Web-Mining Agents

Probabilistic Reasoning over Sequential Structures

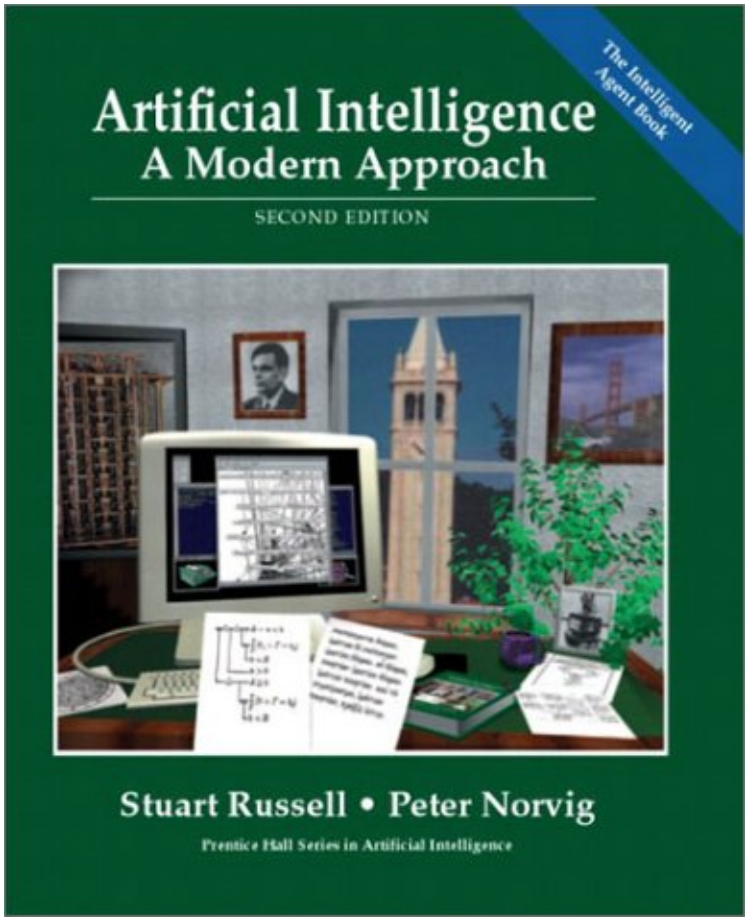
Prof. Dr. Ralf Möller

Universität zu Lübeck

Institut für Informationssysteme

Tanya Braun (Übungen)

Literature

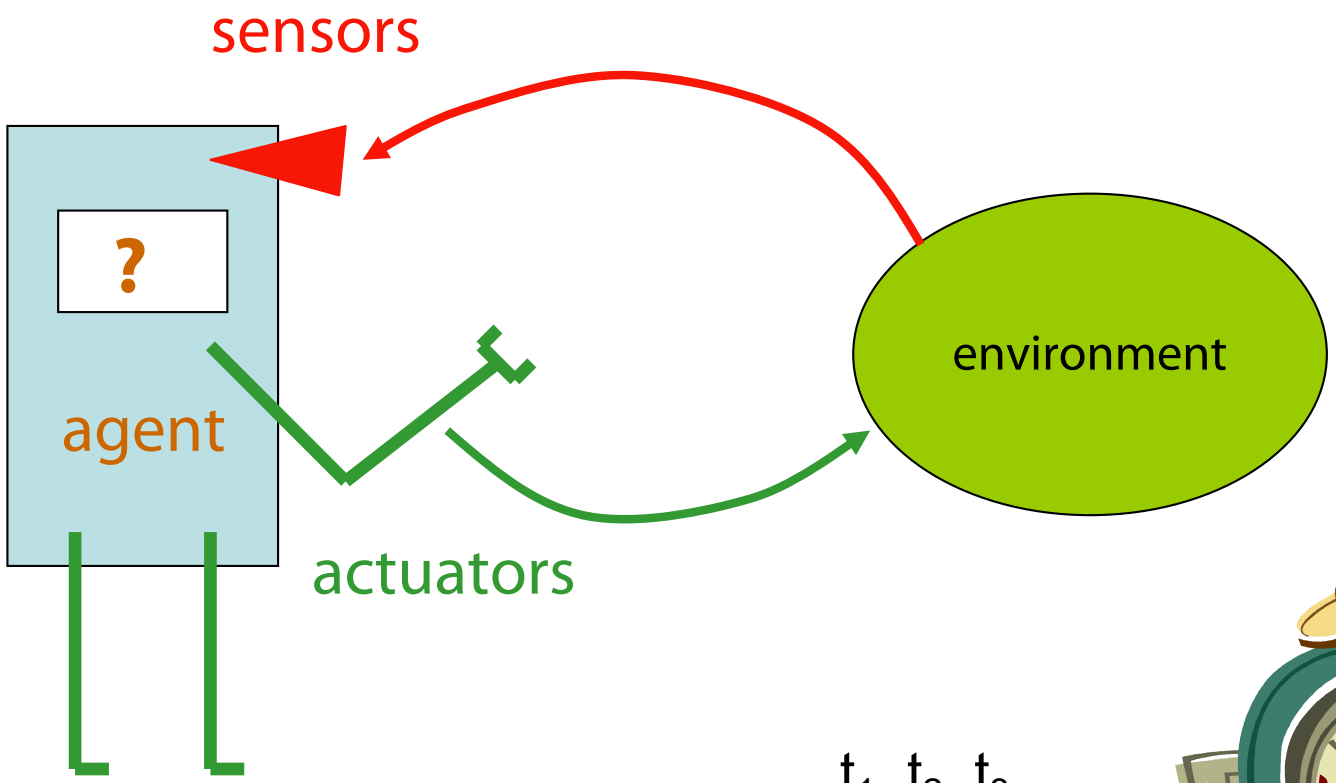


http://aima.cs.berkeley.edu

Structures (ordinal, interval, ratio scale)

- Temporal structures
- Sentence structures
- …
- Genomic structures
- …

Temporal Probabilistic Agent



$t_1, t_2, t_3, \ldots$

Probabilistic Temporal Models

- Dynamic Bayesian Networks (DBNs)
- Hidden Markov Models (HMMs)
- Kalman Filters

Time and Uncertainty

- The world changes, we need to track and predict it
- Examples: diabetes management, traffic monitoring
- Basic idea: copy state and evidence variables for each time step
- $\mathbf{X}_t$ – set of unobservable state variables at time t
  - e.g., BloodSugar_t, StomachContents_t
- $\mathbf{E}_t$ – set of evidence variables at time t
  - e.g., $\text{MeasuredBloodSugar}_t$, PulseRate_t, FoodEaten_t
- Assumes discrete time steps

States and Observations

- Process of change viewed as series of snapshots, each describing the state of the world at a particular time
- Time slice involves a set of random variables indexed by t:
  - the set of unobservable state variables $\mathbf{X}_t$
  - the set of observable evidence variable $\mathbf{E}_t$
- The observation at time t is $\mathbf{E}_t = \mathbf{e}_t$ for some set of values e_t
- The notation $\mathbf{X}_{a:b}$ denotes the set of variables from $\mathbf{X}_a$ to $\mathbf{X}_b$

Dynamic Bayesian Networks

- How can we model dynamic situations with a Bayesian network?

- Example: Is it raining today?

$$X_t = \{R_t\}$$

$$E_t = \{U_t\}$$

⇨ next step: specify dependencies among the variables.

The term “dynamic” means we are modeling a dynamic system, not that the network structure changes over time.

DBN - Representation

- Problem:

  1. Necessity to specify an unbounded number of conditional probability tables, one for each variable in each slice,
  2. Each one might involve an unbounded number of parents.

- Solution:

  1. Assume that changes in the world state are caused by a stationary process (unmoving process over time).

$$P(U_t / Parent(U_t))$$ is the same for all t

Generalization of DBNs

- Time is just one sequential structures
- Can generalize to any dynamic structure "expansion"
  - Sentences
  - Spatial structures
  - …

DBN - Representation

- Solution cont.:

  2. Use **Markov assumption** - The current state depends on only in a finite history of previous states.

     Using the first-order Markov process:

$$P(X_t / X_{0:t-1}) = P(X_t / X_{t-1})$$

Transition Model

In addition to restricting the parents of the state variable *Xt*, we must restrict the parents of the evidence variable *Et*

$$P(E_t / X_{0:t}, E_{0:t-1}) = P(E_t / X_t)$$

Sensor Model

Stationary Process/Markov Assumption

- Markov Assumption: X_t depends on some previous X_is
- First-order Markov process:

  $P(X_t|X_{0:t-1}) = P(X_t|X_{t-1})$

  - kth order: depends on previous k time steps
- Sensor Markov assumption:

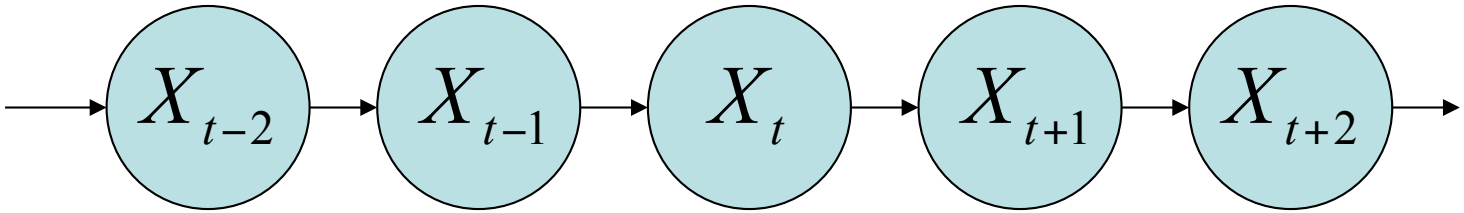
  $P(E_t|X_{0:t}, E_{0:t-1}) = P(E_t|X_t)$

- Assume stationary process: transition model:
  - $P(X_t|X_{t-1})$ and sensor model $P(E_t|X_t)$ are the same for all t
  - Changes in the world state governed by laws not changing over time

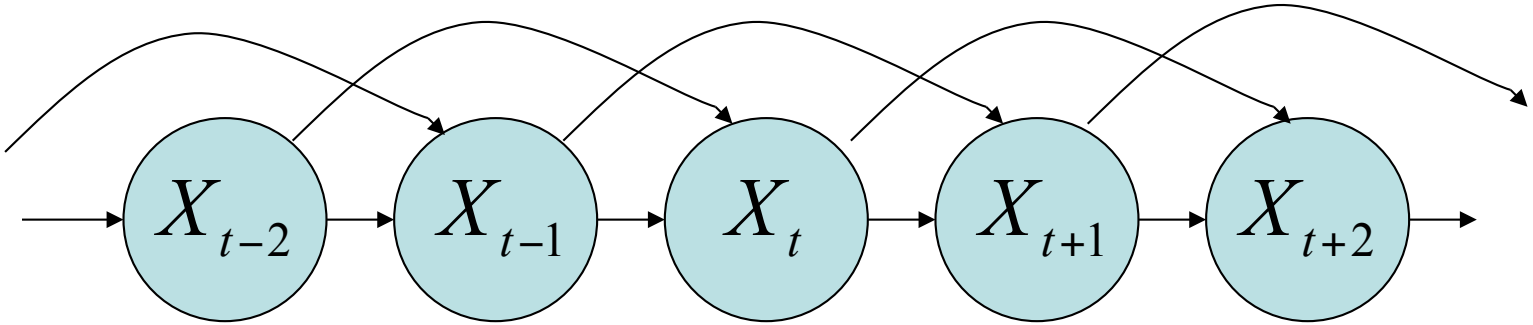
Dynamic Bayesian Networks

- There are two possible fixes if the approximation is too inaccurate:
  - Increasing the order of the Markov process model. For example, adding $Rain_{t-2}$ as a parent of $Rain_t$, which might give slightly more accurate predictions.
  - Increasing the set of state variables. For example, adding $Season_t$ to allow to incorporate historical records of rainy seasons, or adding $Temperature_t$, $Humidity_t$ and $Pressure_t$ to allow to use a physical model of rainy conditions.

Dynamic Bayesian Network



Bayesian network structure corresponding to a first-order of Markov process with state defined by the variables Xt.



