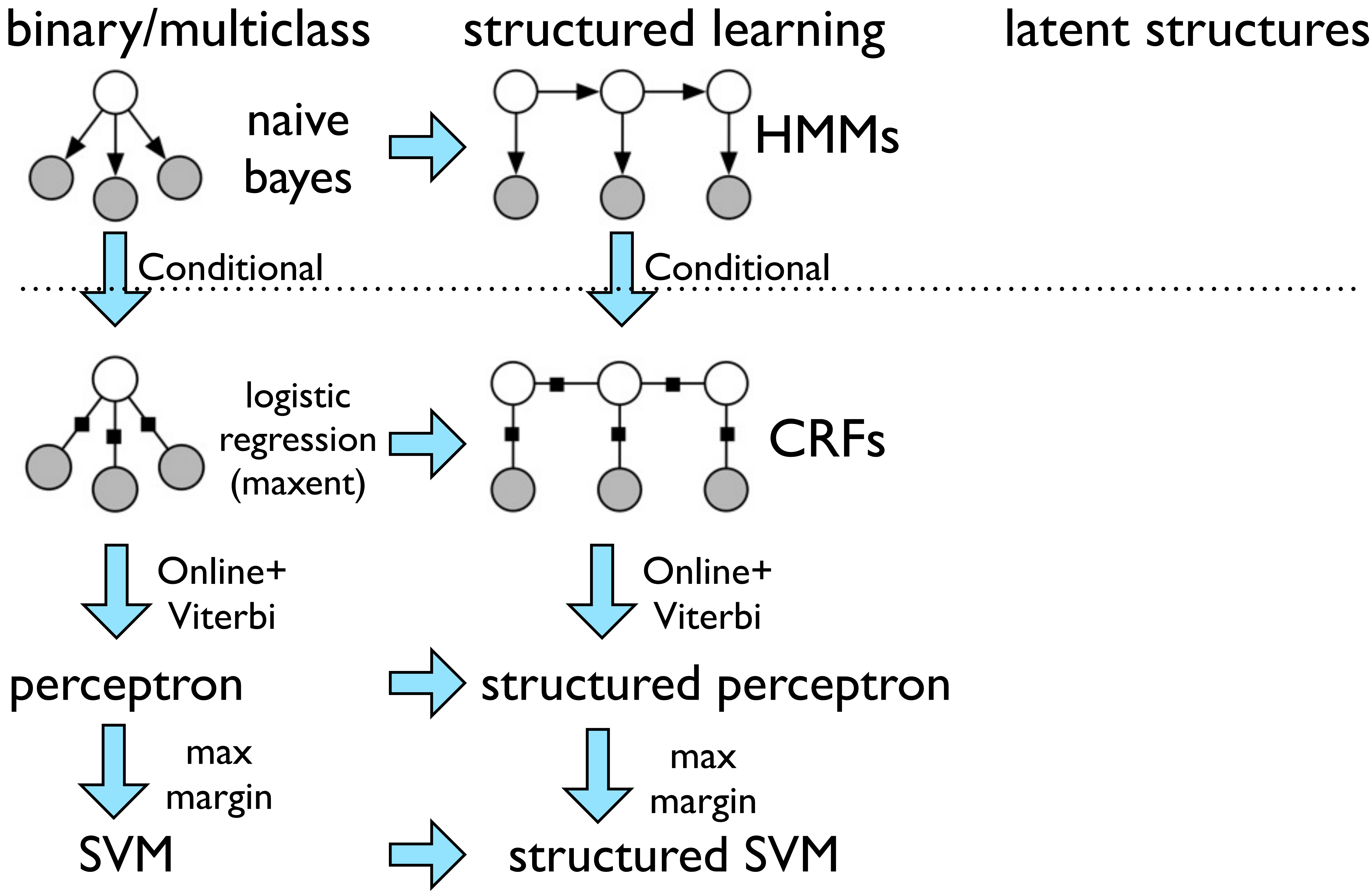
argmax(state, size)

Alaska

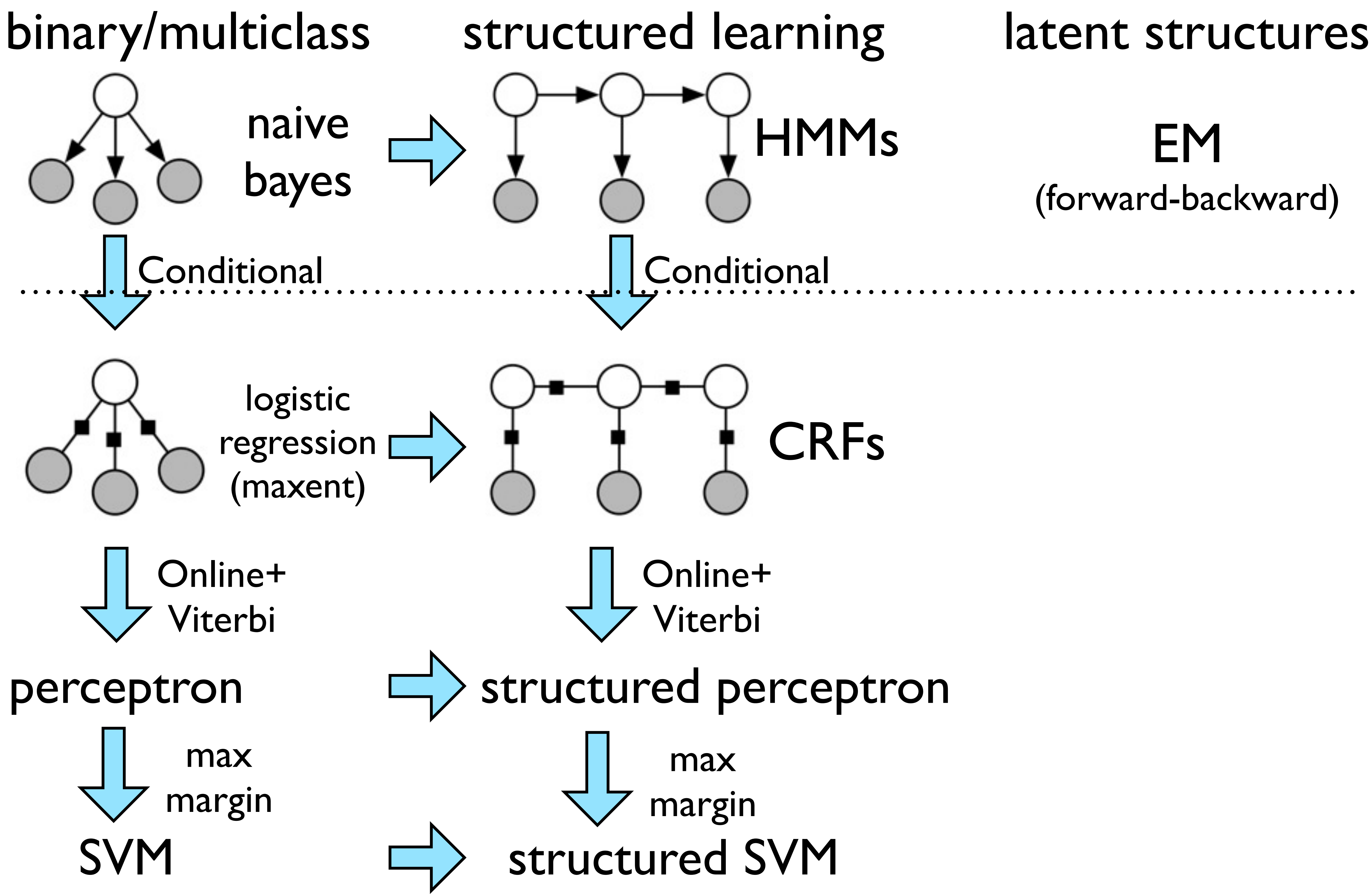
QA pairs

(Clark et al 2010;
Liang et al 2013;
Kwiatkowski et al 2013)

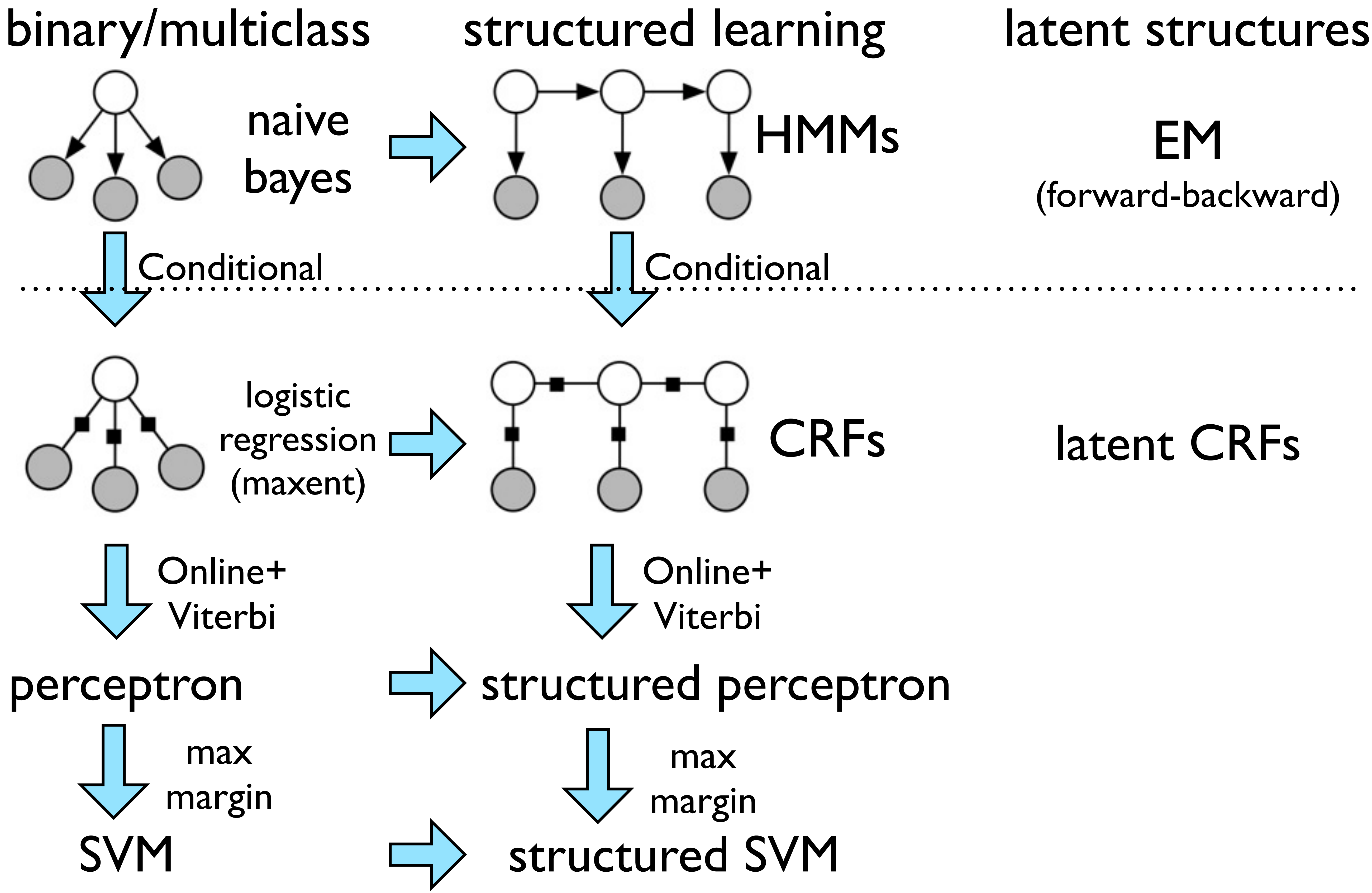
Learning Latent Structures



Learning Latent Structures

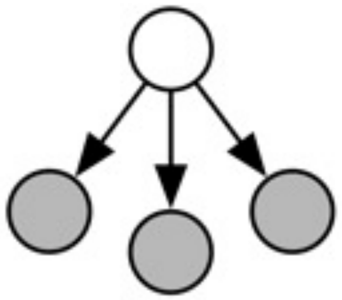


Learning Latent Structures

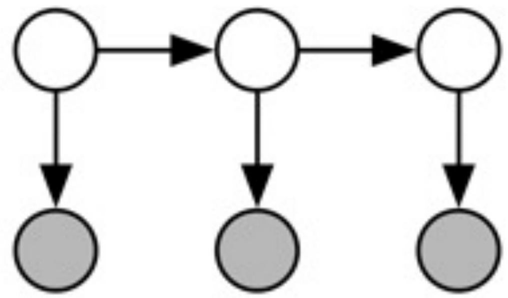


Learning Latent Structures

binary/multiclass | structured learning | latent structures



naive bayes ⇒

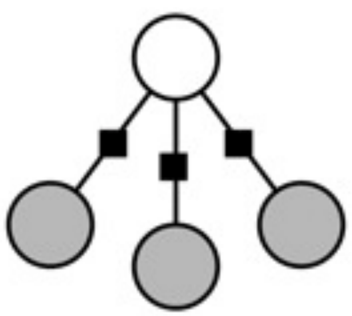


HMMs

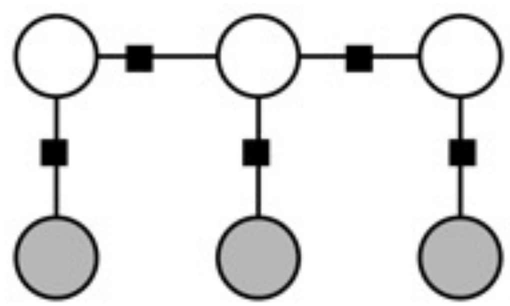
EM
(forward-backward)

Conditional

Conditional



logistic regression (maxent) ⇒



CRFs

latent CRFs

Online+ Viterbi

Online+ Viterbi

perceptron ⇒ structured perceptron

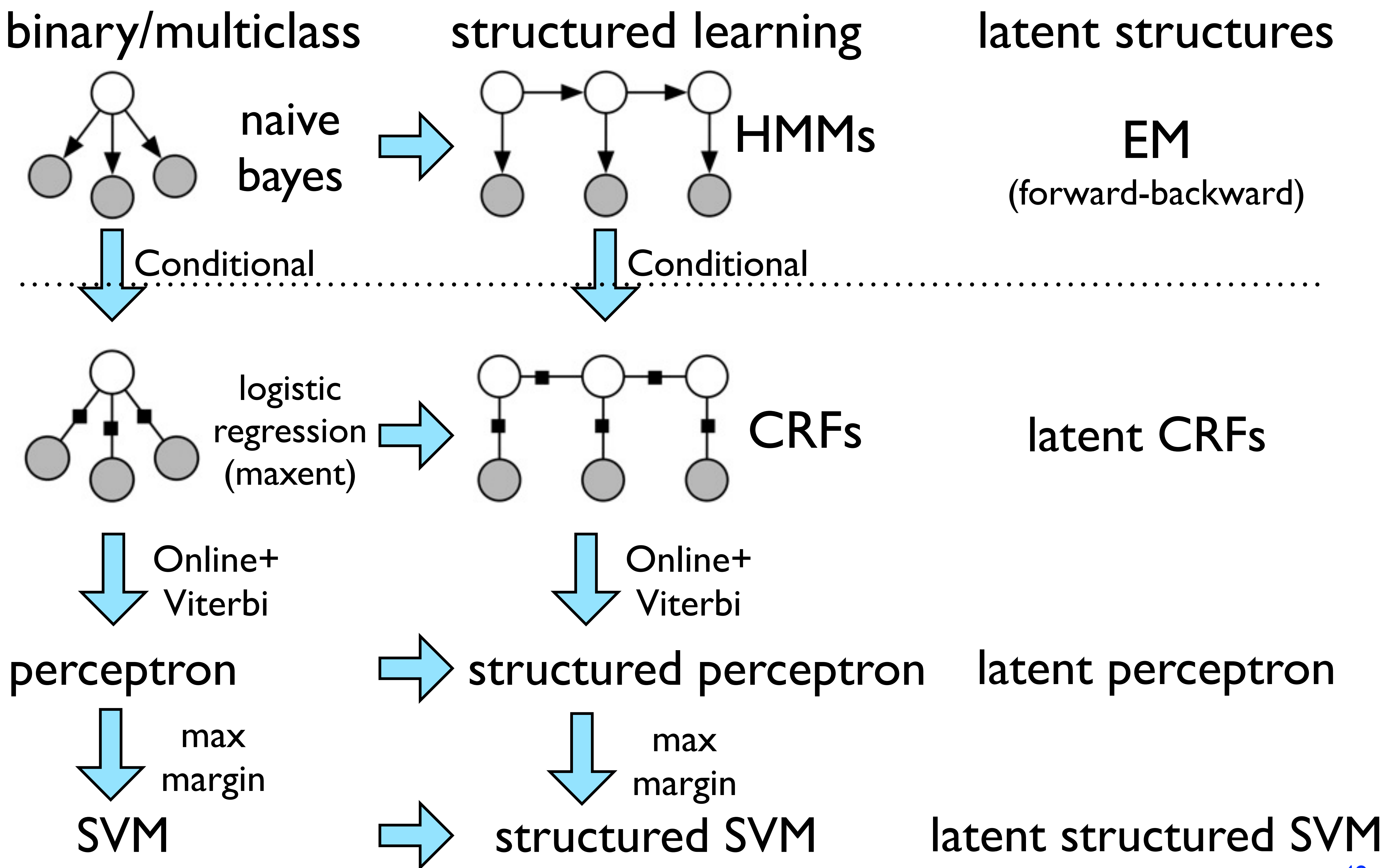
latent perceptron

max margin

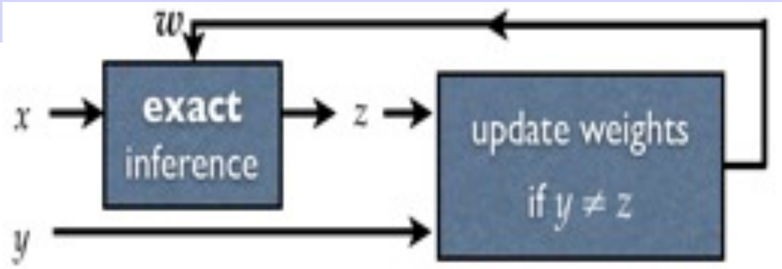
max margin

SVM ⇒ structured SVM

Learning Latent Structures



Latent Structured Perceptron



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

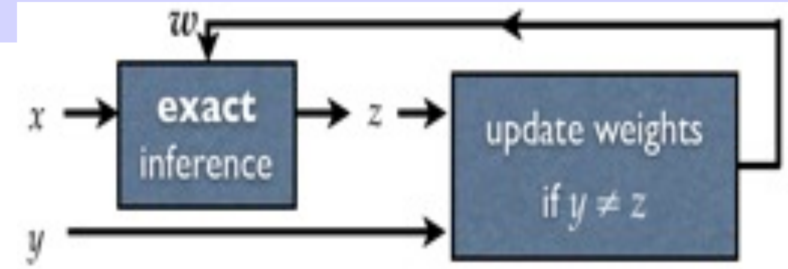
training example

那	人	咬	了	狗	x

the	man	bit	the	dog	y

(Liang et al 2006;
Yu et al 2013)

Latent Structured Perceptron



- no explicit positive signal
  - hallucinate the “correct” derivation by current weights

training example

那	人	咬	了	狗

x

the	man	bit	the	dog

y

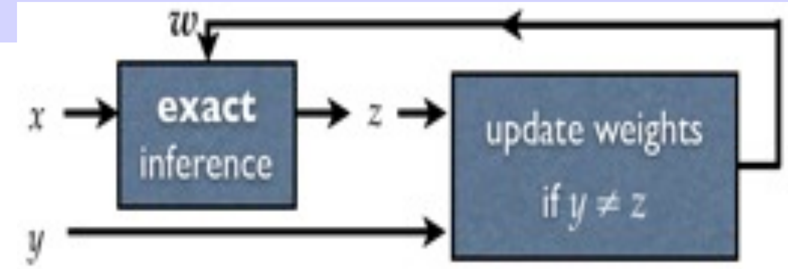
during online learning...

那	人	咬	了	狗

x

(Liang et al 2006;
Yu et al 2013)

Latent Structured Perceptron



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

training example

那	人	咬	了	狗

x

the	man	bit	the	dog

y

during online learning...

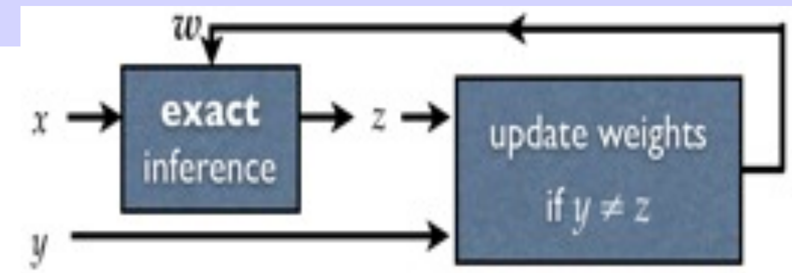
那	人	咬	了	狗

x

full
search
space

(Liang et al 2006;
Yu et al 2013)

Latent Structured Perceptron



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

training example

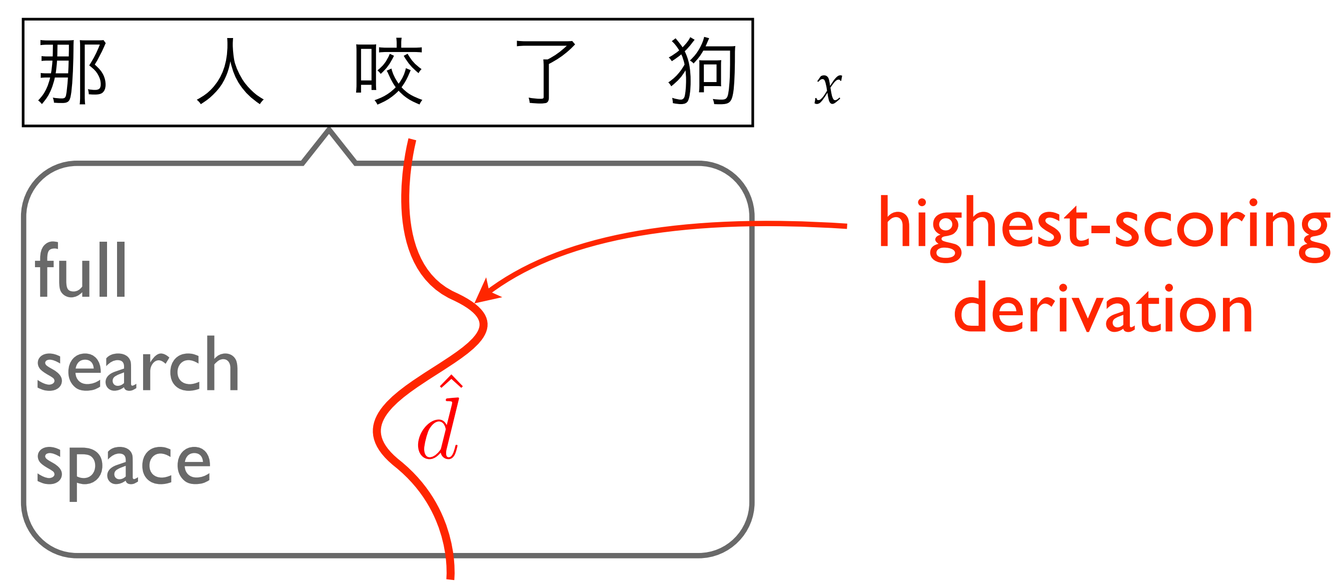
那	人	咬	了	狗

x

the	man	bit	the	dog

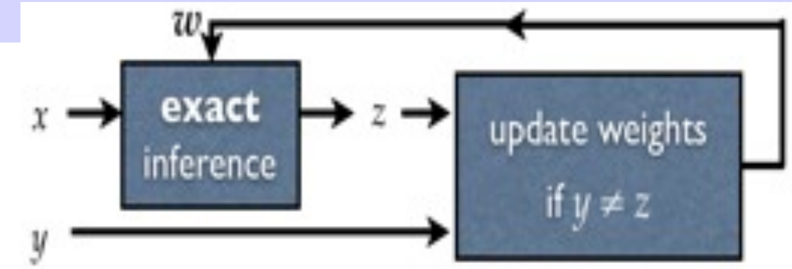
y

during online learning...



(Liang et al 2006;
Yu et al 2013)

Latent Structured Perceptron



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

training example

那	人	咬	了	狗

x

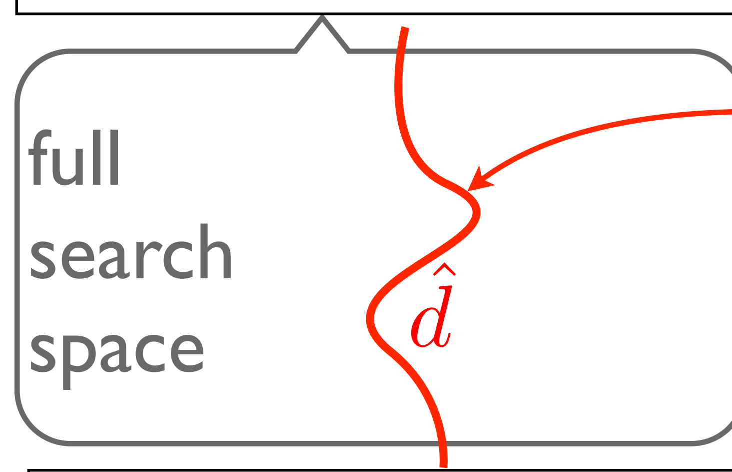
the	man	bit	the	dog

y

during online learning...

那	人	咬	了	狗

x



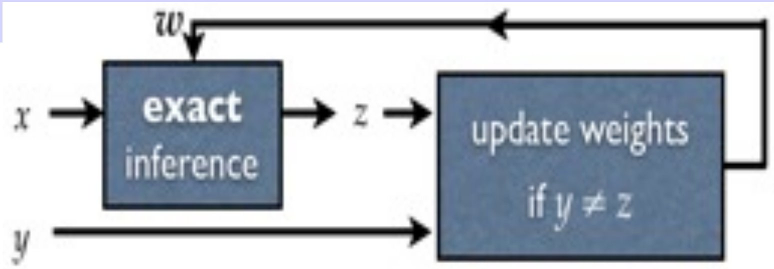
highest-scoring derivation

the	dog	bit	the	man

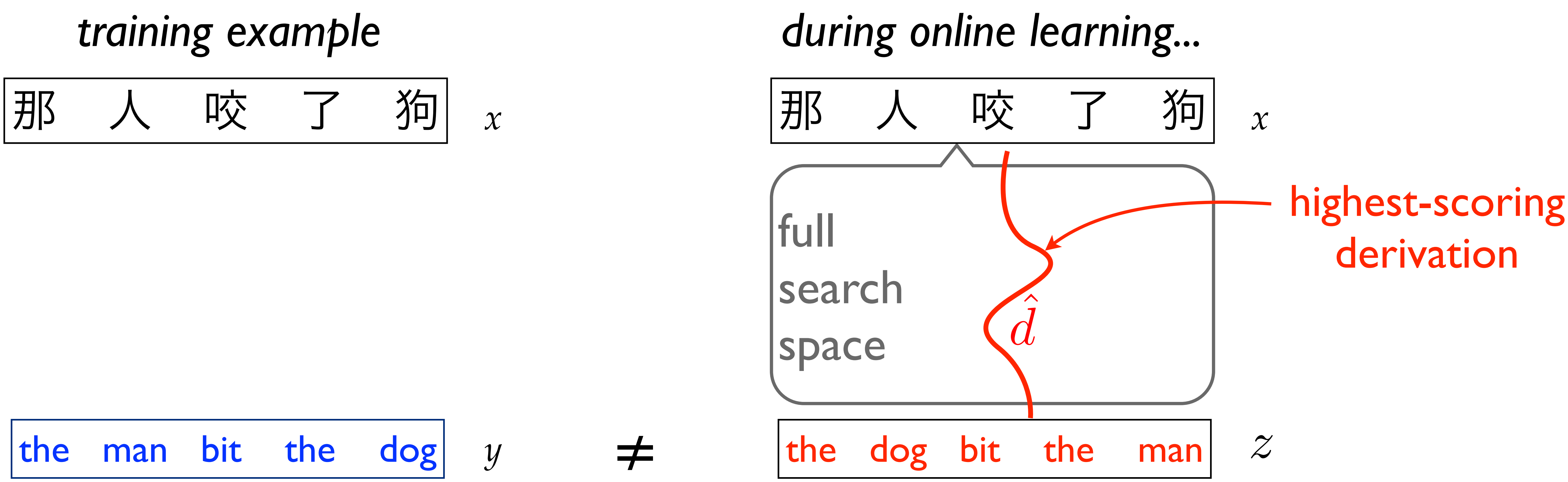
z

(Liang et al 2006;
Yu et al 2013)

Latent Structured Perceptron

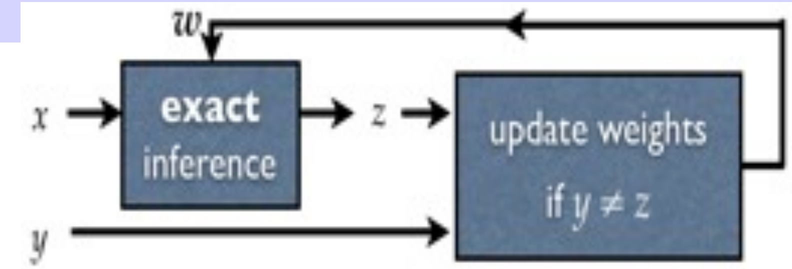


- no explicit positive signal
  - hallucinate the "correct" derivation by current weights



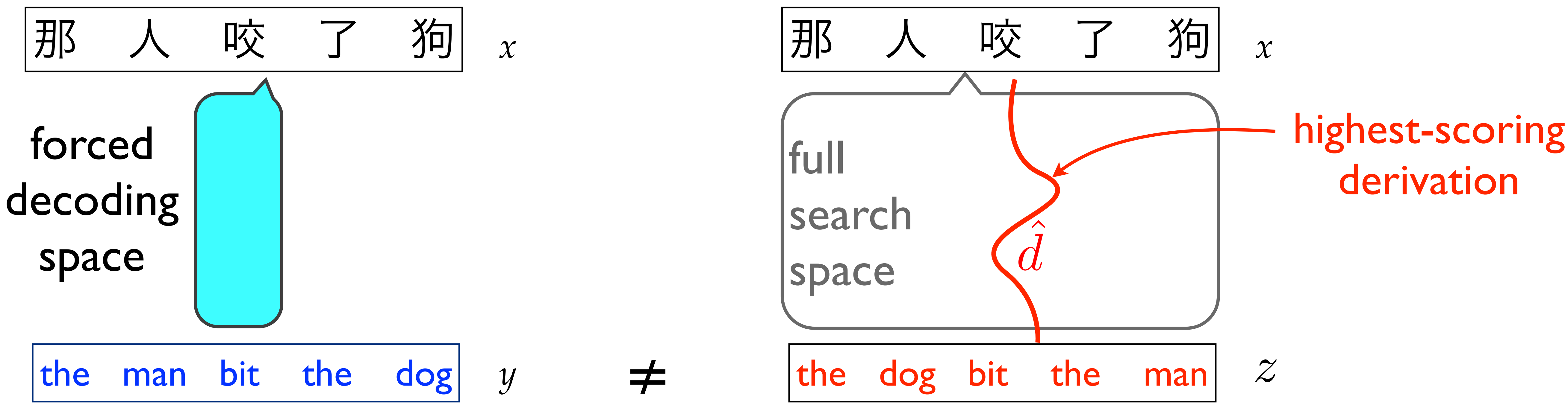
(Liang et al 2006;
Yu et al 2013)

Latent Structured Perceptron



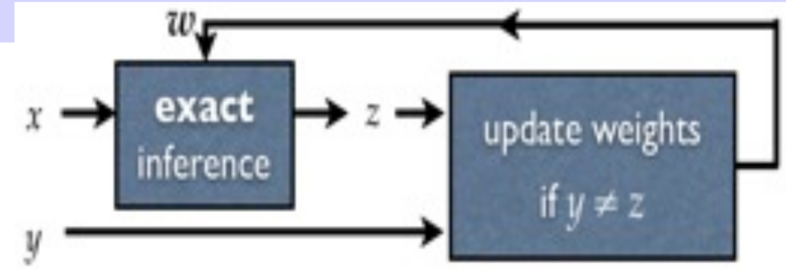
- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

training example *during online learning...*

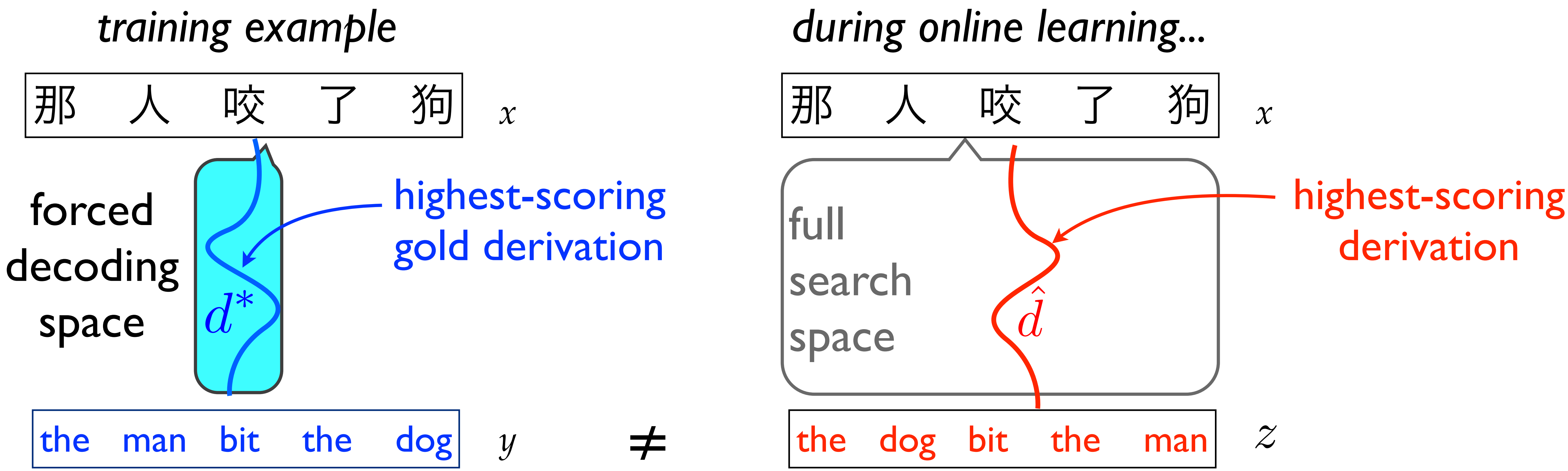


(Liang et al 2006; Yu et al 2013)

Latent Structured Perceptron

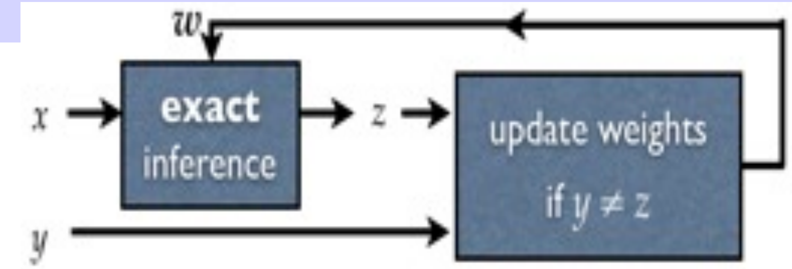


- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

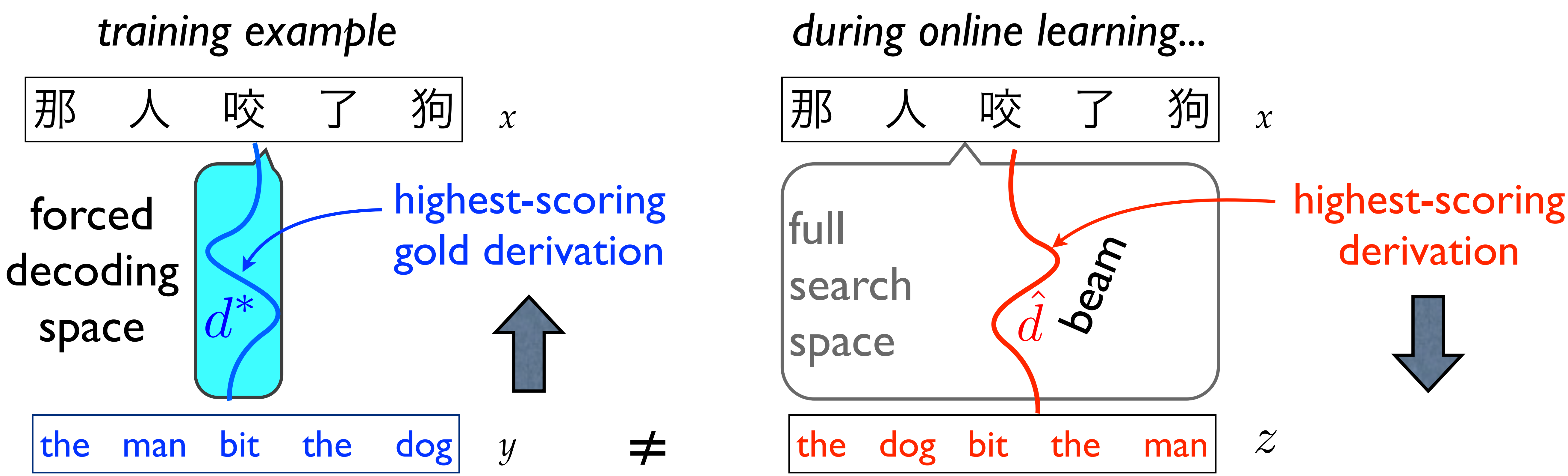


(Liang et al 2006;
Yu et al 2013)

Latent Structured Perceptron



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights



$$\mathbf{w} \leftarrow \mathbf{w} + \Phi(x, d^*) - \Phi(x, \hat{d})$$

reward correct

penalize wrong

(Liang et al 2006; Yu et al 2013)

Unconstrained Search

- example: beam search phrase-based decoding

Bushi yu Shalong juxing le huitan → *???*

Unconstrained Search

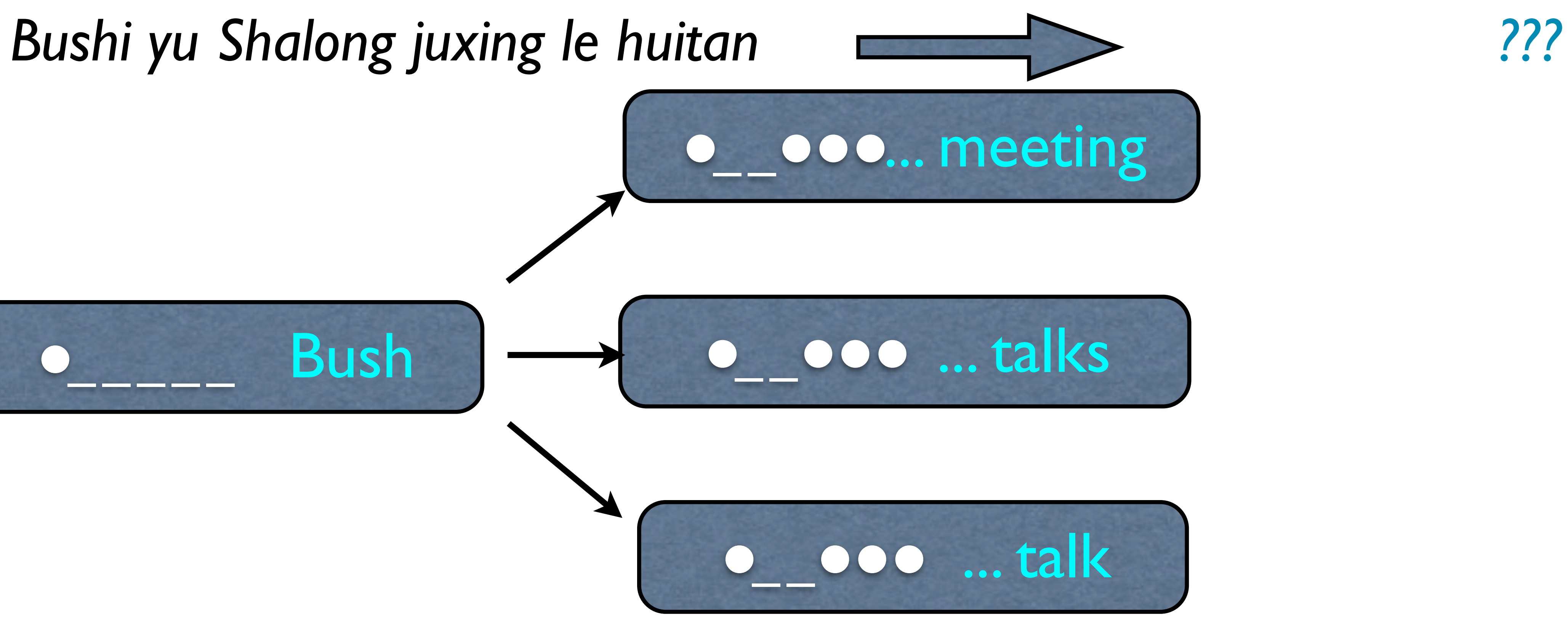
- example: beam search phrase-based decoding