A second order of Markov process

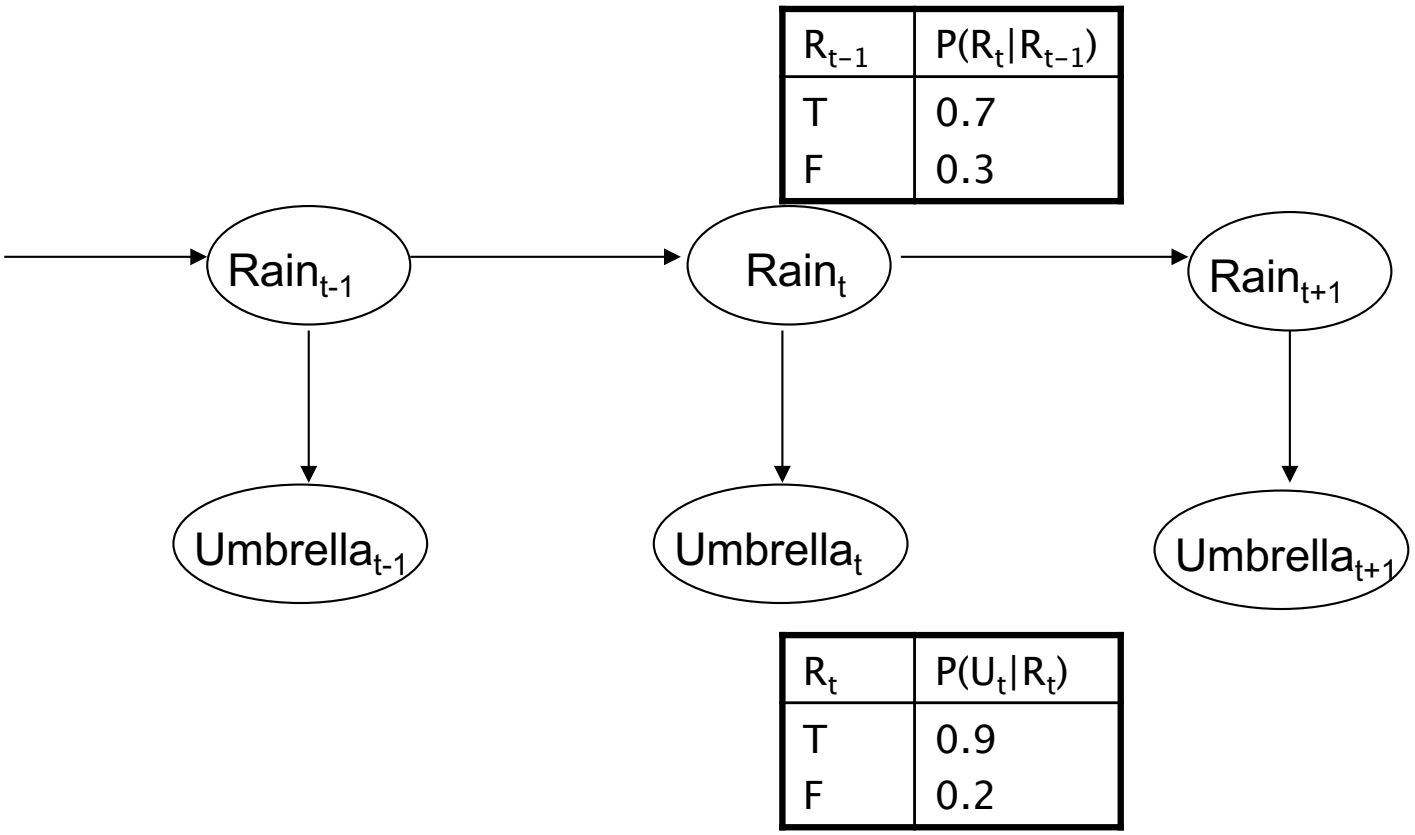
Complete Joint Distribution

- Given:
  - Transition model: $P(X_t|X_{t-1})$
  - Sensor model: $P(E_t|X_t)$
  - Prior probability: $P(X_0)$

- Then we can specify complete joint distribution:

$$P(X_0, X_1, \ldots, X_t, E_1, \ldots, E_t) = P(X_0) \prod_{i=1}^{t} P(X_i \mid X_{i-1}) P(E_i \mid X_i)$$

Example


R_{t-1} | $P(R_t|R_{t-1})$
T | 0.7
F | 0.3
$Rain_{t-1}$
$Rain_t$
$Rain_{t+1}$
$Umbrella_{t-1}$
$Umbrella_t$
$Umbrella_{t+1}$
R_t | $P(U_t|R_t)$
T | 0.9
F | 0.2

Inference Tasks

- **Filtering:** What is the probability that it is raining today, given all the umbrella observations up through today?
- **Prediction:** What is the probability that it will rain the day after tomorrow, given all the umbrella observations up through today?
- **Smoothing:** What is the probability that it rained yesterday, given all the umbrella observations through today?
- **Most likely explanation / most probable explanation:** if the umbrella appeared the first three days but not on the fourth, what is the most likely weather sequence to produce these umbrella sightings?

DBN – Basic Inference

- Filtering or Monitoring:

Compute the belief state - the posterior distribution over the *current* state, given all evidence to date.

$$P(X_t / e_{1:t})$$

Filtering is what a rational agent needs to do in order to keep track of the current state so that the rational decisions can be made.

DBN – Basic Inference

- Filtering cont.

Given the results of filtering up to time *t*, one can easily compute the result for *t+1* from the new evidence e_{t+1}

$$P(X_{t+1} / e_{1:t+1}) = f(e_{t+1}, P(X_t / e_{1:t+1}))$$ (for some function *f*)

$$= P(X_{t+1} / e_{1:t}, e_{t+1})$$ (dividing up the evidence)

$$= \alpha P(e_{t+1} / X_{t+1}, e_{1:t}) P(X_{t+1} / e_{1:t})$$ (using Bayes' Theorem)

$$= \alpha P(e_{t+1} / X_{t+1}) P(X_{t+1} / e_{1:t})$$ (by the Markov property of evidence)

α is a normalizing constant used to make probabilities sum up to 1.

DBN – Basic Inference

- Filtering cont.

The second term $P(X_{t+1} / e_{1:t})$ represents a one-step prediction of the next step, and the first term $P(e_{t+1} / X_{t+1})$ updates this with the new evidence.

Now we obtain the one-step prediction for the next step by conditioning on the current state Xt:

$$P(X_{t+1} / e_{1:t+1}) = \alpha P(e_{t+1} / X_{t+1}) \sum_{X_t} P(X_{t+1} / x_t, e_{1:t}) P(x_t / e_{1:t})$$

$$= \alpha P(e_{t+1} / X_{t+1}) \sum_{X_t} P(X_{t+1} / \boldsymbol{x_t}) P(x_t / e_{1:t})$$

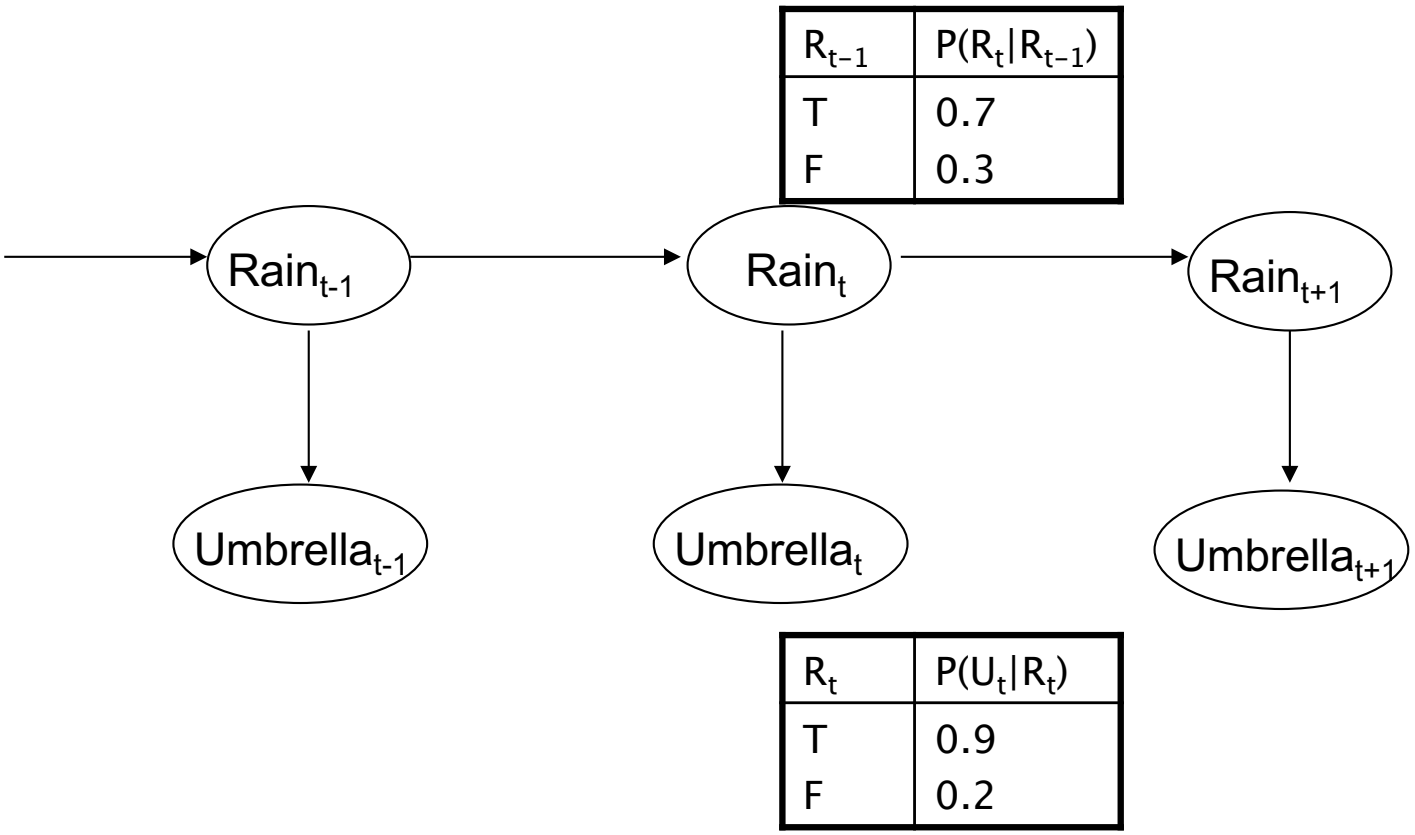
(using the Markov property)

Forward Messages

$\mathbf{f}_{1:t+1} = \textsc{Forward}(\mathbf{f}_{1:t}, \mathbf{e}_{t+1})$ where $\mathbf{f}_{1:t} = \mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$

Time and space **constant** (independent of t)

Example


R_{t-1} | $P(R_t|R_{t-1})$
T | 0.7
F | 0.3
$Rain_{t-1}$
$Rain_t$
$Rain_{t+1}$
$Umbrella_{t-1}$
$Umbrella_t$
$Umbrella_{t+1}$
R_t | $P(U_t|R_t)$
T | 0.9
F | 0.2

DBN – Basic Inference

Illustration for two steps in the Umbrella example:

- On day 1, the umbrella appears so U1=true. The prediction from t=0 to t=1 is

$$P(R_1) = \sum_{r_0} P(R_1 / r_0) P(r_0)$$

and updating it with the evidence for t=1 gives

$$P(R_1 / u_1) = \alpha P(u_1 / R_1) P(R_1)$$

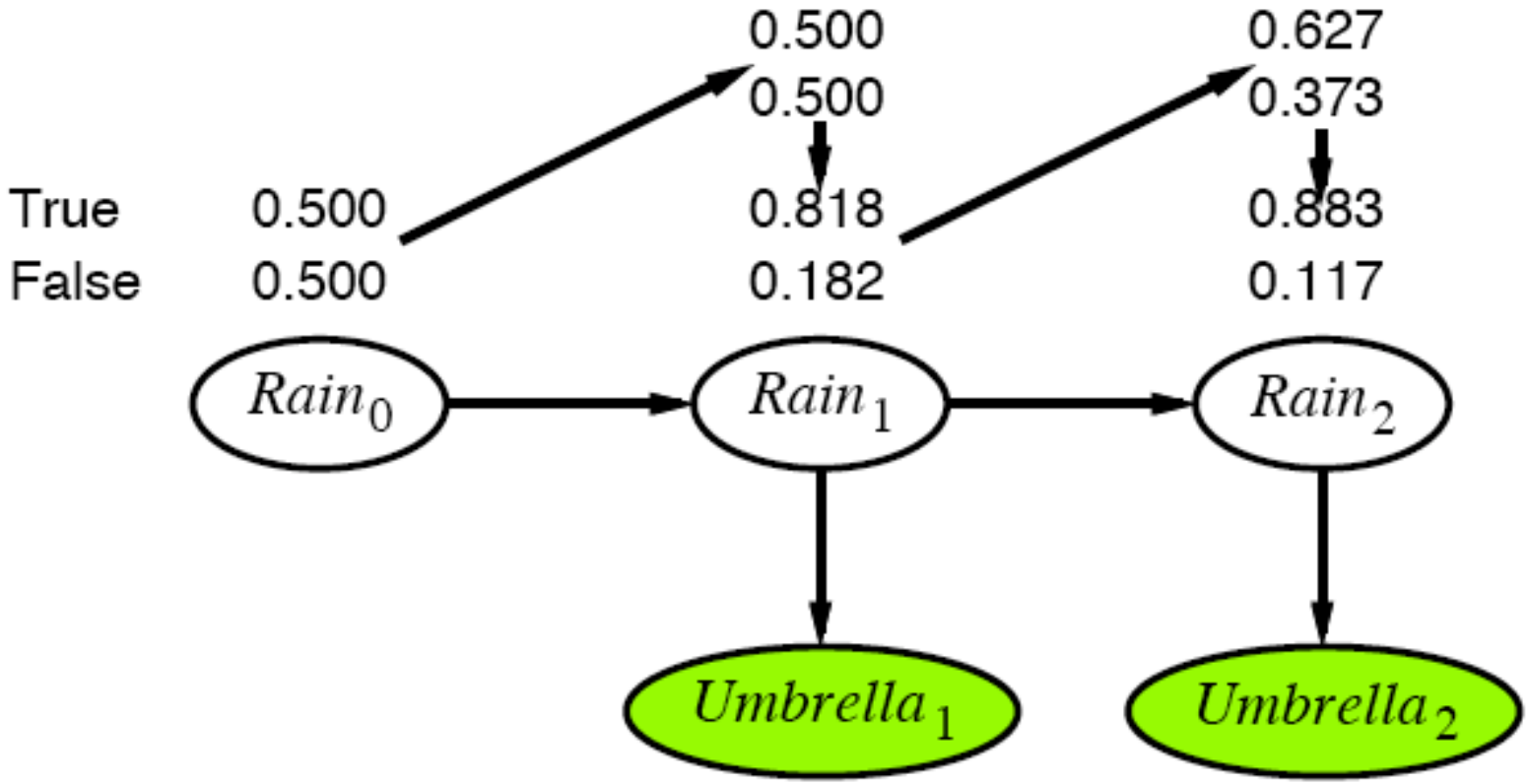
- On day 2, the umbrella appears so U2=true. The prediction from t=1 to t=2 is

$$P(R_2 / u_1) = \sum_{r_1} P(R_2 / r_1) P(r_1 / u_1)$$

and updating it with the evidence for t=2 gives

$$P(R_2 / u_1, u_2) = \alpha P(u_2 / R_2) P(R_2 / u_1)$$

Example cntd.



DBN – Basic Inference

- Prediction:

  Compute the posterior distribution over the *future* state, given all evidence to date.

$$P(X_{t+k} / e_{1:t})$$

for some k>0

  The task of prediction can be seen simply as filtering without the addition of new evidence.

DBN – Basic Inference

- Smoothing or hindsight:

  Compute the posterior distribution over the *past* state, given all evidence up to the present.

$$P(X_k / e_{1:t})$$ for some k such that 0 ≤ k < t.

Hindsight provides a better estimate of the state than was available at the time, because it incorporates more evidence.

Smoothing

Divide evidence $\mathbf{e}_{1:t}$ into $\mathbf{e}_{1:k}$, $\mathbf{e}_{k+1:t}$:

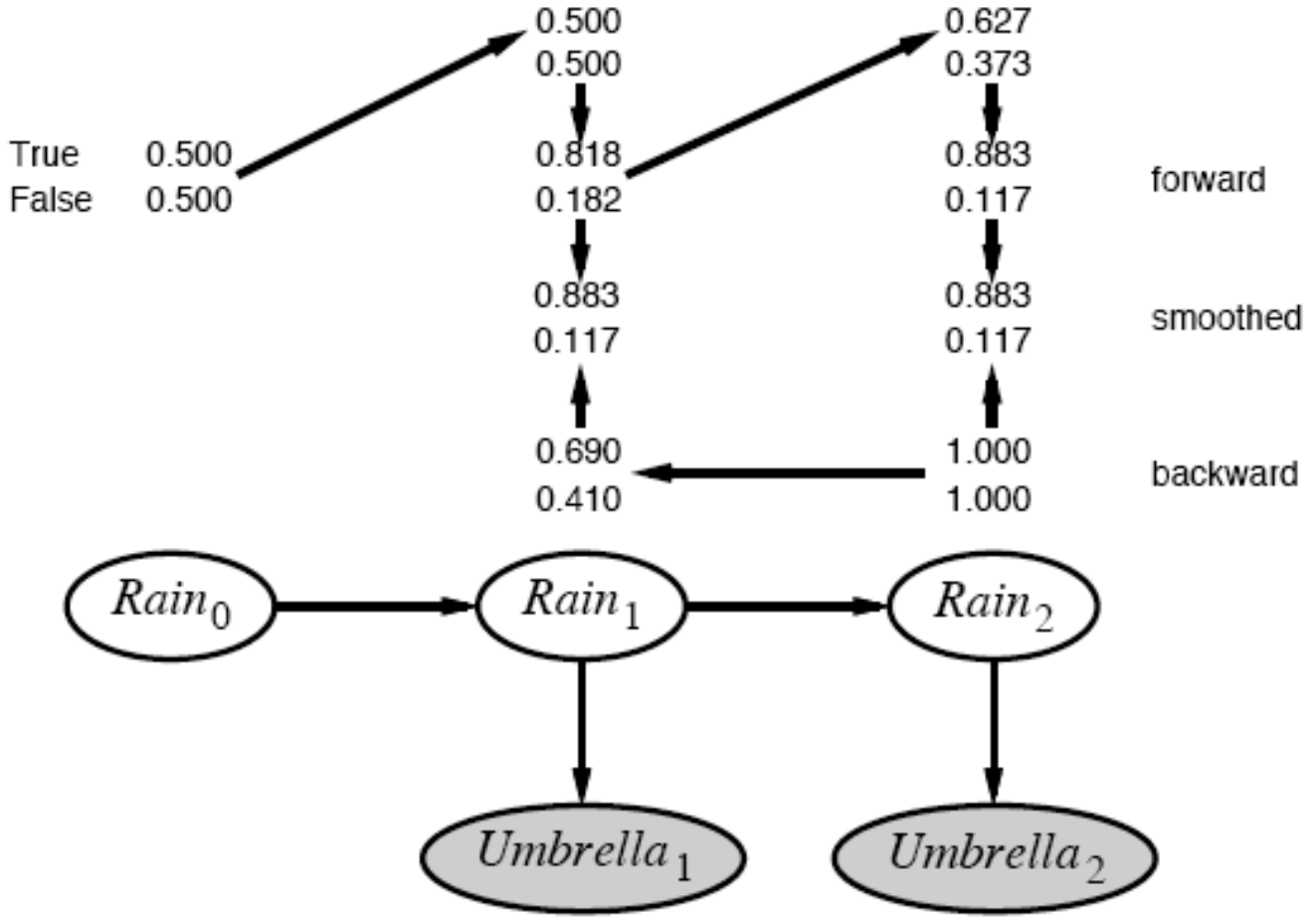
$$\begin{aligned}\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:t}) &= \mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:k},\mathbf{e}_{k+1:t}) \\ &= \alpha\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:k})\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k,\mathbf{e}_{1:k}) \\ &= \alpha\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:k})\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k) \\ &= \alpha\mathbf{f}_{1:k}\mathbf{b}_{k+1:t}\end{aligned}$$

Backward message computed by a backwards recursion:

$$\begin{aligned}\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k) &= \textstyle\sum_{\mathbf{x}_{k+1}}\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k,\mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k) \\ &= \textstyle\sum_{\mathbf{x}_{k+1}}P(\mathbf{e}_{k+1:t}|\mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k) \\ &= \textstyle\sum_{\mathbf{x}_{k+1}}P(\mathbf{e}_{k+1}|\mathbf{x}_{k+1})P(\mathbf{e}_{k+2:t}|\mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k)\end{aligned}$$

Forward–backward algorithm: cache forward messages along the way
Time linear in t (polytree inference), space $O(t|\mathbf{f}|)$

Example contd.



DBN – Basic Inference

- Most likely explanation:

Compute the sequence of states that is most likely to have generated a given sequence of observation.

$$\arg\max_{x_{1:t}} P(X_{1:t} \mid e_{1:t})$$

Algorithms for this task are useful in many applications, including, e.g., speech recognition.

Web-Mining Agents
Probabilistic Reasoning over Time

Prof. Dr. Ralf Möller
Universität zu Lübeck
Institut für Informationssysteme

Tanya Braun (Übungen)

Most-likely explanation

Most likely sequence $\neq$ sequence of most likely states!!!!

Most likely path to each $\mathbf{x}_{t+1}$
= most likely path to **some** $\mathbf{x}_t$ plus one more step

$$\max_{\mathbf{x}_1 \ldots \mathbf{x}_t} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{X}_{t+1} | \mathbf{e}_{1:t+1})$$
$$= \mathbf{P}(\mathbf{e}_{t+1} | \mathbf{X}_{t+1}) \max_{\mathbf{x}_t} \left(\mathbf{P}(\mathbf{X}_{t+1} | \mathbf{x}_t) \max_{\mathbf{x}_1 \ldots \mathbf{x}_{t-1}} P(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t | \mathbf{e}_{1:t}) \right)$$

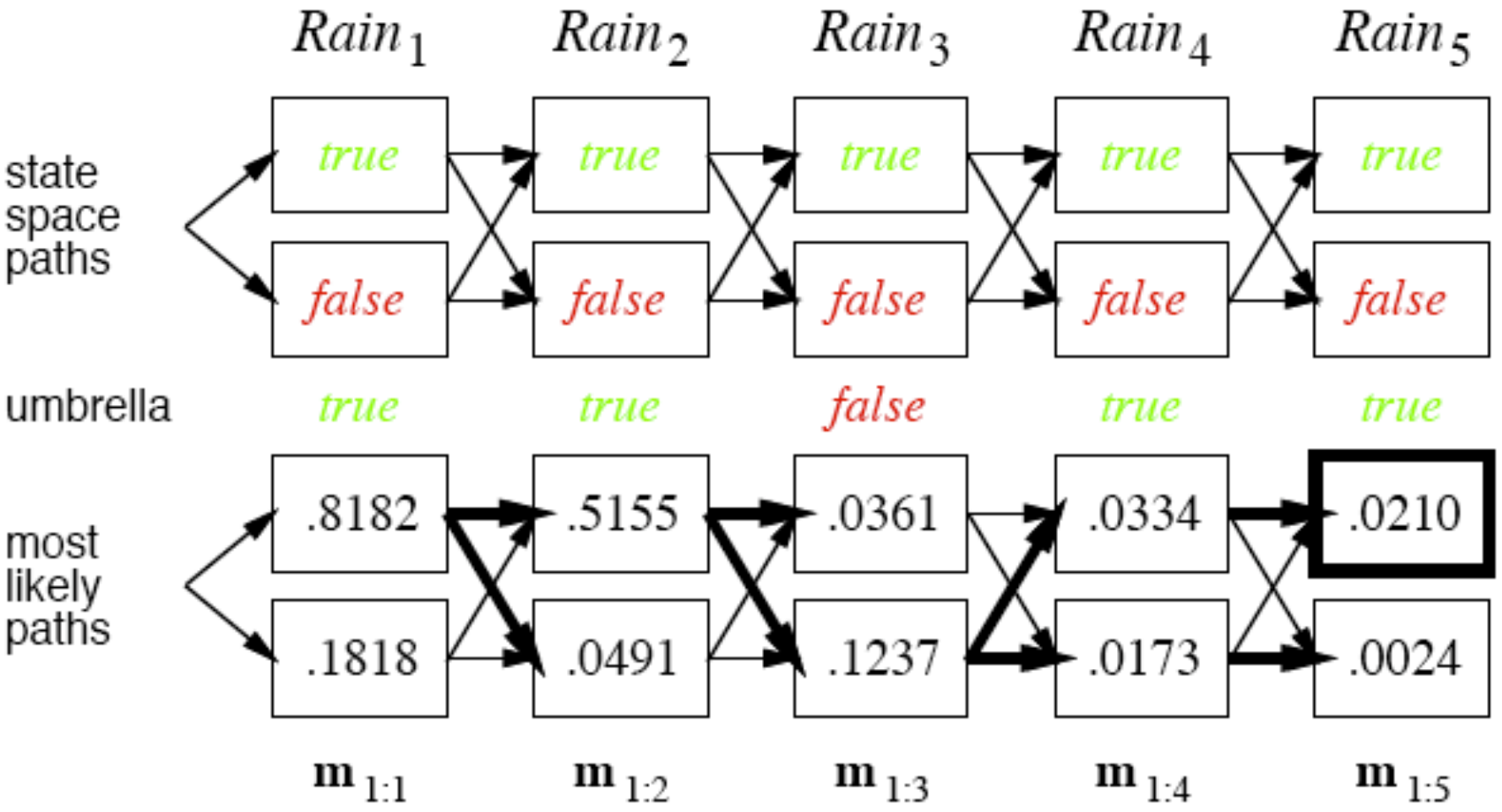
Identical to filtering, except $\mathbf{f}_{1:t}$ replaced by

$$\mathbf{m}_{1:t} = \max_{\mathbf{x}_1 \ldots \mathbf{x}_{t-1}} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{X}_t | \mathbf{e}_{1:t}),$$

I.e., $\mathbf{m}_{1:t}(i)$ gives the probability of the most likely path to state i.
Update has sum replaced by max, giving the Viterbi algorithm:

$$\mathbf{m}_{1:t+1} = \mathbf{P}(\mathbf{e}_{t+1} | \mathbf{X}_{t+1}) \max_{\mathbf{x}_t} (\mathbf{P}(\mathbf{X}_{t+1} | \mathbf{x}_t) \mathbf{m}_{1:t})$$