Bushi yu Shalong juxing le huitan → *???*

●_____ Bush

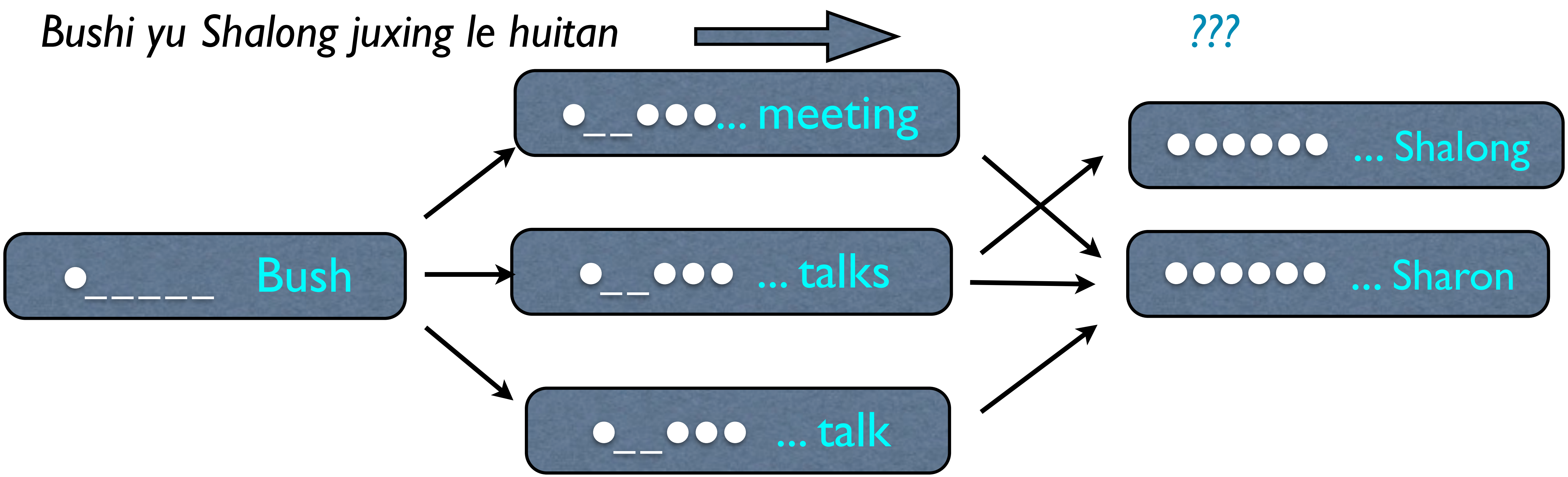
Unconstrained Search

- example: beam search phrase-based decoding



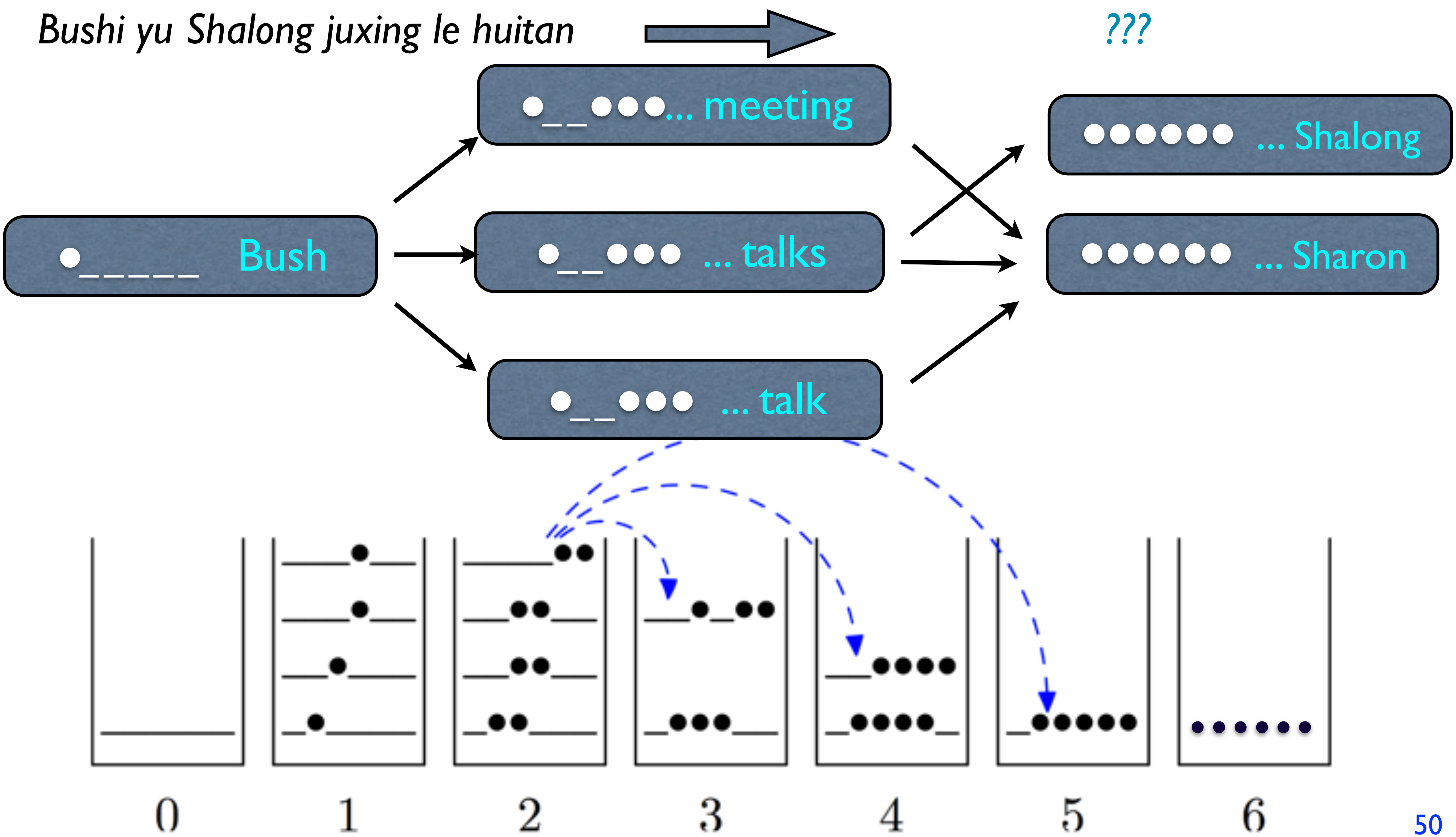
Unconstrained Search

- example: beam search phrase-based decoding



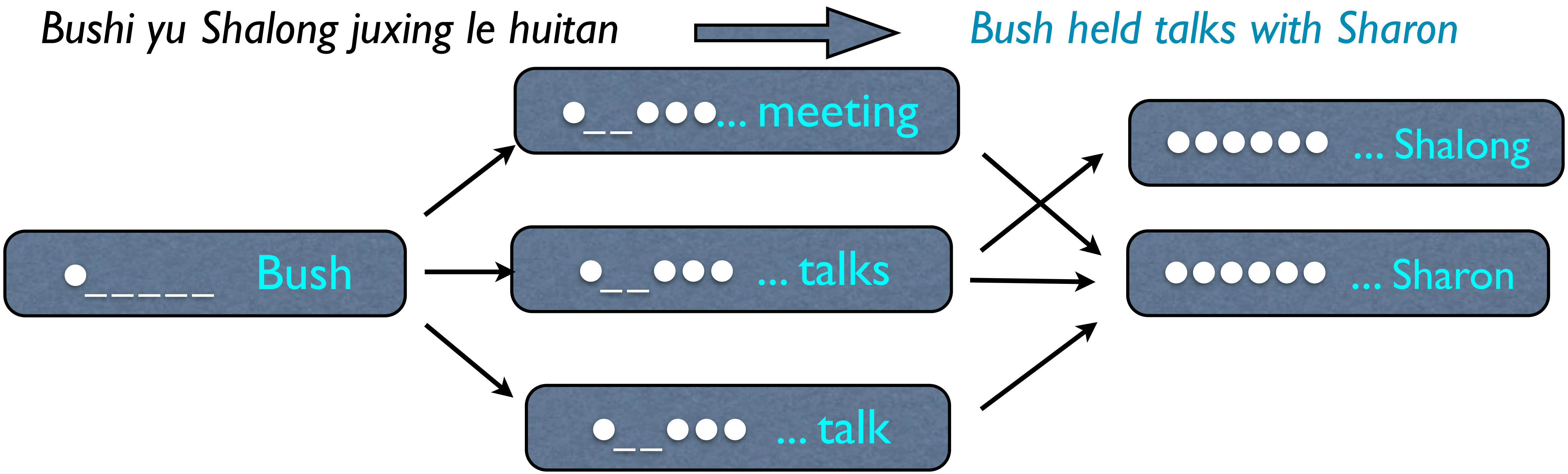
Unconstrained Search

- example: beam search phrase-based decoding



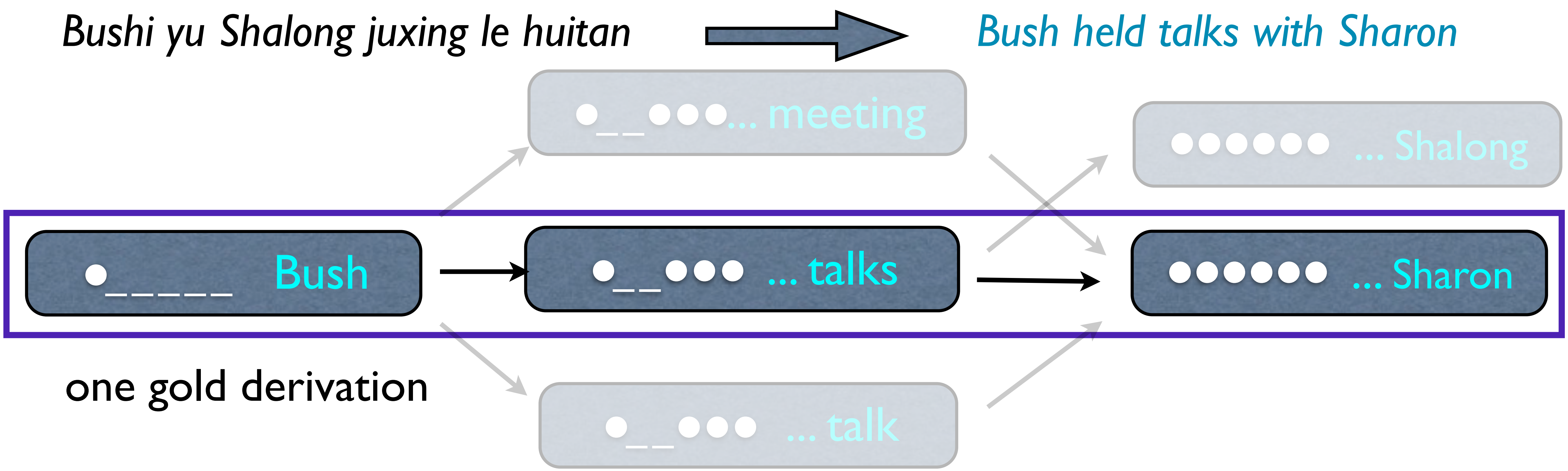
Constrained Search

- forced decoding: must produce the exact reference translation



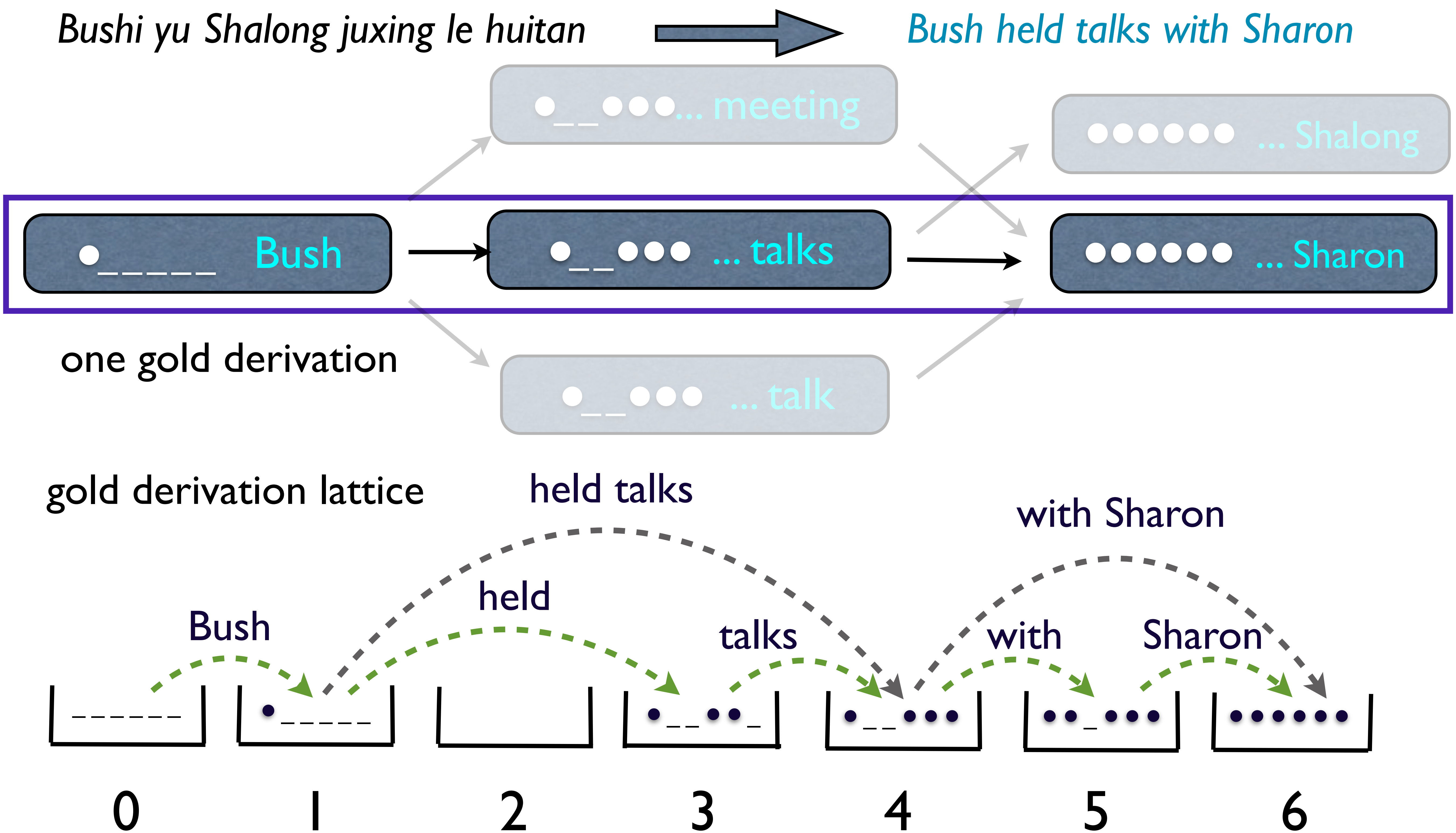
Constrained Search

- forced decoding: must produce the exact reference translation

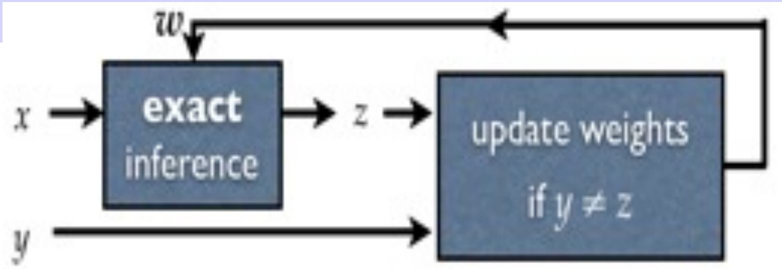


Constrained Search

- forced decoding: must produce the exact reference translation



Search Errors in Decoding



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

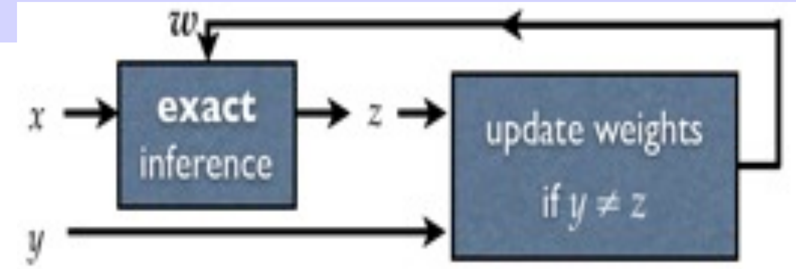
training example

那	人	咬	了	狗	x

the	man	bit	the	dog	y

(Liang et al 2006;
Yu et al 2013)

Search Errors in Decoding



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

training example

那	人	咬	了	狗	x

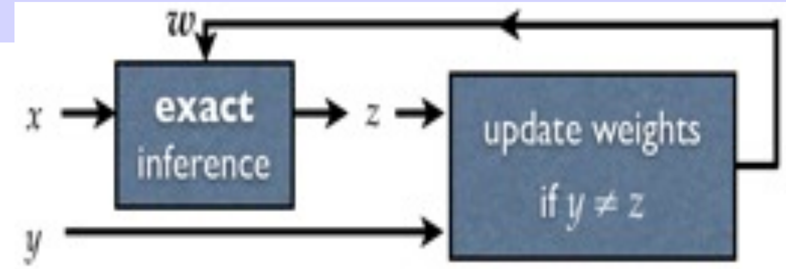
the	man	bit	the	dog	y

during online learning...

那	人	咬	了	狗	x

(Liang et al 2006;
Yu et al 2013)

Search Errors in Decoding



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

training example

那	人	咬	了	狗

x

the	man	bit	the	dog

y

during online learning...

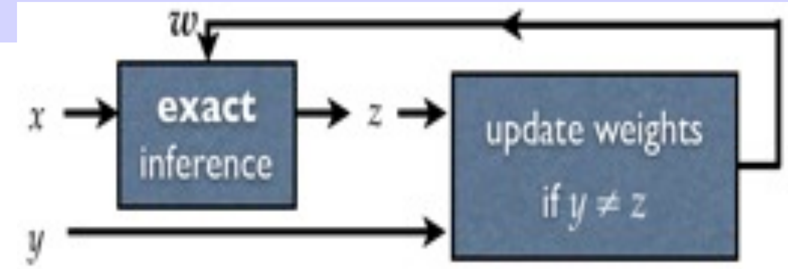
那	人	咬	了	狗

x

full
search
space

(Liang et al 2006;
Yu et al 2013)

Search Errors in Decoding



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

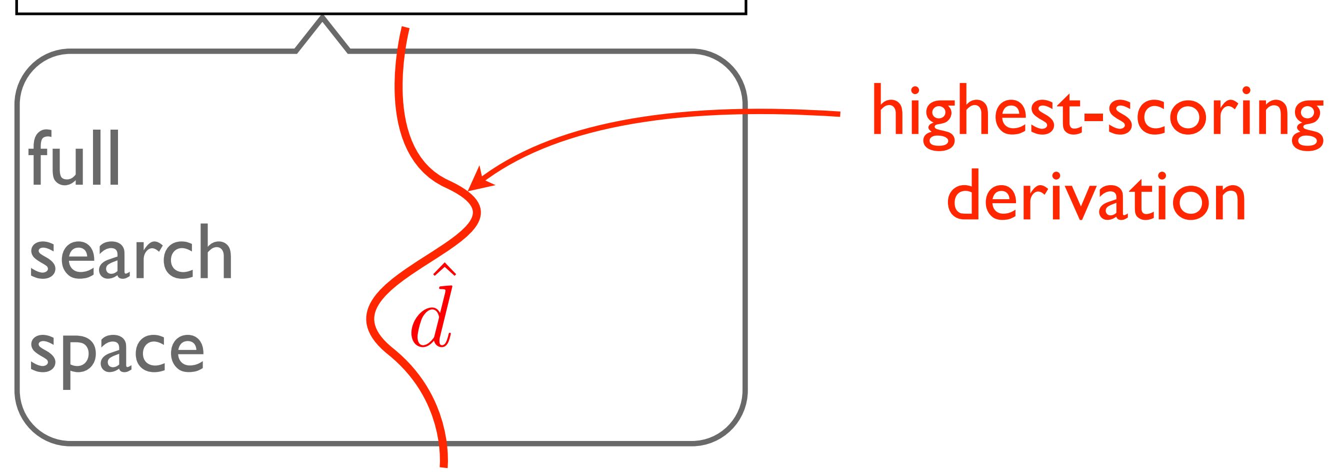
training example

那	人	咬	了	狗	x

the	man	bit	the	dog	y

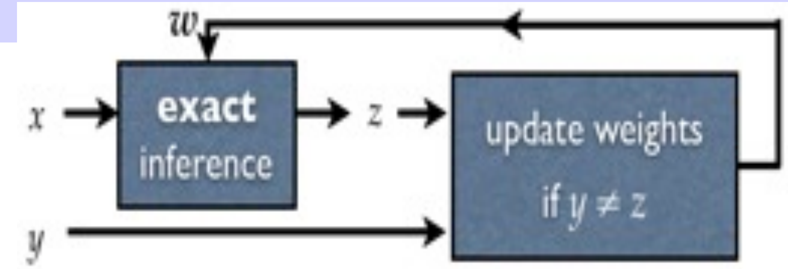
during online learning...

那	人	咬	了	狗	x



(Liang et al 2006;
Yu et al 2013)

Search Errors in Decoding



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

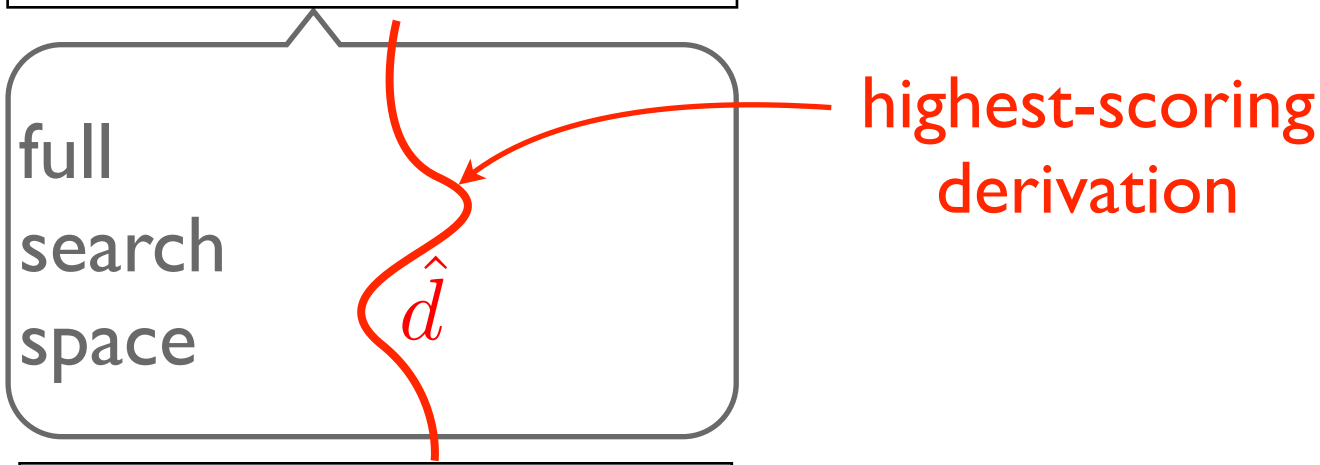
training example

那 人 咬 了 狗 x

the man bit the dog y

during online learning...

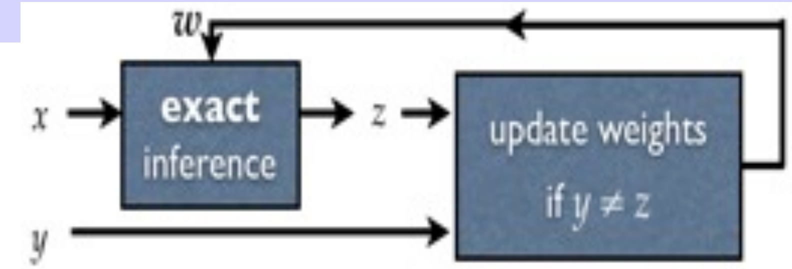
那 人 咬 了 狗 x



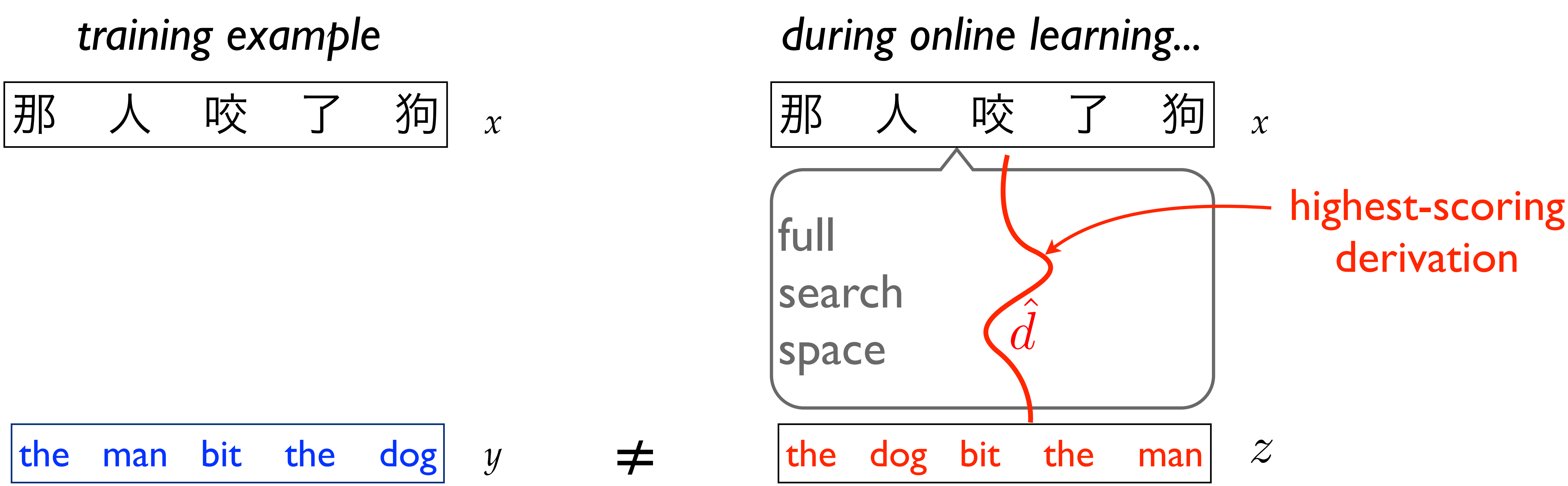
the dog bit the man z

(Liang et al 2006;
Yu et al 2013)

Search Errors in Decoding

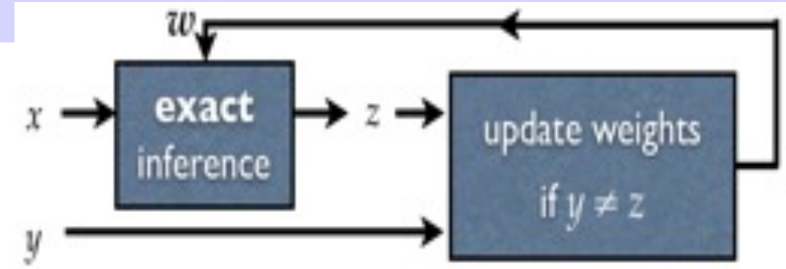


- no explicit positive signal
  - hallucinate the “correct” derivation by current weights

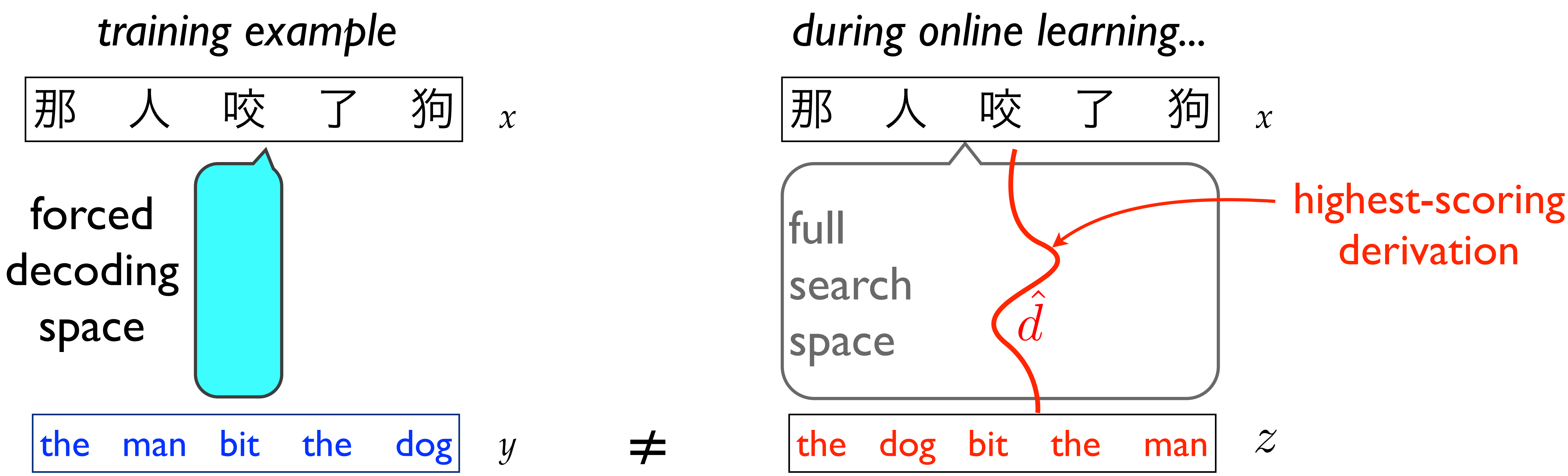


(Liang et al 2006;
Yu et al 2013)

Search Errors in Decoding

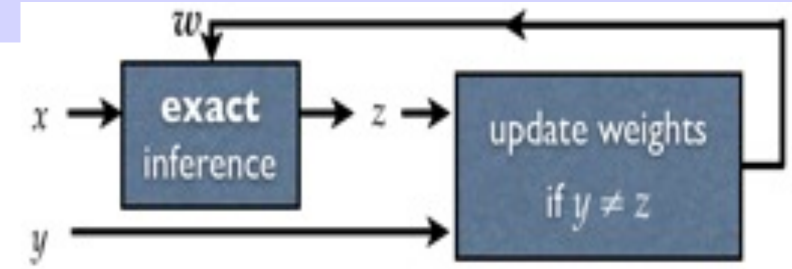


- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

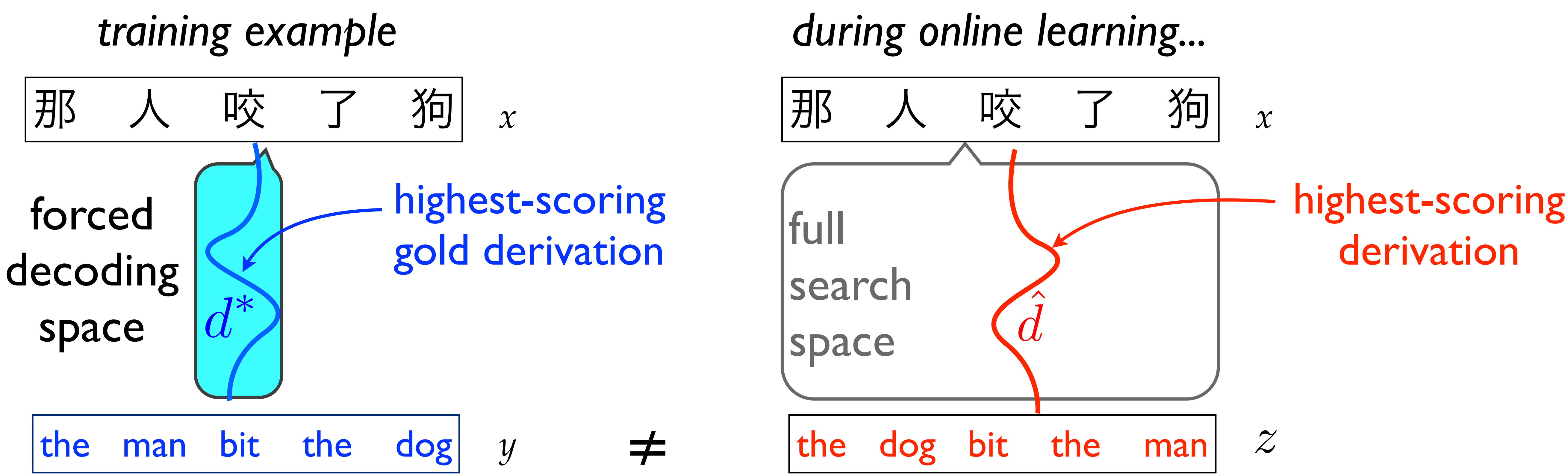


(Liang et al 2006; Yu et al 2013)

Search Errors in Decoding

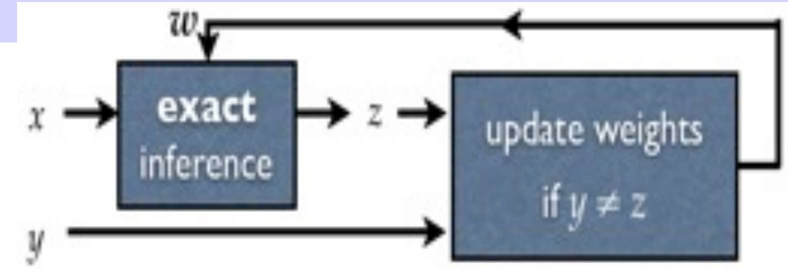


- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

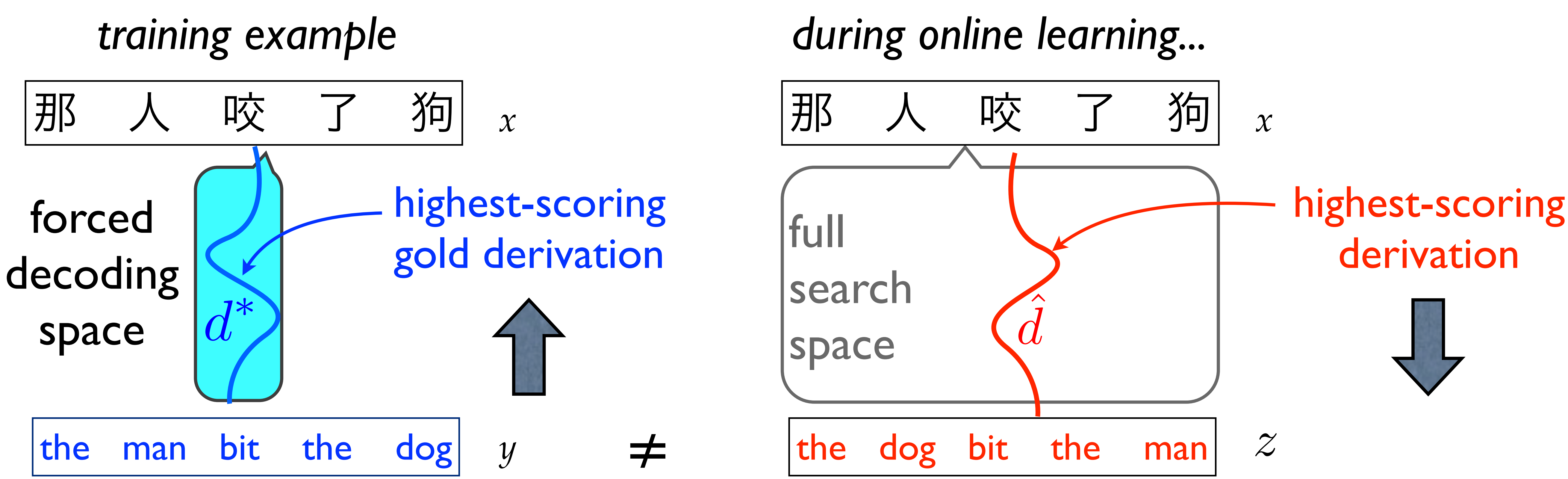


(Liang et al 2006;
Yu et al 2013)

Search Errors in Decoding



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights



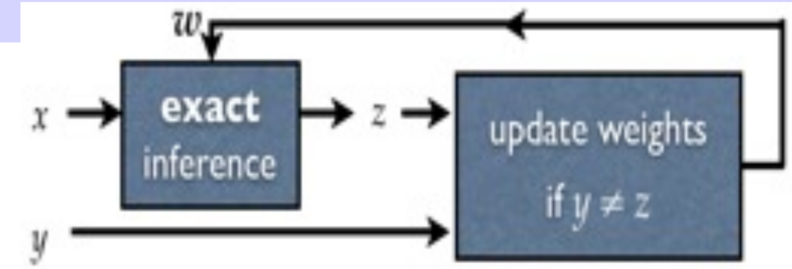
$$\mathbf{w} \leftarrow \mathbf{w} + \Phi(x, d^*) - \Phi(x, \hat{d})$$

reward correct

penalize wrong

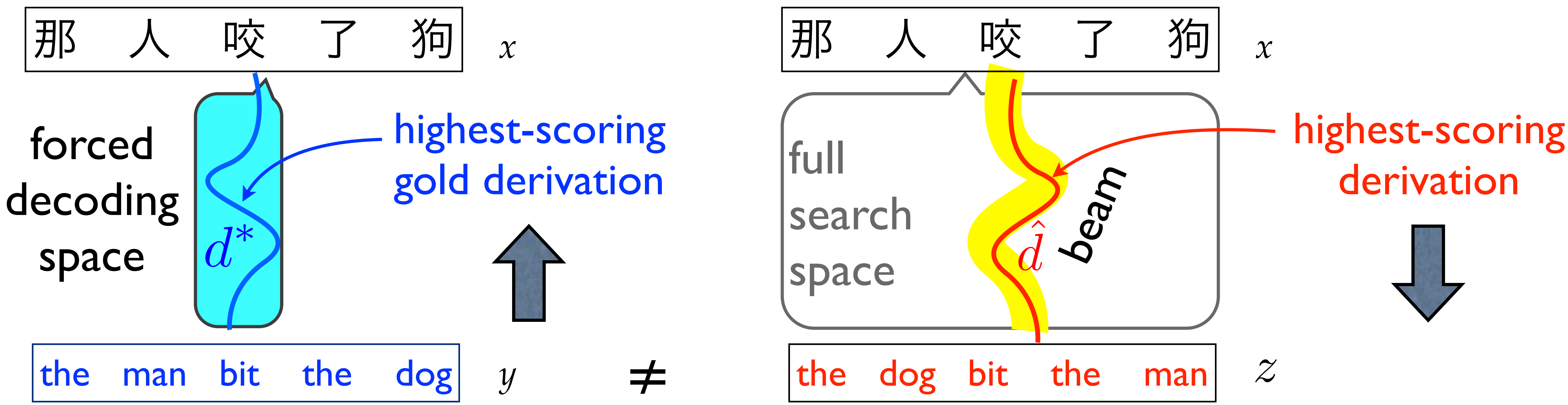
(Liang et al 2006; Yu et al 2013)

Search Errors in Decoding



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

training example *during online learning...*



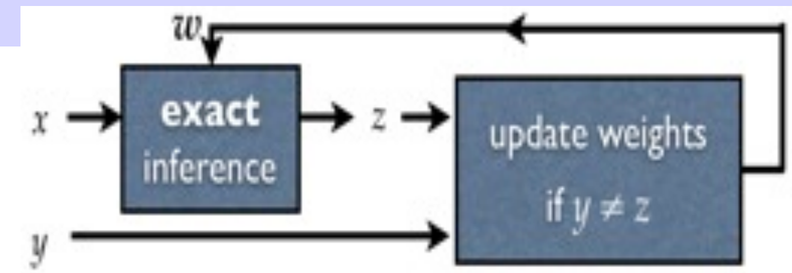
$$\mathbf{w} \leftarrow \mathbf{w} + \Phi(x, d^*) - \Phi(x, \hat{d})$$

reward correct penalize wrong

problem: search errors

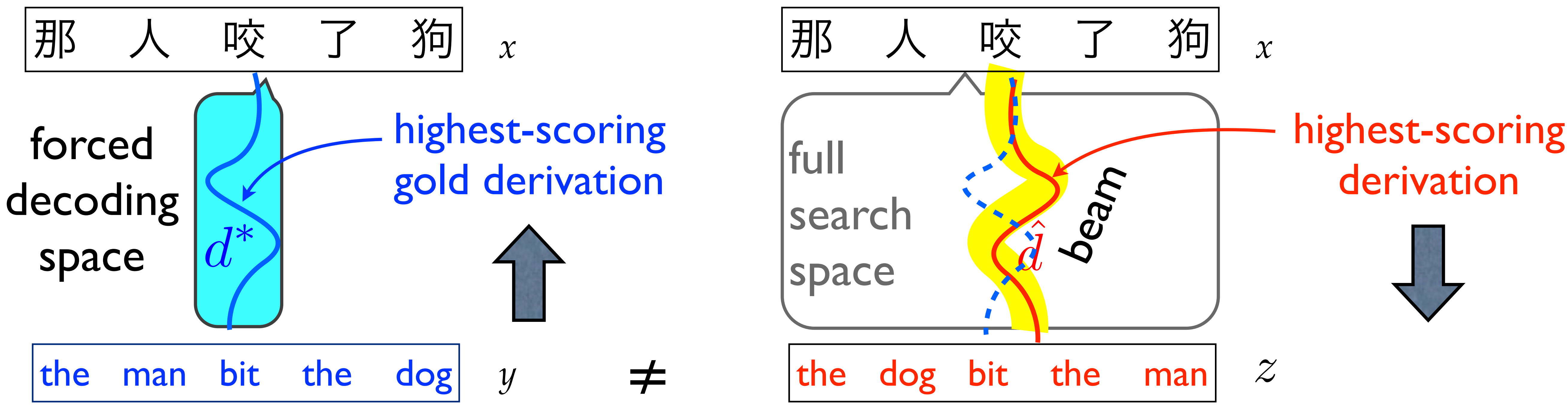
(Liang et al 2006; Yu et al 2013)