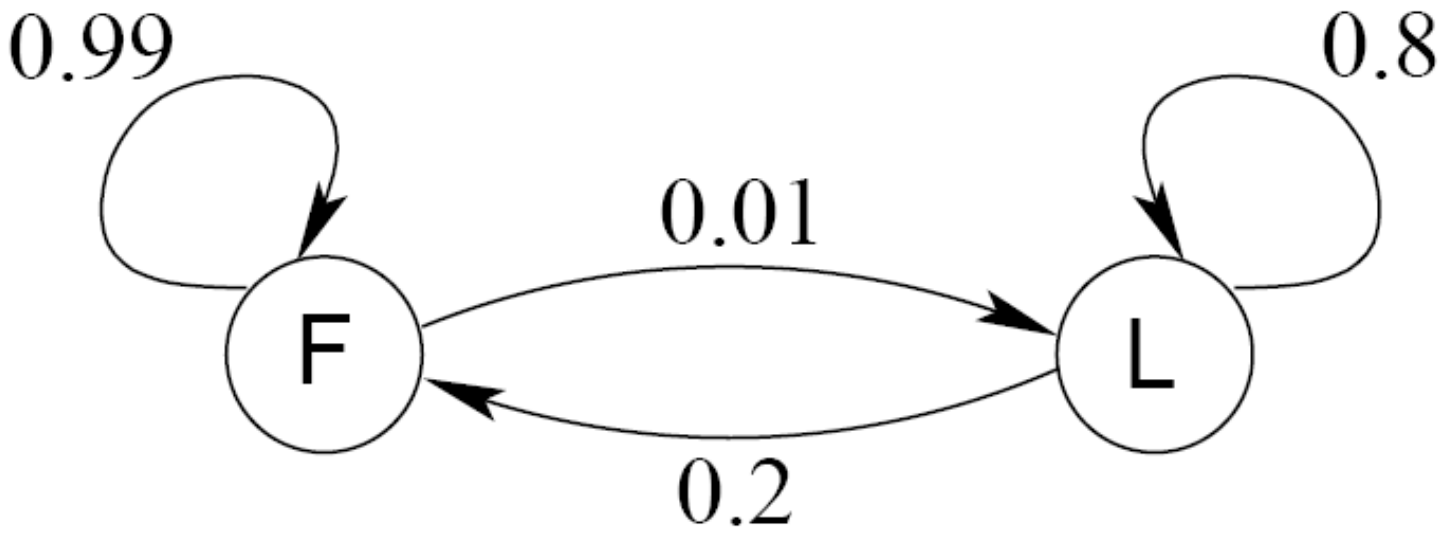
Rain/Umbrella Example



The occasionally dishonest casino

- A casino uses a fair die most of the time, but occasionally switches to a loaded one
  - Fair die: Prob(1) = Prob(2) = . . . = Prob(6) = 1/6
  - Loaded die: Prob(1) = Prob(2) = . . . = Prob(5) = 1/10, Prob(6) = ½
  - These are the ***emission*** probabilities
- ***Transition probabilities***
  - Prob(Fair → Loaded) = 0.01
  - Prob(Loaded → Fair) = 0.2
  - Transitions between states modeled by a Markov process

Transition model for the casino



Slide by Changui Yan

The occasionally dishonest casino

- Known:
  - The structure of the model
  - The transition probabilities
- Hidden: What the casino did
  - FFFFFLLLLLLLFFFF...
- Observable: The series of die tosses
  - 3415256664666153...
- What we must infer:
  - When was a fair die used?
  - When was a loaded one used?
    - The answer is a sequence FFFFFFFLLLLLLFFF...

Making the inference

- Model assigns a probability to each explanation of the observation:

  $P(326|FFL)$
  $= P(3|F) \cdot P(F \rightarrow F) \cdot P(2|F) \cdot P(F \rightarrow L) \cdot P(6|L)$
  $= 1/6 \cdot 0.99 \cdot 1/6 \cdot 0.01 \cdot ½$

- **Maximum Likelihood:** Determine which explanation is most likely
  - Find the path *most likely* to have produced the observed sequence
- **Total probability:** Determine probability that observed sequence was produced by the model
  - Consider *all* paths that could have produced the observed sequence

Notation

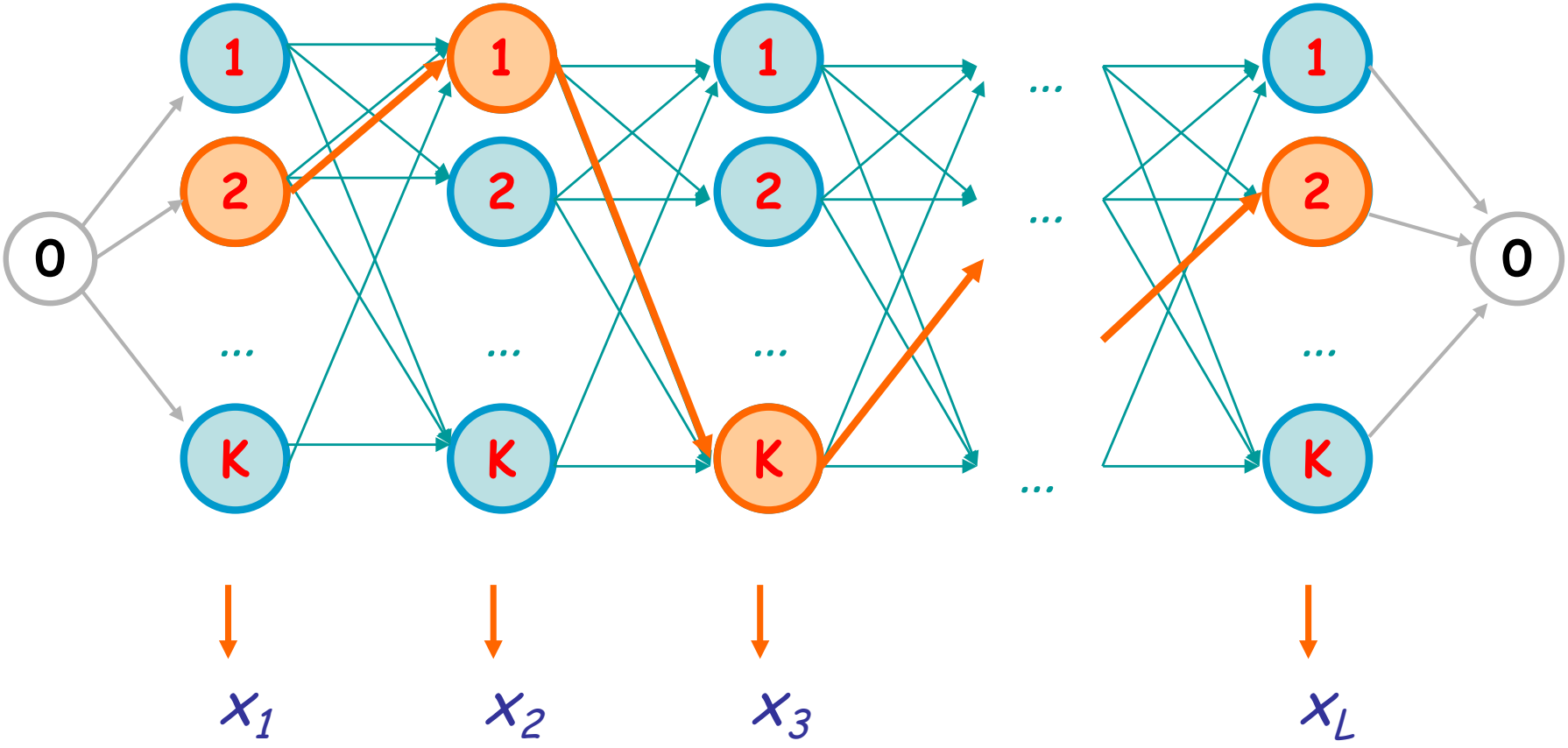
- x is the sequence of symbols emitted by model
  - x_i is the symbol emitted at time i
- A ***path***, π, is a sequence of states
  - The i-th state in π is π_i
- a_{kr} is the probability of making a transition from state k to state r:

$$a_{kr} = \Pr(\pi_i = r \mid \pi_{i-1} = k)$$

- $e_k(b)$ is the probability that symbol b is emitted when in state k

$$e_k(b) = \Pr(x_i = b \mid \pi_i = k)$$

A “parse” of a sequence



$$\Pr(x,\pi) = a_{0\pi_1} \prod_{i=1}^{L} e_{\pi_i}(x_i) \cdot a_{\pi_i \pi_{i+1}}$$

Slide by Changui Yan

The occasionally dishonest casino

$$x = \langle x_1, x_2, x_3 \rangle = \langle 6, 2, 6 \rangle$$

$\pi^{(1)} = FFF$

$$\Pr(x, \pi^{(1)}) = a_{0F} e_F(6) a_{FF} e_F(2) a_{FF} e_F(6)$$
$$= 0.5 \times \frac{1}{6} \times 0.99 \times \frac{1}{6} \times 0.99 \times \frac{1}{6}$$
$$\approx 0.00227$$

$\pi^{(2)} = LLL$

$$\Pr(x, \pi^{(2)}) = a_{0L} e_L(6) a_{LL} e_L(2) a_{LL} e_L(6)$$
$$= 0.5 \times 0.5 \times 0.8 \times 0.1 \times 0.8 \times 0.5$$
$$= 0.008$$

$\pi^{(3)} = LFL$

$$\Pr(x, \pi^{(3)}) = a_{0L} e_L(6) a_{LF} e_F(2) a_{FL} e_L(6) a_{L0}$$
$$= 0.5 \times 0.5 \times 0.2 \times \frac{1}{6} \times 0.01 \times 0.5$$
$$\approx 0.0000417$$

Slide by Changui Yan

The most probable path

The most likely path π^* satisfies

$$\pi^* = \arg\max_{\pi} \Pr(x, \pi)$$

To find π^*, consider all possible ways the last symbol of x could have been emitted

Let

$$v_k(i) = \text{Prob. of path } \langle \pi_1, \cdots, \pi_i \rangle \text{ most likely}$$

$$\text{to emit } \langle x_1, \ldots, x_i \rangle \text{ such that } \pi_i = k$$

Then

$$v_k(i) = e_k(x_i) \max_r \left(v_r(i-1) a_{rk} \right)$$

The Viterbi Algorithm

- Initialization $(i = 0)$

$$v_0(0) = 1, \quad v_k(0) = 0 \text{ for } k > 0$$

- Recursion ($i = 1, \ldots, L$): For each state k

$$v_k(i) = e_k(x_i) \max_r \left(v_r(i-1) a_{rk} \right)$$

- Termination:

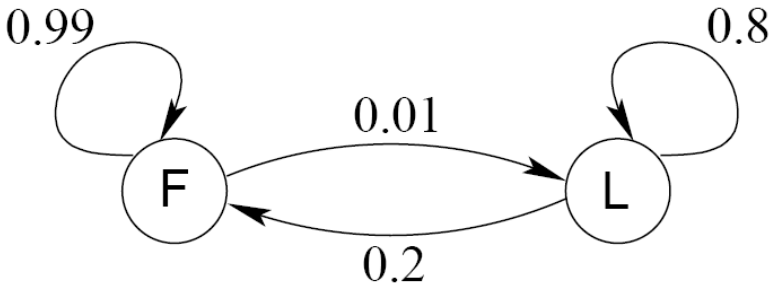
$$\Pr(x, \pi^*) = \max_k \left(v_k(Length) a_{k0} \right)$$

To find π^*, use trace-back, as in dynamic programming

Viterbi: Example

		ε	6	2	6
				x	
	B	**1**	**0**	**0**	**0**
π	F	**0**	$(1/6)\times(1/2)$ $= 1/12$	$(1/6)\times\max\{(1/12)\times 0.99,$ $(1/4)\times 0.2\}$ $= 0.01375$	$(1/6)\times\max\{0.01375\times 0.99,$ $0.02\times 0.2\}$ $= 0.00226875$
	L	**0**	$(1/2)\times(1/2)$ $= 1/4$	$(1/10)\times\max\{(1/12)\times 0.01,$ $(1/4)\times 0.8\}$ $= 0.02$	$(1/2)\times\max\{0.01375\times 0.01,$ $0.02\times 0.8\}$ $= 0.08$

$$v_k(i) = e_k(x_i)\max_r\left(v_r(i-1)a_{rk}\right)$$



Viterbi gets it right more often than not

```
Rolls      315116246446644245321131631164152133625144543631656626566666
Die        FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLL
Viterbi    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLL

Rolls      651166453132651245636664631636663162326455235266666625151631
Die        LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLLFFFFFFFFF
Viterbi    LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFF

Rolls      222555441666566563564324364131513465146353411126414626253356
Die        FFFFFFFFLLLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL
Viterbi    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL

Rolls      366163666466232534413661661163252562462255265252664353533336
Die        LLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Viterbi    LLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

Rolls      233121625364414432335163243633665562466662632666612355245242
Die        FFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFF
Viterbi    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLFFFFFFFFF
```

Slide by Changui Yan

Dynamic Bayesian Networks

- *Learning* requires the full smoothing inference, rather than filtering, because it provides better estimates of the state of the process.

- Learning the parameters of a BN is done using Expectation – Maximization (EM) Algorithms. Iterative optimization method to estimate some unknowns parameters.

Acknowledgements:

The following slides are taken from
CS 388 Natural Language Processing:
Part-Of-Speech Tagging,
Sequence Labeling, and
Hidden Markov Models (HMMs)

Raymond J. Mooney

University of Texas at Austin

Application 1: Part Of Speech Tagging

- Annotate each word in a sentence with a part-of-speech marker.
- Lowest level of syntactic analysis.

John saw the saw and decided to take it to the table.
NNP VBD DT NN CC VBD TO VB PRP IN DT NN

- Useful for subsequent syntactic parsing and word sense disambiguation.

English Parts of Speech

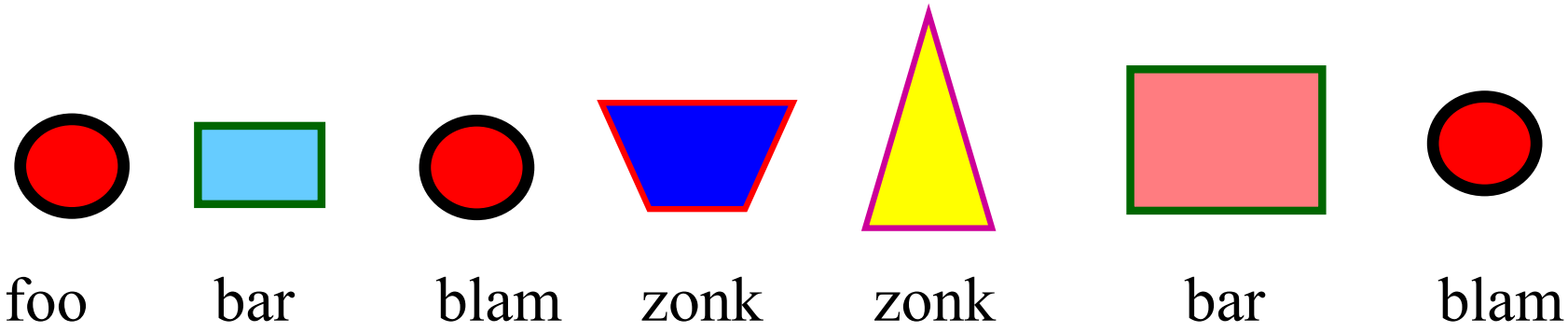
- Noun (person, place or thing)
  - Singular (NN): dog, fork
  - Plural (NNS): dogs, forks
  - Proper (NNP, NNPS): John, Springfields
  - Personal pronoun (PRP): I, you, he, she, it
  - Wh-pronoun (WP): who, what
- Verb (actions and processes)
  - Base, infinitive (VB): eat
  - Past tense (VBD): ate
  - Gerund (VBG): eating
  - Past participle (VBN): eaten
  - Non 3rd person singular present tense (VBP): eat
  - 3rd person singular present tense: (VBZ): eats
  - Modal (MD): should, can
  - To (TO): to (to eat)

English Parts of Speech (cont.)

- Adjective (modify nouns)
  - Basic (JJ): red, tall
  - Comparative (JJR): redder, taller
  - Superlative (JJS): reddest, tallest
- Adverb (modify verbs)
  - Basic (RB): quickly
  - Comparative (RBR): quicker
  - Superlative (RBS): quickest
- Preposition (IN): on, in, by, to, with
- Determiner:
  - Basic (DT) a, an, the
  - WH-determiner (WDT): which, that
- Coordinating Conjunction (CC): and, but, or,
- Particle (RP): off (took off), up (put up)

In General: Sequence Labeling Problem

- Many NLP problems can viewed as sequence labeling.
- Each token in a sequence is assigned a label.
- Labels of tokens are dependent on the labels of other tokens in the sequence, particularly their neighbors (not i.i.d).



Application 2: Information Extraction

- Identify phrases in language that refer to specific types of entities and relations in text.
- Named entity recognition is the task of identifying names of people, places, organizations, etc. in text.

  people organizations places

  – Michael Dell is the CEO of Dell Computer Corporation and lives in Austin Texas.
- Extract pieces of information relevant to a specific application, e.g. used car ads:

  make model year mileage price

  – For sale, 2002 Toyota Prius, 20,000 mi, $15K or best offer. Available starting July 30, 2006.

Semantic Role Labeling

- For each clause, determine the semantic role played by each noun phrase that is an argument to the verb.

  agent patient source destination instrument

  – John drove Mary from Austin to Dallas in his Toyota Prius.

  – The hammer broke the window.

- Also referred to a "case role analysis," "thematic analysis," and "shallow semantic parsing"

Application 3: Bioinformatics

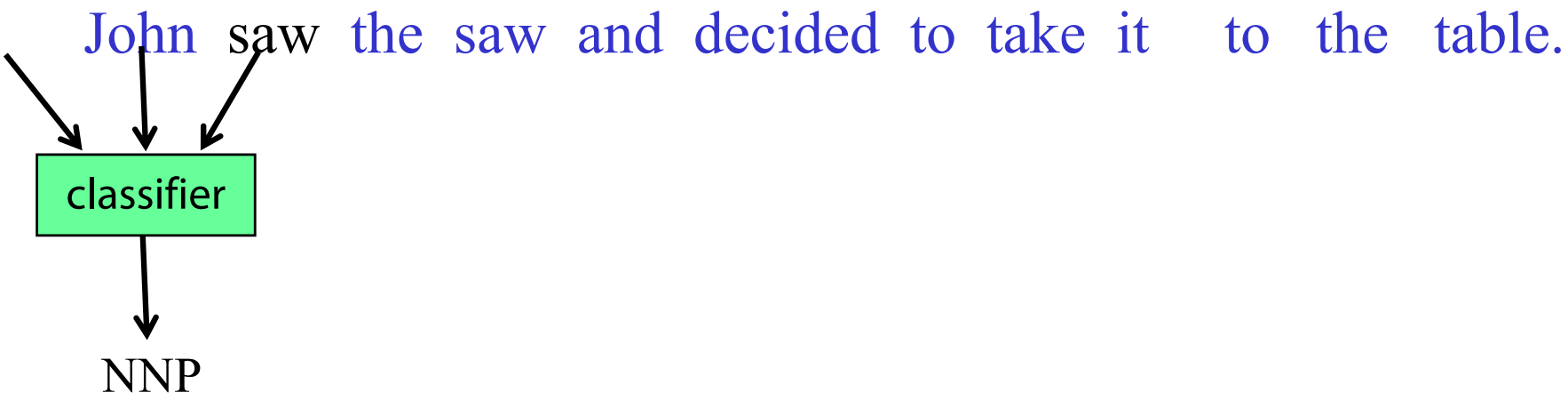
- Sequence labeling also valuable in labeling genetic sequences in genome analysis.

  extron intron

  – AGCTAACGTTCGATACGGATTACAGCCT

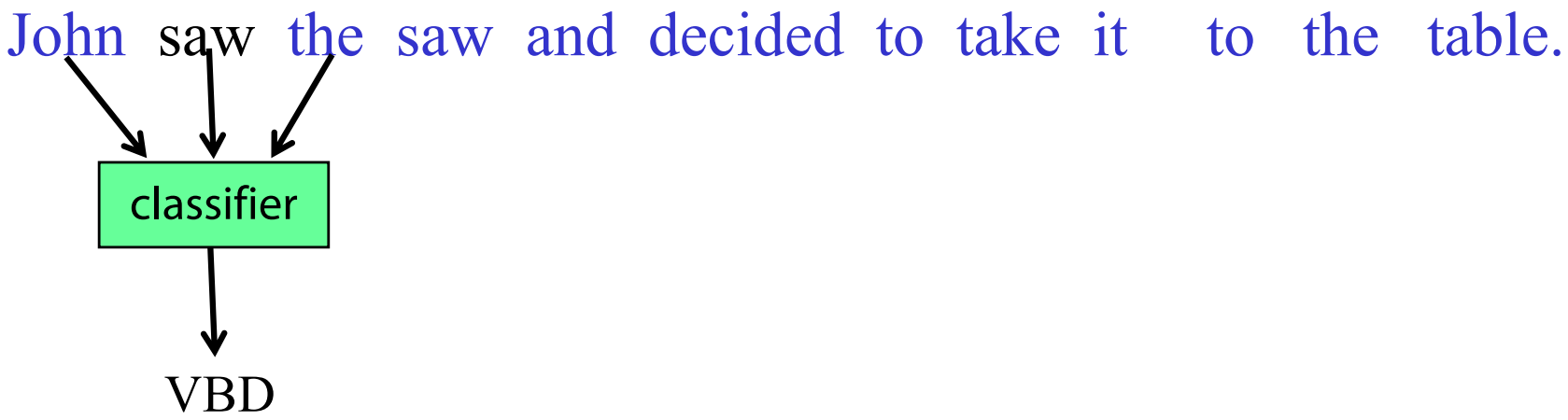
Back to App1: Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



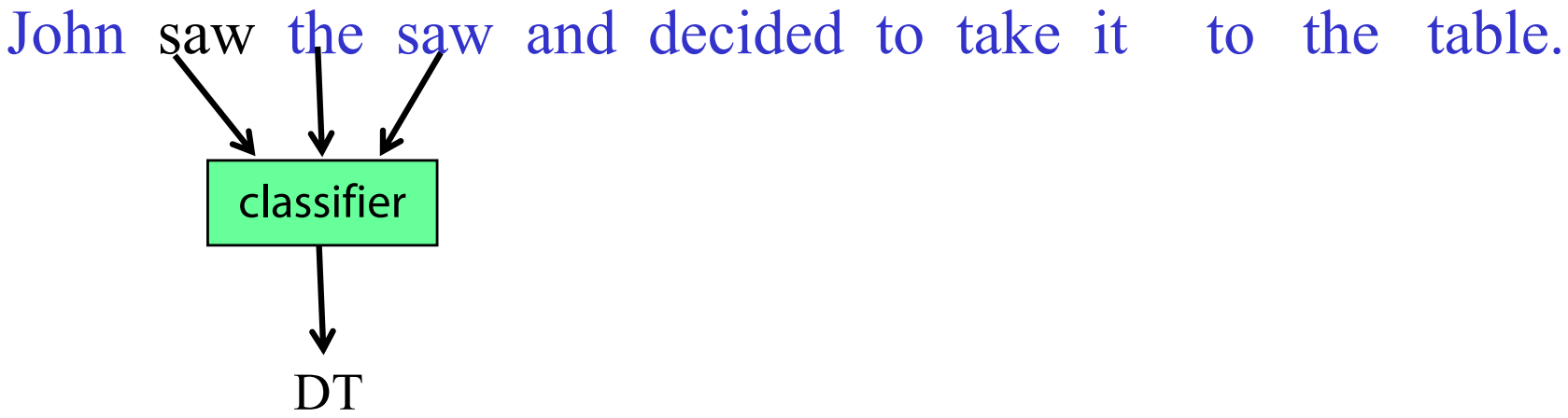
Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



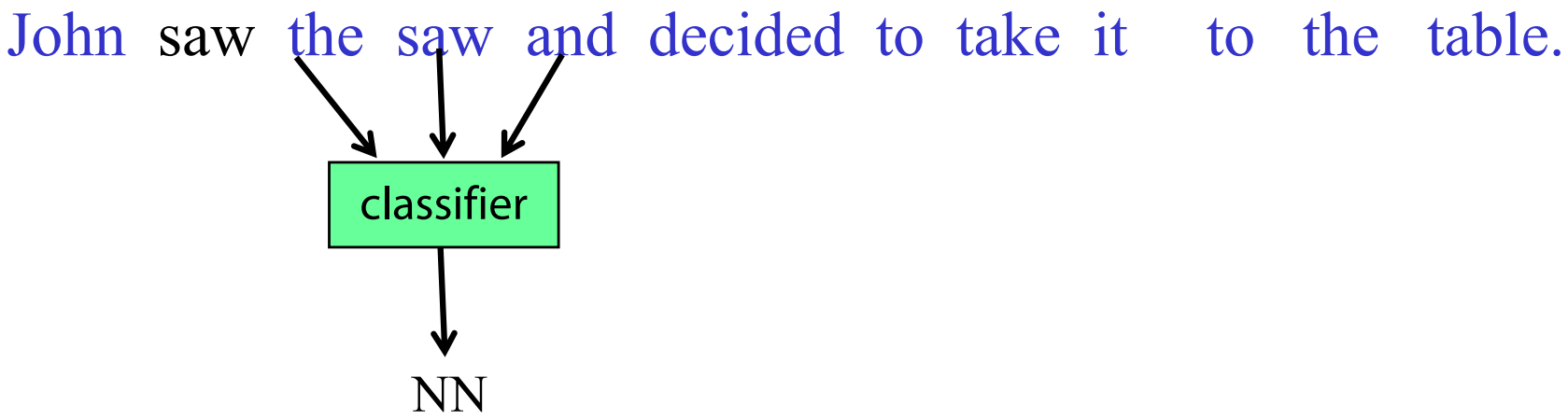
Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