Search Errors in Decoding



- no explicit positive signal
  - hallucinate the "correct" derivation by current weights

training example *during online learning...*



$$\mathbf{w} \leftarrow \mathbf{w} + \Phi(x, d^*) - \Phi(x, \hat{d})$$

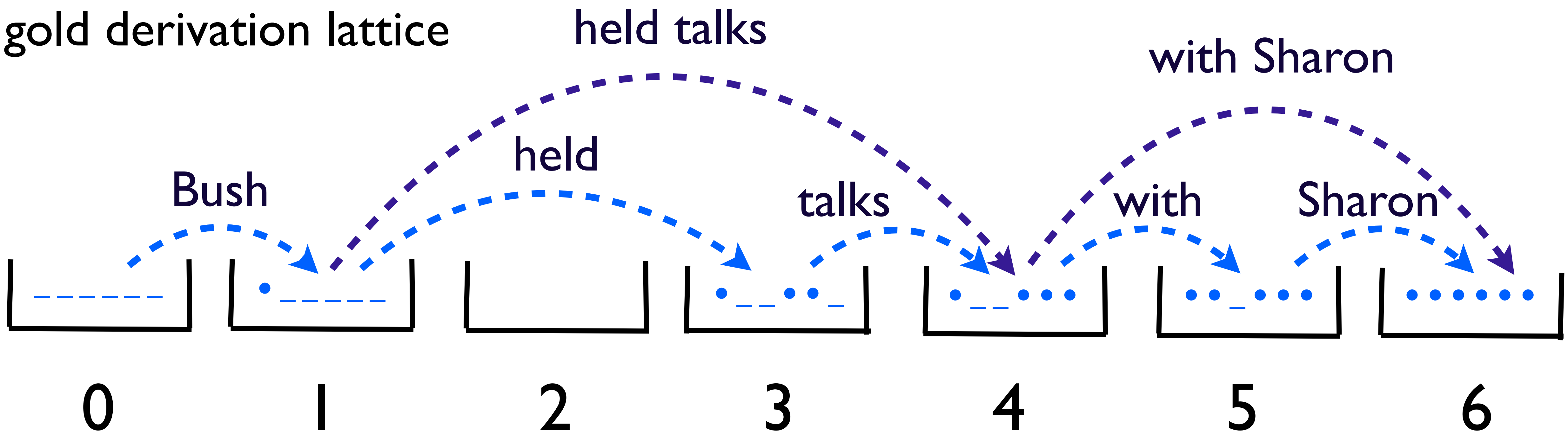
reward correct penalize wrong

problem: search errors

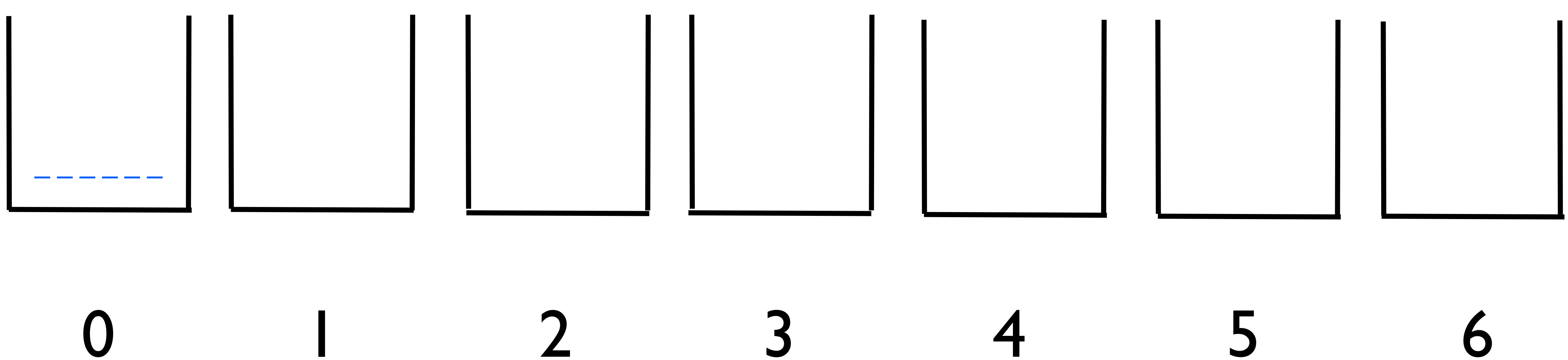
(Liang et al 2006; Yu et al 2013)

Search Error: Gold Derivations Pruned

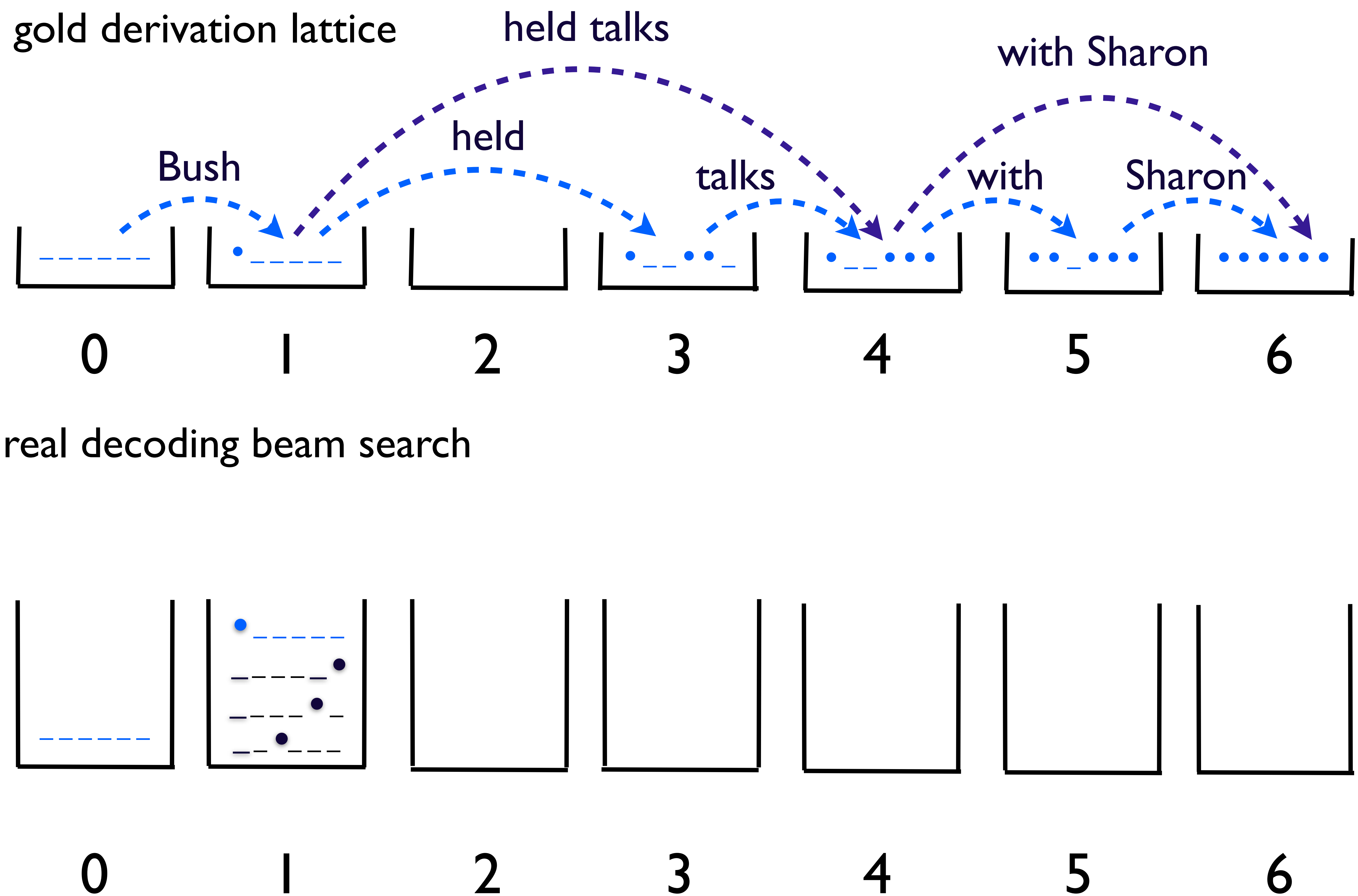
gold derivation lattice



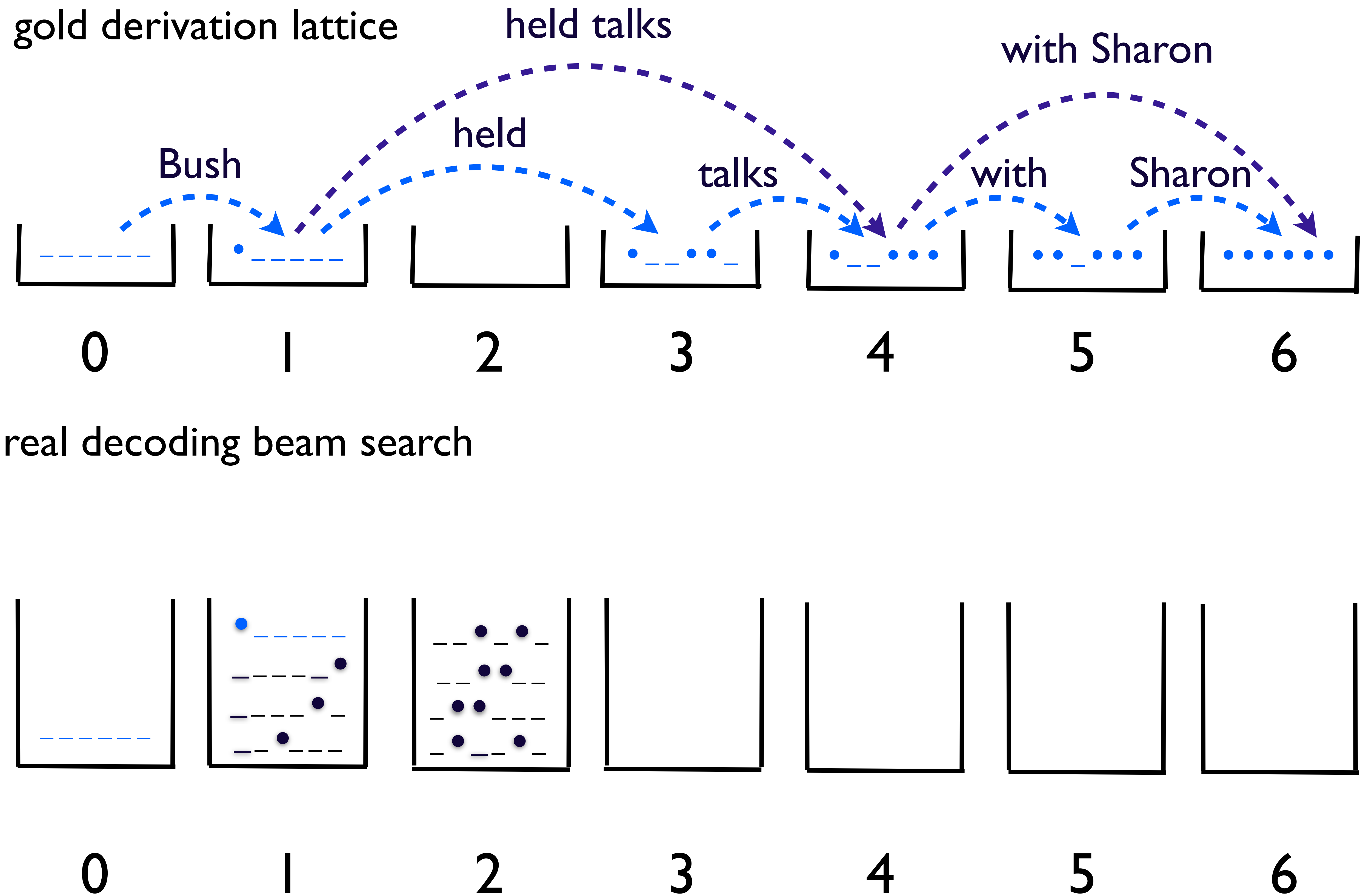
real decoding beam search



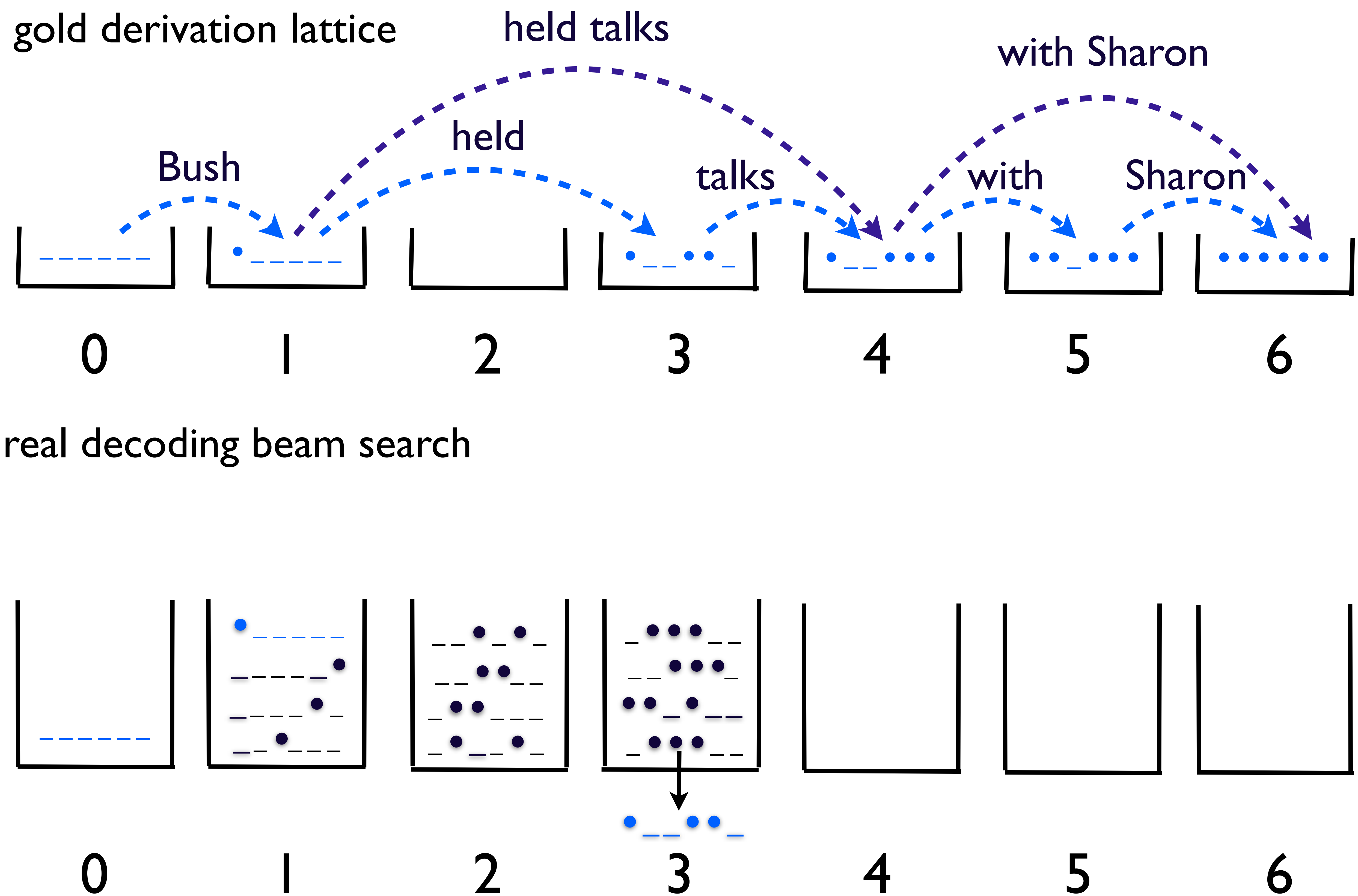
Search Error: Gold Derivations Pruned



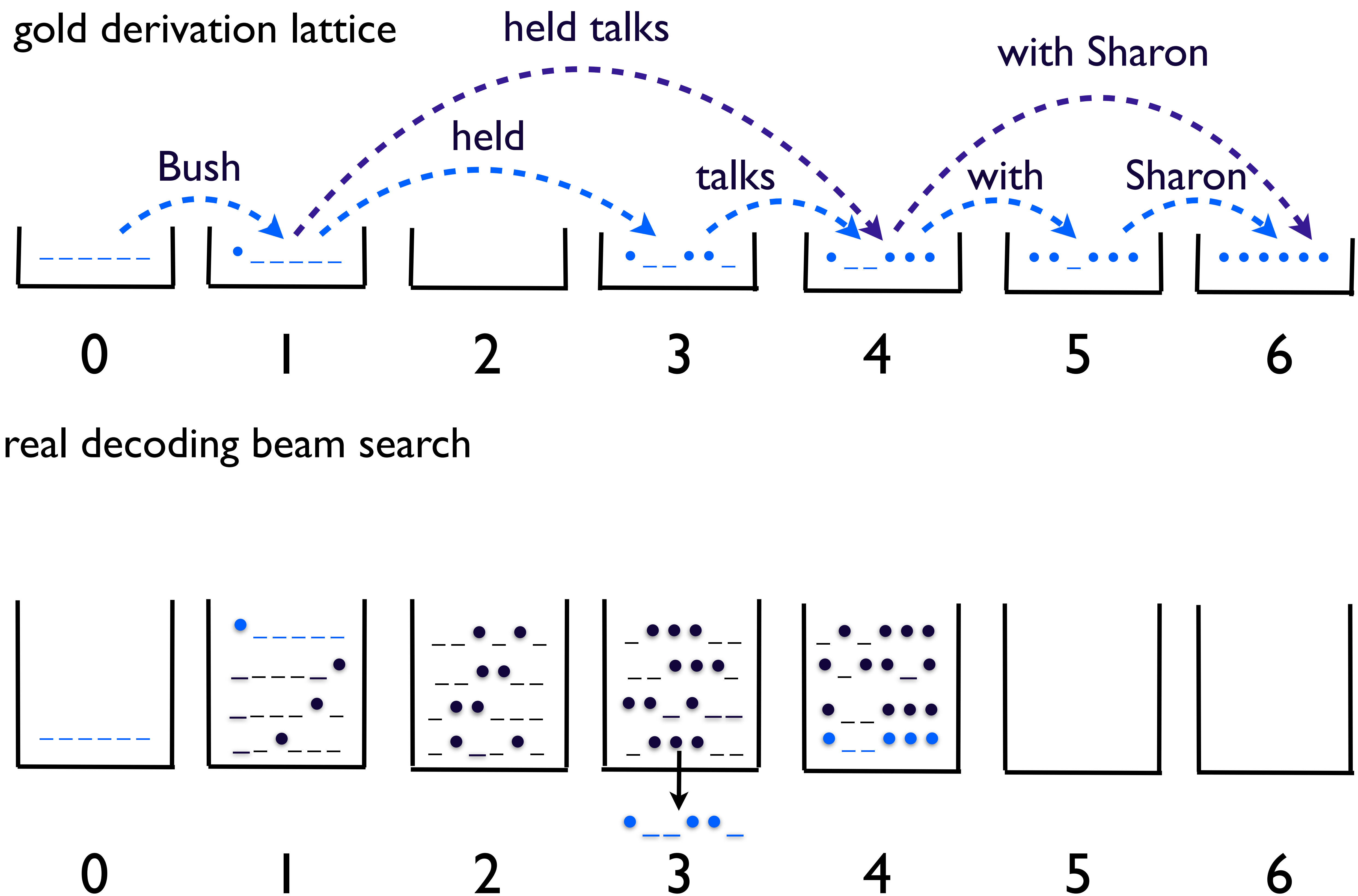
Search Error: Gold Derivations Pruned



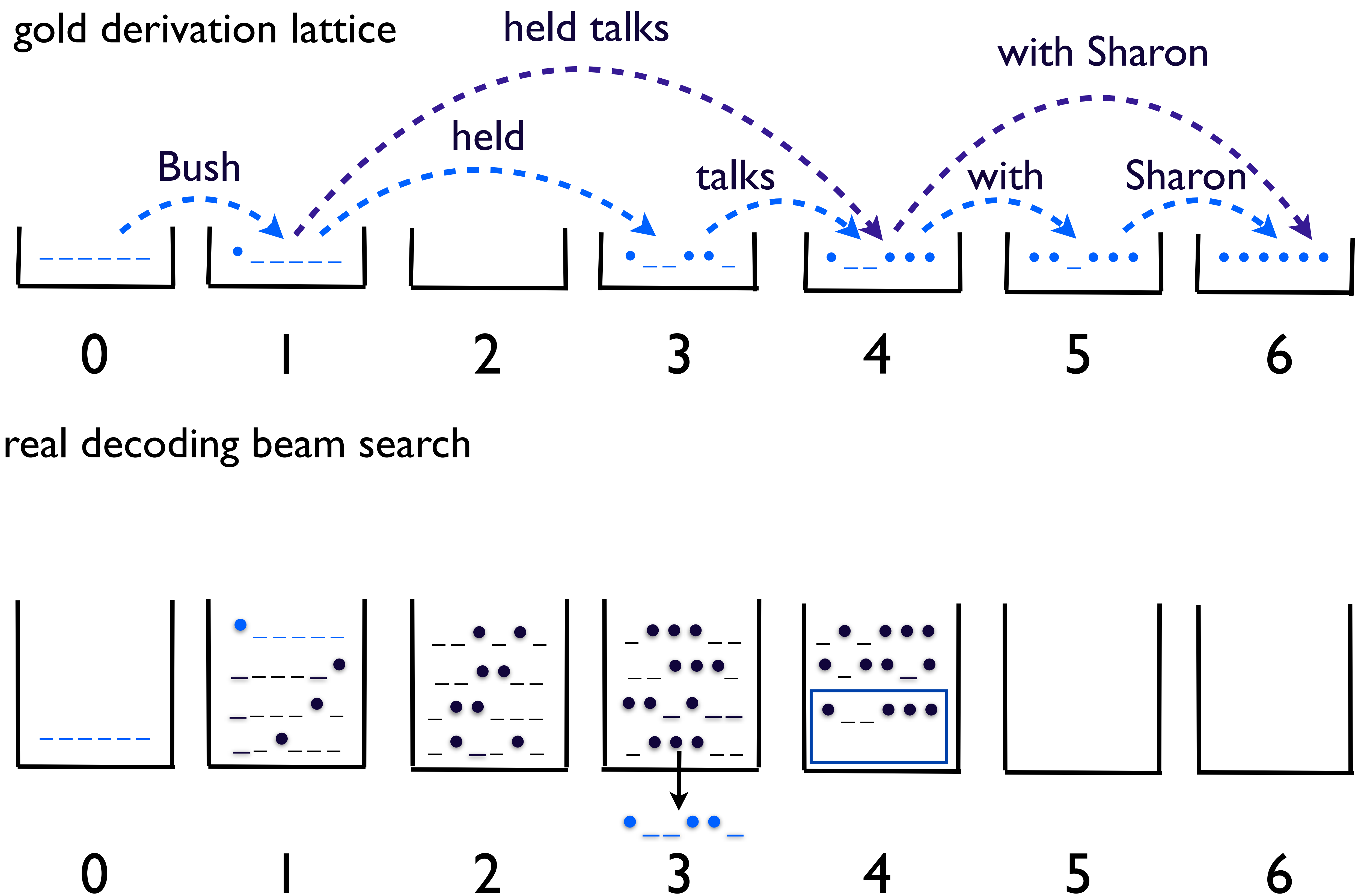
Search Error: Gold Derivations Pruned



Search Error: Gold Derivations Pruned

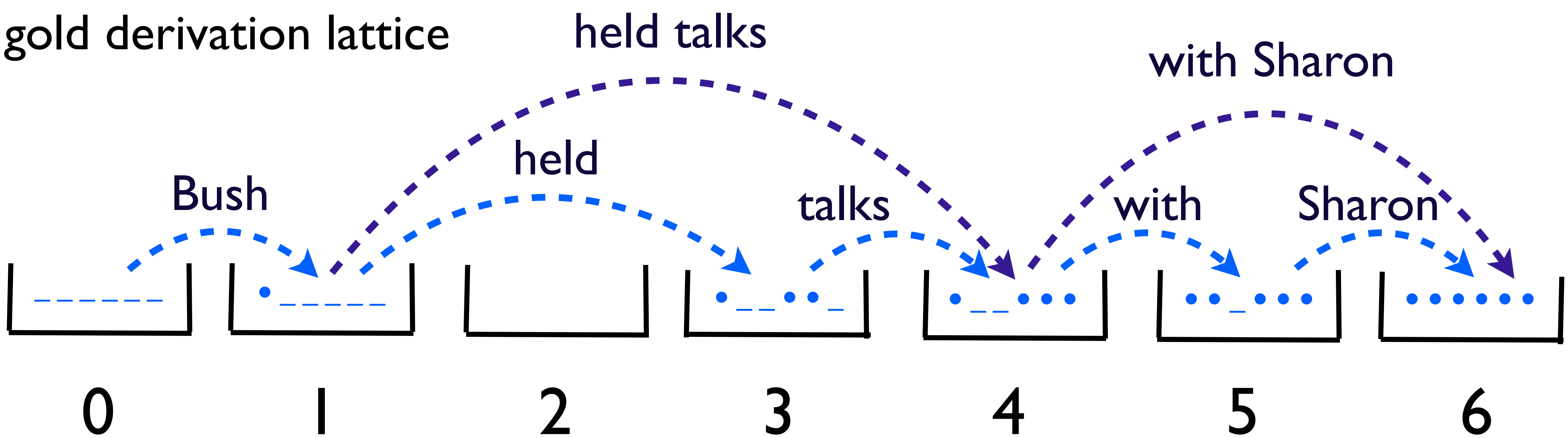


Search Error: Gold Derivations Pruned



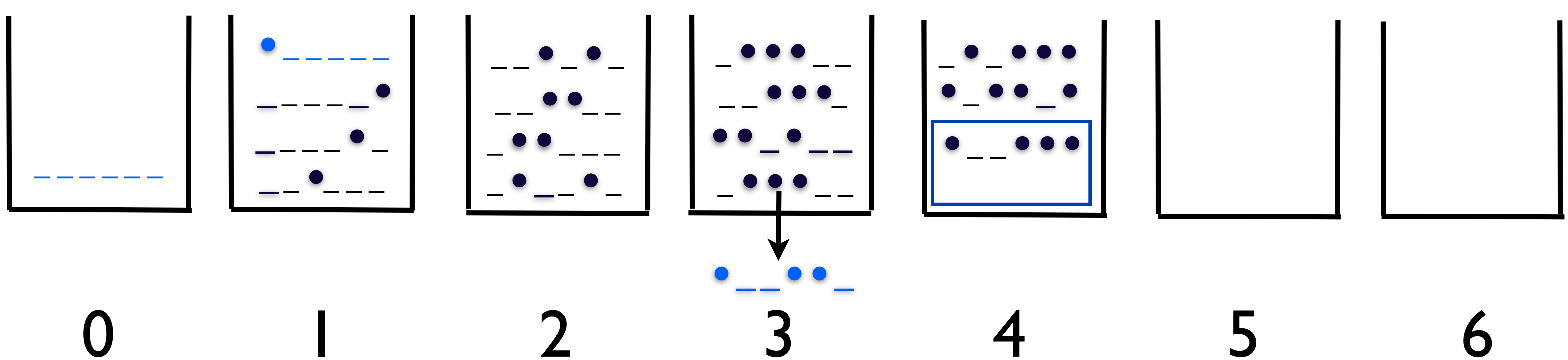
Search Error: Gold Derivations Pruned

gold derivation lattice

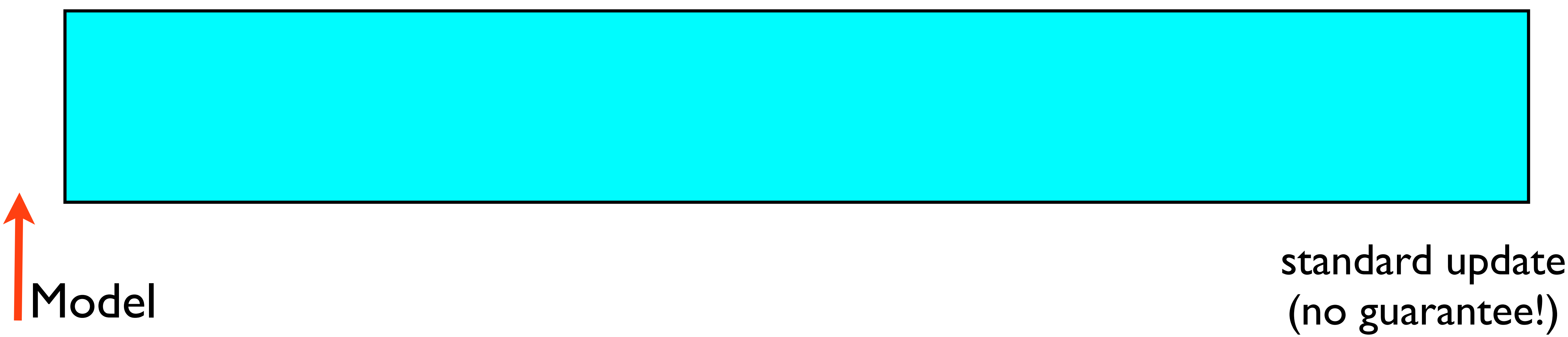


real decoding beam search

should address search errors here!

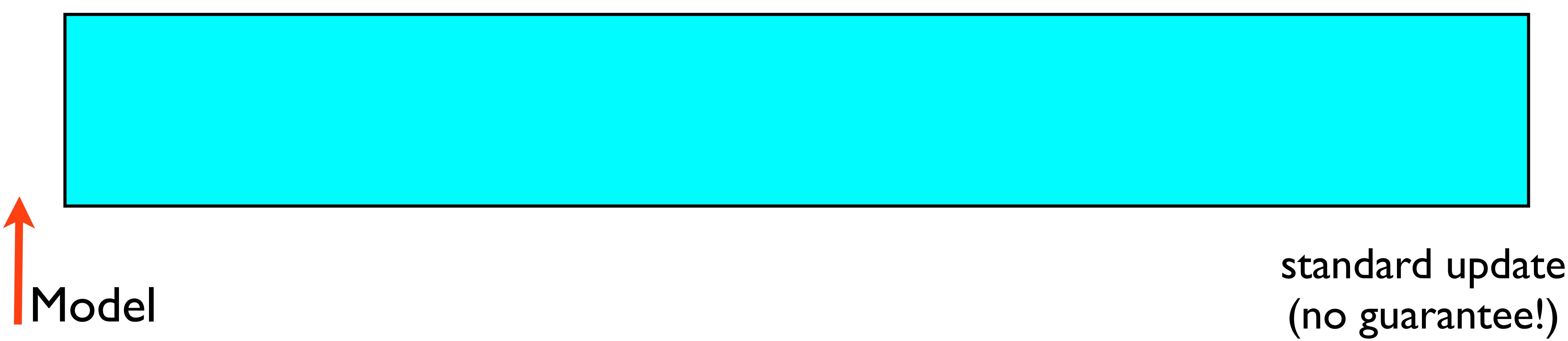


Fixing Search Error 1: Early Update



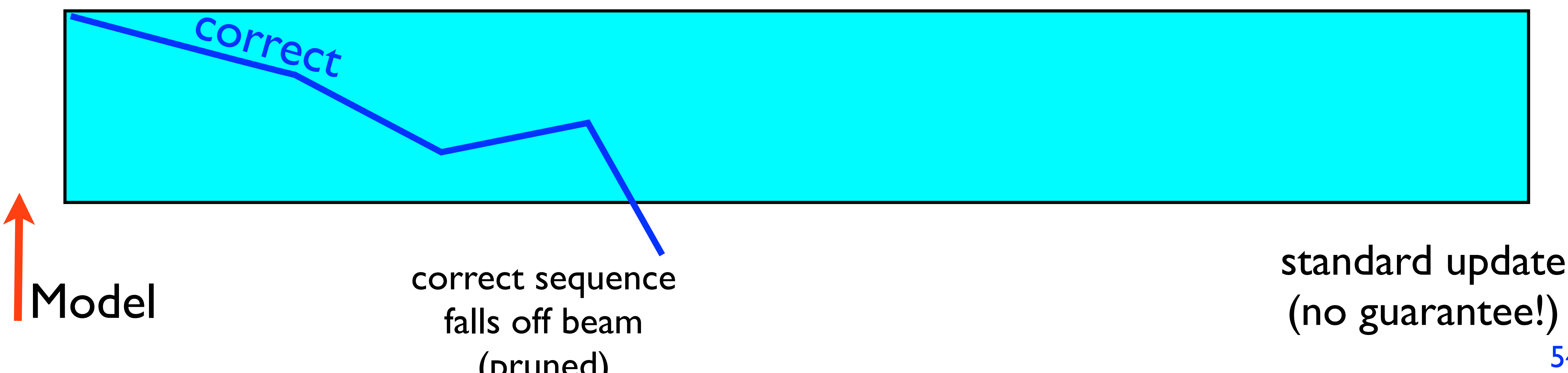
Fixing Search Error 1: Early Update

- early update (Collins/Roark'04) when the correct falls off beam
  - up to this point the incorrect prefix should score higher
  - that's a "violation" we want to fix; proof in (Huang et al 2012)



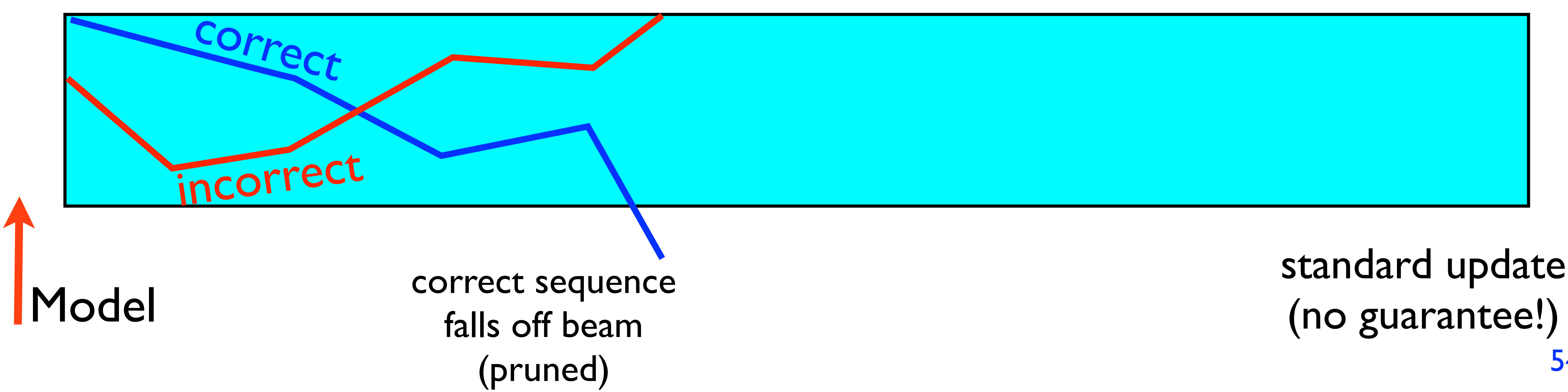
Fixing Search Error 1: Early Update

- early update (Collins/Roark'04) when the correct falls off beam
  - up to this point the incorrect prefix should score higher
  - that's a "violation" we want to fix; proof in (Huang et al 2012)



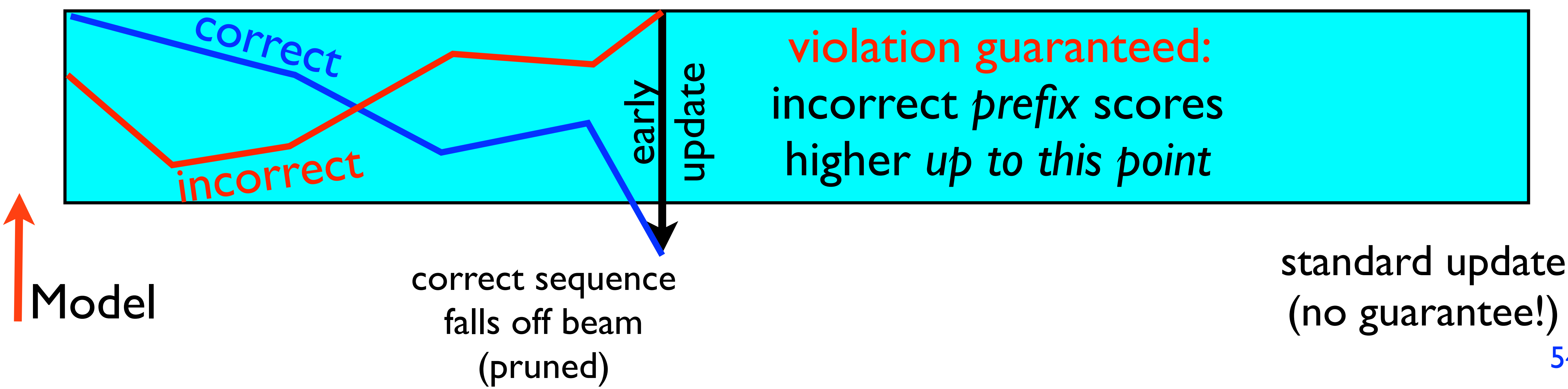
Fixing Search Error 1: Early Update

- early update (Collins/Roark'04) when the correct falls off beam
  - up to this point the incorrect prefix should score higher
  - that's a "violation" we want to fix; proof in (Huang et al 2012)



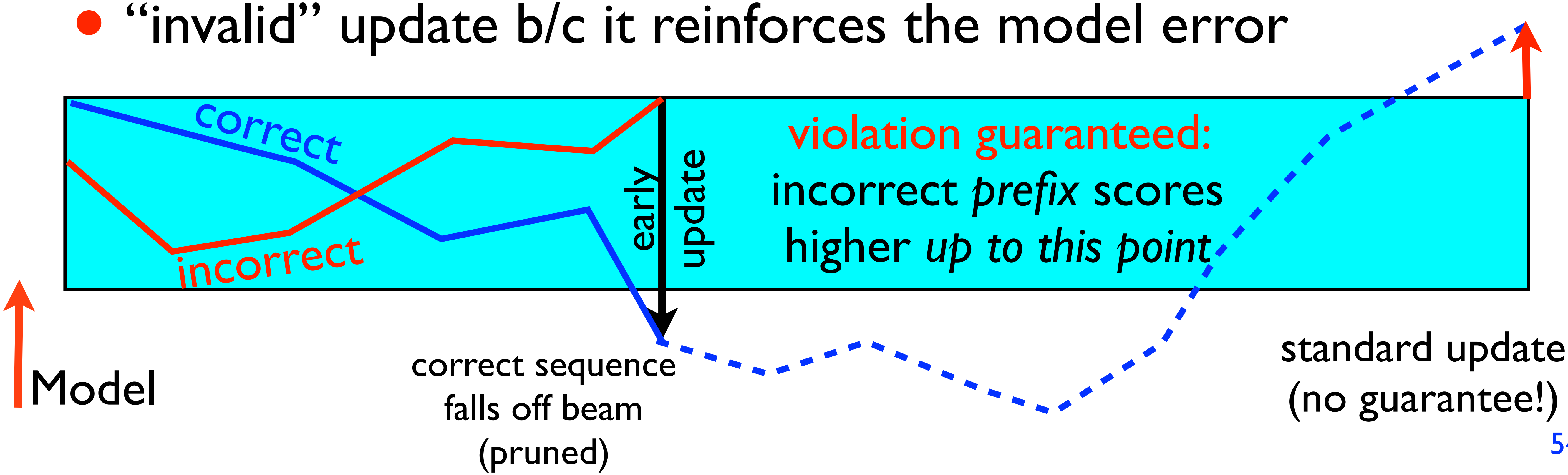
Fixing Search Error 1: Early Update

- early update (Collins/Roark'04) when the correct falls off beam
  - up to this point the incorrect prefix should score higher
  - that's a "violation" we want to fix; proof in (Huang et al 2012)



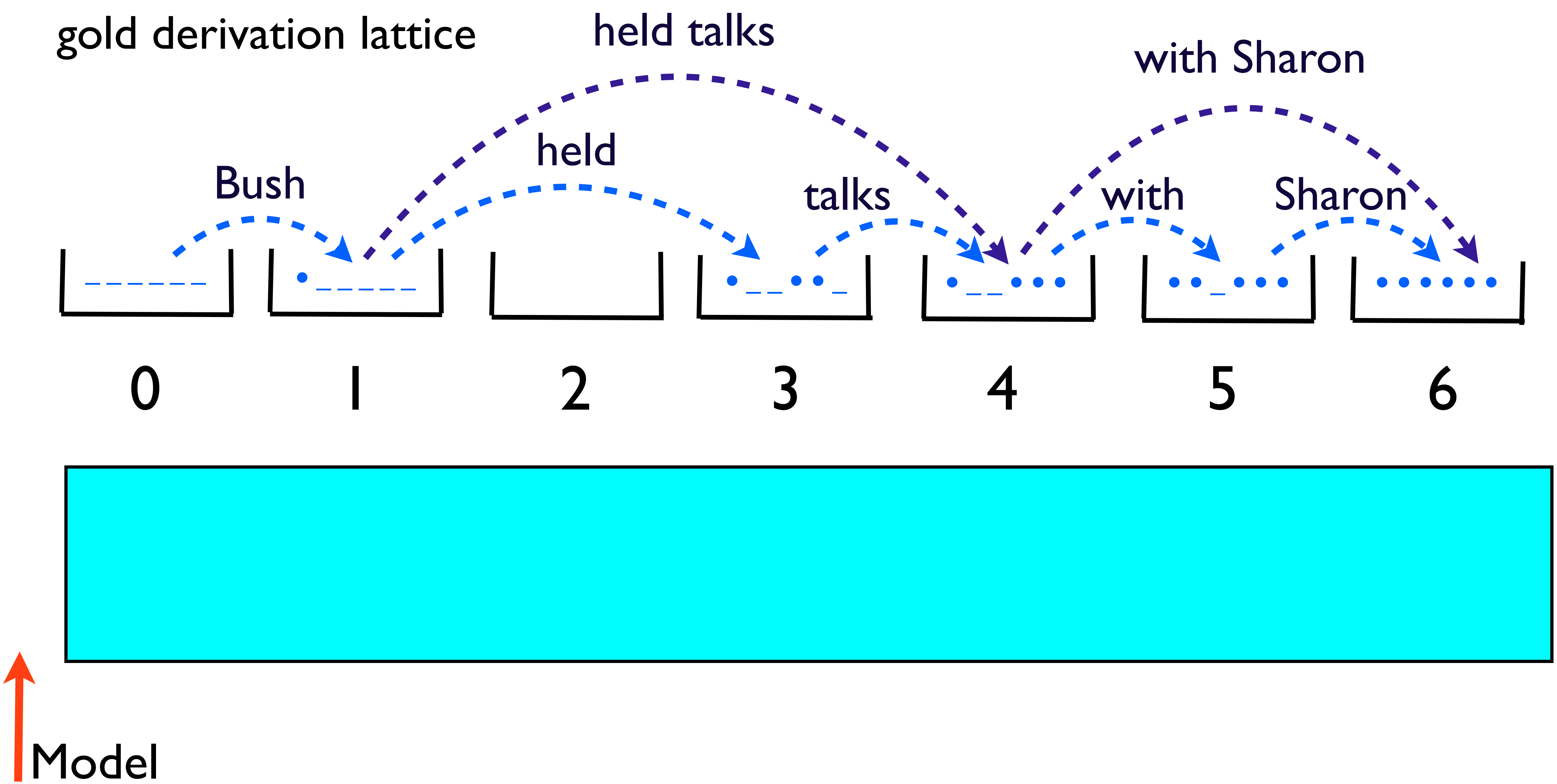
Fixing Search Error 1: Early Update

- early update (Collins/Roark'04) when the correct falls off beam
  - up to this point the incorrect prefix should score higher
  - that's a "violation" we want to fix; proof in (Huang et al 2012)
- standard perceptron does not guarantee violation
  - the correct sequence (pruned) might score higher at the end!
  - "invalid" update b/c it reinforces the model error



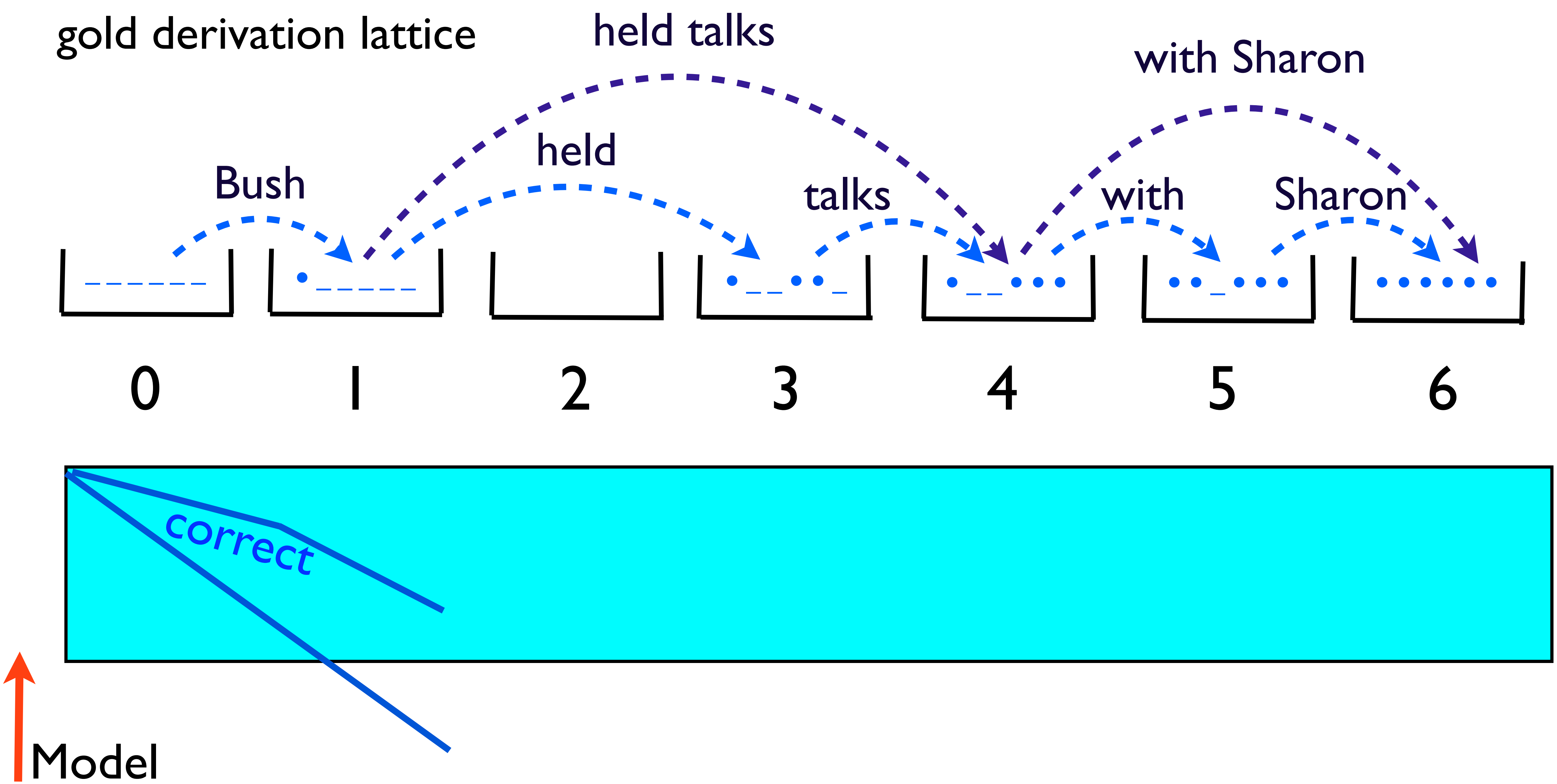
Early Update w/ Latent Variable

- the gold-standard derivations are ***not*** annotated
  - we treat any reference-producing derivation as good



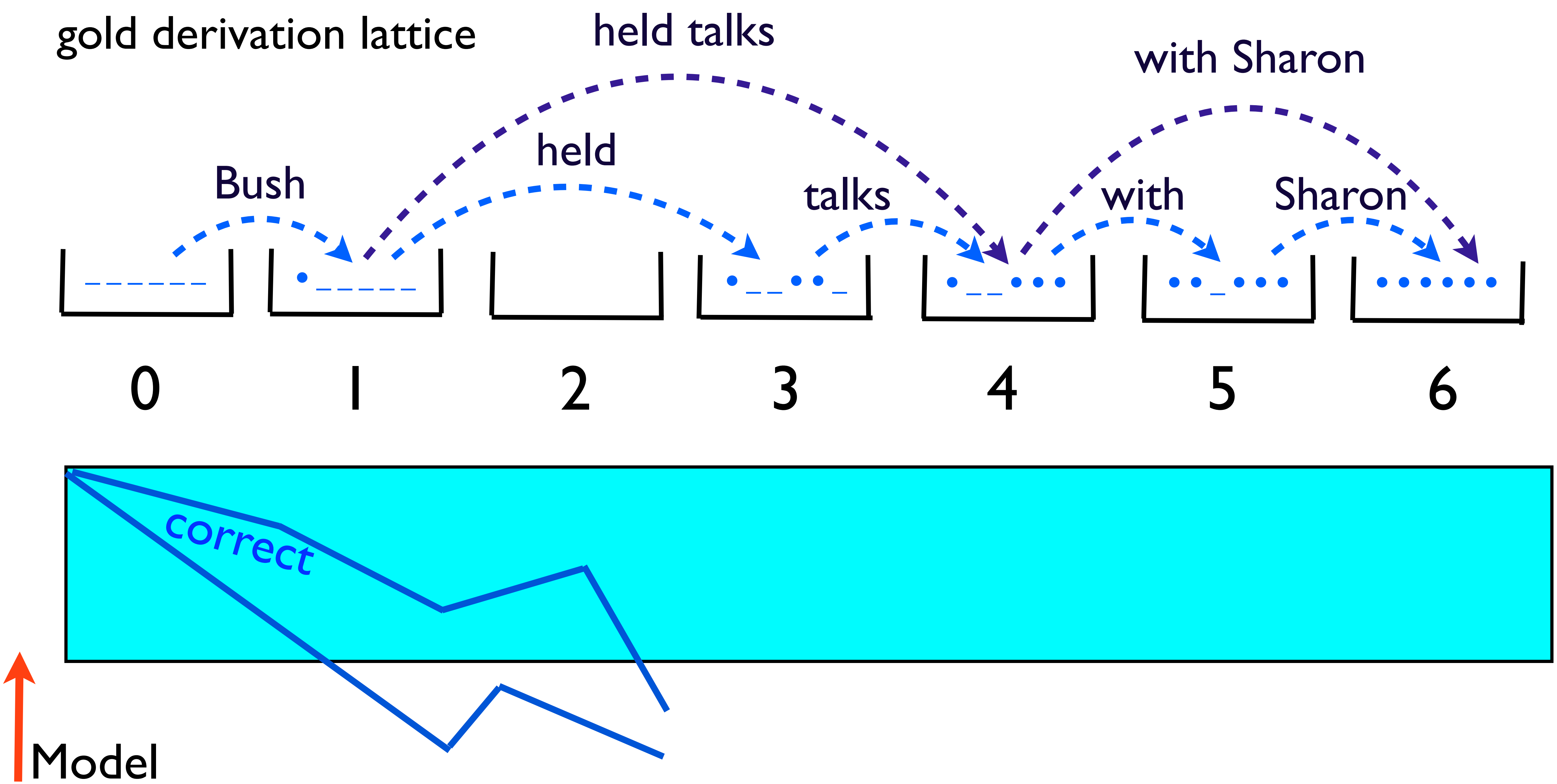
Early Update w/ Latent Variable

- the gold-standard derivations are ***not*** annotated
  - we treat any reference-producing derivation as good



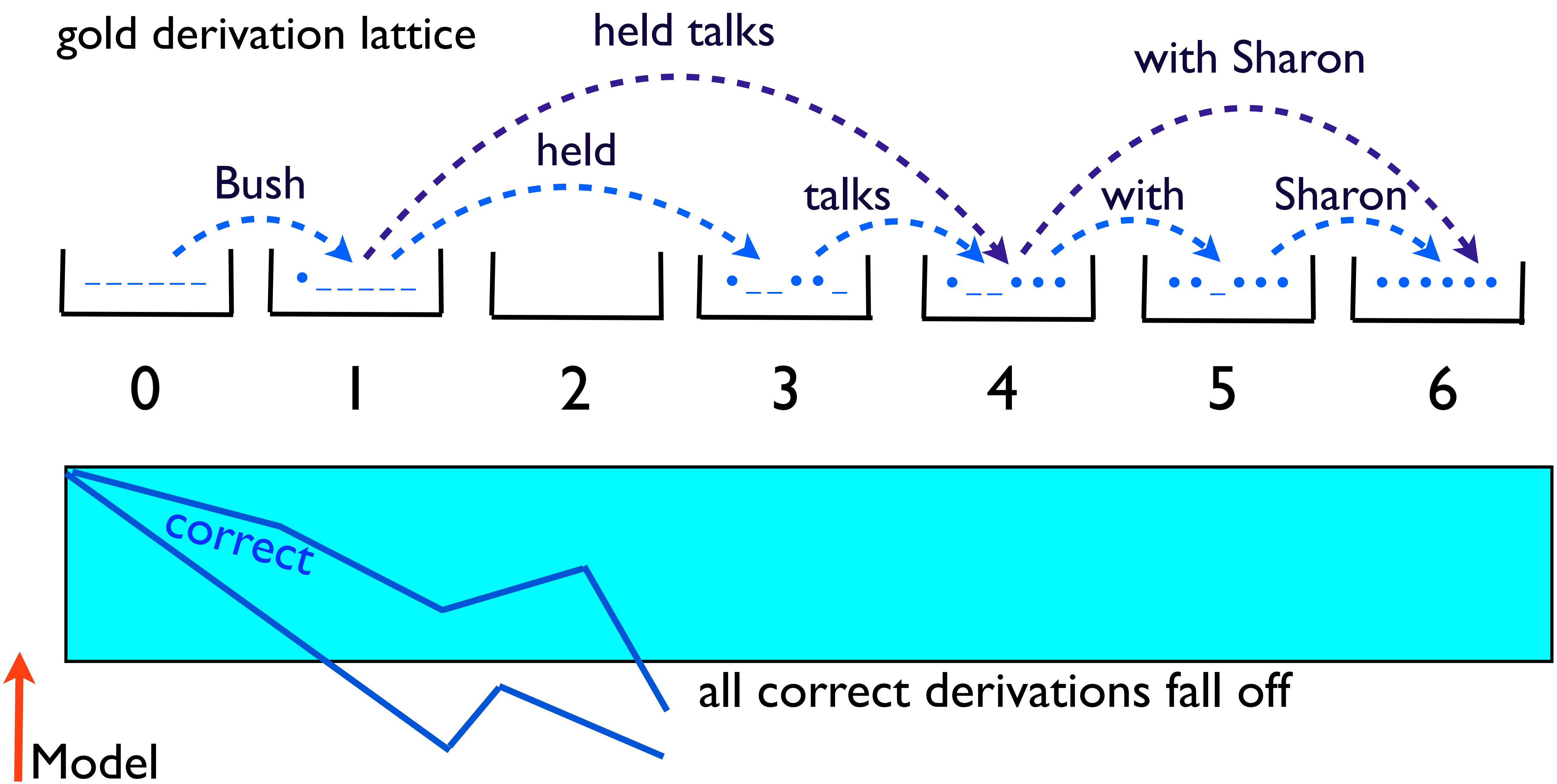
Early Update w/ Latent Variable

- the gold-standard derivations are ***not*** annotated
  - we treat any reference-producing derivation as good



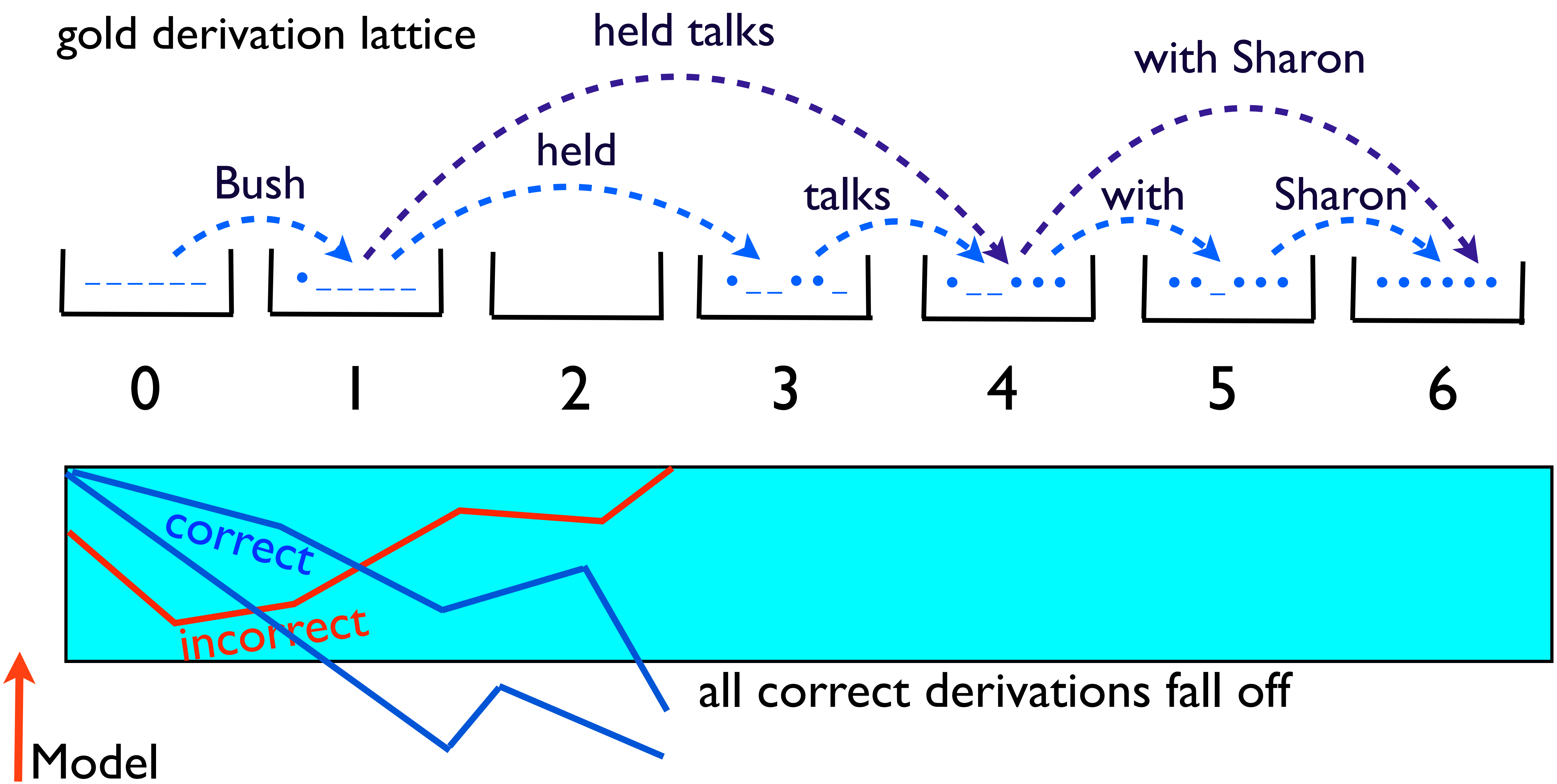
Early Update w/ Latent Variable

- the gold-standard derivations are ***not*** annotated
  - we treat any reference-producing derivation as good



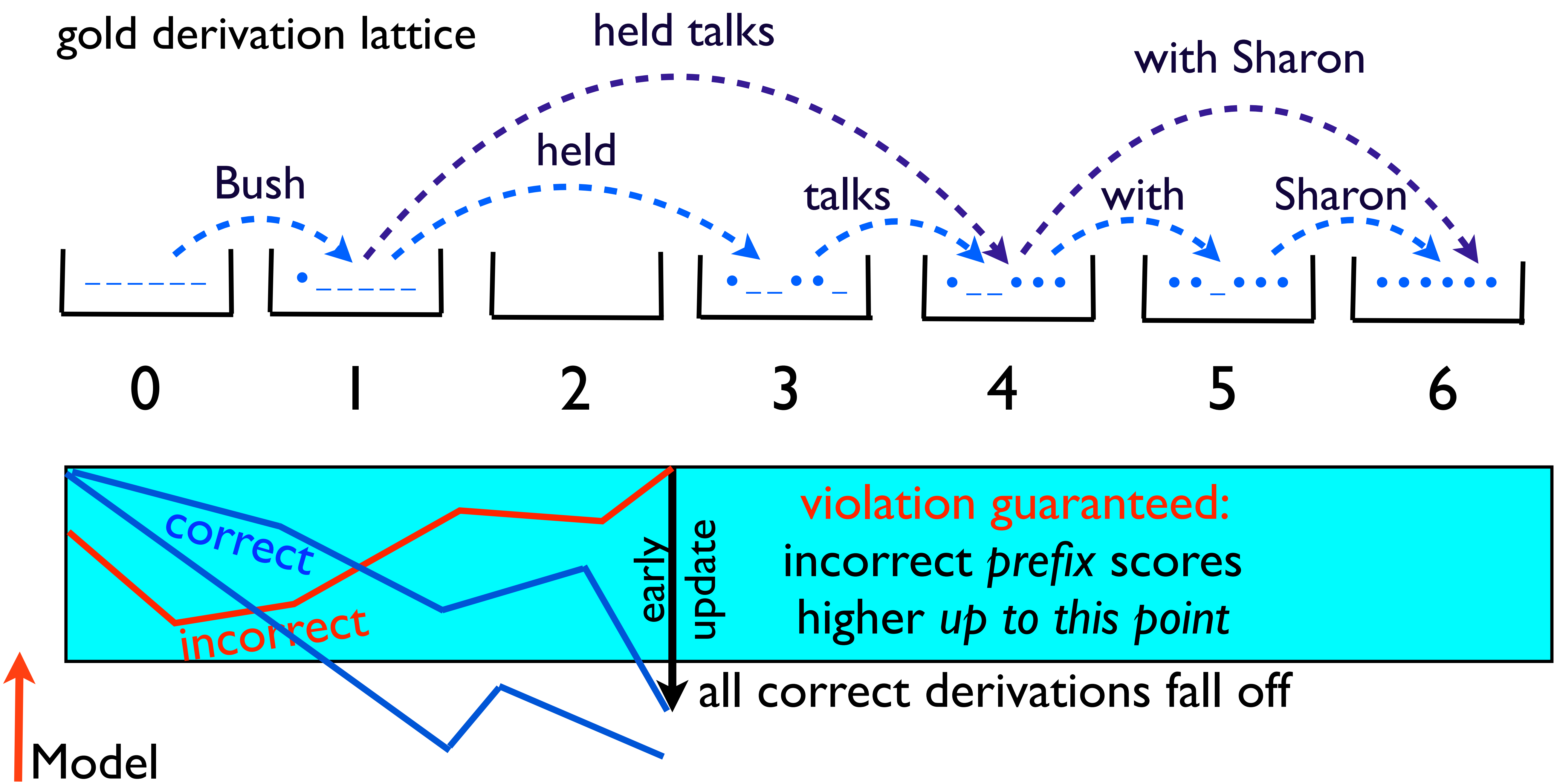
Early Update w/ Latent Variable

- the gold-standard derivations are ***not*** annotated
  - we treat any reference-producing derivation as good



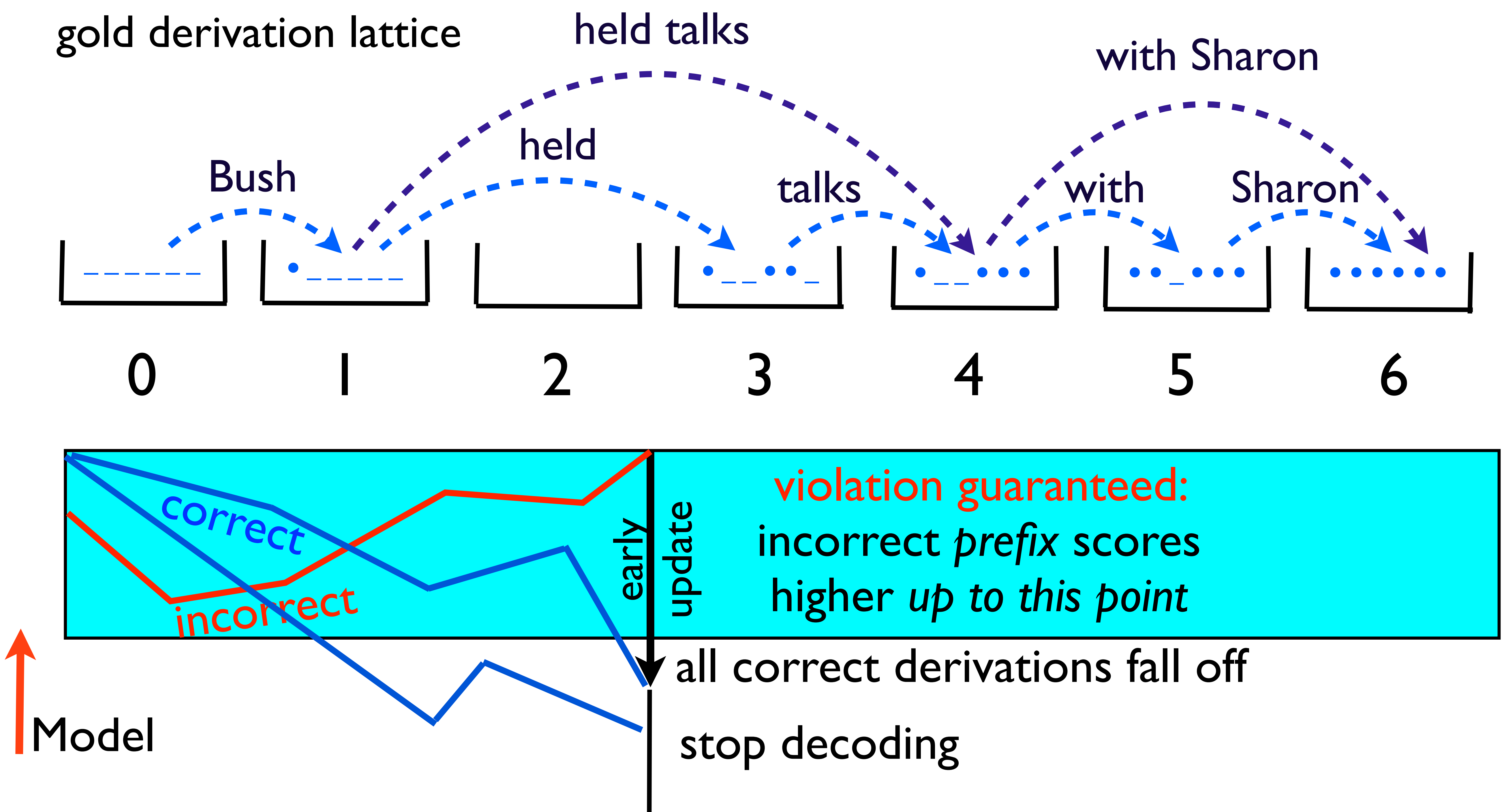
Early Update w/ Latent Variable

- the gold-standard derivations are ***not*** annotated
  - we treat any reference-producing derivation as good

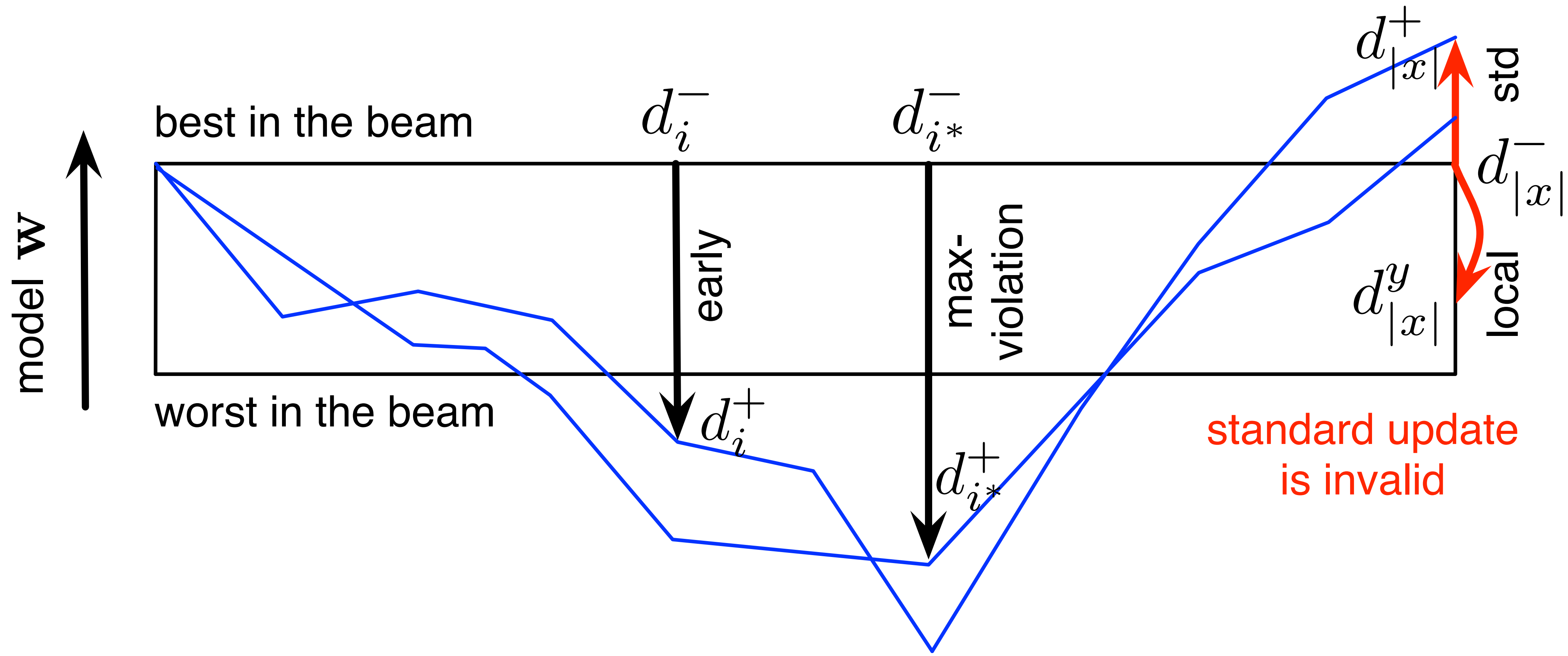


Early Update w/ Latent Variable

- the gold-standard derivations are ***not*** annotated
  - we treat any reference-producing derivation as good

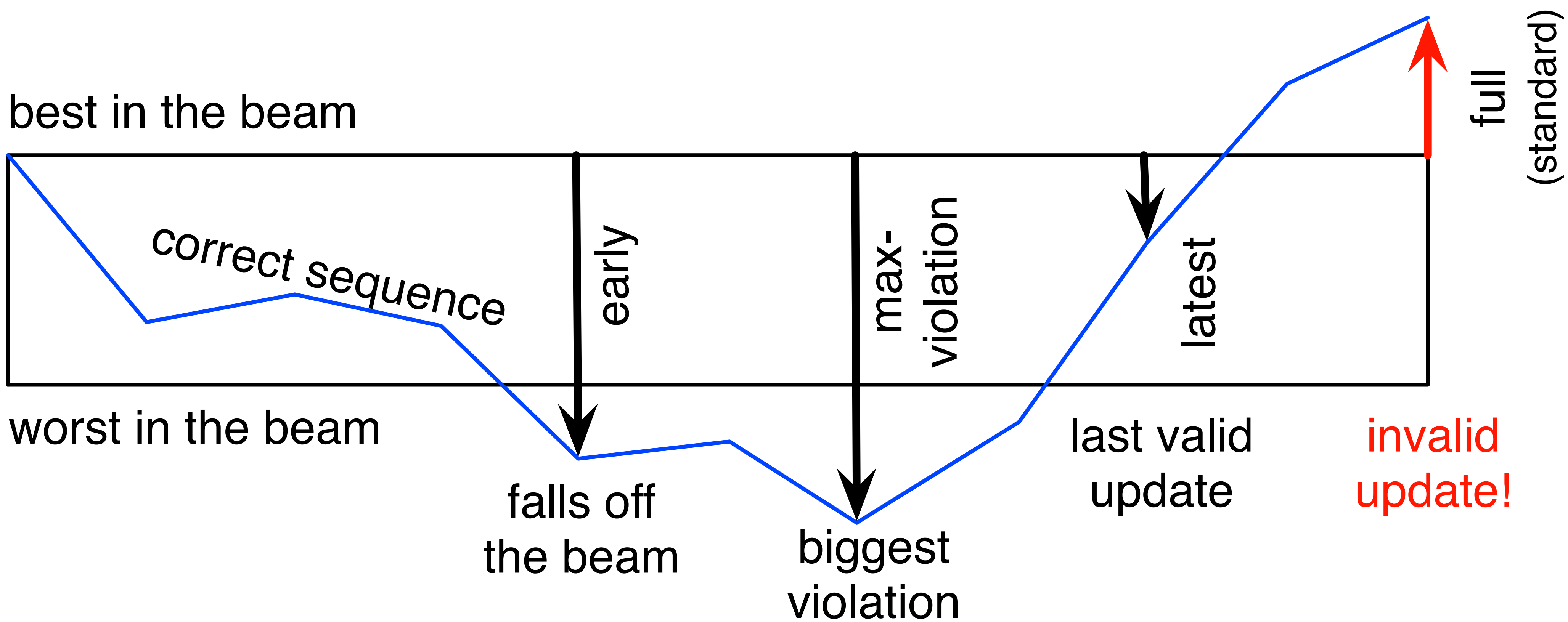


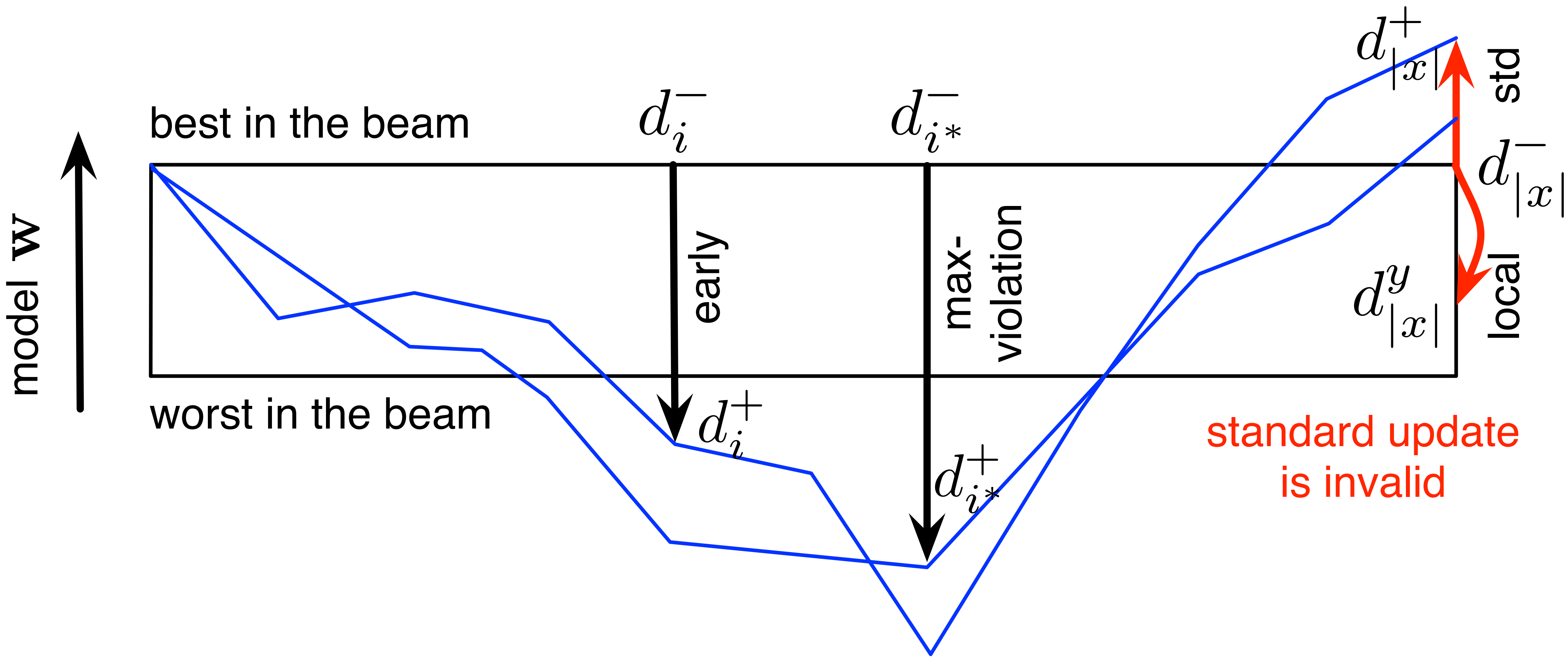
Fixing Search Error 2: Max-Violation



- early update works but learns slowly due to partial updates
- max-violation: use the prefix where violation is maximum
  - "worst-mistake" in the search space
  - now extended to handle latent-variable

Latent-Variable Perceptron





Roadmap of Techniques

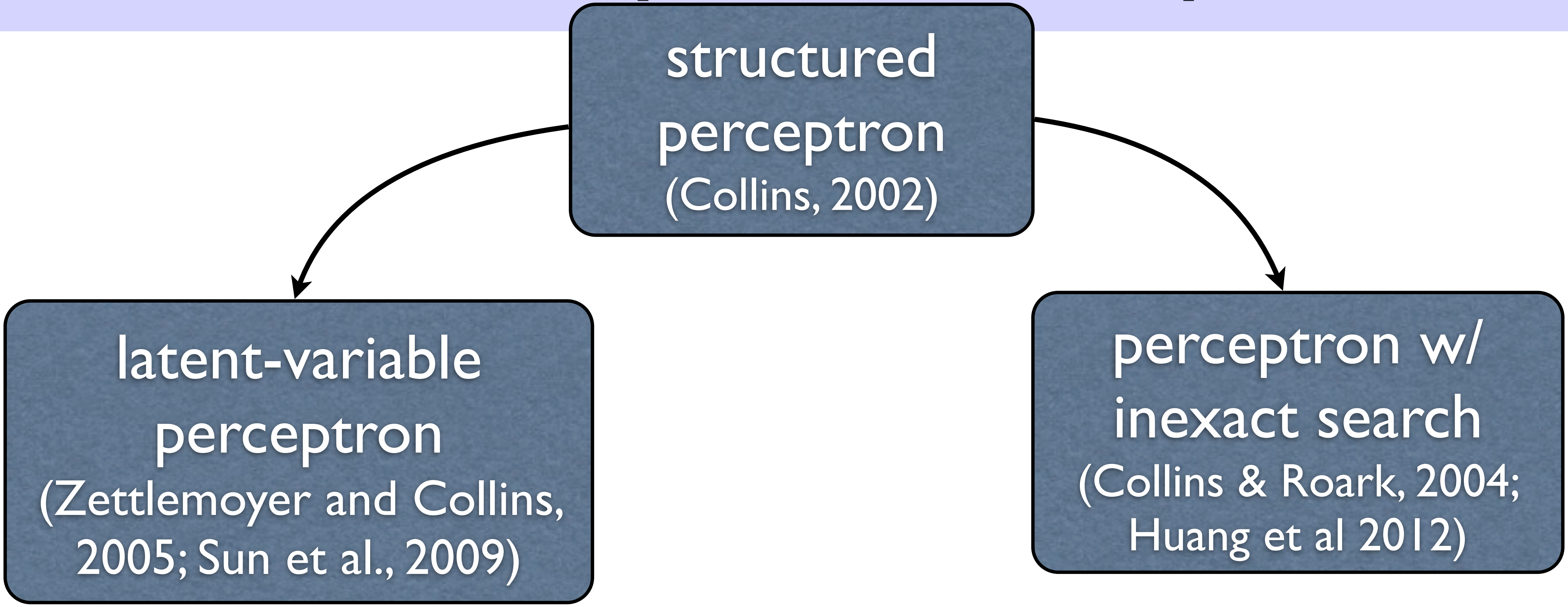
structured
perceptron
(Collins, 2002)

Roadmap of Techniques

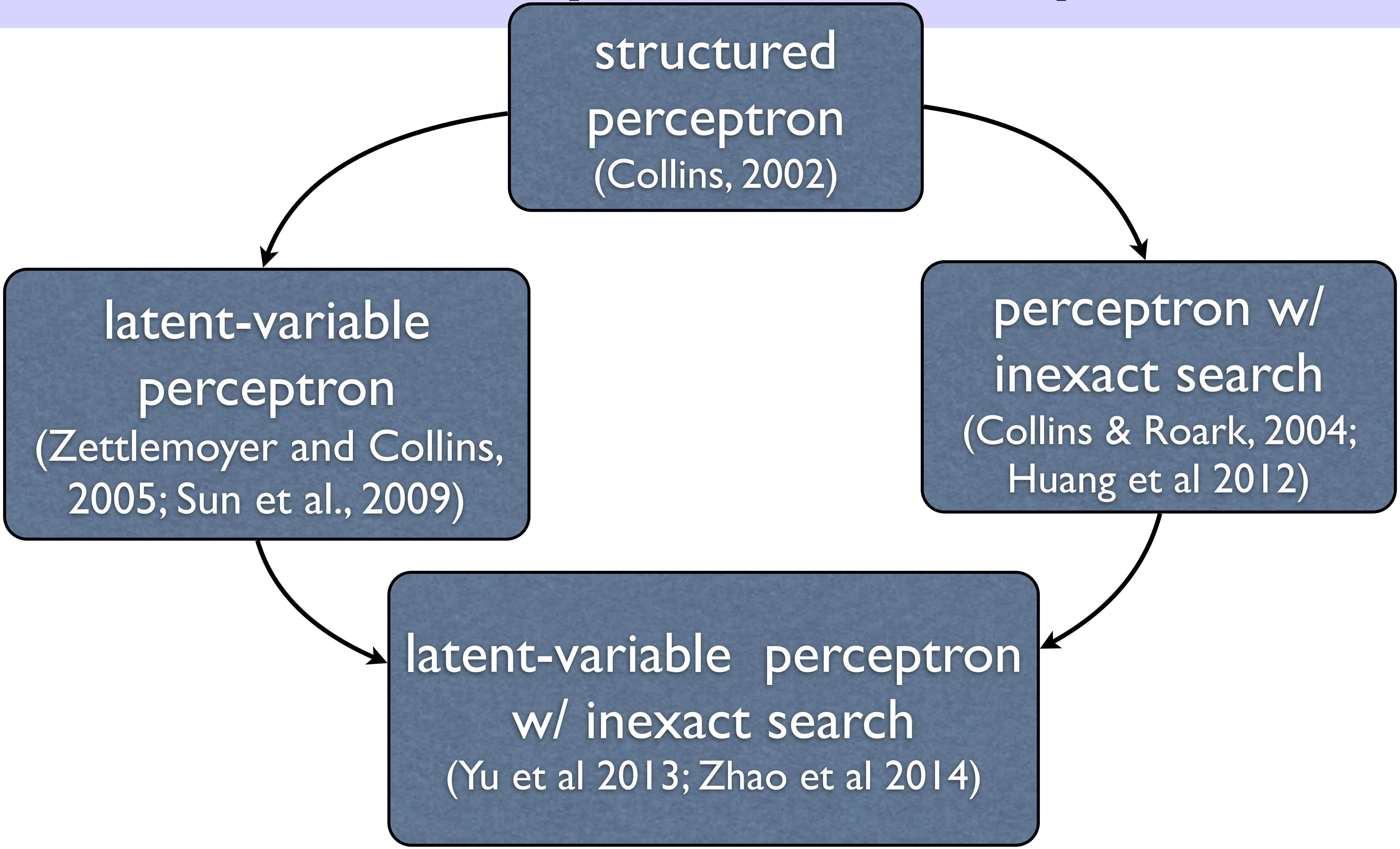
structured perceptron
(Collins, 2002)

latent-variable perceptron
(Zettlemoyer and Collins, 2005; Sun et al., 2009)