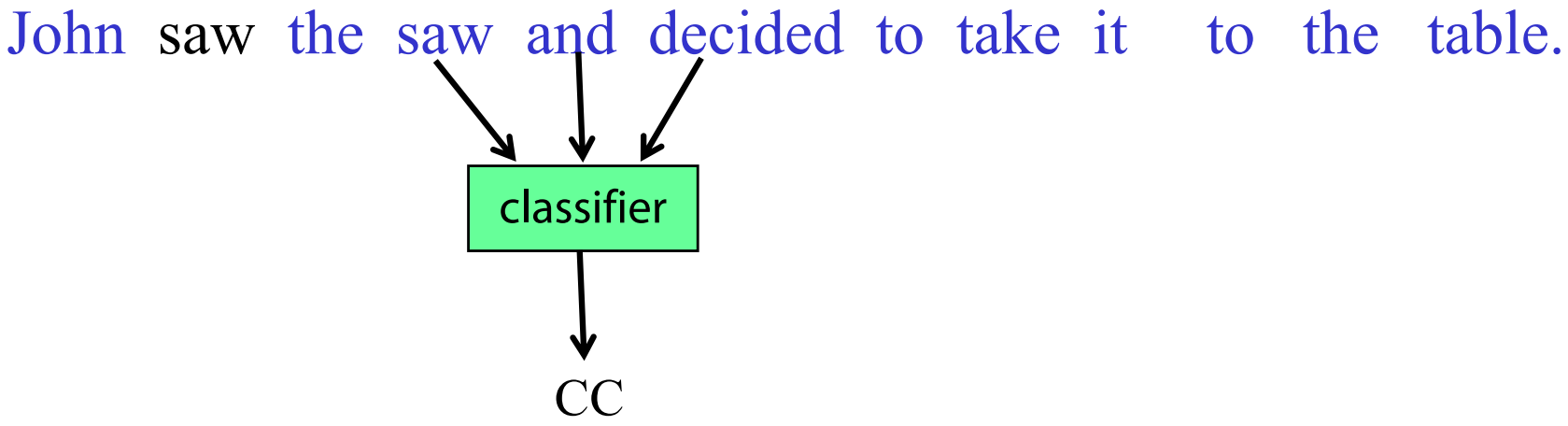
Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



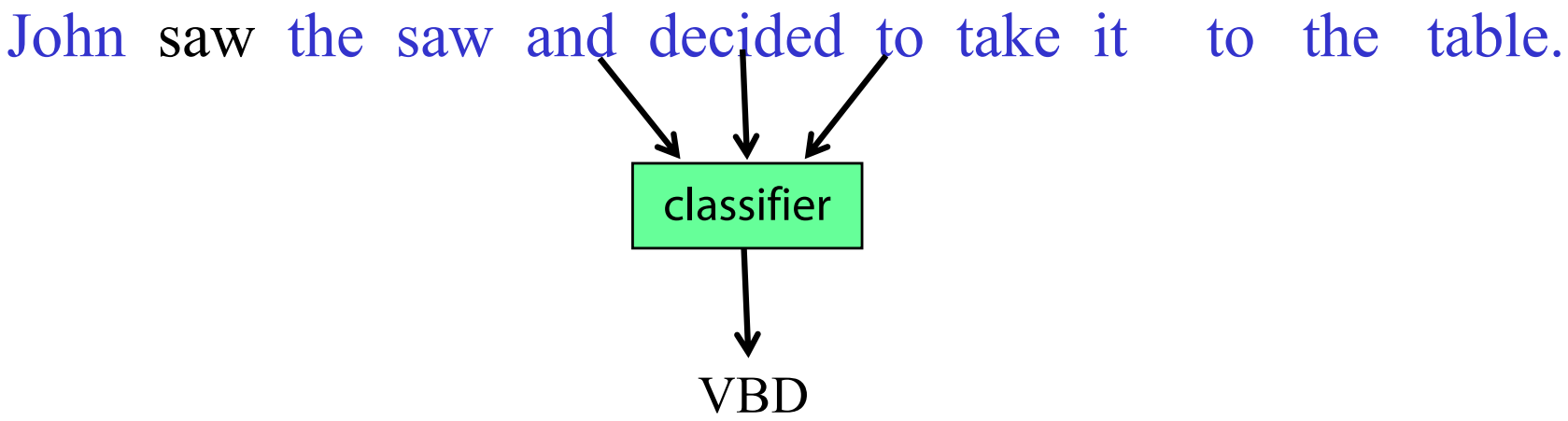
Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



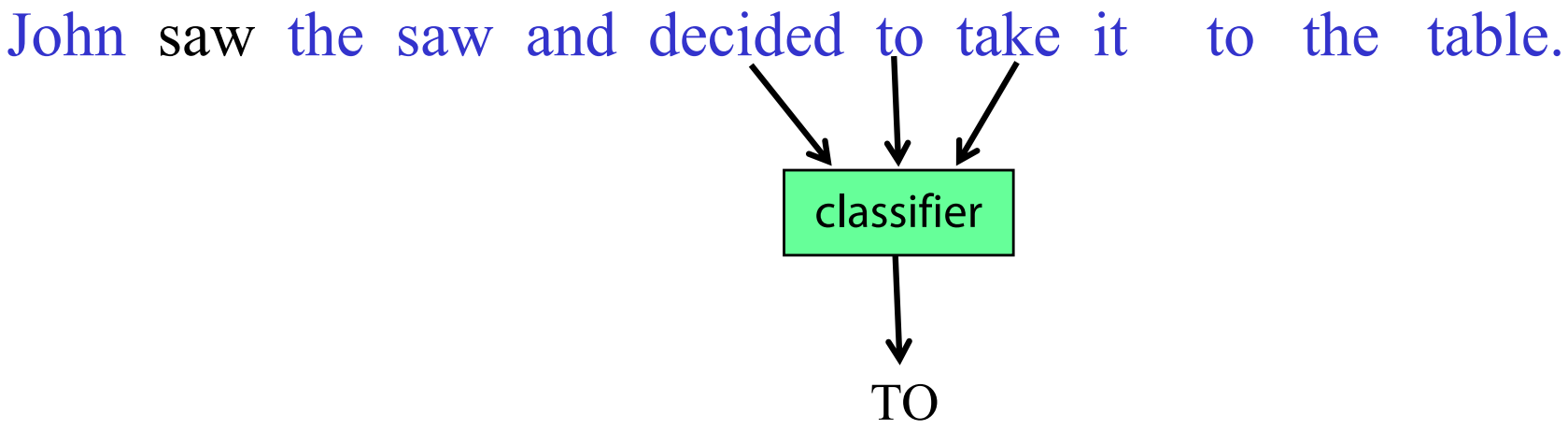
Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



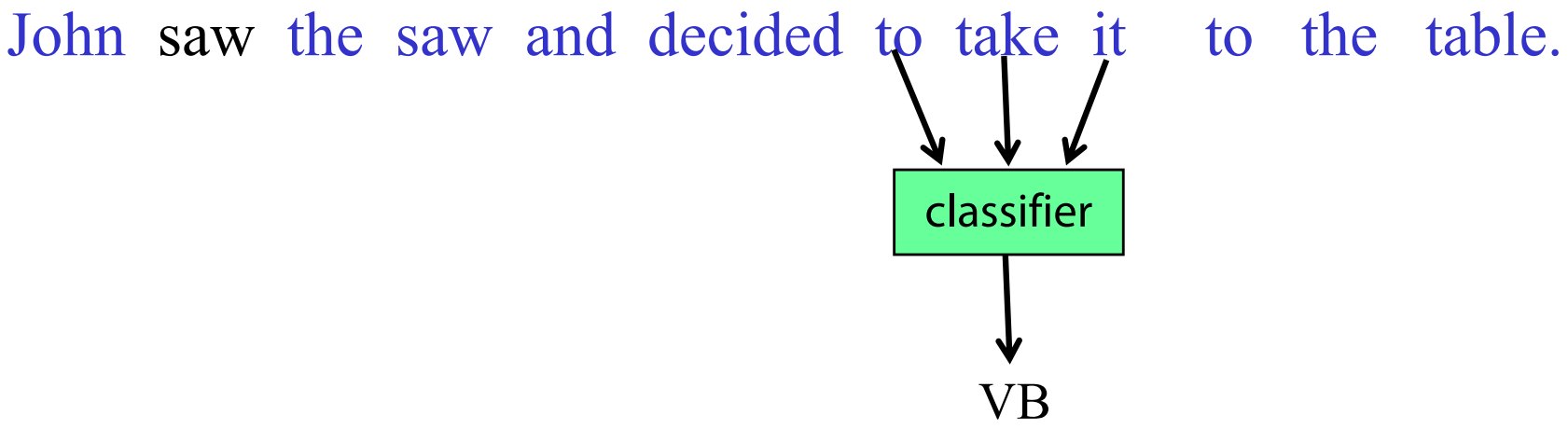
Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



Sequence Labeling as Classification

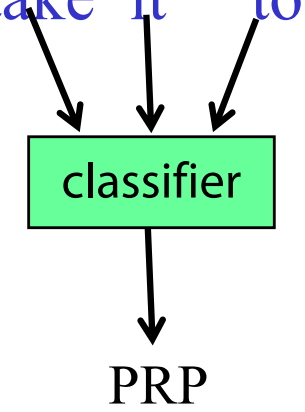
- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



Sequence Labeling as Classification

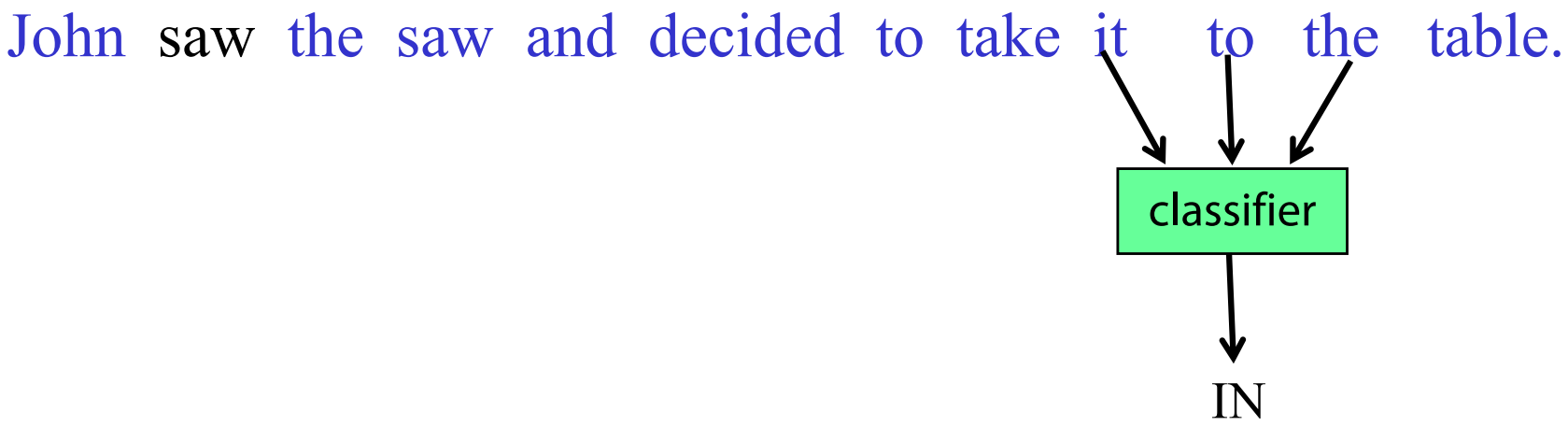
- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).

John saw the saw and decided to take it to the table.



Sequence Labeling as Classification

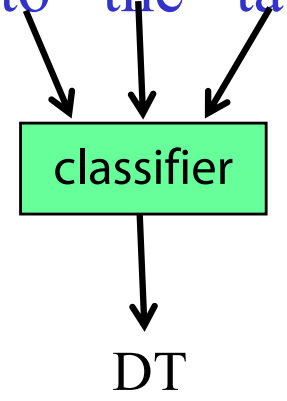
- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).

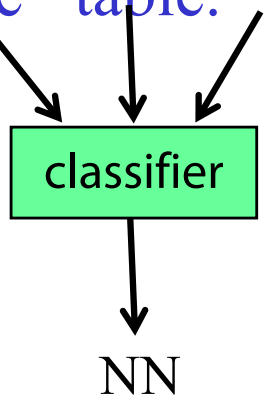
John saw the saw and decided to take it to the table.



Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).

John saw the saw and decided to take it to the table.

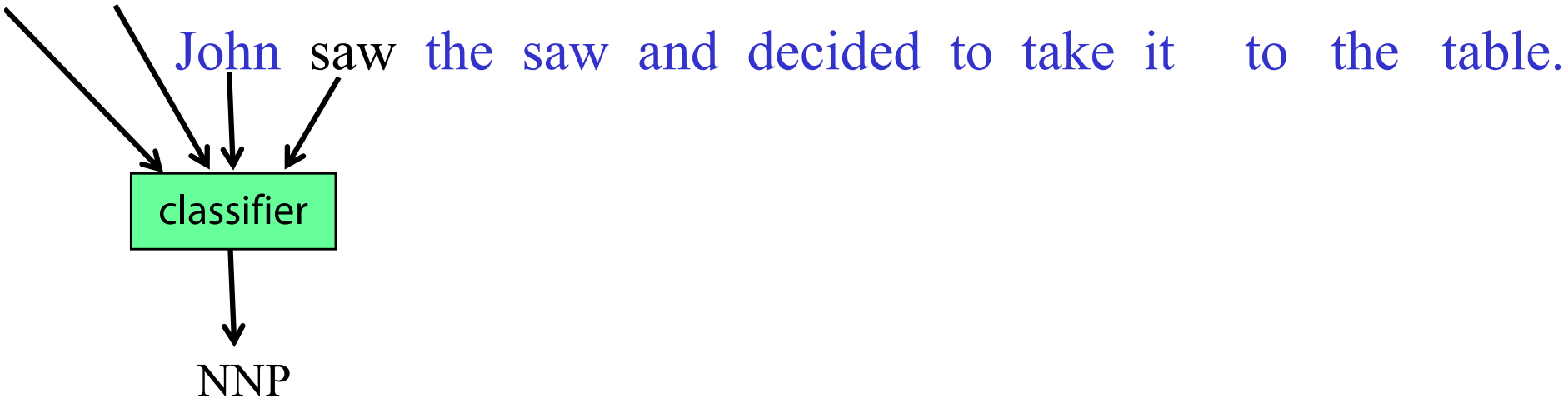


Sequence Labeling as Classification

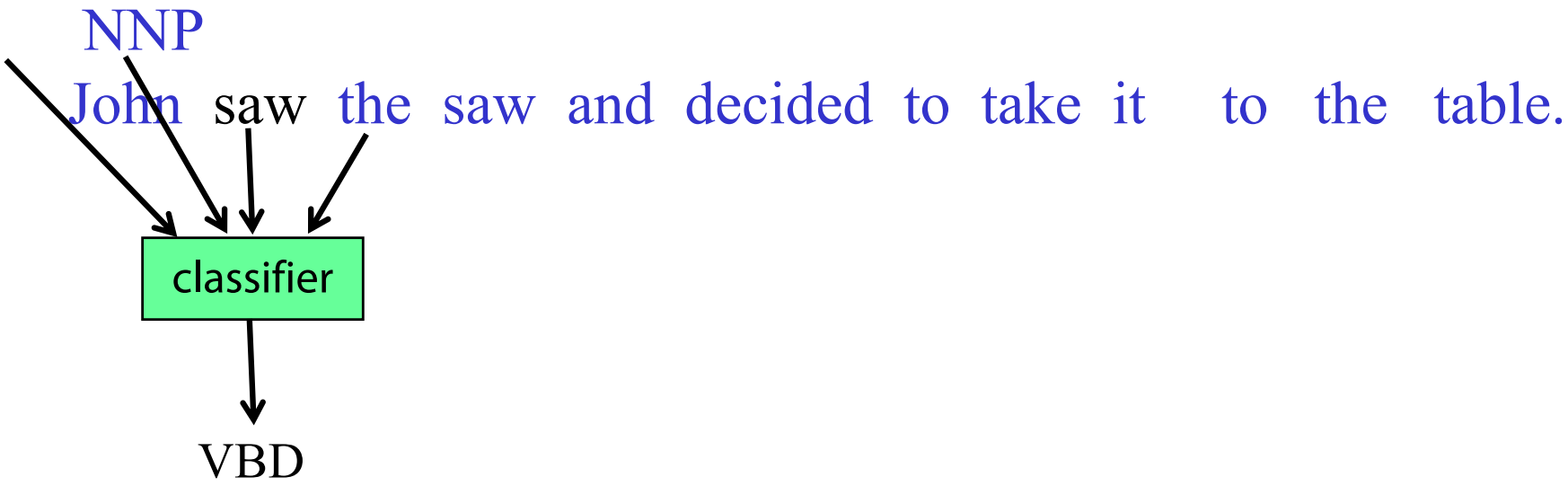
Using Outputs as Inputs

- Better input features are usually the categories of the surrounding tokens, but these are not available yet.
- Can use category of either the preceding or succeeding tokens by going forward or back and using previous output.