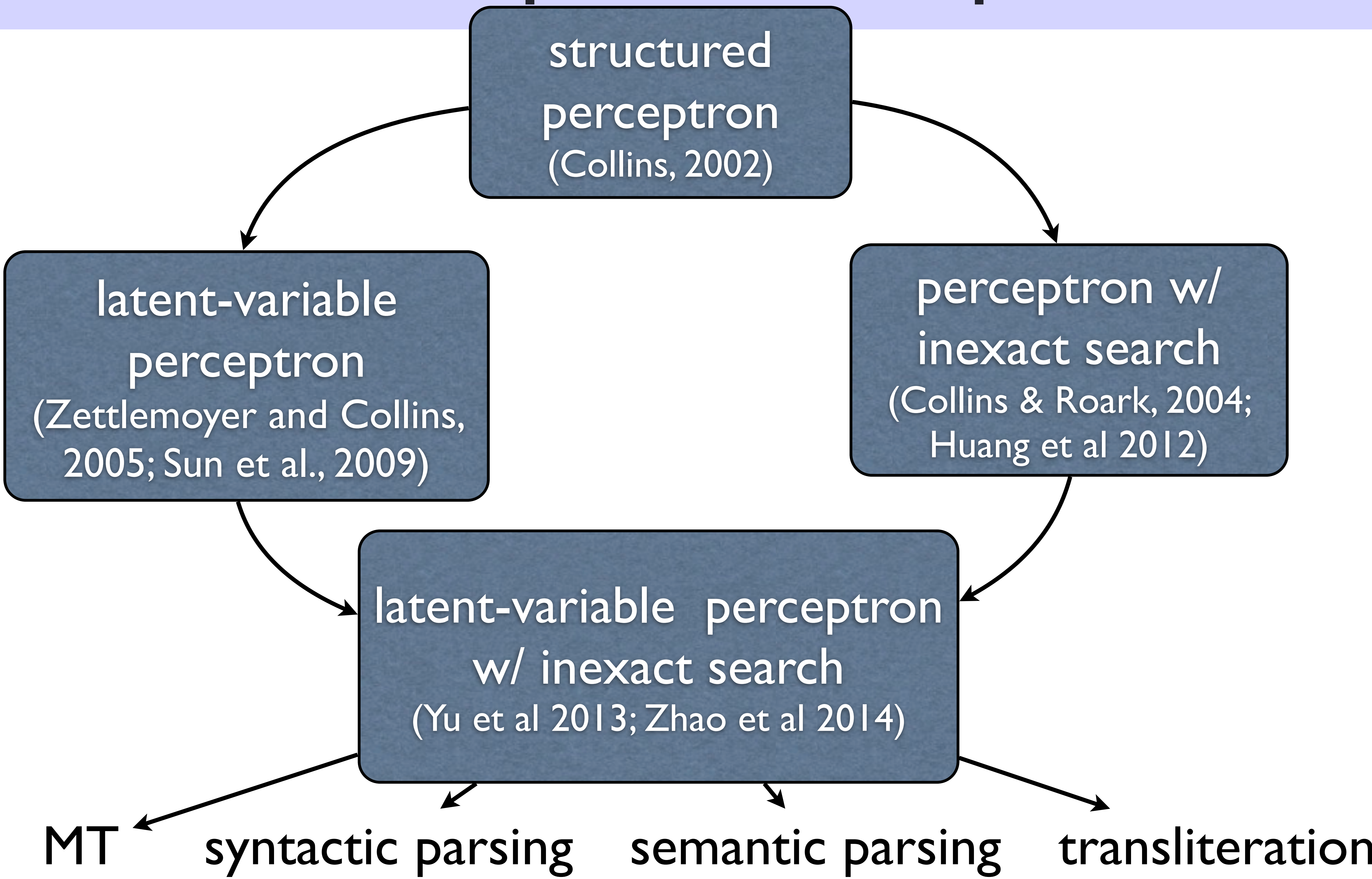
Roadmap of Techniques



Roadmap of Techniques

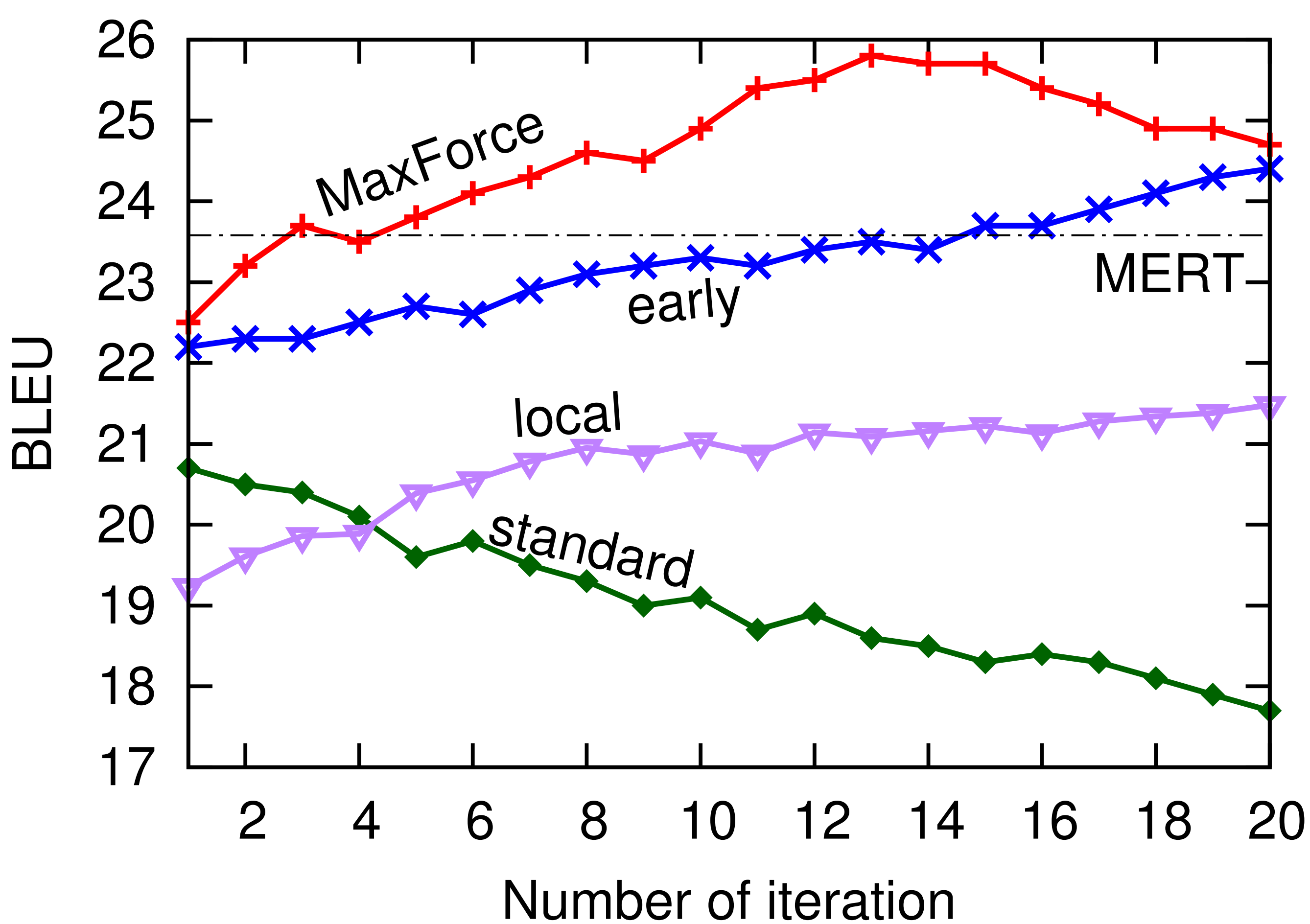


Roadmap of Techniques



Experiments: Discriminative Training for MT

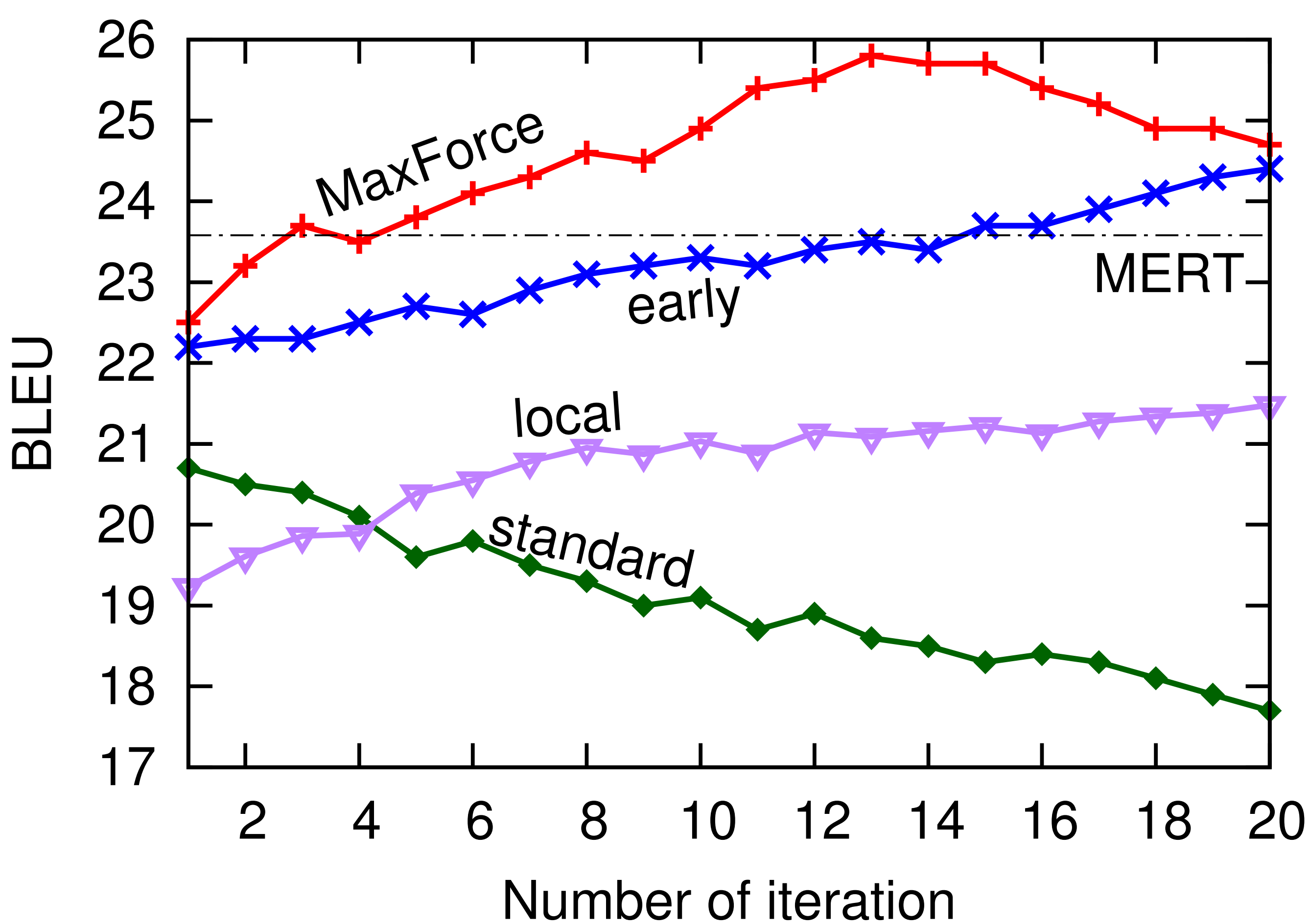
- standard update (Liang et al's "bold") works poorly
  - b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update



Experiments: Discriminative Training for MT

- standard update (Liang et al's "bold") works poorly
  - b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update

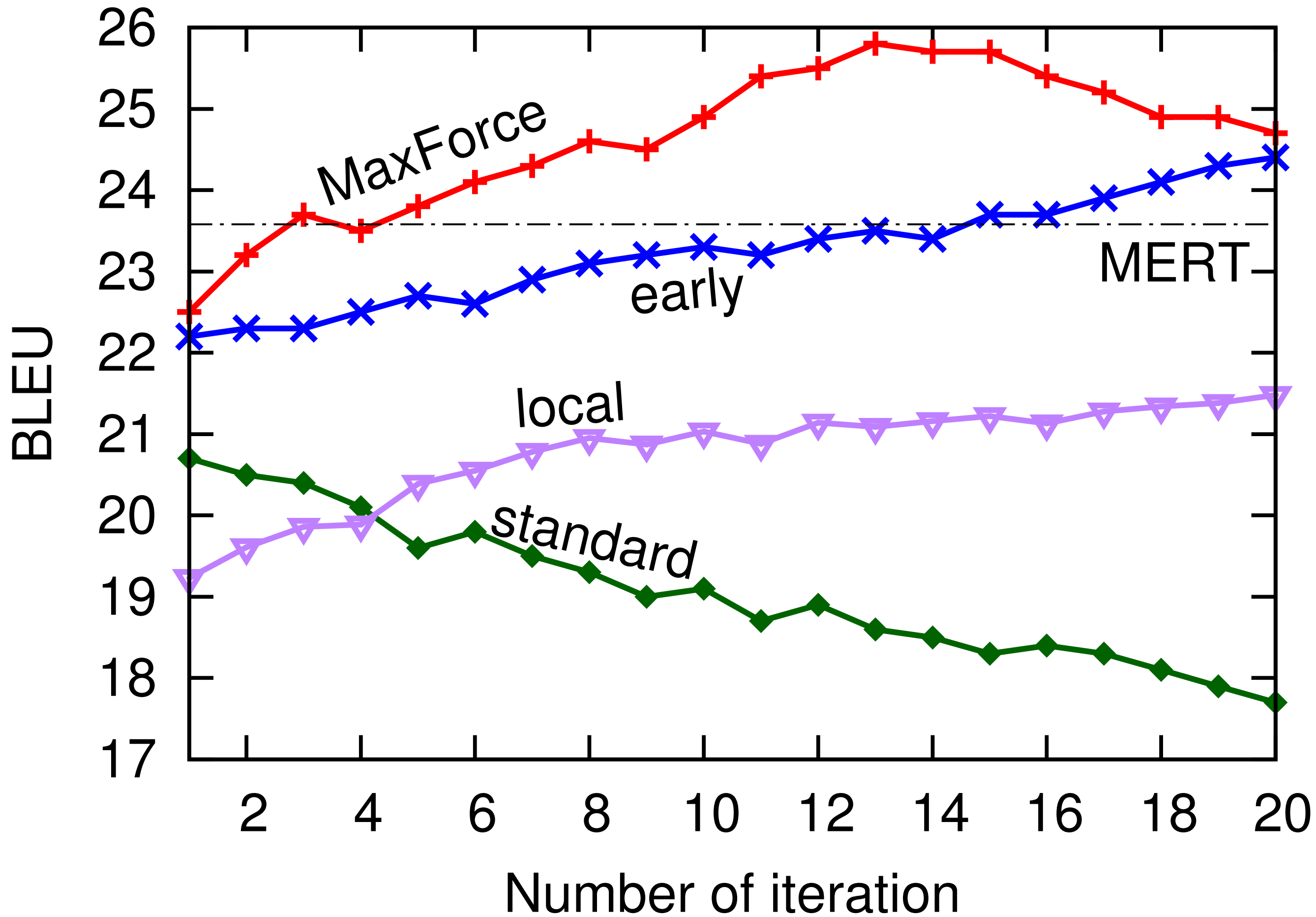
this explains why Liang et al '06 failed
std ~ "bold"; local ~ "local"

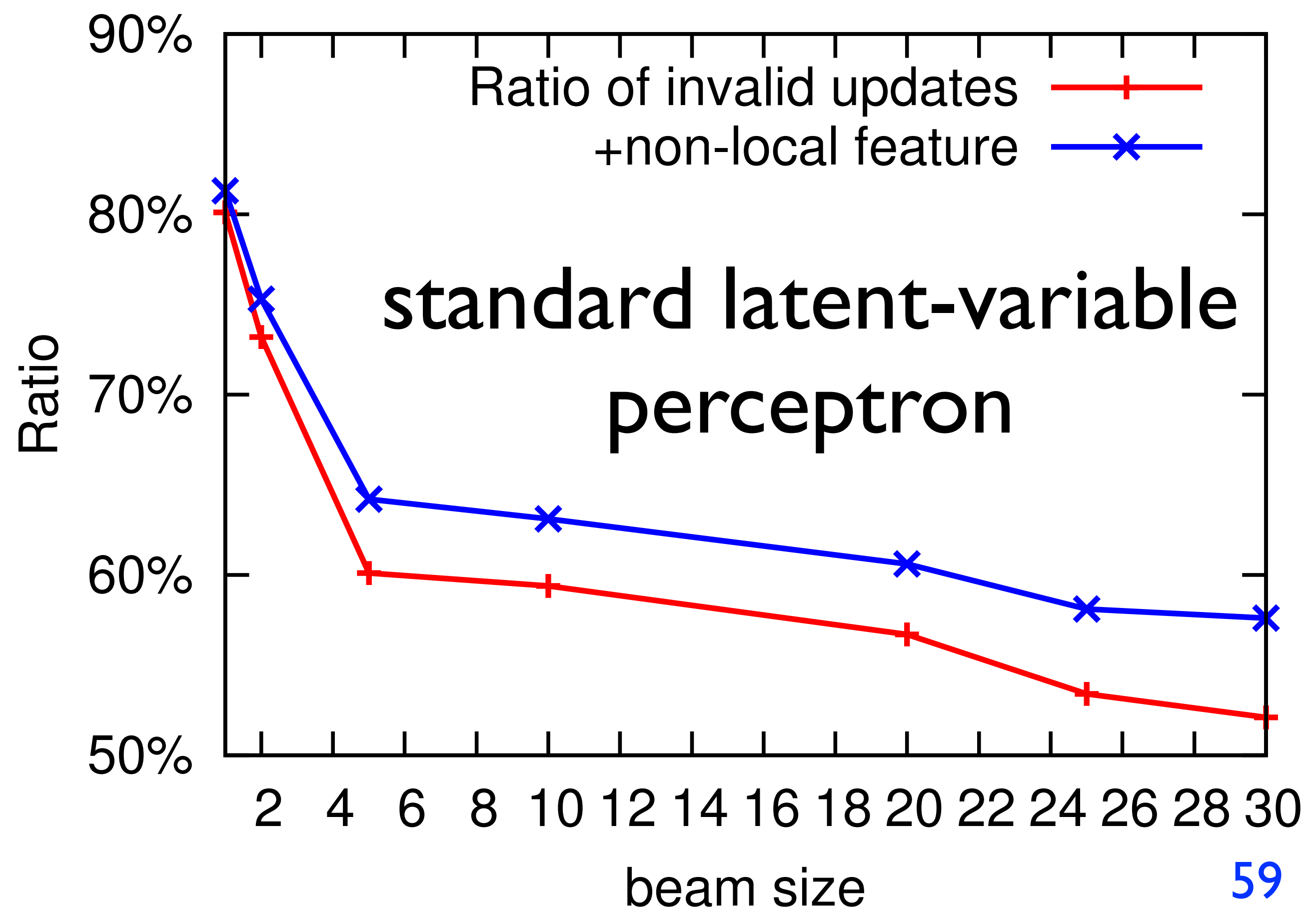


Experiments: Discriminative Training for MT

- standard update (Liang et al's "bold") works poorly
  - b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update

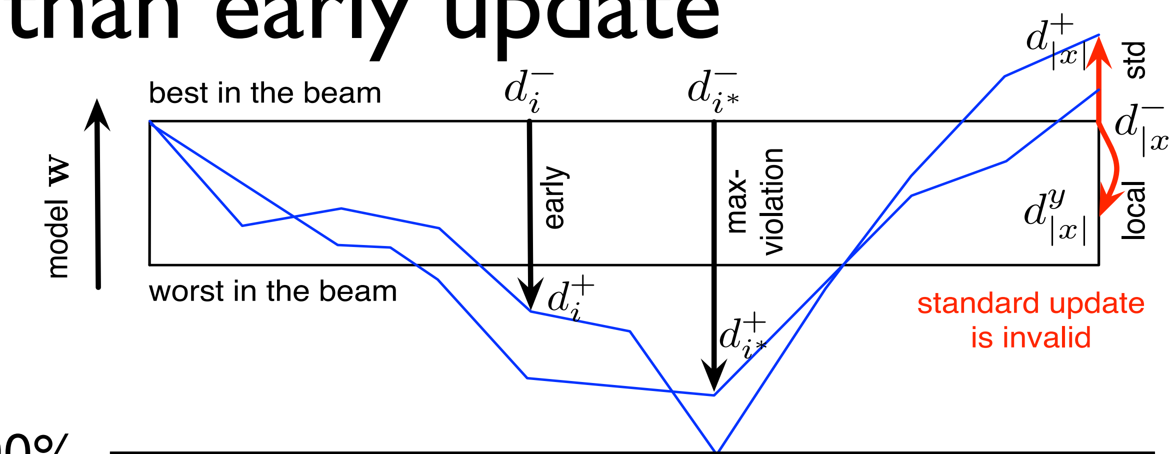
this explains why Liang et al '06 failed
std ~ "bold"; local ~ "local"



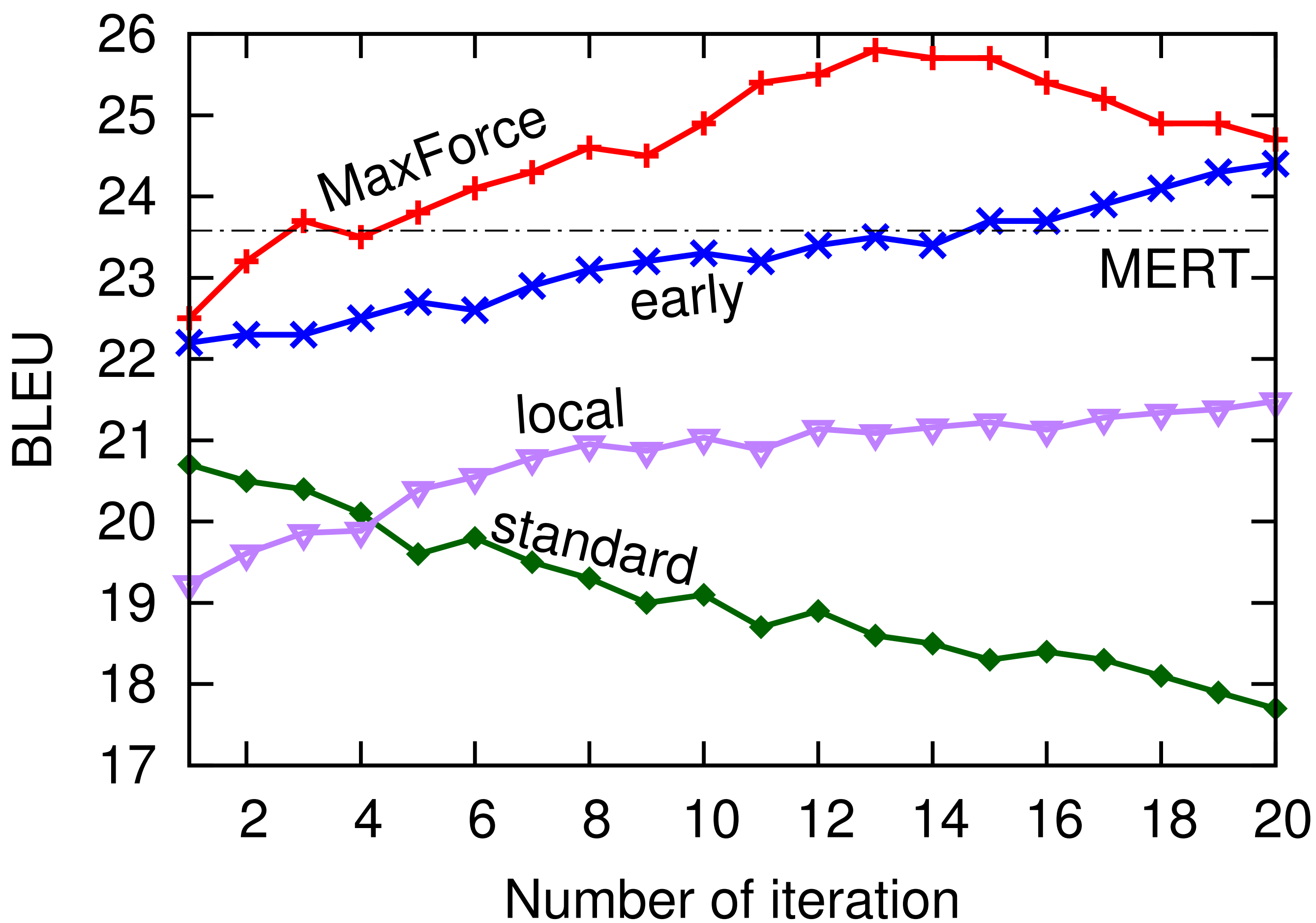


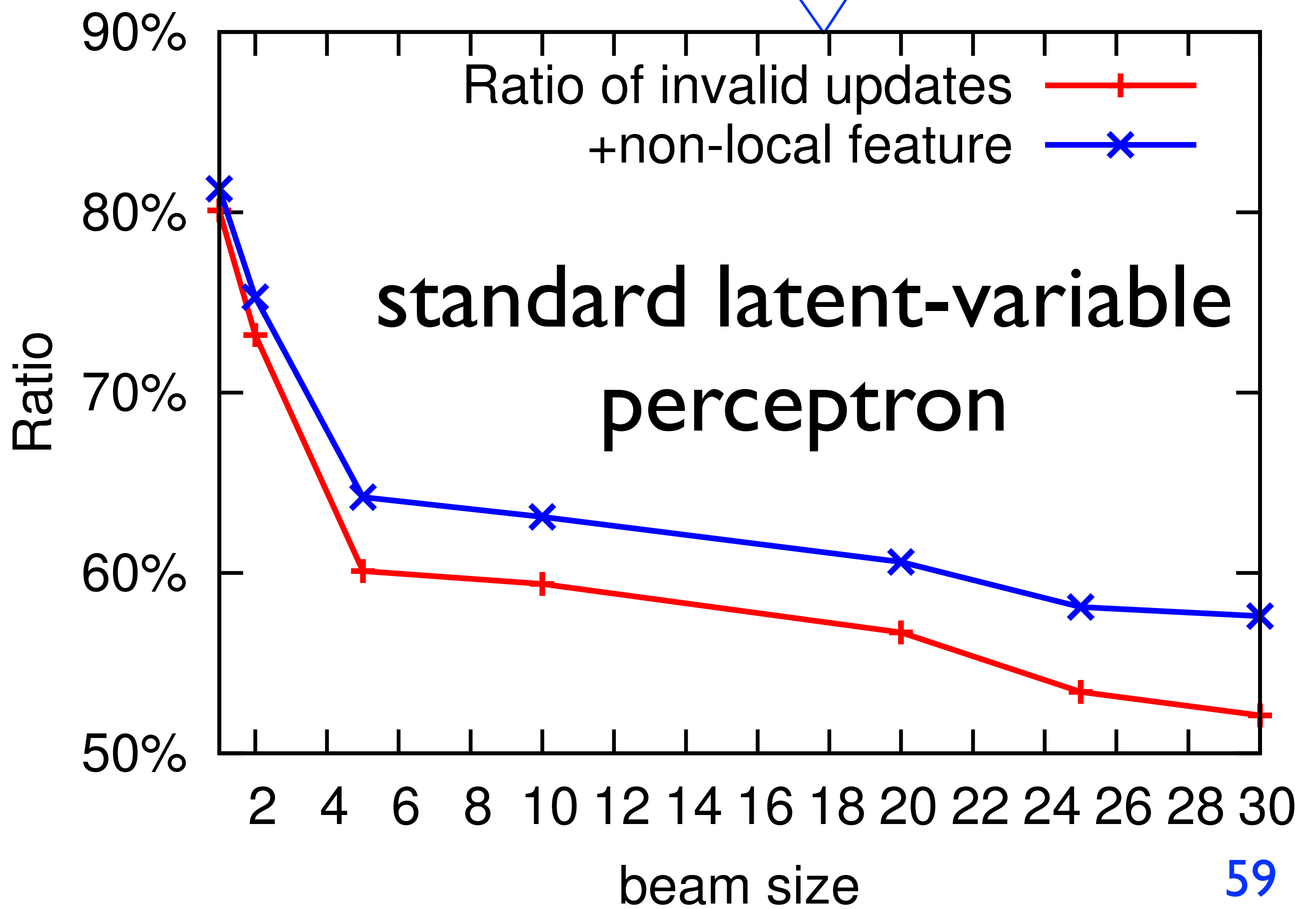
Experiments: Discriminative Training for MT

- standard update (Liang et al's "bold") works poorly
  - b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update



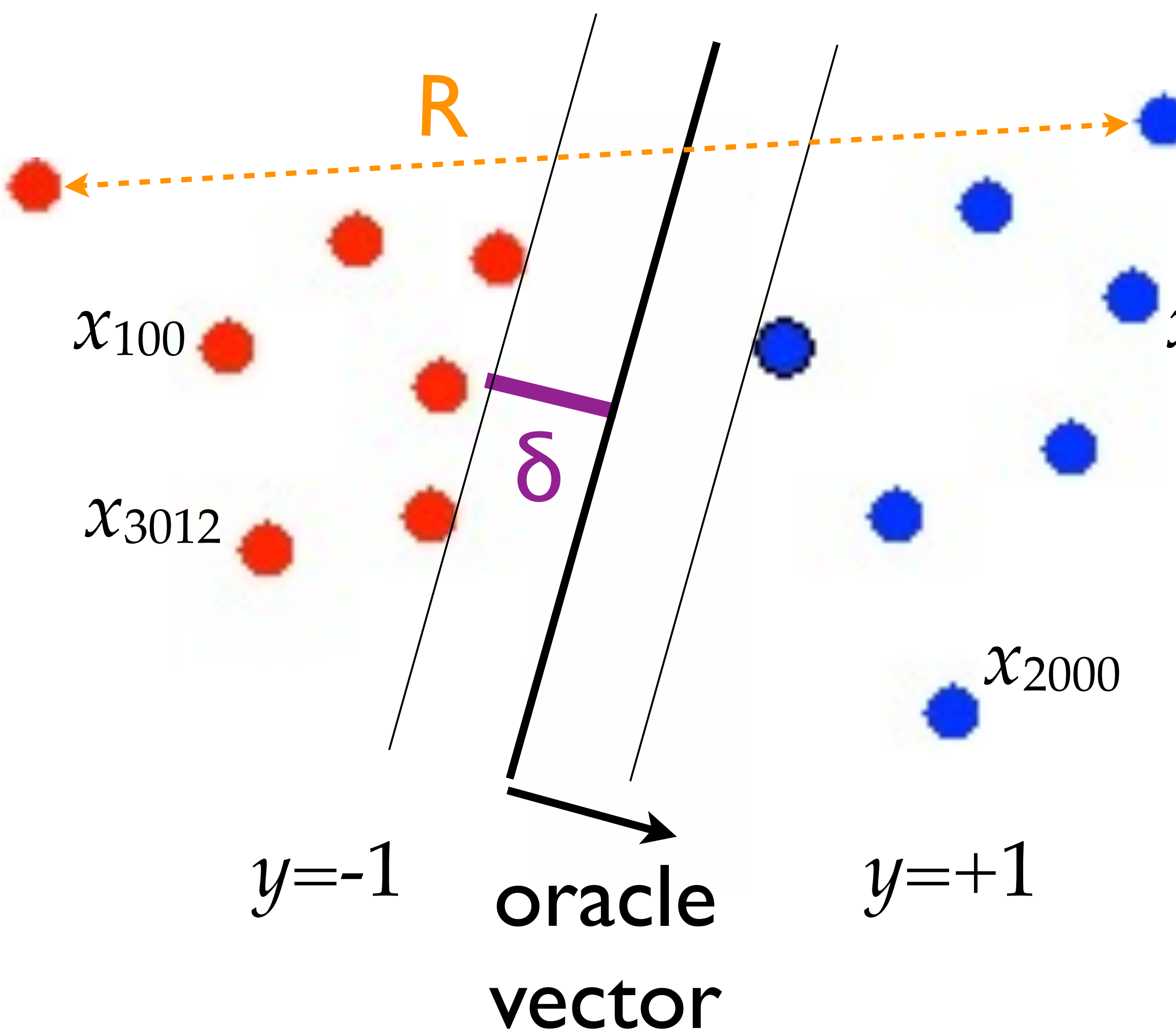
this explains why Liang et al '06 failed
std ~ "bold"; local ~ "local"





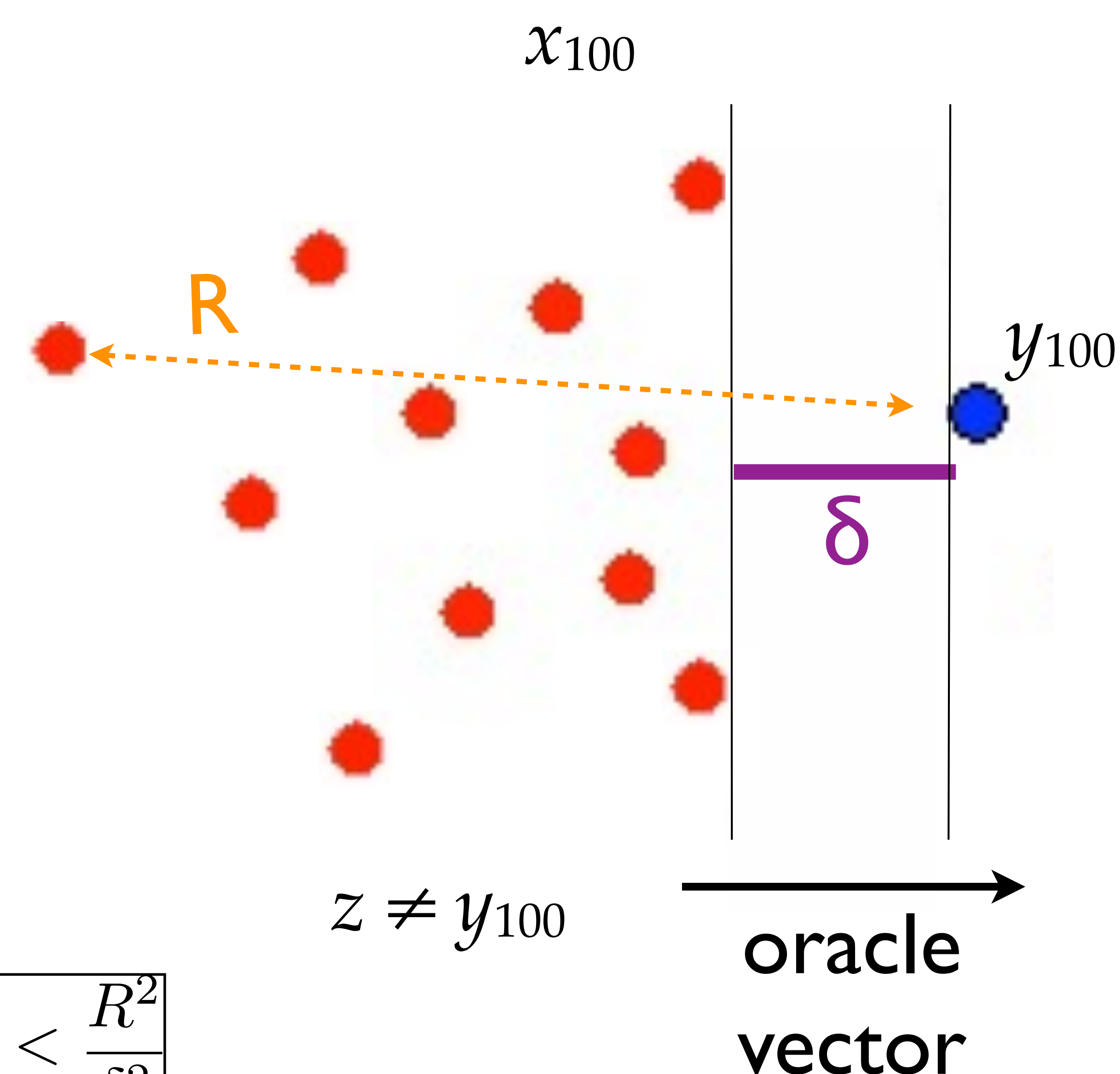
Open Problems in Theory

- latent-variable structured perceptron:
  - does it converge? under what conditions?
- latent-variable structured perceptron with inexact search
  - does it converge? under what conditions?



(Novikoff, 1962)

$$\boxed{\#\text{updates} \leq \frac{R^2}{\delta^2}}$$

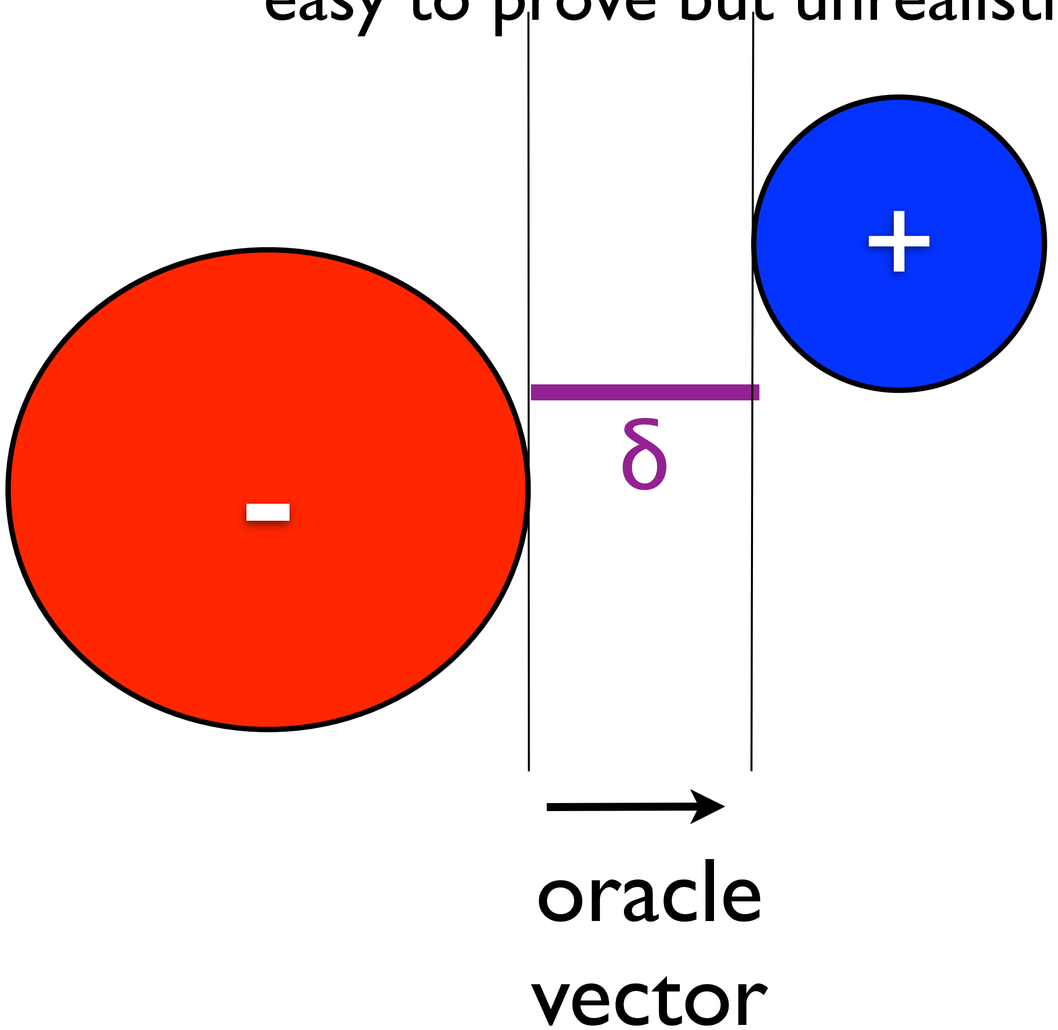


(Collins, 2002)

Open Problems in Theory

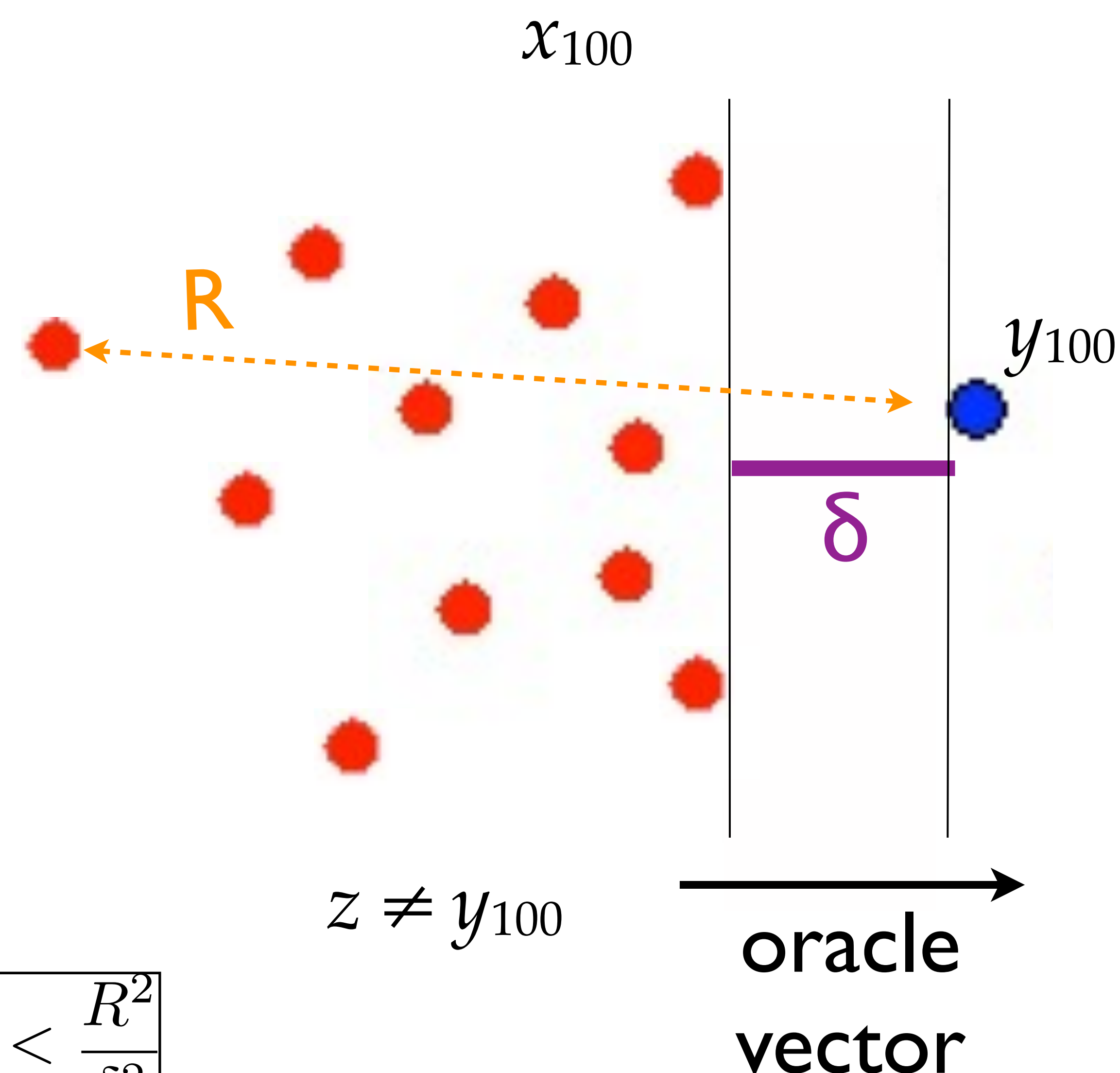
- latent-variable structured perceptron:
  - does it converge? under what conditions?
- latent-variable structured perceptron with inexact search
  - does it converge? under what conditions?

easy to prove but unrealistic



(Sun et al, 2009)

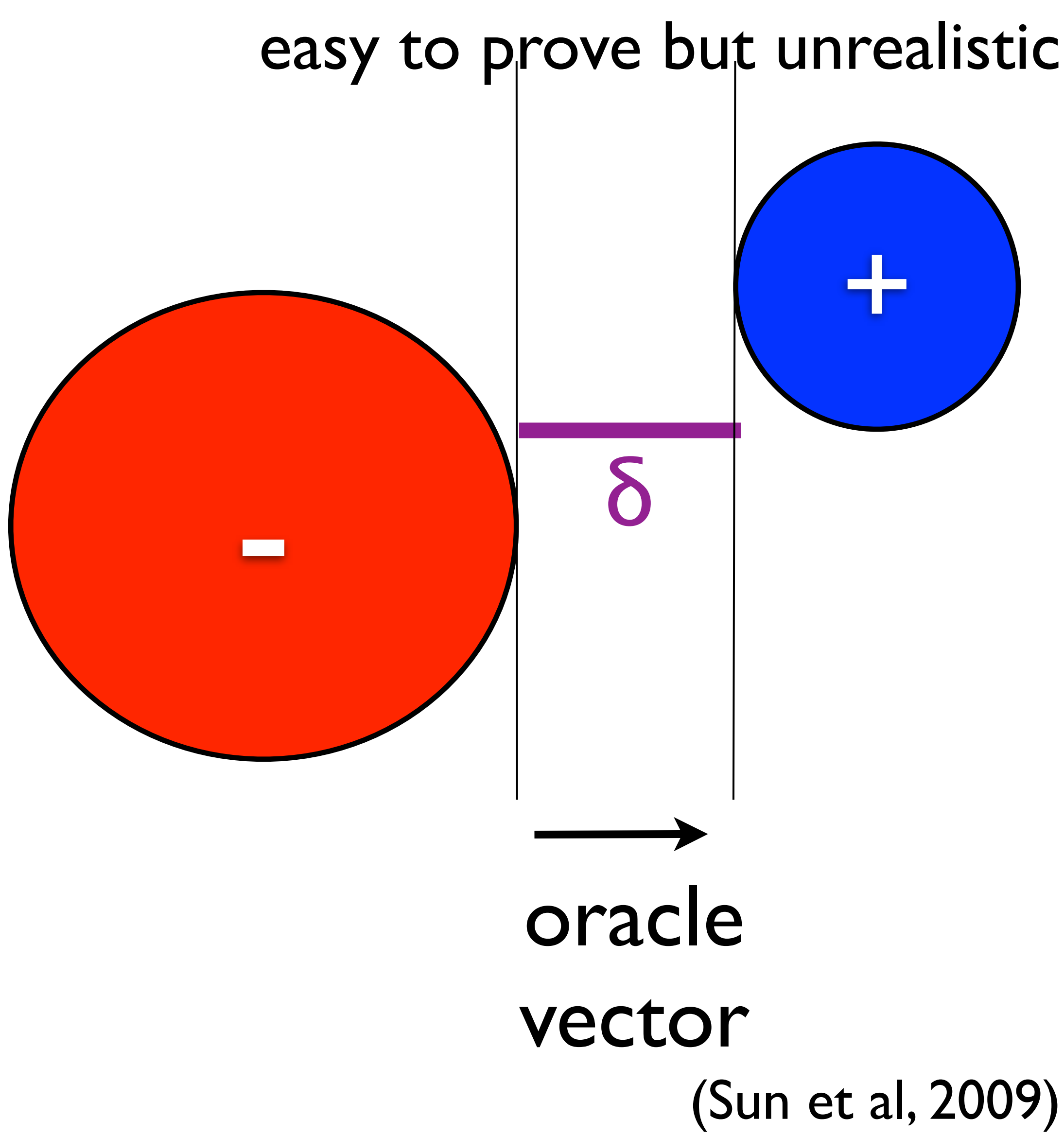
$$\boxed{\#\text{updates} \leq \frac{R^2}{\delta^2}}$$



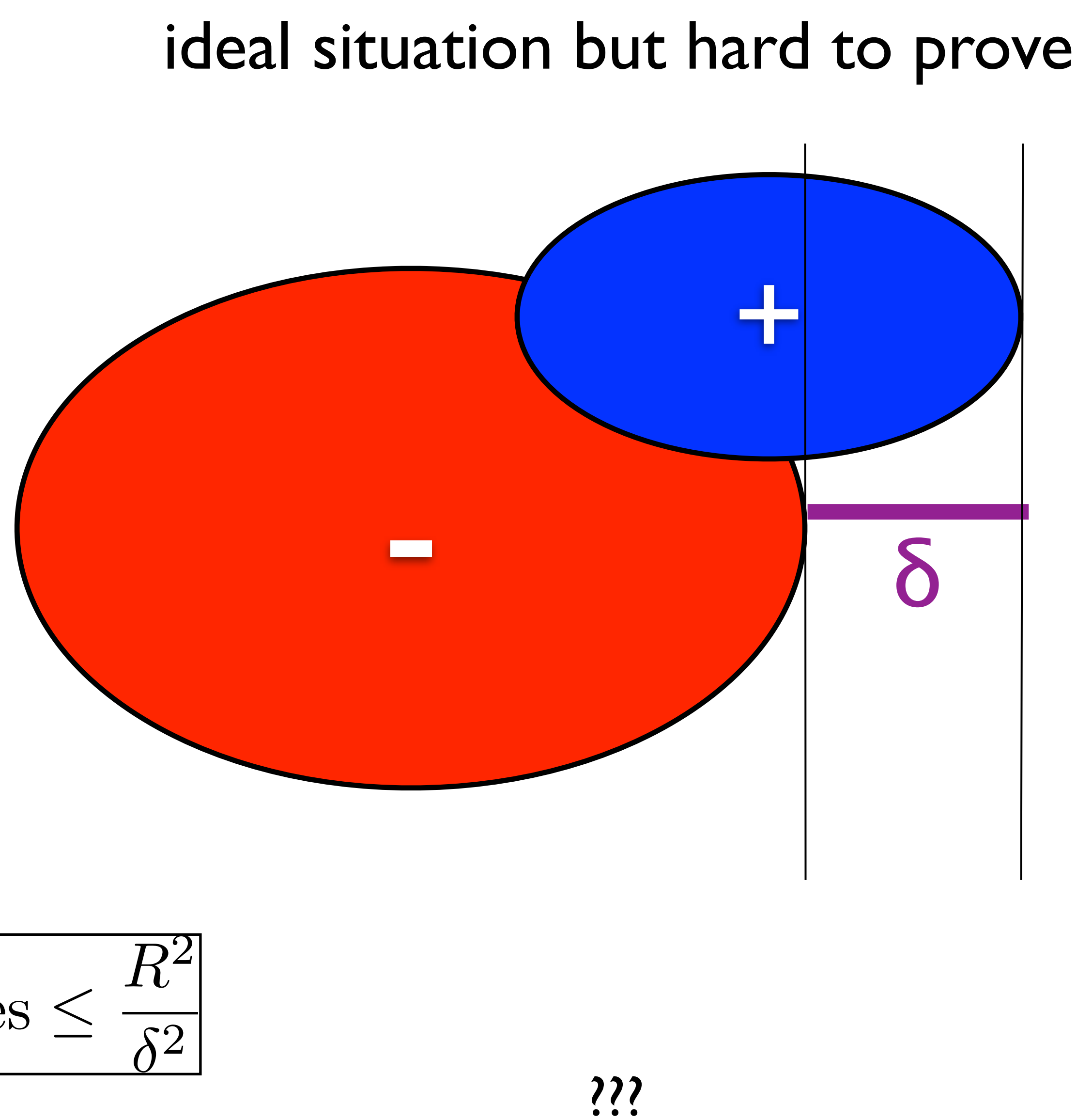
(Collins, 2002)

Open Problems in Theory

- latent-variable structured perceptron:
  - does it converge? under what conditions?
- latent-variable structured perceptron with inexact search
  - does it converge? under what conditions?



$$\boxed{\#\text{updates} \leq \frac{R^2}{\delta^2}}$$

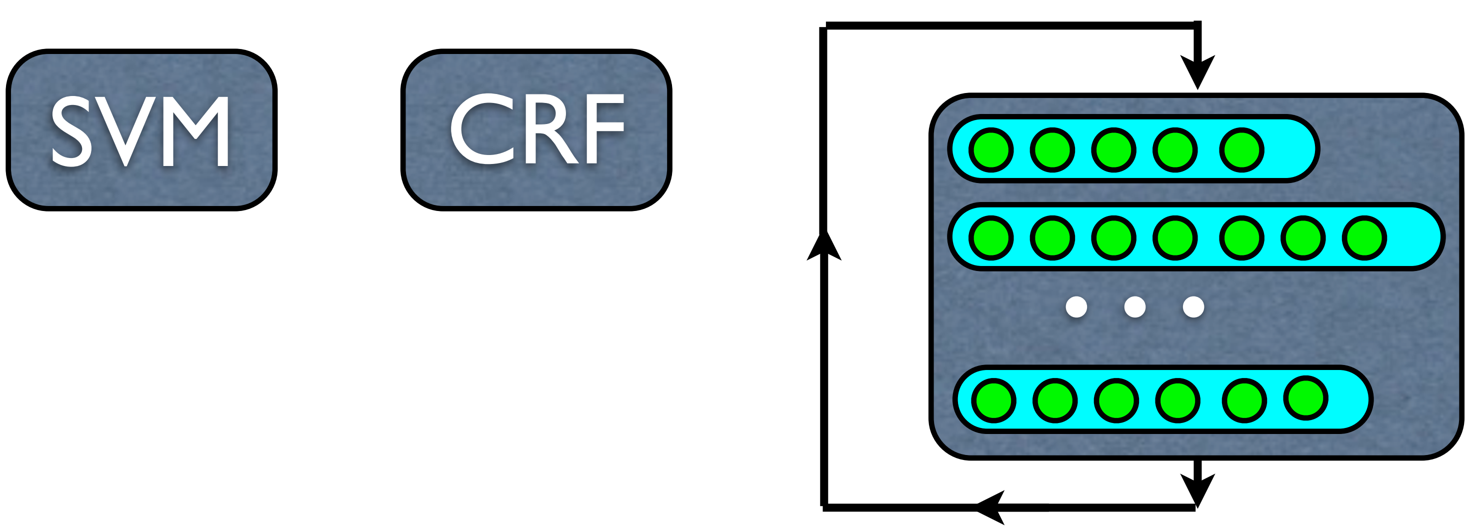


Tutorial Outline

- Overview of Structured Learning
  - Challenges in Scalability
- Structured Perceptron
  - convergence proof
- Structured Perceptron with Inexact Search
- Latent-Variable Perceptron
- Parallelizing Online Learning (Perceptron & MIRA)

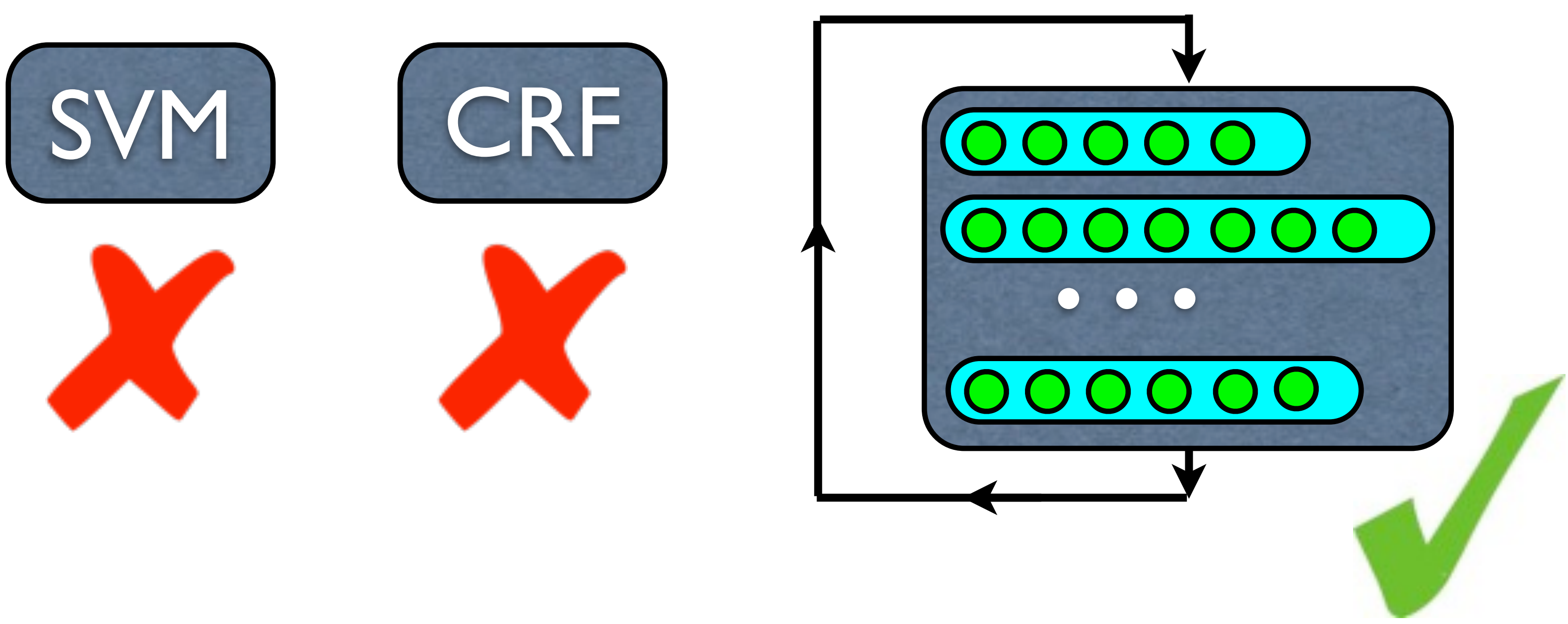
Online Learning from Big Data

- online learning has linear-time guarantee
  - contrast: popular methods such as SVM/CRF are superlinear
  - but online learning can still be too slow if data is too big
    - even with the fastest inexact search (e.g. greedy)
- how to parallelize online learning for big data?



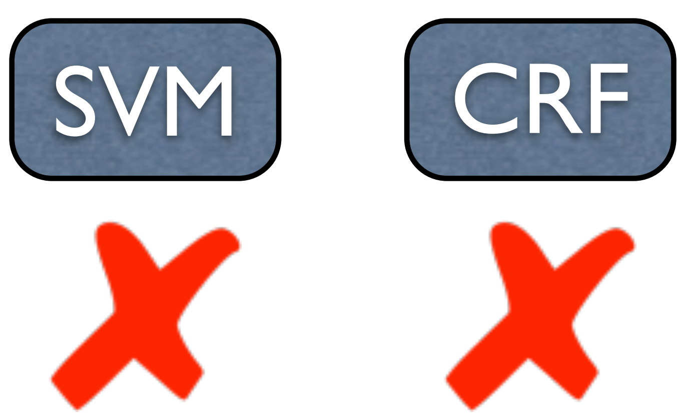
Online Learning from Big Data

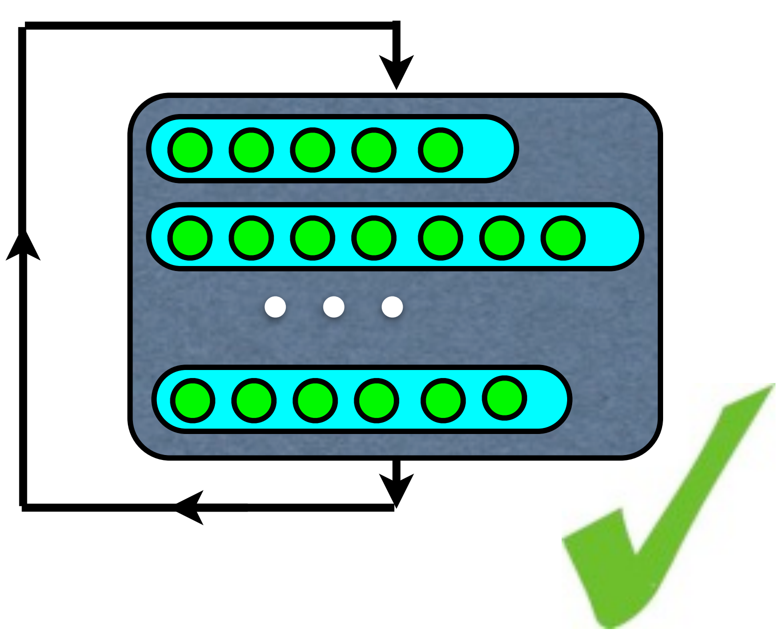
- online learning has linear-time guarantee
  - contrast: popular methods such as SVM/CRF are superlinear
  - but online learning can still be too slow if data is too big
    - even with the fastest inexact search (e.g. greedy)
- how to parallelize online learning for big data?

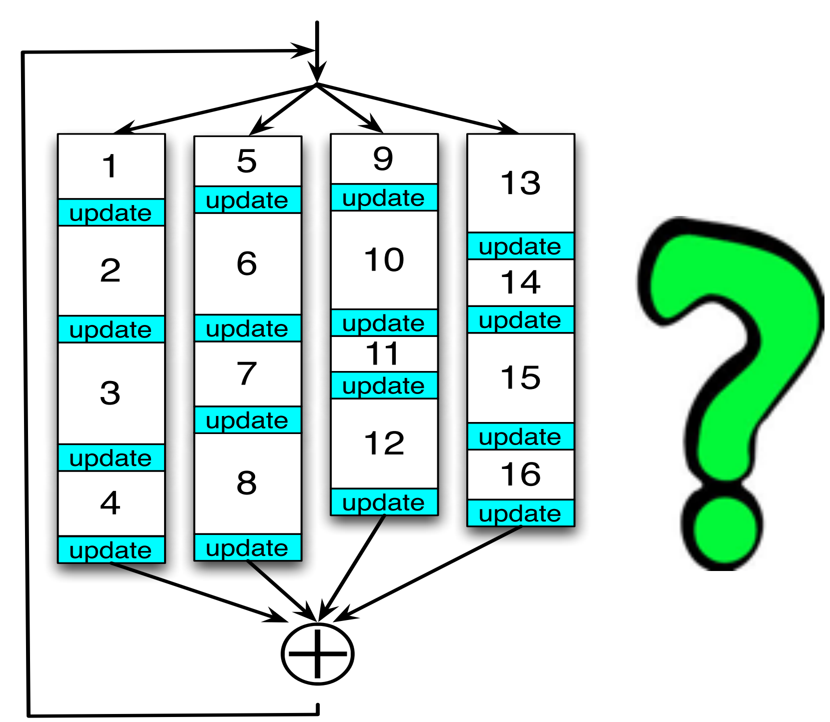


Online Learning from Big Data

- online learning has linear-time guarantee
  - contrast: popular methods such as SVM/CRF are superlinear
  - but online learning can still be too slow if data is too big
    - even with the fastest inexact search (e.g. greedy)
- how to parallelize online learning for big data?

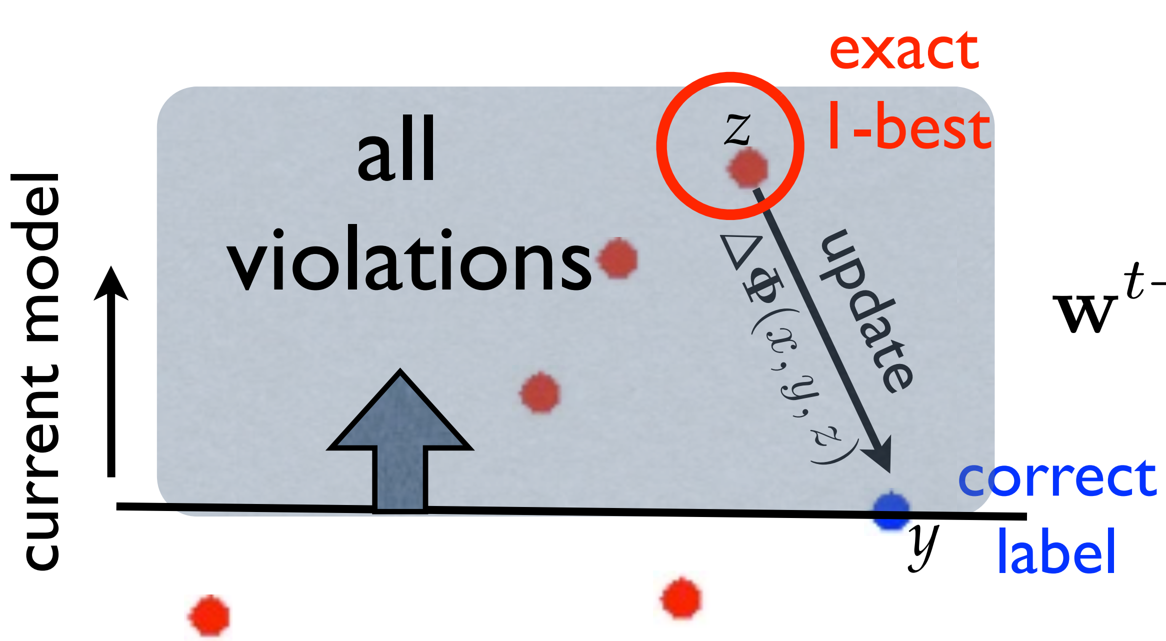






Aside: Perceptron => MIRA

- perceptron is simple but...
  - only "aims to" fixes one mistake (violation) on each example
  - but structured prediction may have many violations
  - may under- or over-correct on a violation
- MIRA (margin infused relaxation algorithm)
  - 1-best MIRA: corrects one violation "just enough" (Crammer 03)
  - *k*-best MIRA: corrects *k* violations at one time (McDonald et al 05)



$$Z = \underset{z \in \mathbf{GEN}(x)}{k\text{-best}} \ \mathbf{w}^t \cdot \mathbf{\Phi}(x, z)$$

$$\mathbf{w}^{t+1} = \underset{\mathbf{w}' : \forall z \in Z, \mathbf{w}' \cdot \Delta\mathbf{\Phi}(x,y,z) \geq \ell(y,z)}{\mathbf{argmin}} \|\mathbf{w}' - \mathbf{w}^t\|^2$$

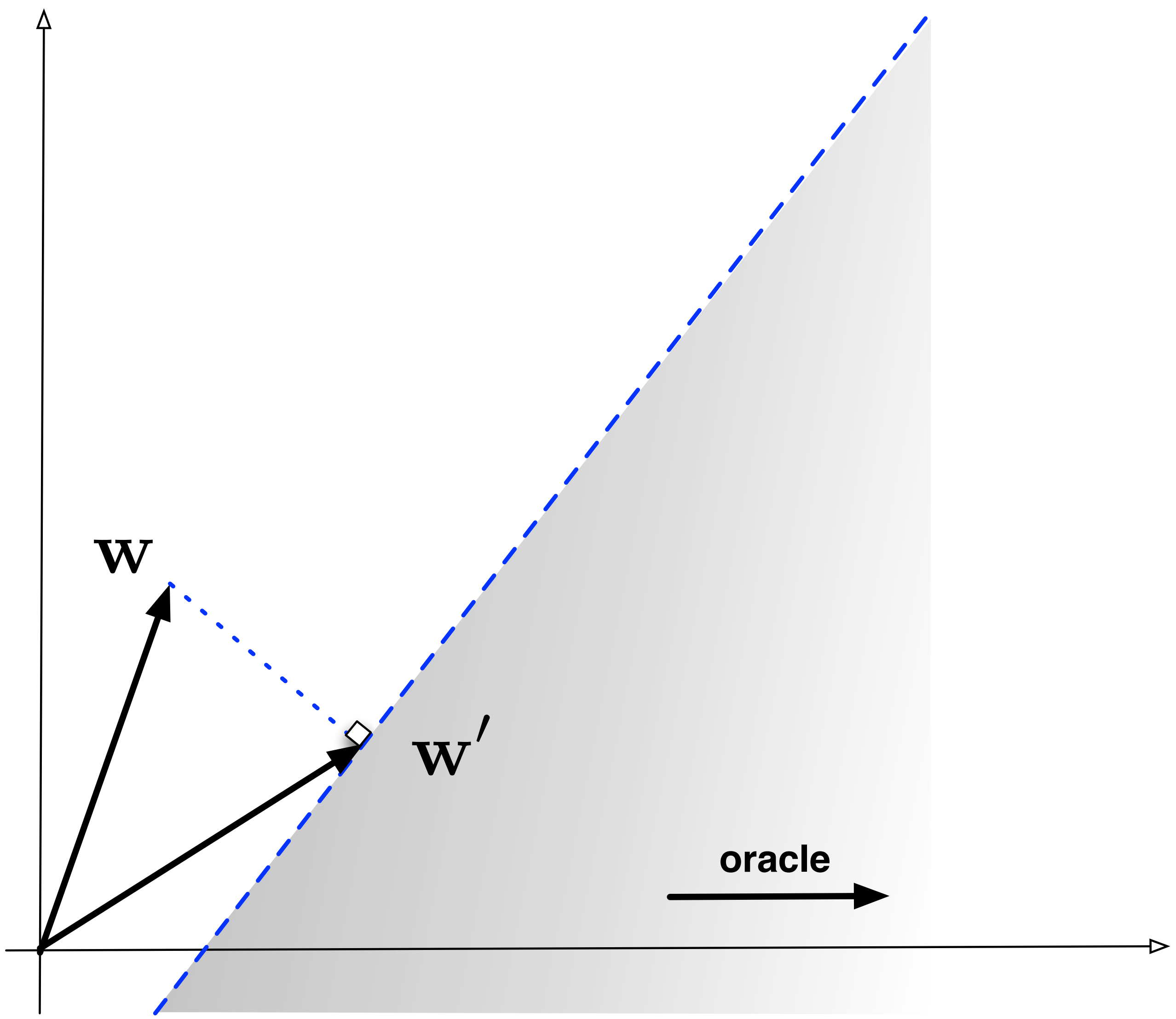
Geometry of 1-best & *k*-best MIRA

$$\mathbf{w}^{t+1} = \underset{\mathbf{w}':\forall z \in Z, \mathbf{w}'\cdot\Delta\boldsymbol{\Phi}(x,y,z) \geq \ell(y,z)}{\mathbf{argmin}} \|\mathbf{w}' - \mathbf{w}^t\|^2$$

1-best MIRA

single constraint

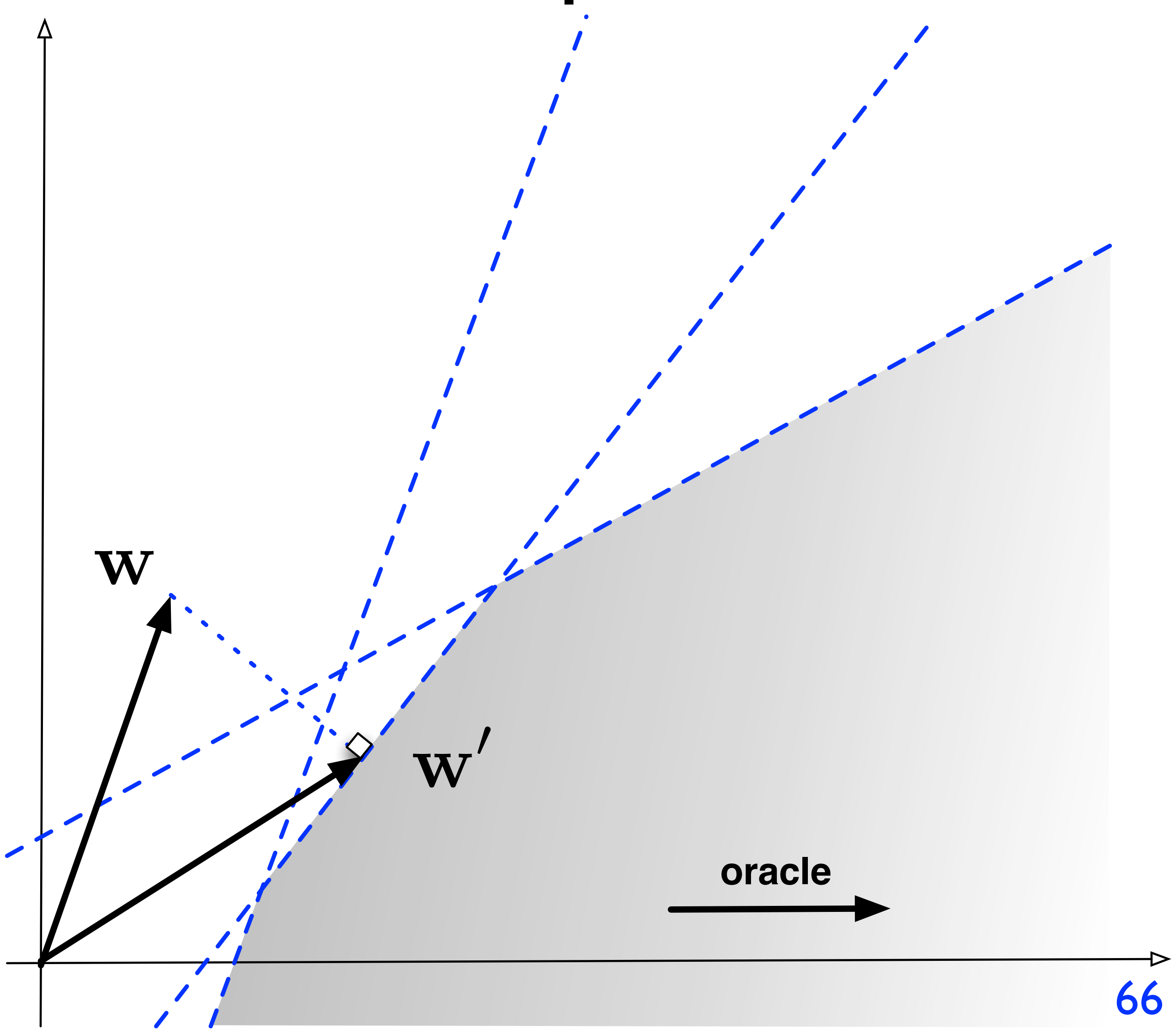
=> analytic solution



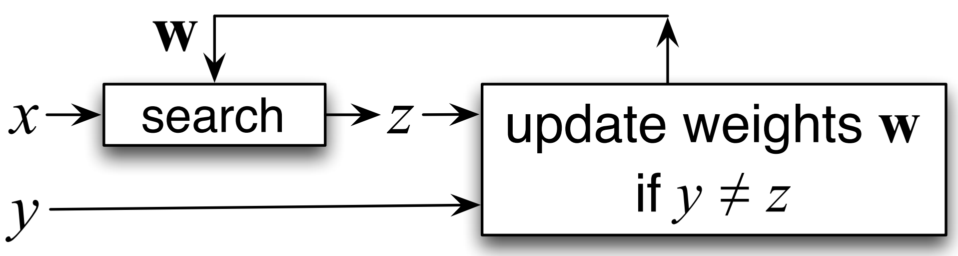
k-best MIRA

up to *k* constraints

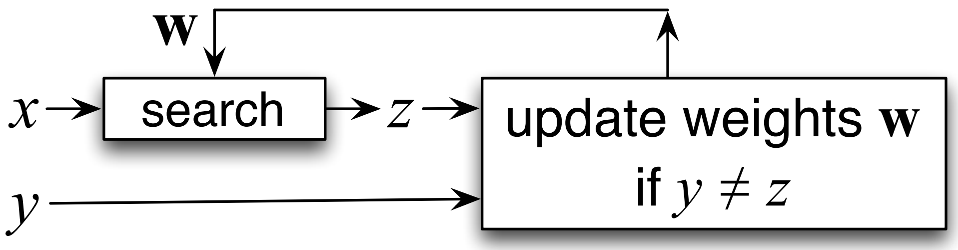
=> convex optimization



Can We Parallelize Online Learning?

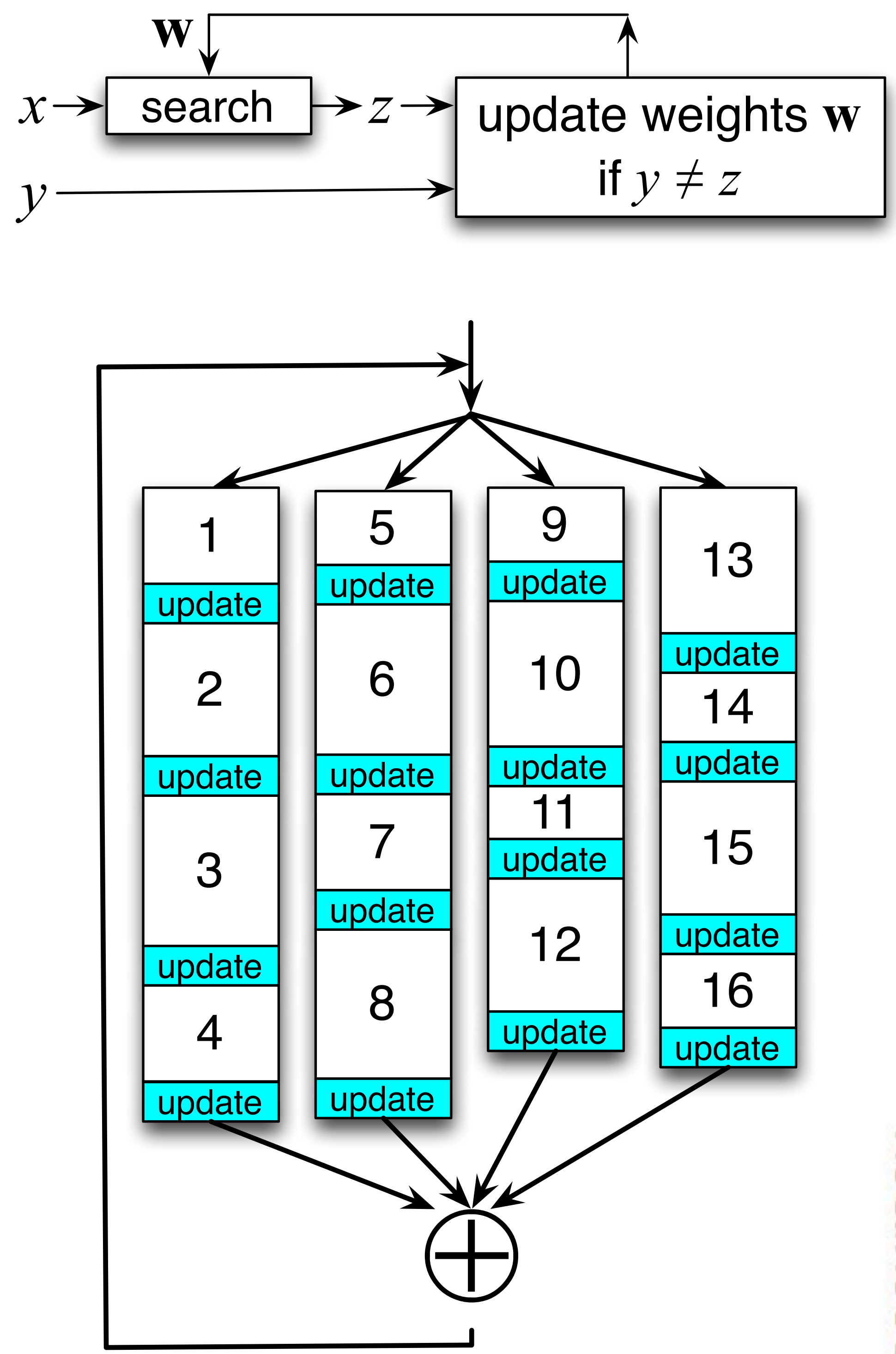


Can We Parallelize Online Learning?



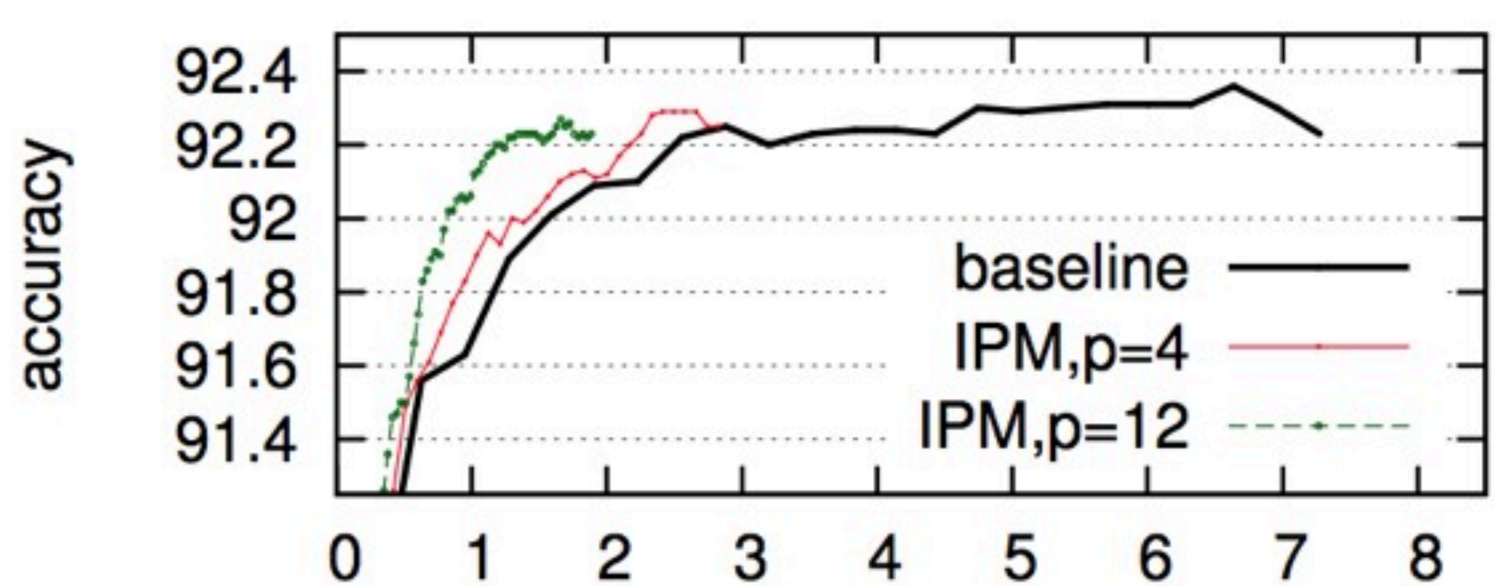
- can we parallelize online learning?
  - harder than parallelizing batch (CRF)
  - losing dependency b/w examples
  - each iteration faster, but accuracy lower

Method 1: Iterative Parameter Mixing

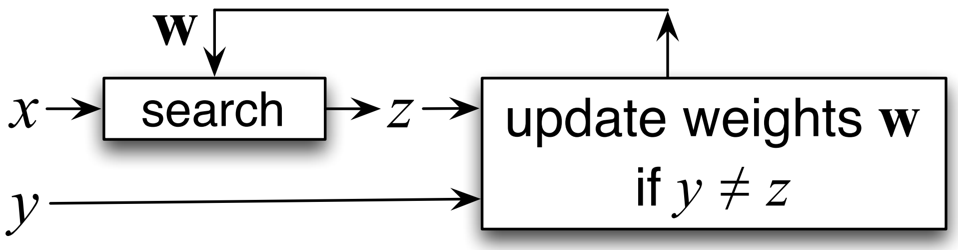


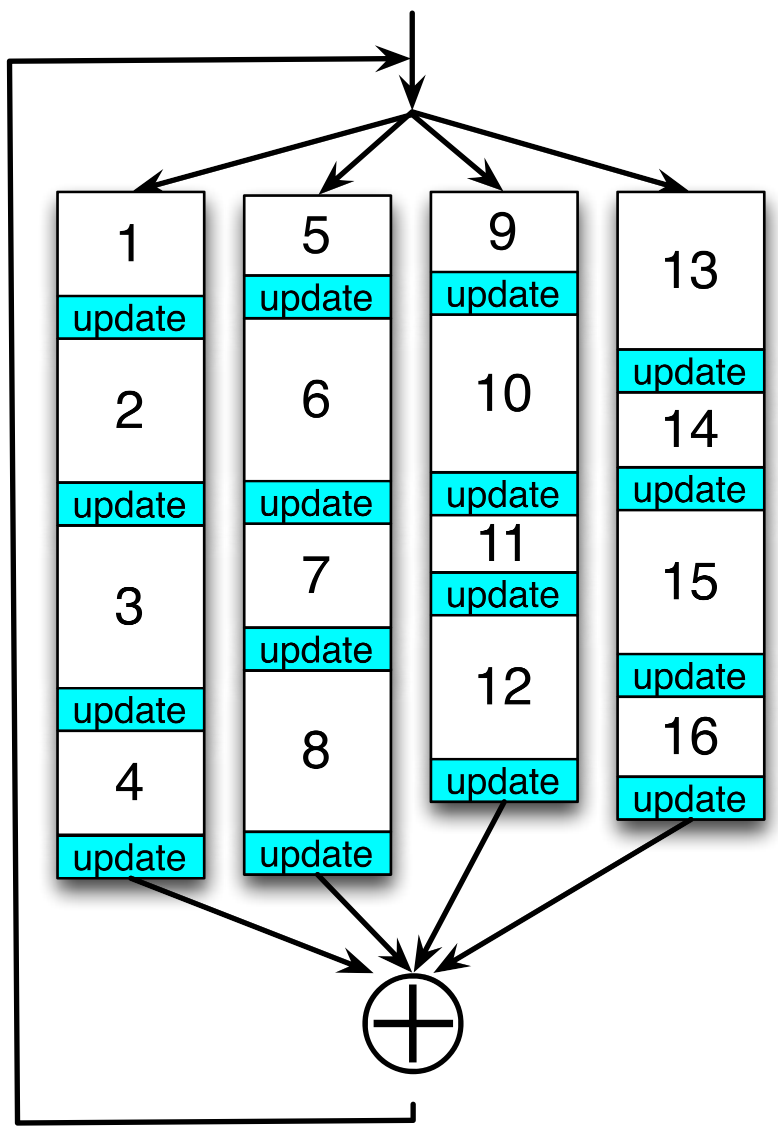
Iterative Parameter Mixing (IPM)
(McDonald et al 2010)

- can we parallelize online learning?
  - harder than parallelizing batch (CRF)
  - losing dependency b/w examples
  - each iteration faster, but accuracy lower
- method 1: IPM (McDonald et al 2010) only ~3-4x faster on 10+ CPUs



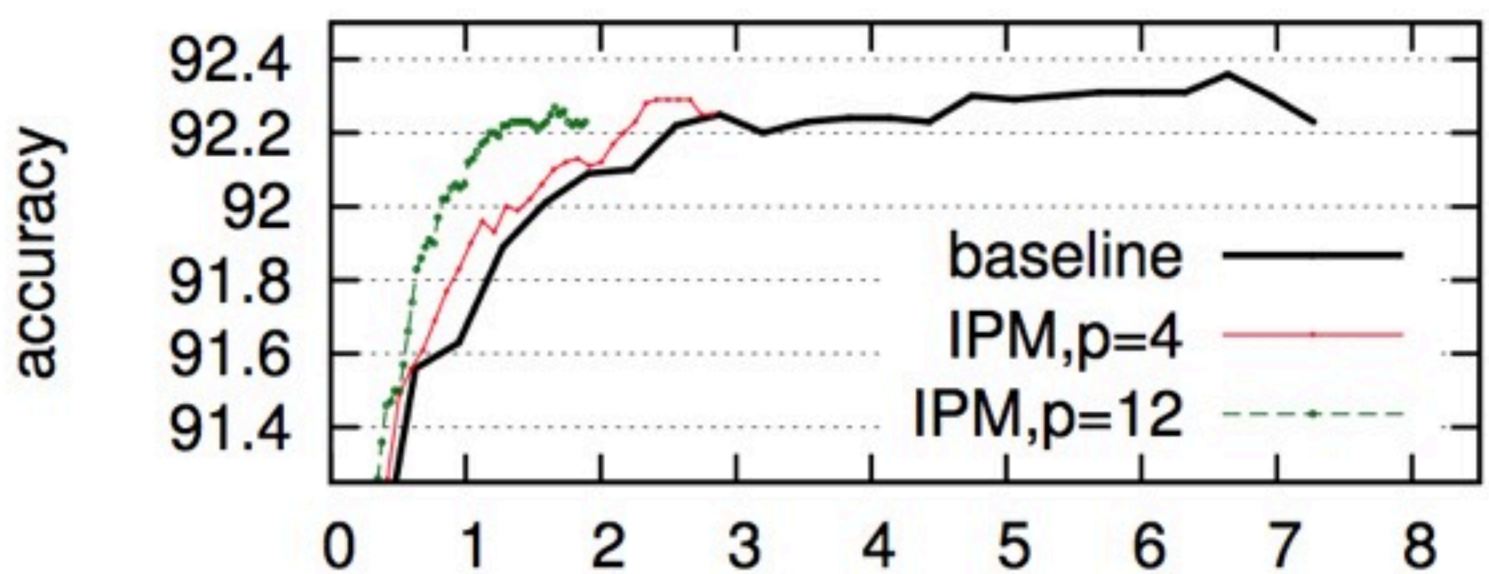
Method 1: Iterative Parameter Mixing





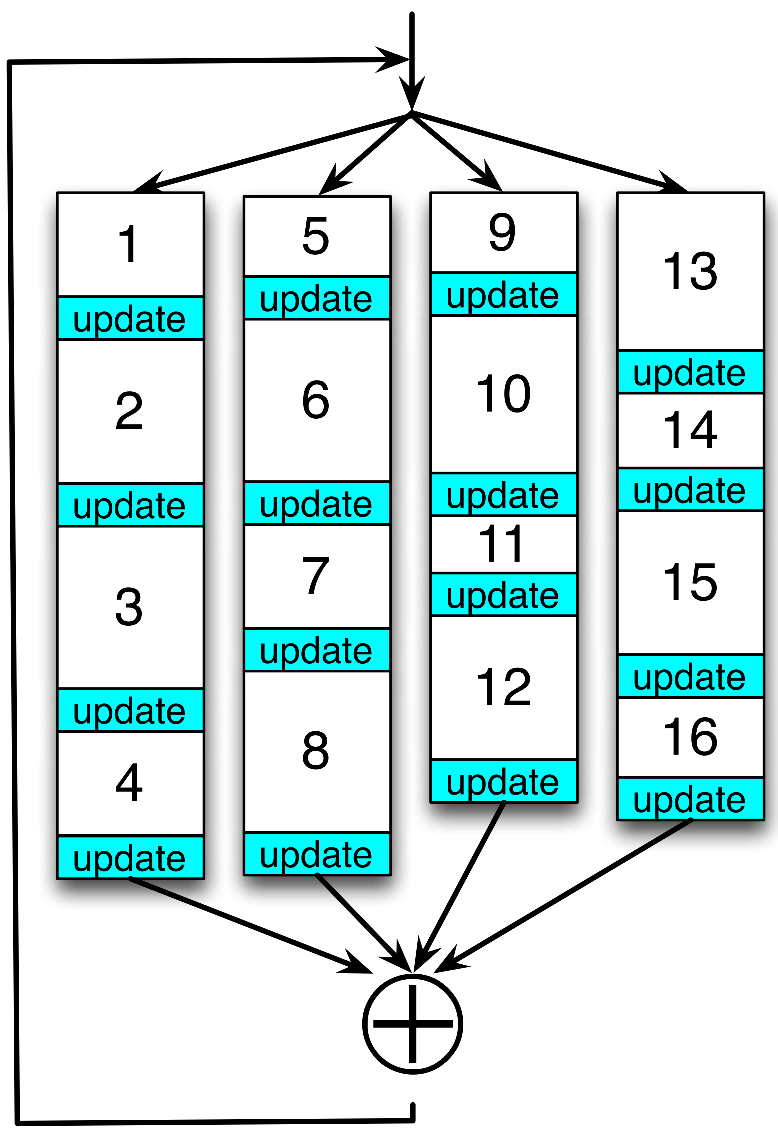
Iterative Parameter Mixing (IPM)
(McDonald et al 2010)

- can we parallelize online learning?
  - harder than parallelizing batch (CRF)
  - losing dependency b/w examples
  - each iteration faster, but accuracy lower
- method 1: IPM (McDonald et al 2010) only ~3-4x faster on 10+ CPUs
- can we do (a lot) better?

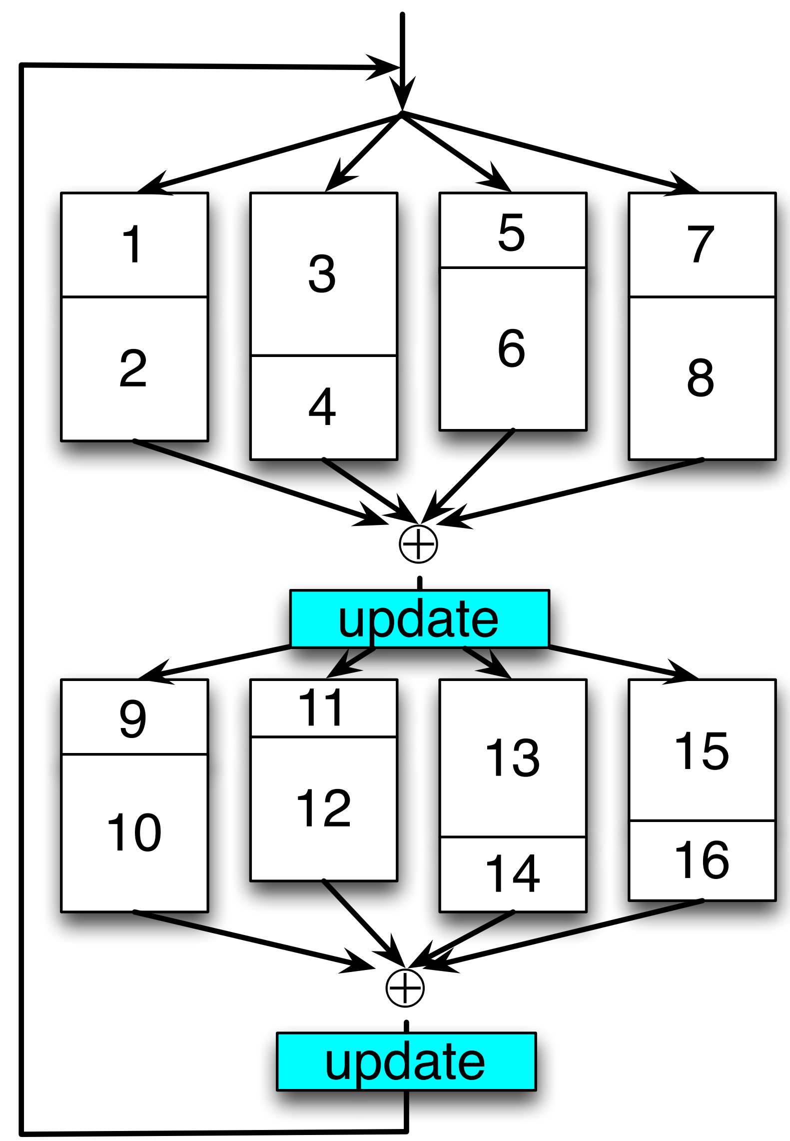


Method 2: Minibatch Parallelization

- decode a minibatch in parallel, update in serial



Iterative Parameter Mixing (IPM)
(McDonald et al 2010)



minibatch
(Zhao and Huang, 2012)

Why Minibatch?

- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
  - "averaging effect" (cf. McDonald et al 2010)
  - easy to prove convergence (still R^2/δ^2)

Why Minibatch?

- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
    - "averaging effect" (cf. McDonald et al 2010)
    - easy to prove convergence (still R^2/δ^2)

update $$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \frac{1}{a}\sum_i \Delta\mathbf{\Phi}(x, y, z)$$

Why Minibatch?

- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
  - "averaging effect" (cf. McDonald et al 2010)
  - easy to prove convergence (still R^2/δ^2)

update $$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \frac{1}{a}\sum_i \Delta\mathbf{\Phi}(x, y, z)$$

lower bound

$$\mathbf{u}\cdot\mathbf{w}^{(k+1)} = \mathbf{u}\cdot\mathbf{w}^{(k)} + \boxed{\frac{1}{a}\sum_i \mathbf{u}\cdot\Delta\mathbf{\Phi}(x, y, z) \geq \delta \quad \text{margin}}$$

$$\mathbf{u}\cdot\mathbf{w}^{(k+1)} \geq k\delta$$

$$\|\mathbf{w}^{(k+1)}\| \geq k\delta$$ (by induction)

Why Minibatch?

- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
  - "averaging effect" (cf. McDonald et al 2010)
  - easy to prove convergence (still R^2/δ^2)

update $$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \frac{1}{a}\sum_i \Delta\mathbf{\Phi}(x, y, z)$$

lower bound

$$\mathbf{u} \cdot \mathbf{w}^{(k+1)} = \mathbf{u} \cdot \mathbf{w}^{(k)} + \boxed{\underbrace{\frac{1}{a}\sum_i \mathbf{u} \cdot \Delta\mathbf{\Phi}(x, y, z)}_{\geq \delta \quad \text{margin}}}$$

$$\mathbf{u} \cdot \mathbf{w}^{(k+1)} \geq k\delta$$

$$\|\mathbf{w}^{(k+1)}\| \geq k\delta$$

(by induction)

upper bound

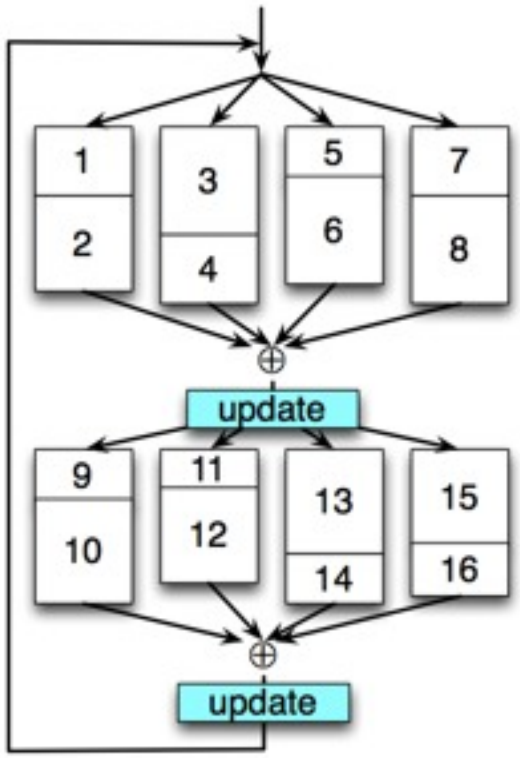
$$\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)}\|^2 + \boxed{\underbrace{\|\frac{1}{a}\sum_i \Delta\mathbf{\Phi}(x, y, z)\|^2}_{\leq R^2 \quad \text{Jensen's}}} + \boxed{\underbrace{\frac{2}{a}\mathbf{w}^{(k)} \cdot \sum_i \Delta\mathbf{\Phi}(x, y, z)}_{\leq 0 \quad \text{violation}}}$$

$$\|\mathbf{w}^{(k+1)}\|^2 \leq kR^2$$

(by induction)

Why Minibatch?

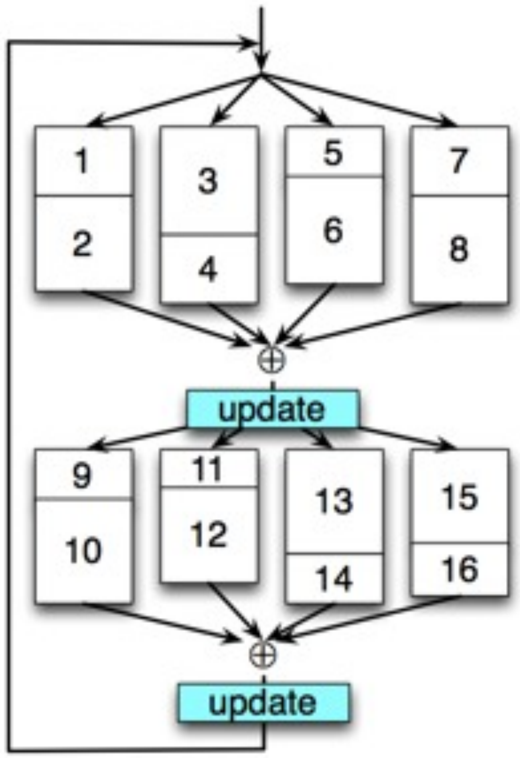
- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
  - "averaging effect" (cf. McDonald et al 2010)
  - easy to prove convergence (still R^2/δ^2)



8x constrains in each update

Why Minibatch?

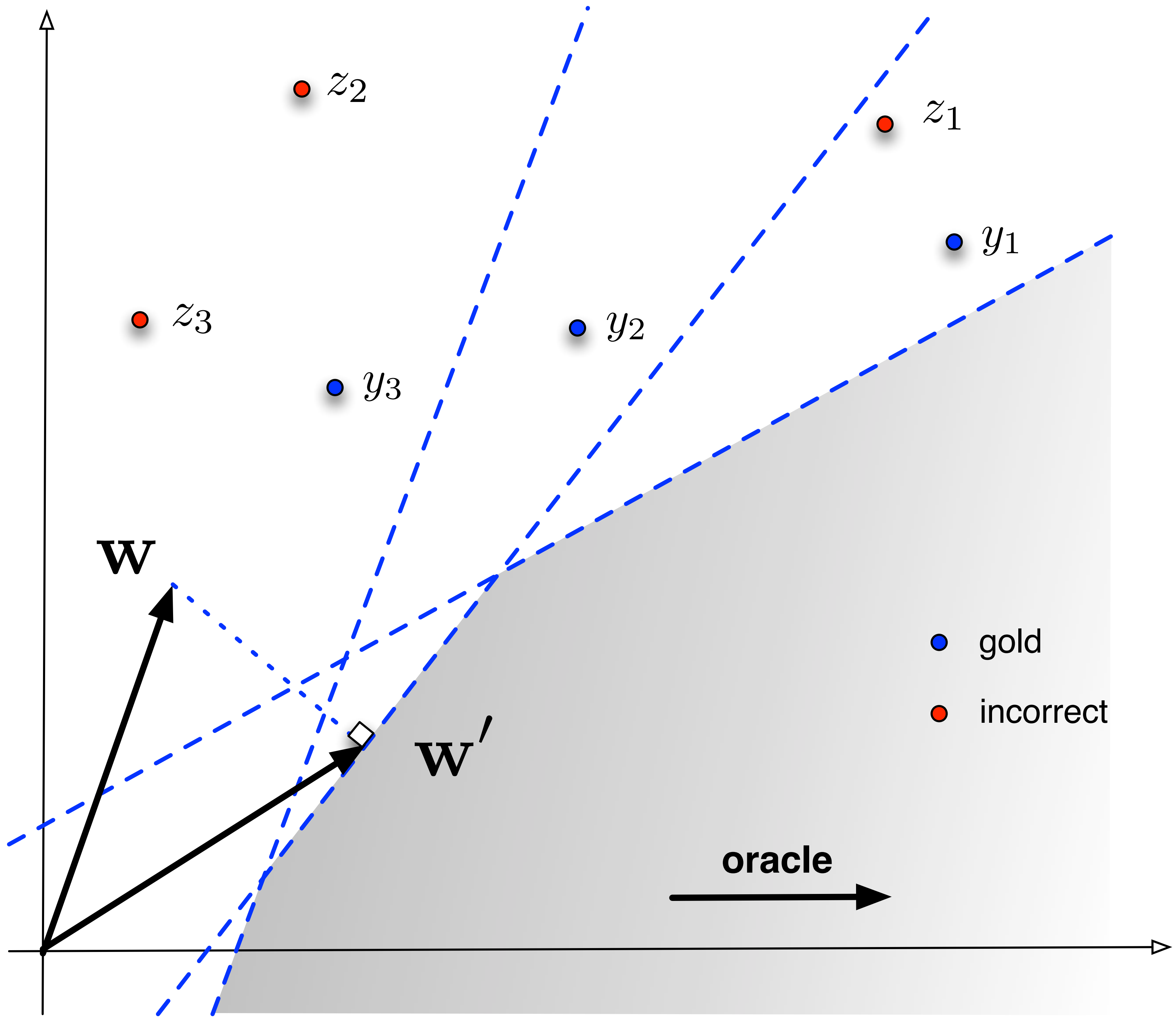
- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
  - "averaging effect" (cf. McDonald et al 2010)
  - easy to prove convergence (still R^2/δ^2)
- MIRA: optimization over more constraints
  - MIRA is an online approximation of SVM
  - minibatch MIRA: better approximation of SVM
    - approaches SVM at maximum batch size

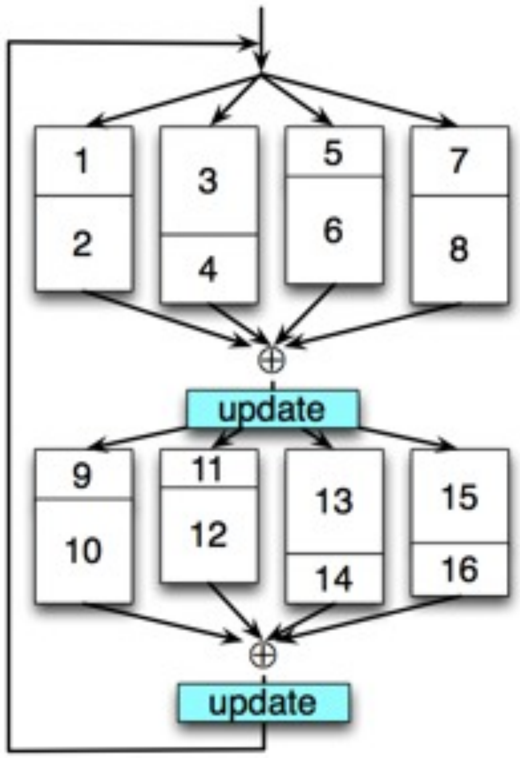


8x constrains in each update

$$\underset{\mathbf{w}'}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}' - \mathbf{w}\|^2$$
$$\text{s.t. } \mathbf{w}' \cdot \Delta\mathbf{\Phi}(x, y, z) \geq \ell(y, z)$$
$$\forall (x, y, z) \in V$$

Geometry of Minibatch MIRA





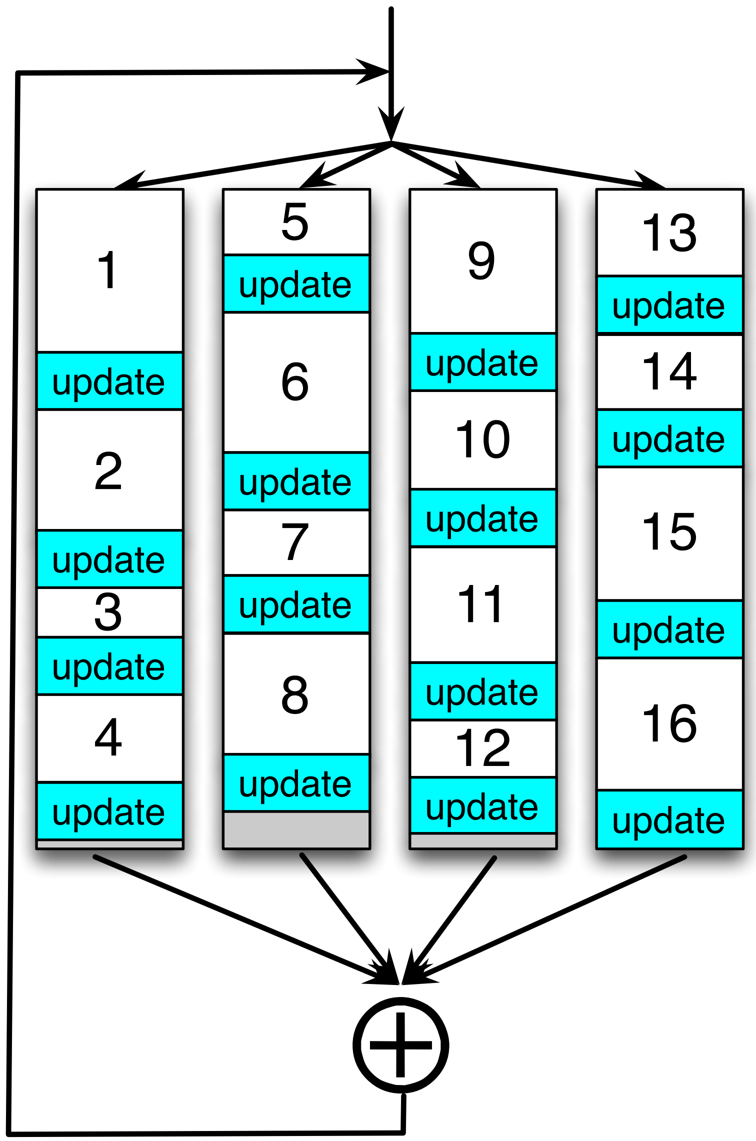
$$\underset{\mathbf{w}'}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}' - \mathbf{w}\|^2$$

$$\text{s.t. } \mathbf{w}' \cdot \Delta\mathbf{\Phi}(x, y, z) \geq \ell(y, z)$$

$$\forall (x, y, z) \in V$$

Load Balancing

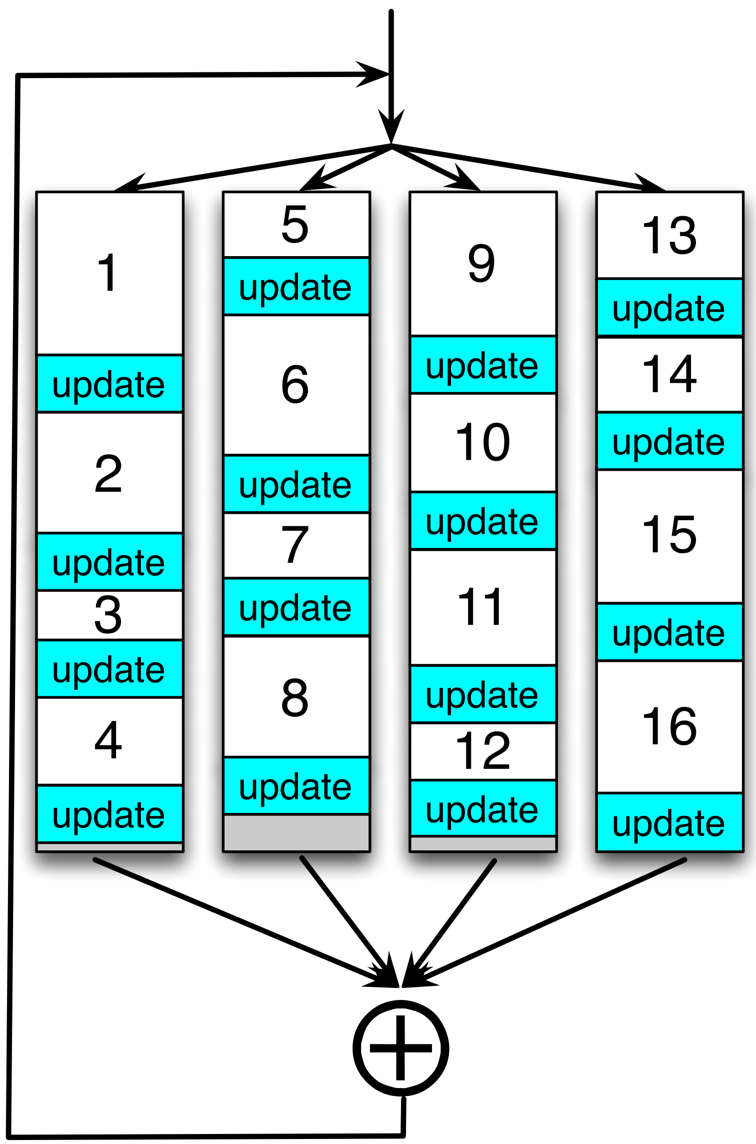
- rearrange each minibatch to minimize wasted time



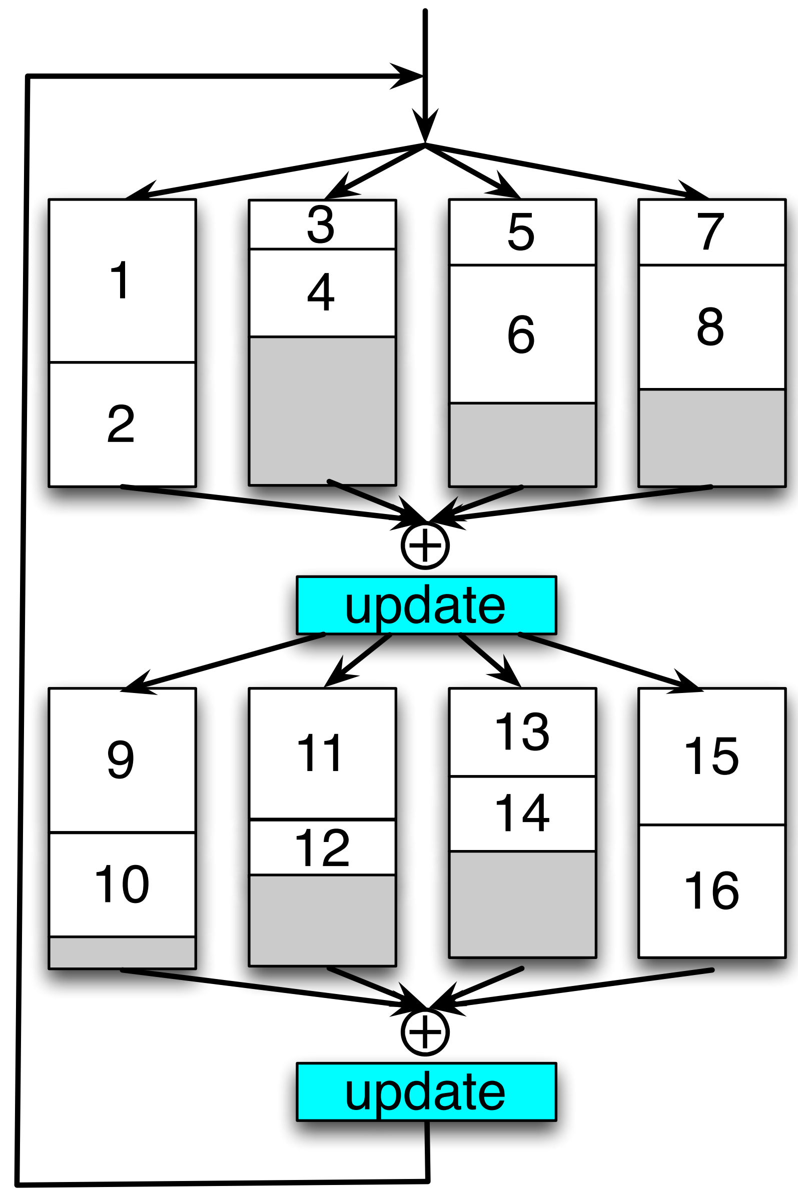
iterative parameter mixing
McDonald et al 2010

Load Balancing

- rearrange each minibatch to minimize wasted time



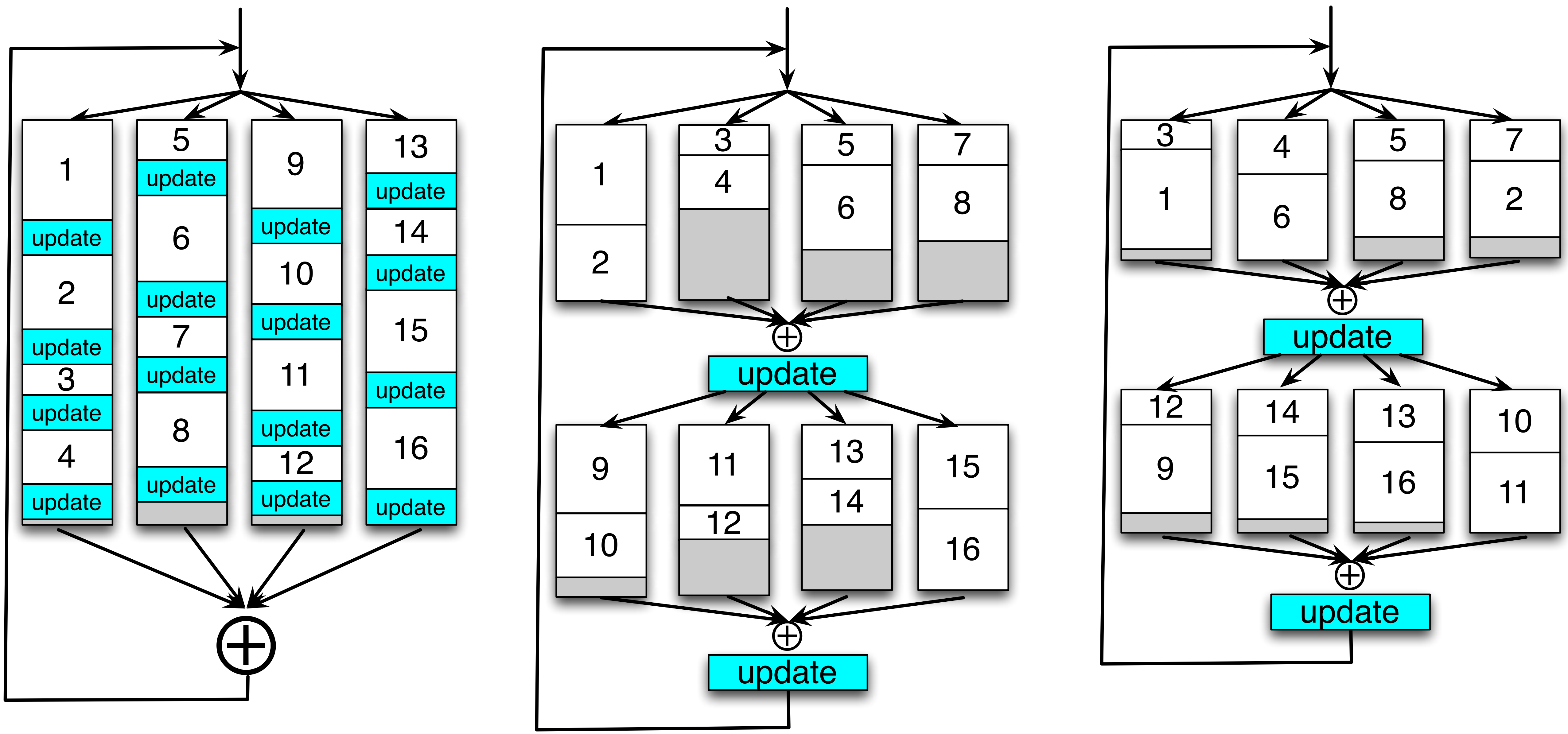
iterative parameter mixing
McDonald et al 2010



minibatch

Load Balancing

- rearrange each minibatch to minimize wasted time

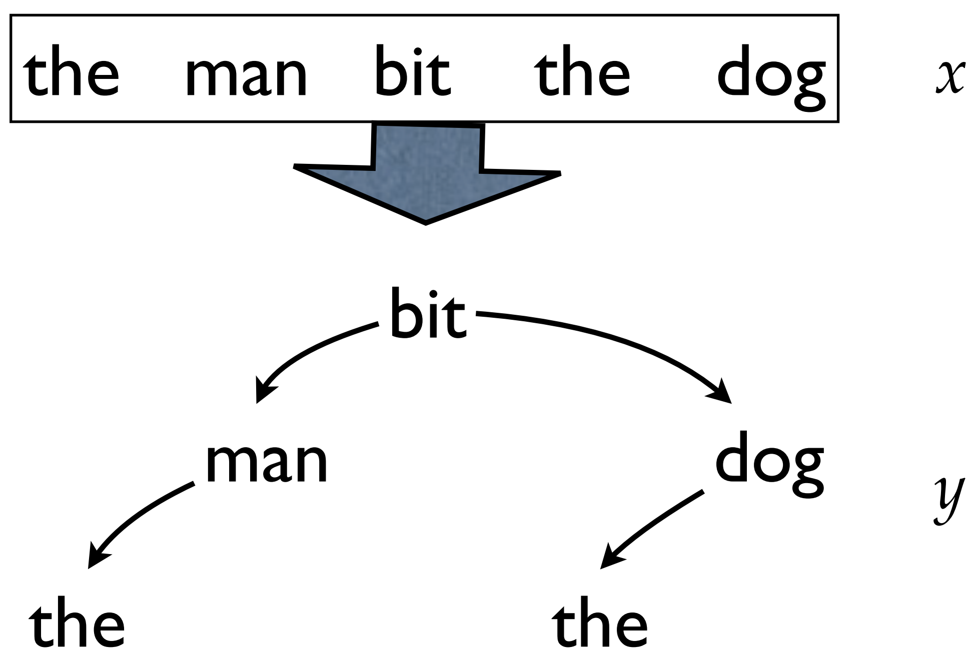


iterative parameter mixing
McDonald et al 2010

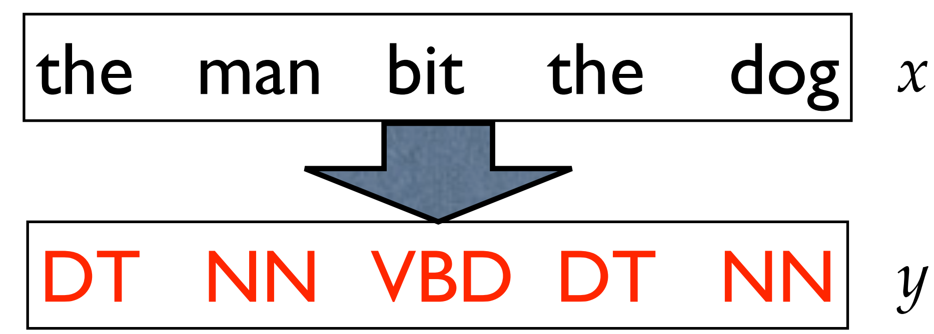
minibatch

load balanced minibatch

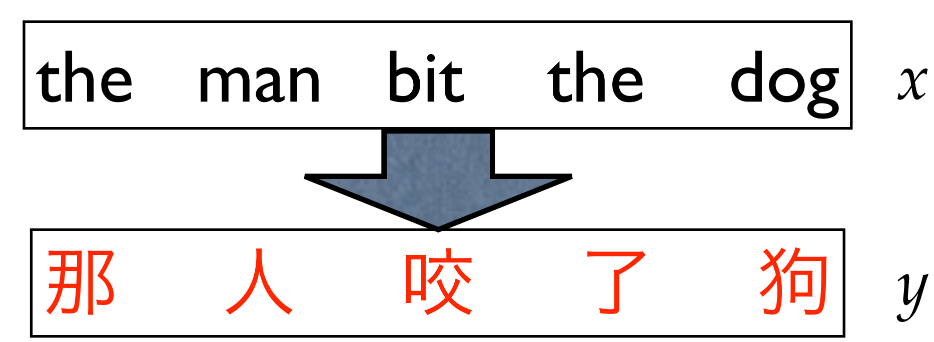
Experiments



incremental parsing



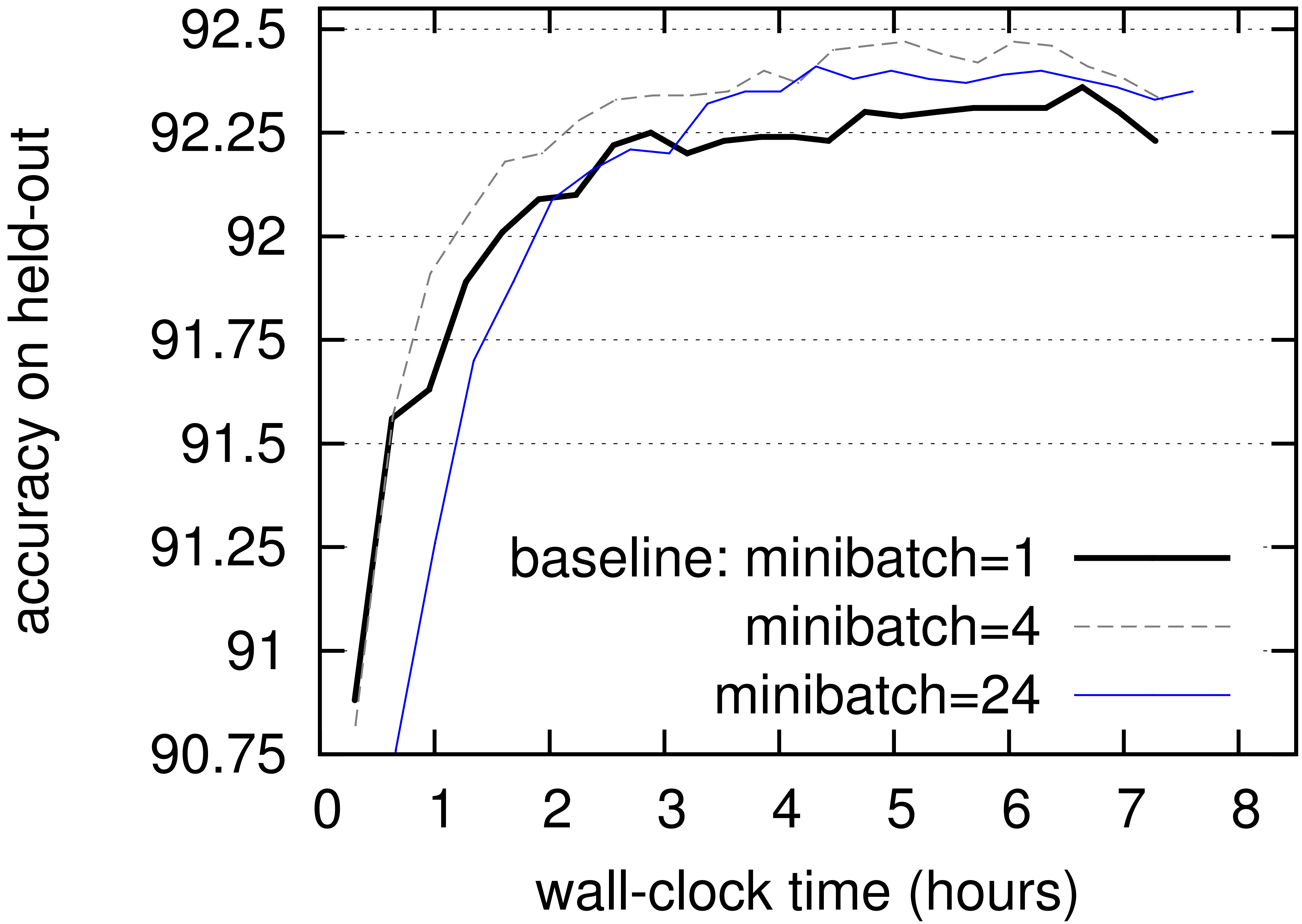
part-of-speech tagging



phrase-based translation

Experiment 1: Parsing with MIRA

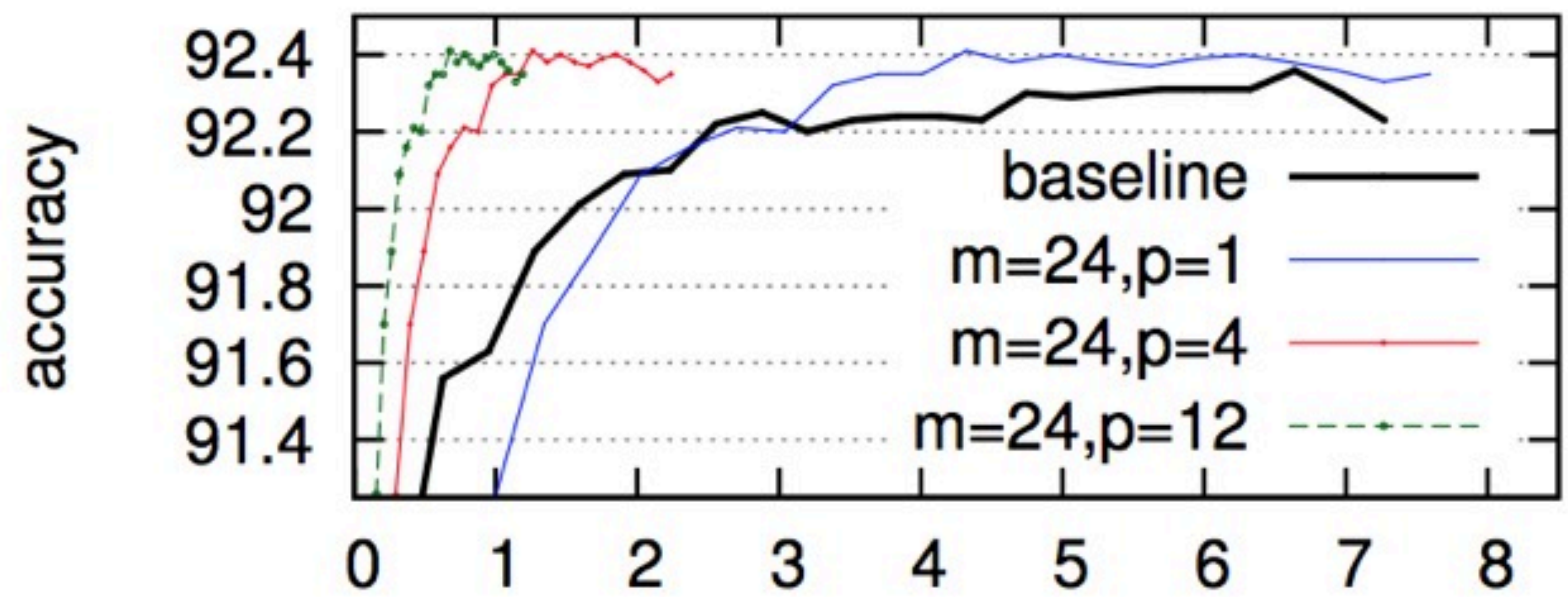
- beam search incremental parsing (Huang et al 2012) with MIRA
- minibatch learns better even in the serial setting (1 CPU)



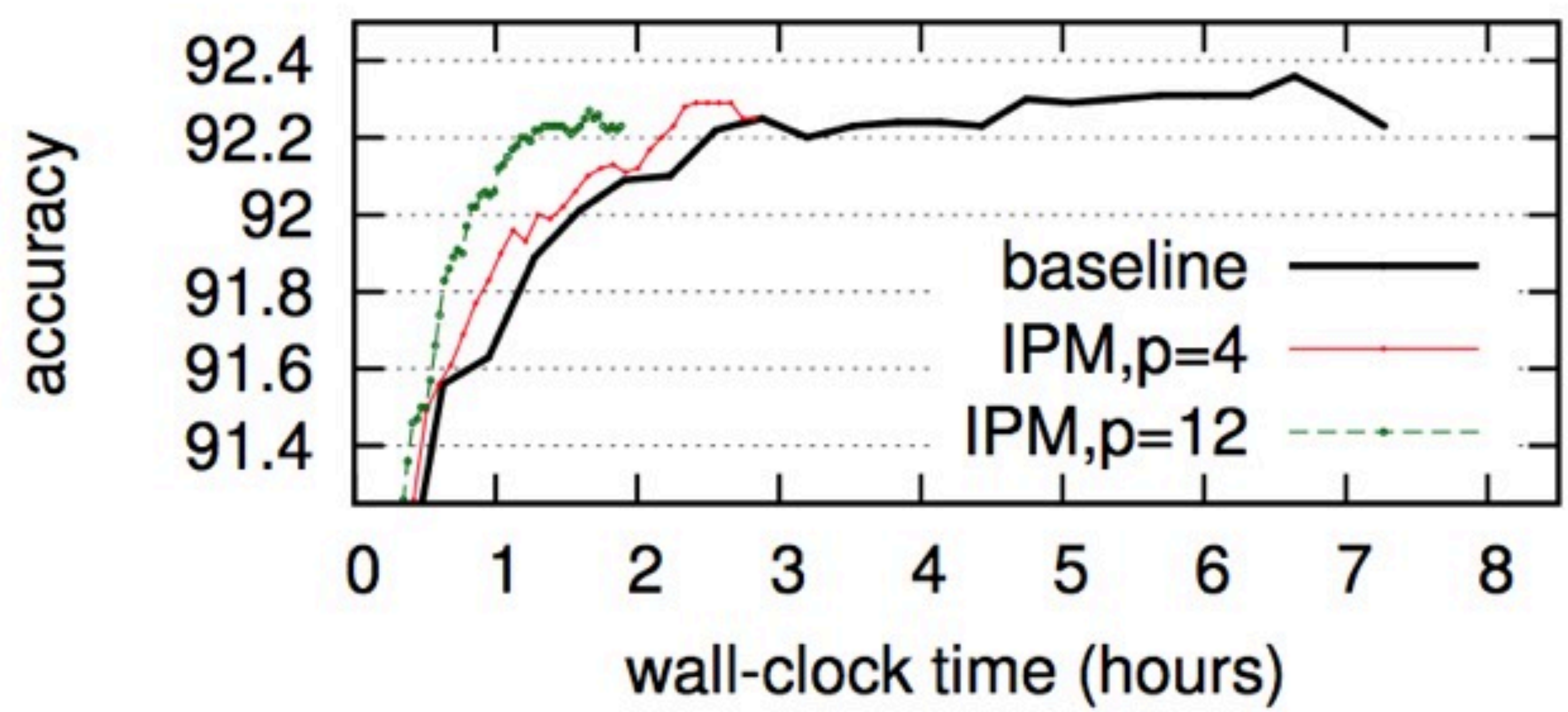
Parallelized Minibatch Faster than IPM

- minibatch is much faster than iterative parameter mixing

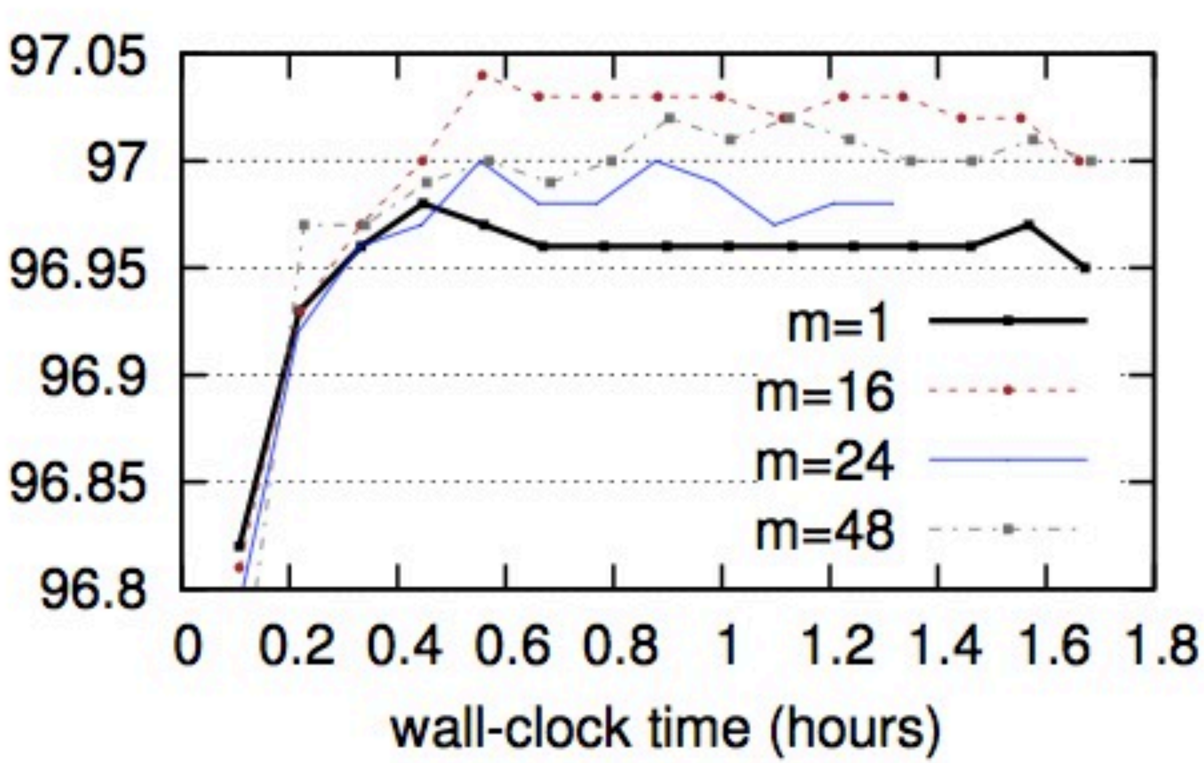
minibatch:
9x on 12 cores

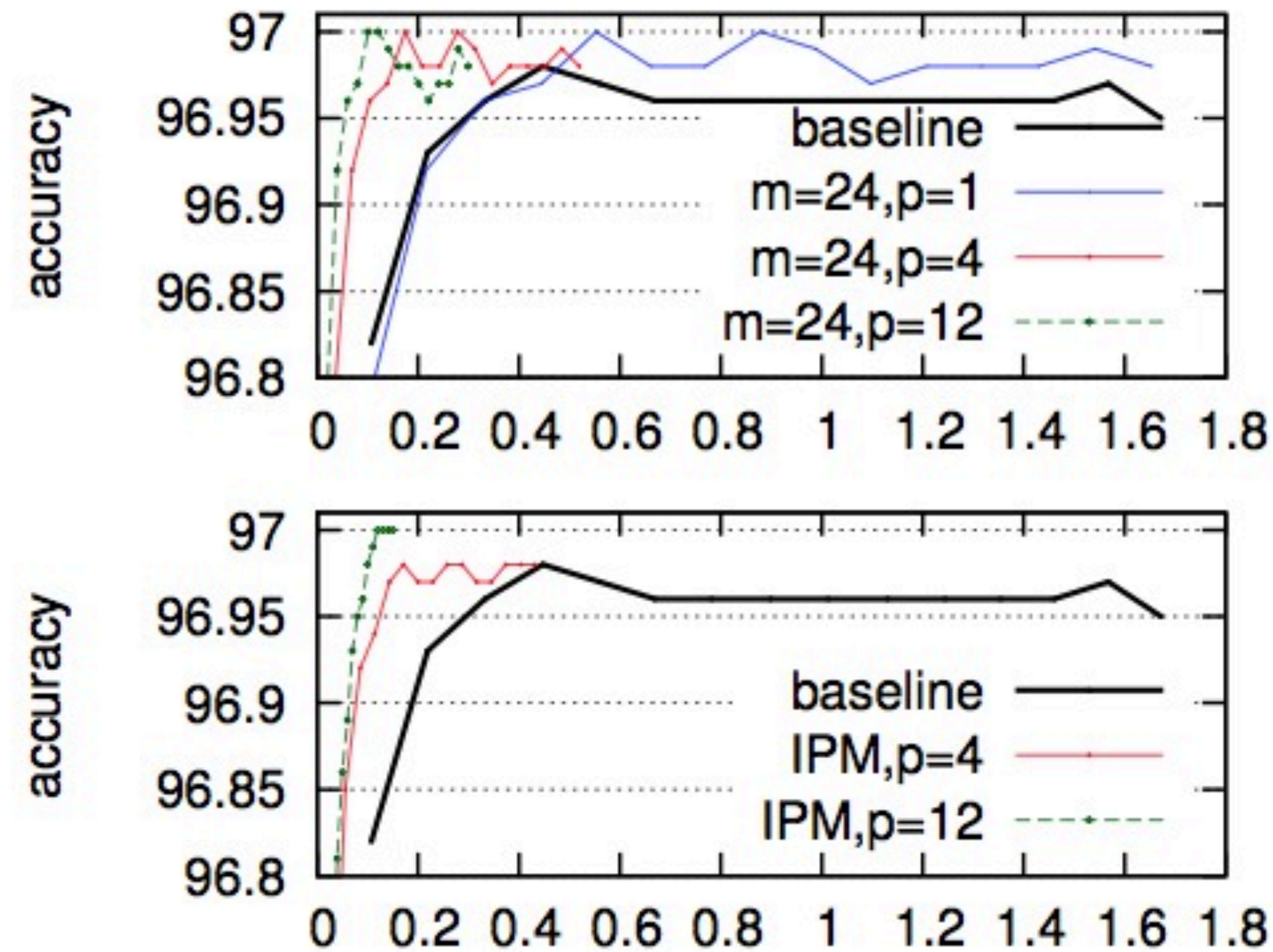


McDonald et al:
3x on 12 cores



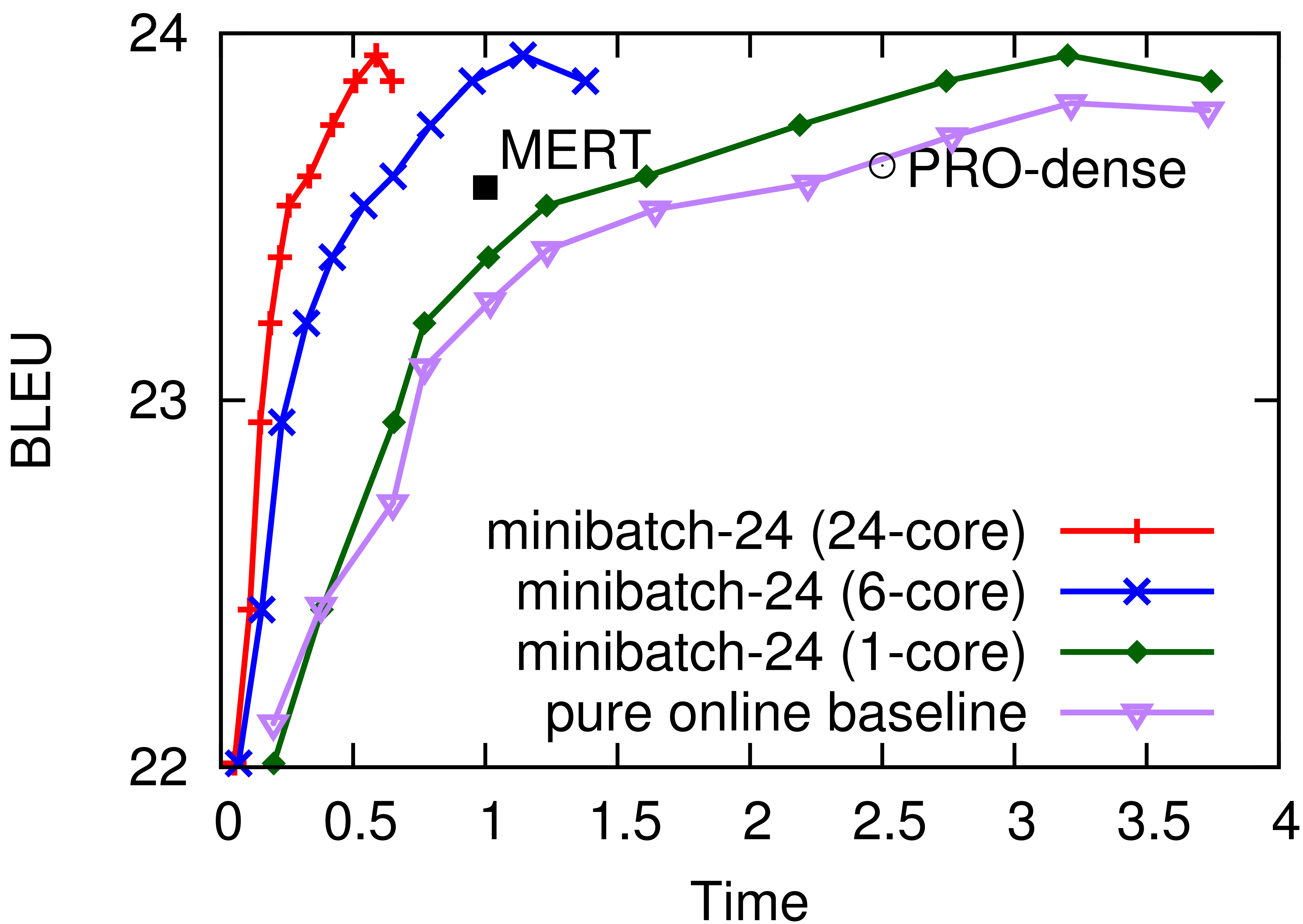
Experiment 2: Tagging (Perceptron)





Experiment 3: Machine Translation

- minibatch leads to 7x speedup on 24 cores



Roadmap of Techniques

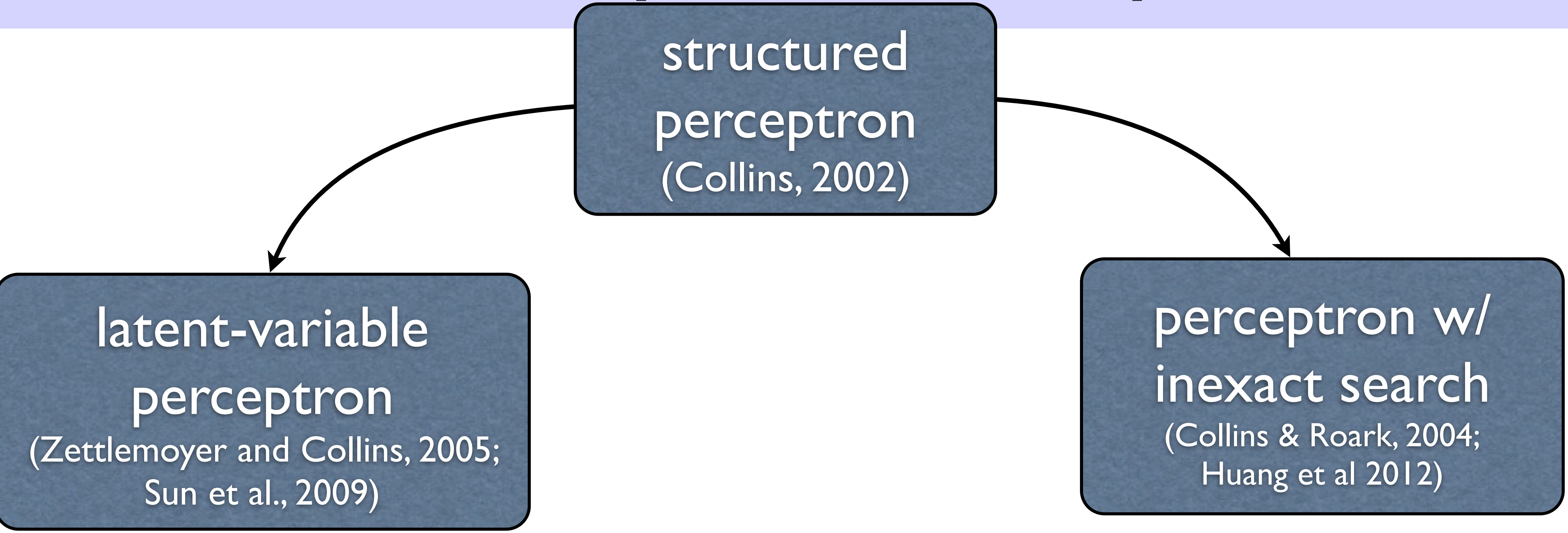
structured
perceptron
(Collins, 2002)

Roadmap of Techniques

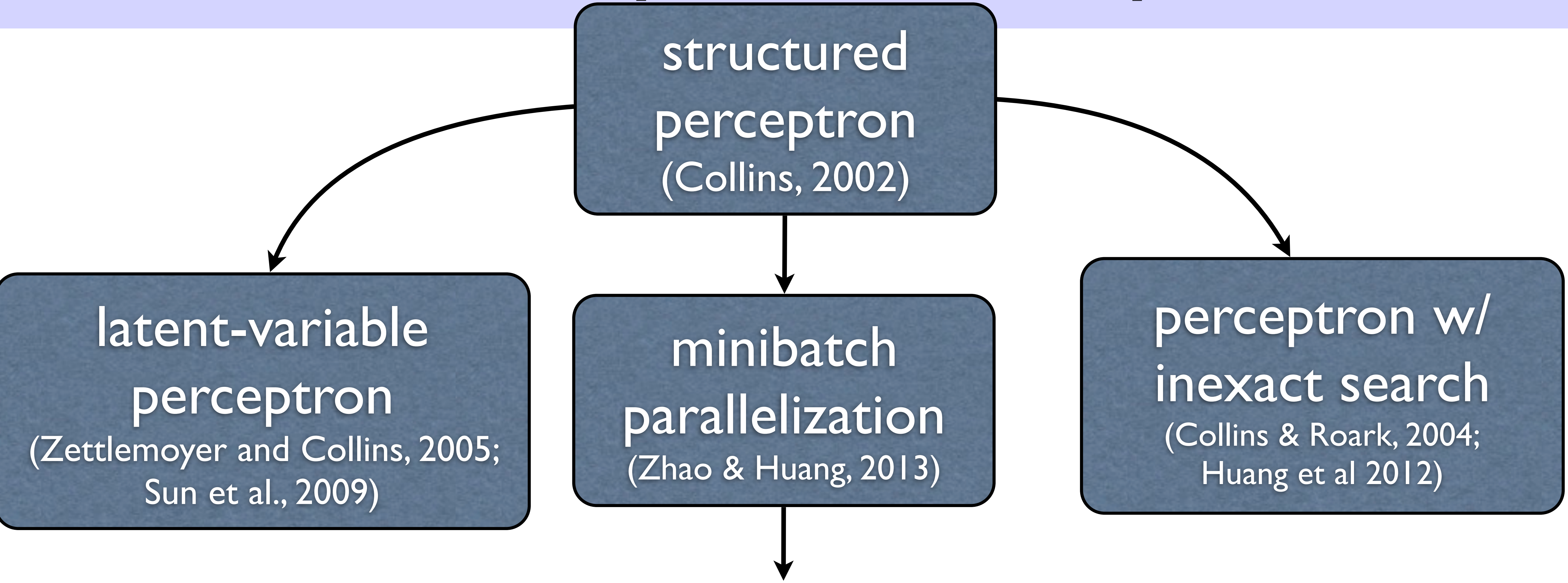
structured perceptron
(Collins, 2002)

latent-variable perceptron
(Zettlemoyer and Collins, 2005; Sun et al., 2009)

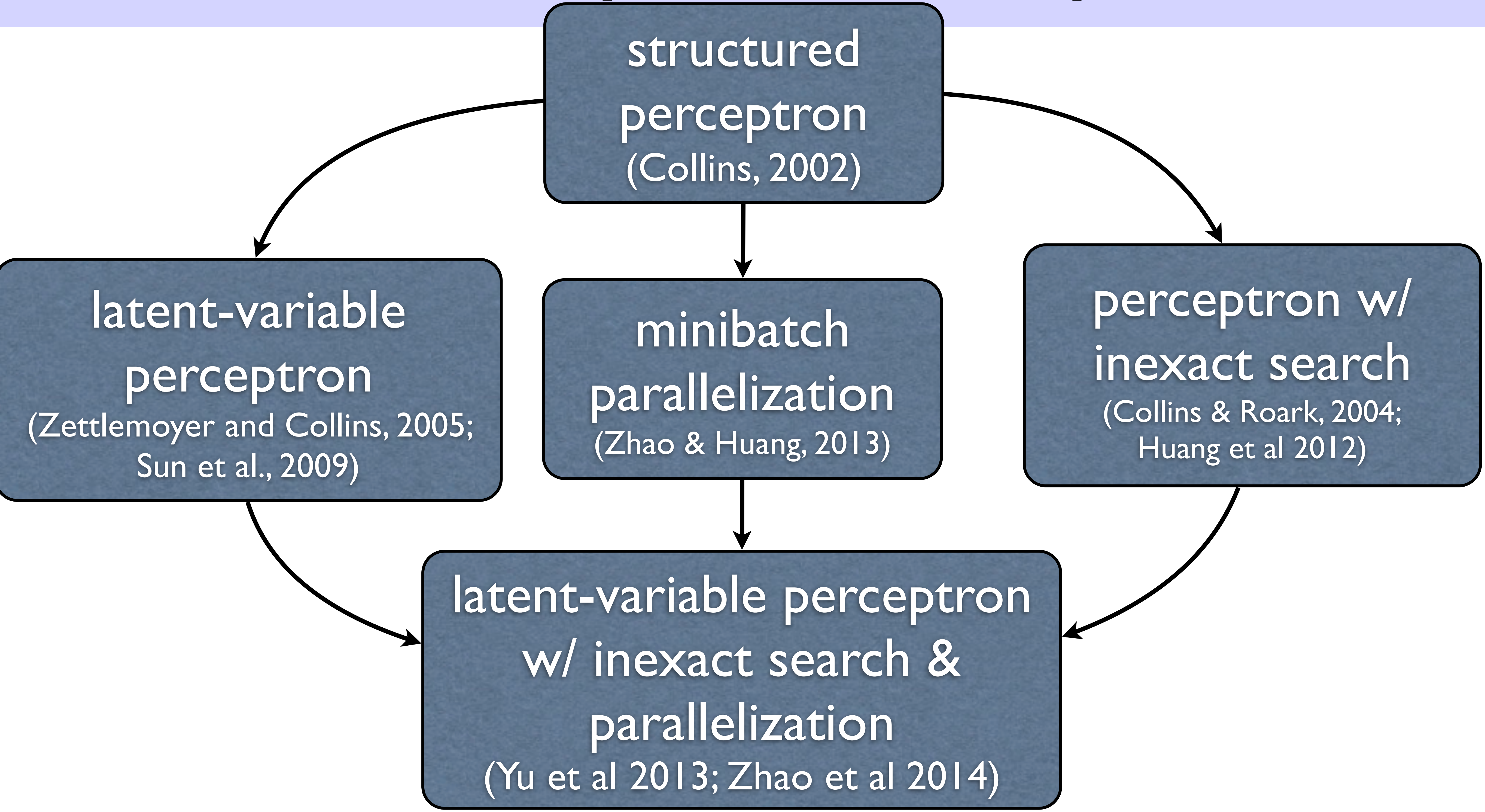
Roadmap of Techniques



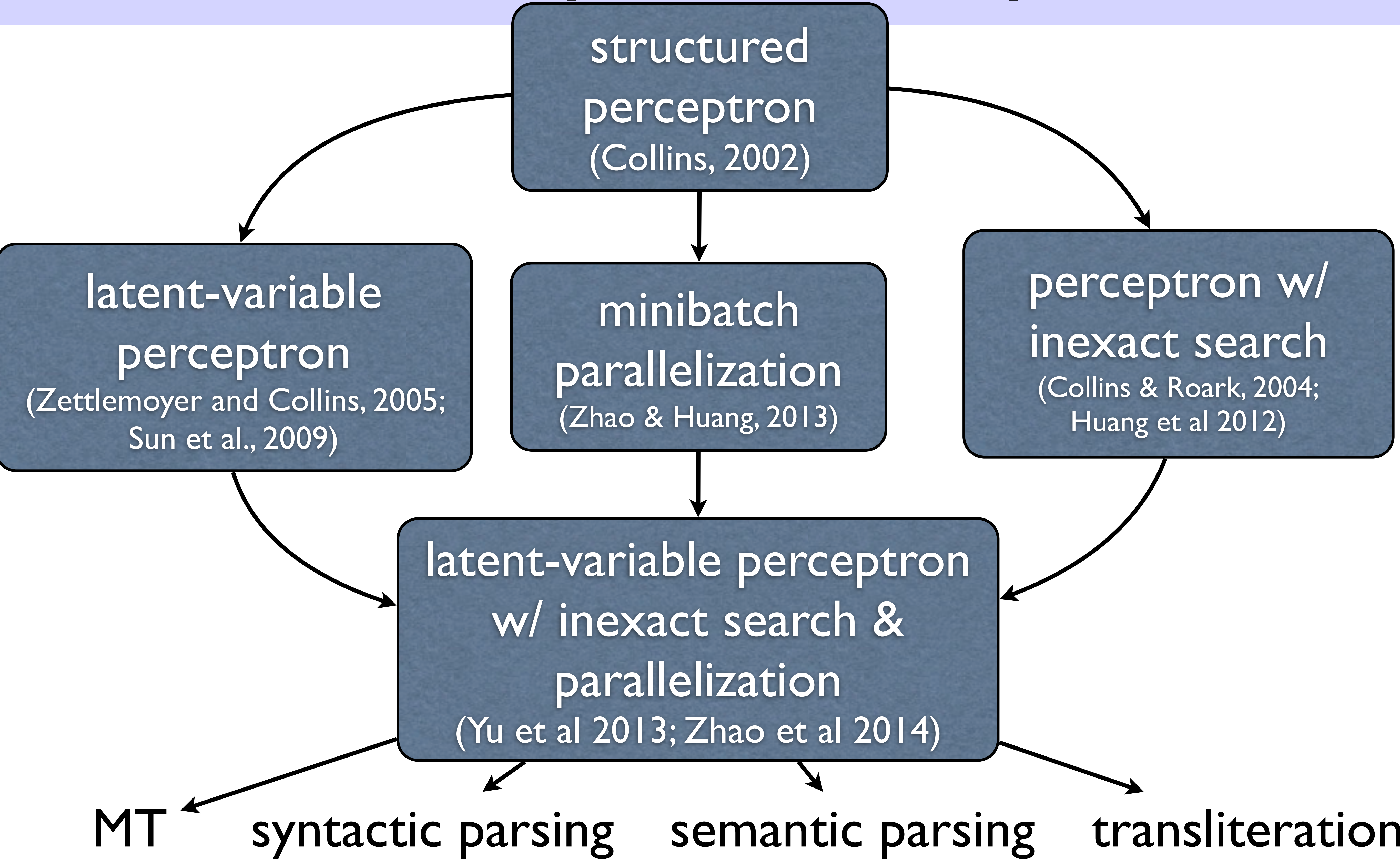
Roadmap of Techniques



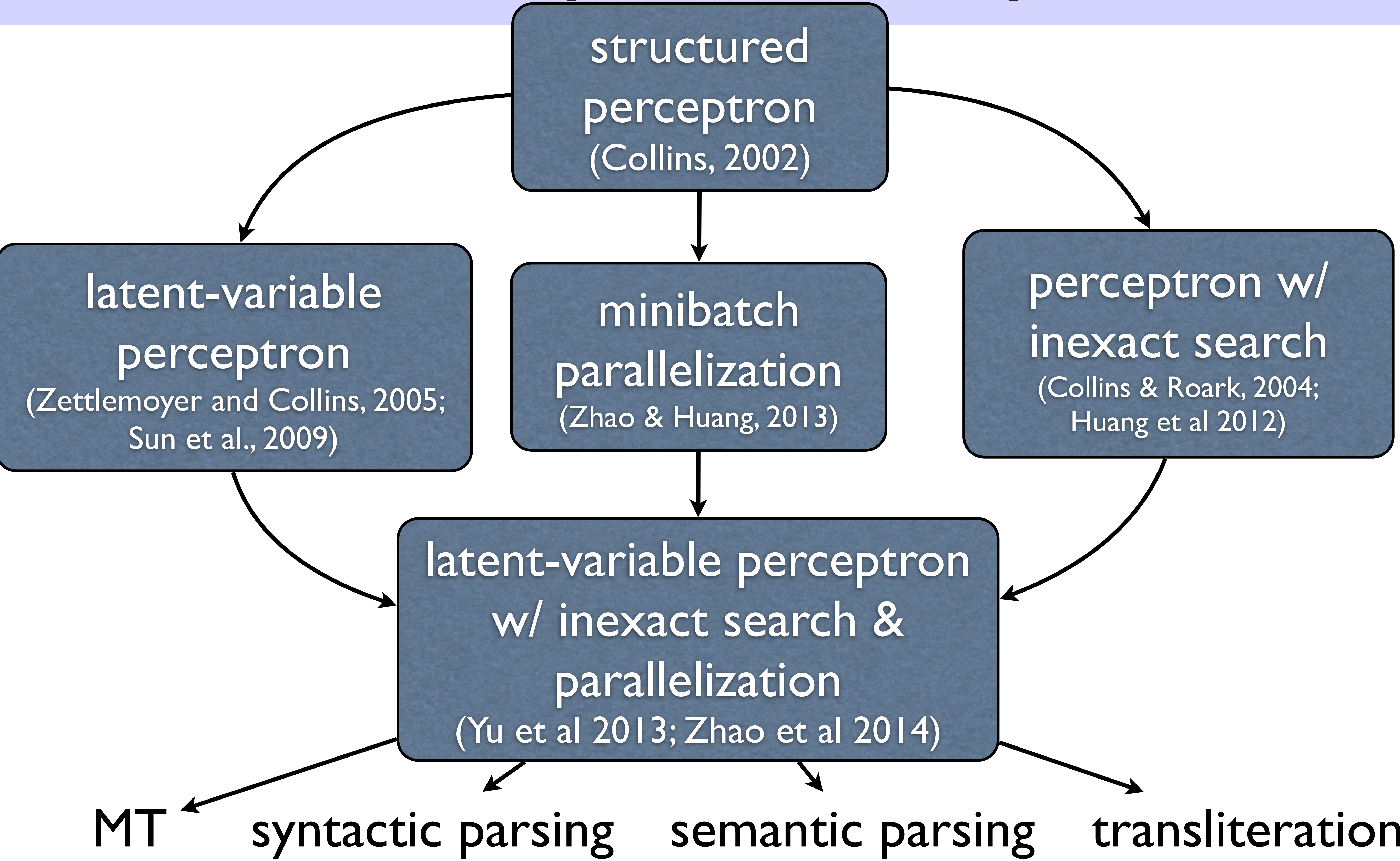
Roadmap of Techniques



Roadmap of Techniques



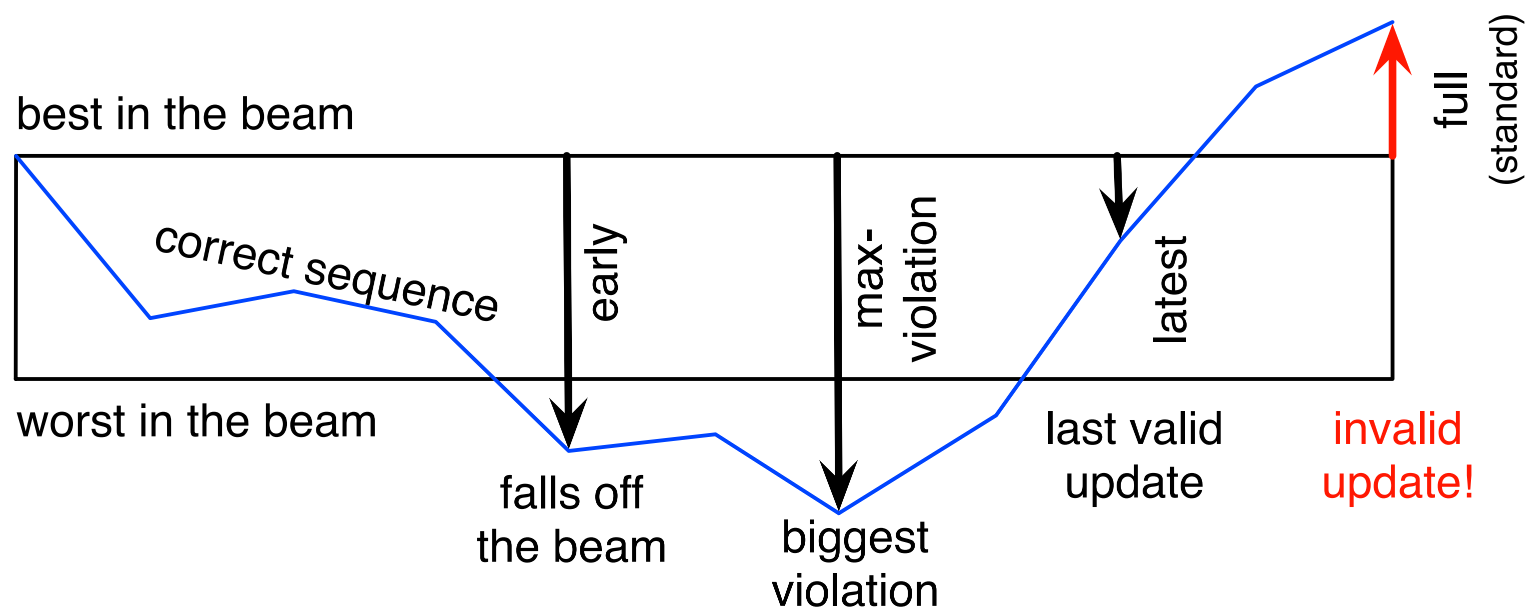
Roadmap of Techniques



(Zhao et al 2014)

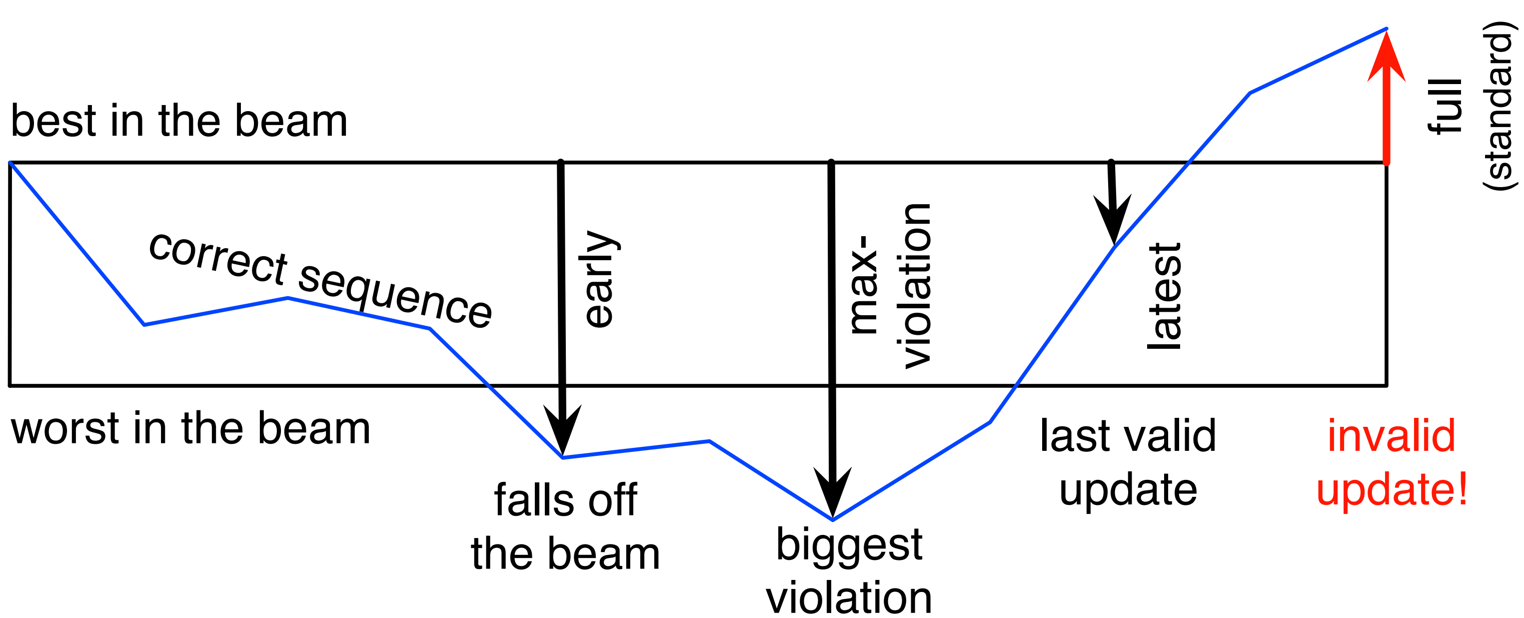
Final Conclusions

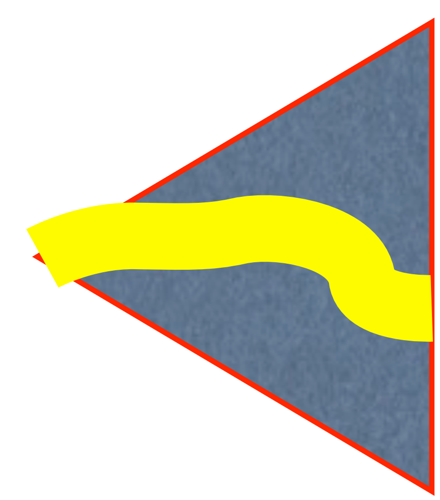
- online structured learning is simple and powerful
- search efficiency is the key challenge
- search errors do interfere with learning
  - but we can use violation-fixing perceptron w/ inexact search
- we can extend perceptron to learn latent structures
- we can parallelize online learning using minibatch



Final Conclusions

- online structured learning is simple and powerful
- search efficiency is the key challenge
- search errors do interfere with learning
  - but we can use violation-fixing perceptron w/ inexact search
- we can extend perceptron to learn latent structures
- we can parallelize online learning using minibatch





Annotated References (1)

- Binary Perceptron
  - original: Rosenblatt, 1959
  - convergence proof: Novikoff, 1962
- Multiclass Perceptron (and voted/average perceptron)
  - Freund and Schapire, 1999
- Structured Perceptron (and inexact search extensions)
  - original: Collins, 2002 (also contains generalization bounds; proofs mostly verbatim from Freund/Schapire, 1999)
  - early-update: Collins and Roark, 2004 (but no justification)
  - max-violation: Huang et al, 2012 (also defines violation-fixing perceptron framework, where early/max-violation are instances)
  - hypergraph inexact search: Zhang et al, 2013 (CKY-style parsing)

Annotated References (2)

- Latent Variable Perceptron (and inexact search extensions)
  - semantic parsing: Zettlemoyer and Collins, 2005
  - machine translation: Liang et al, 2006
  - separability condition: Sun et al, 2009
  - inexact search: Yu et al, 2013
  - hiero w/ inexact search: Zhao et al, 2014
- MIRA
  - 1-best MIRA: Crammer and Singer, 2003
  - *k*-best MIRA: McDonald et al, 2005

Annotated References (3)

- Parallelizing Online Learning
  - iterative parameter mixing: McDonald et al, 2010
  - minibatch: Zhao and Huang, 2013
- Other References
  - CRF: Lafferty et al, 2001
  - M^3N (structured SVM): Taskar et al, 2003
  - LaSO: Daumé and Marcu, 2005
  - averaging trick: Daumé, 2006, Ph.D. thesis