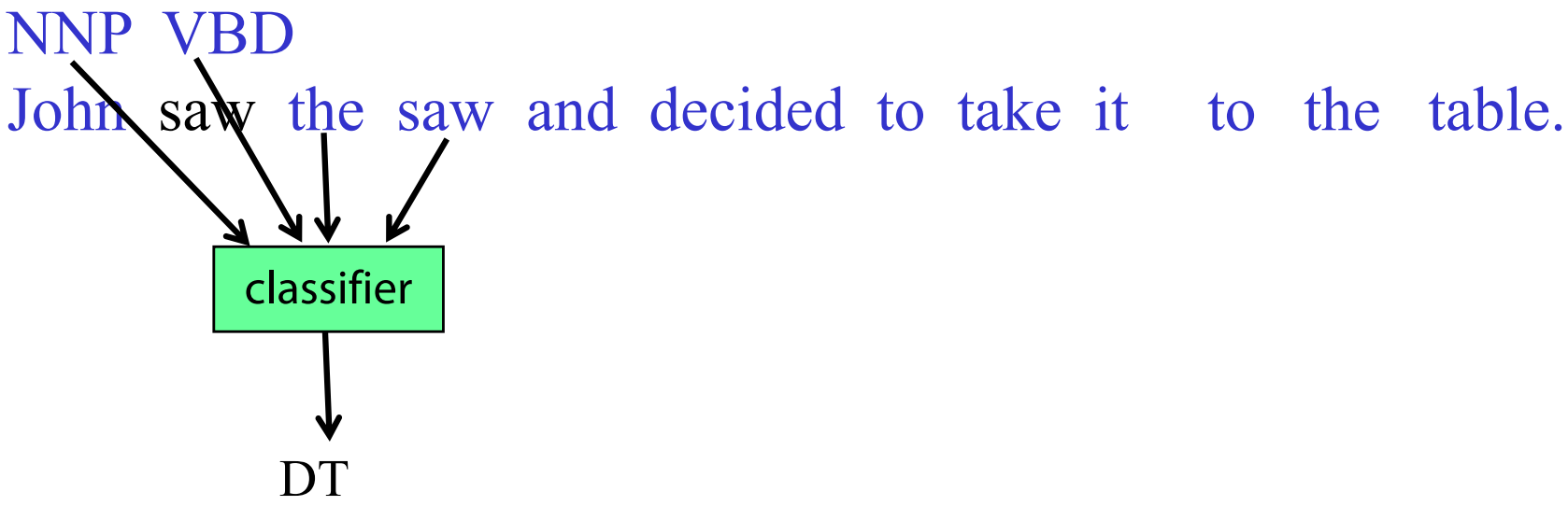
Forward Classification



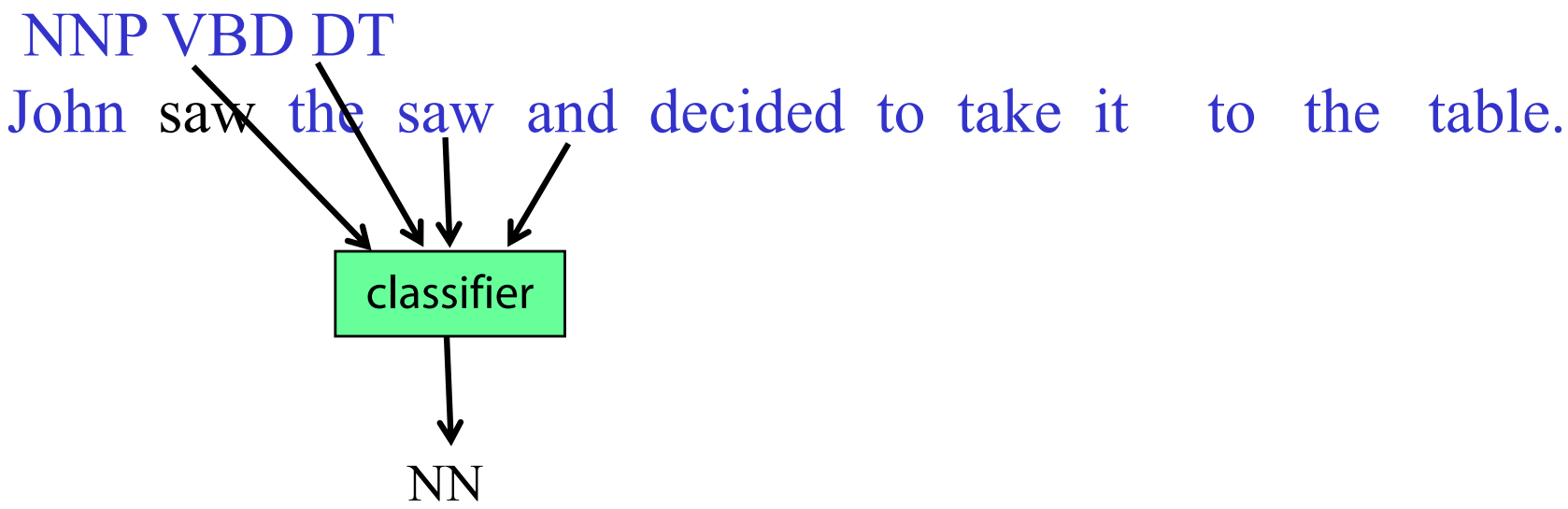
Forward Classification



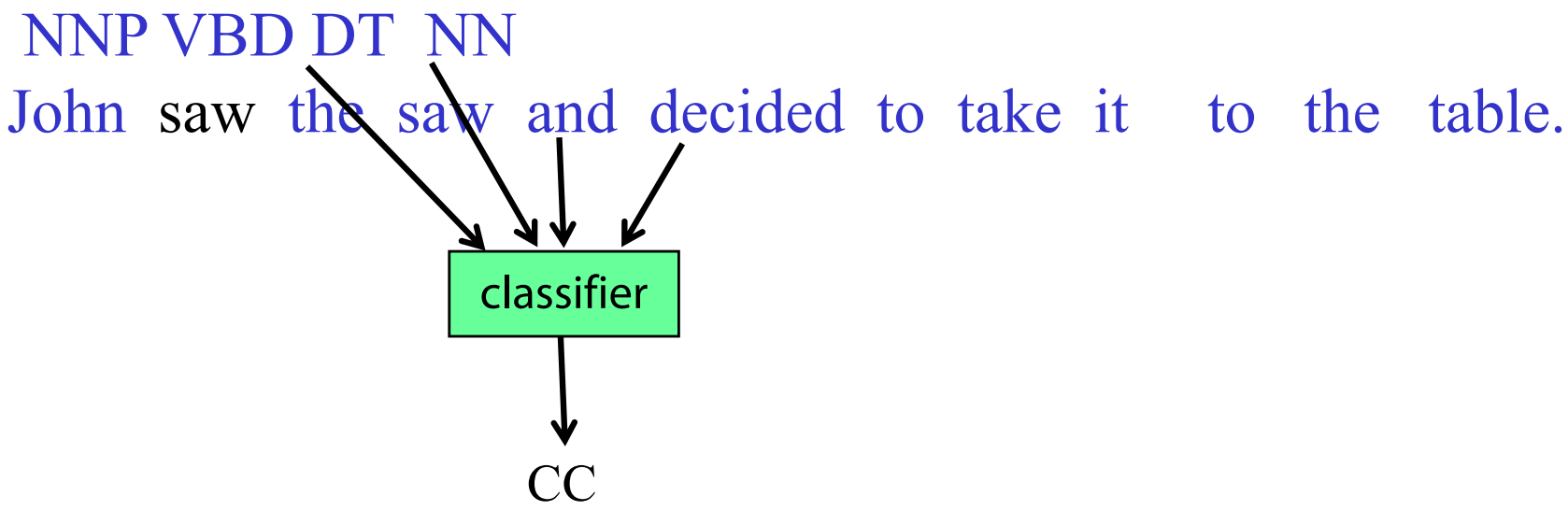
Forward Classification



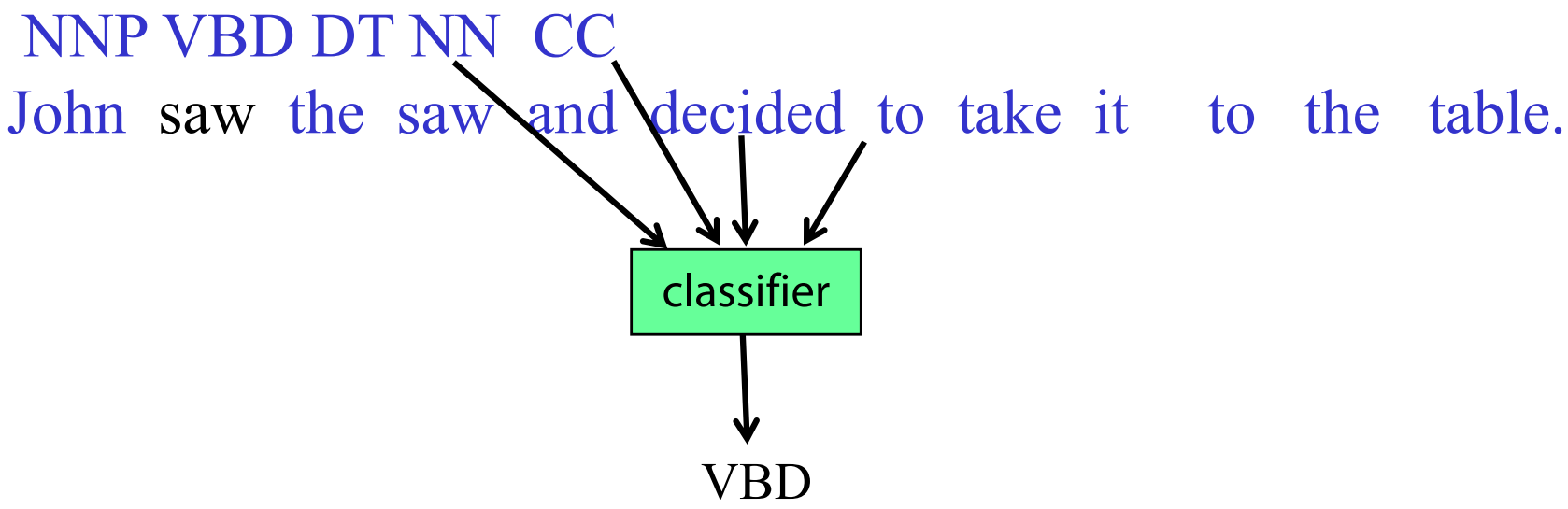
Forward Classification



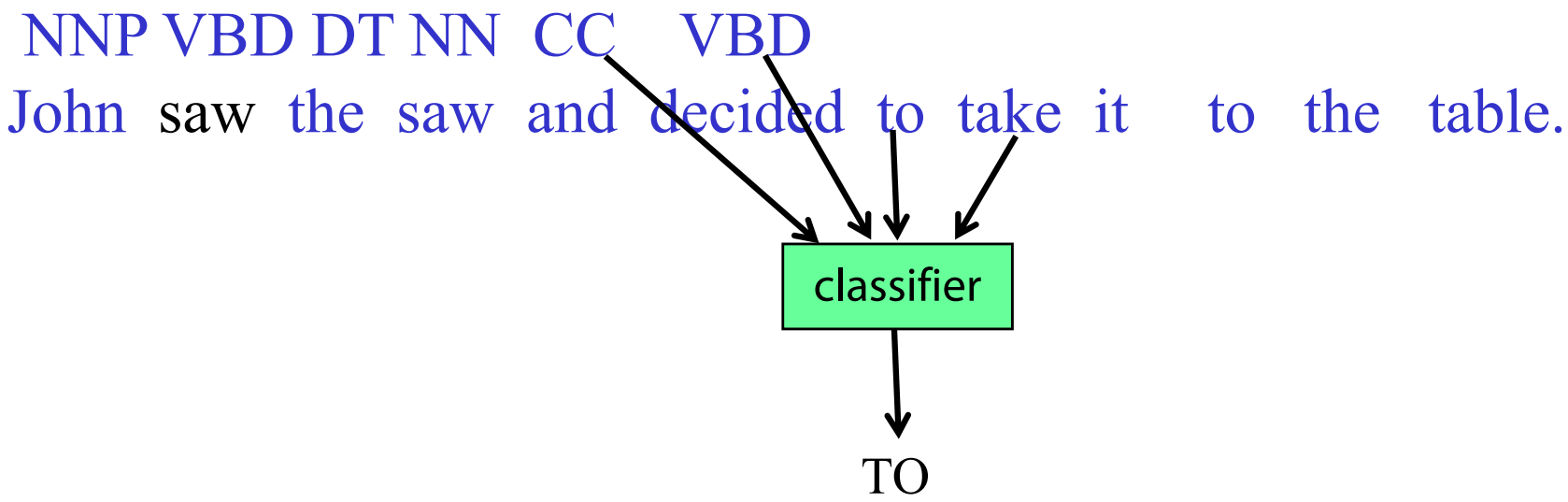
Forward Classification



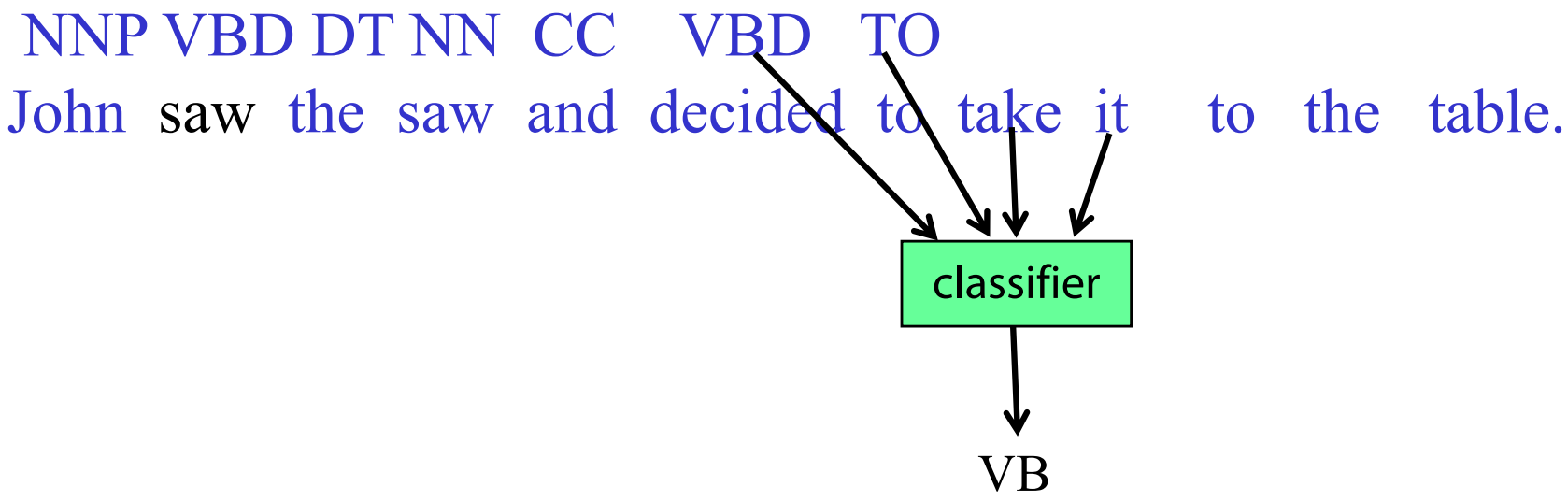
Forward Classification



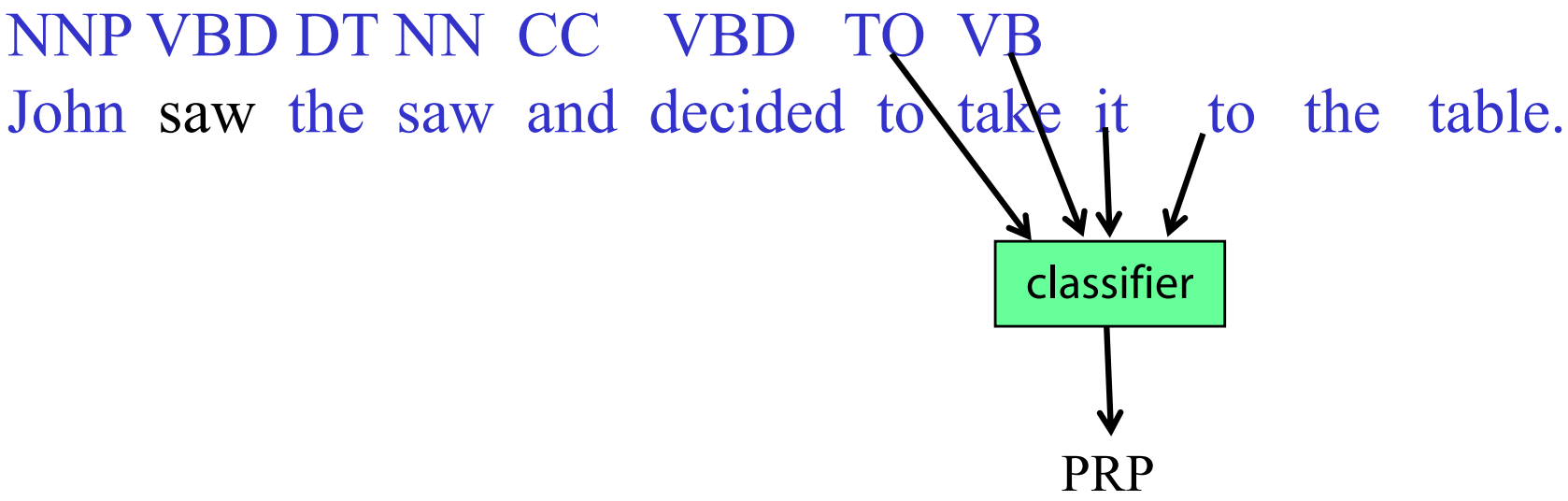
Forward Classification



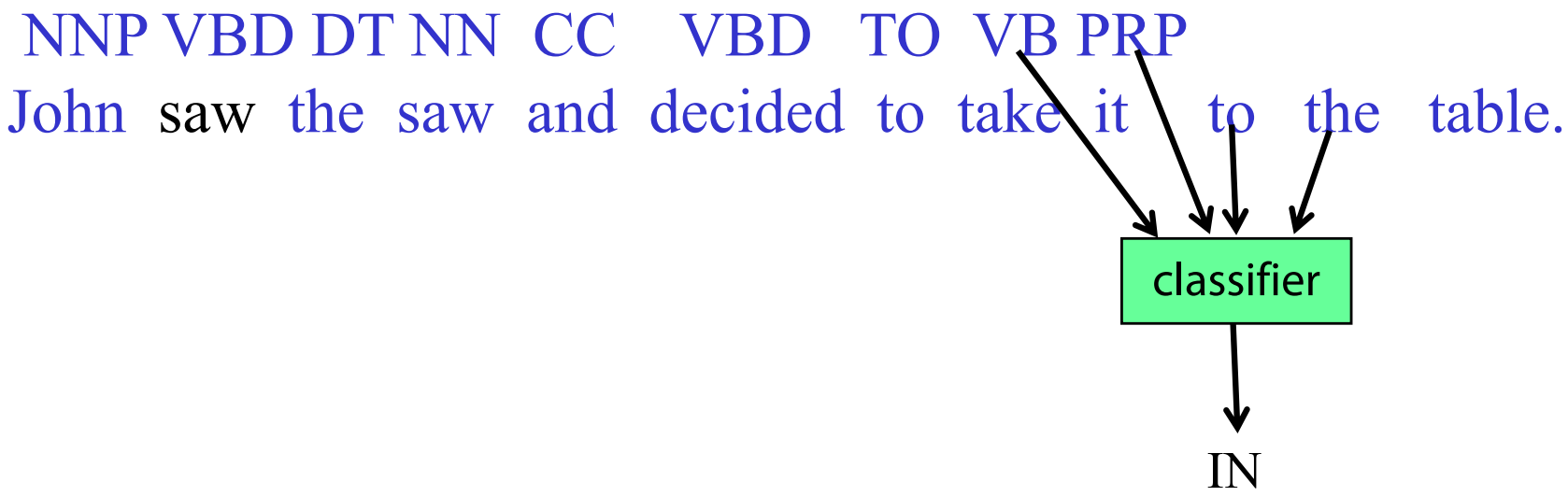
Forward Classification



Forward Classification

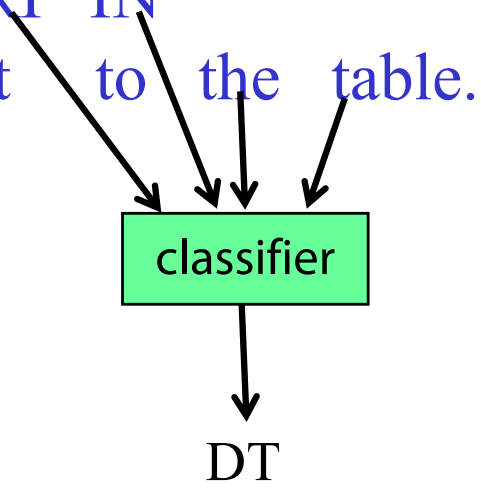


Forward Classification



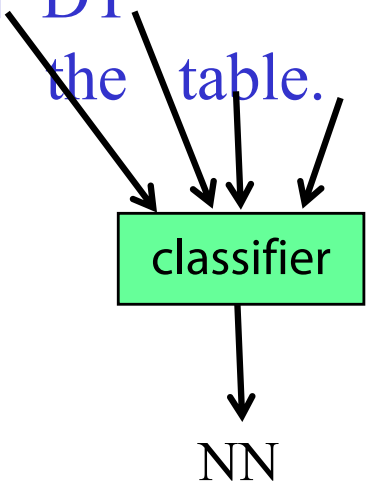
Forward Classification

NNP VBD DT NN CC VBD TO VB PRP IN
John saw the saw and decided to take it to the table.



Forward Classification

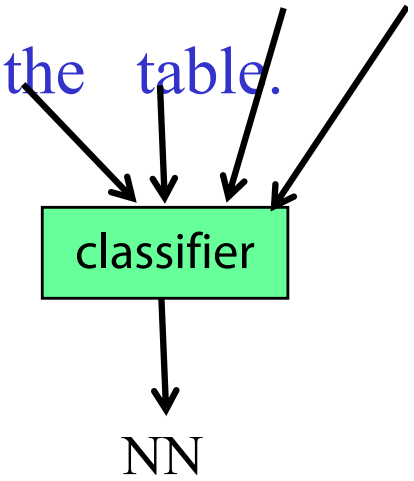
NNP VBD DT NN CC VBD TO VB PRP IN DT
John saw the saw and decided to take it to the table.



Backward Classification

- Disambiguating “to” in this case would be even easier backward.

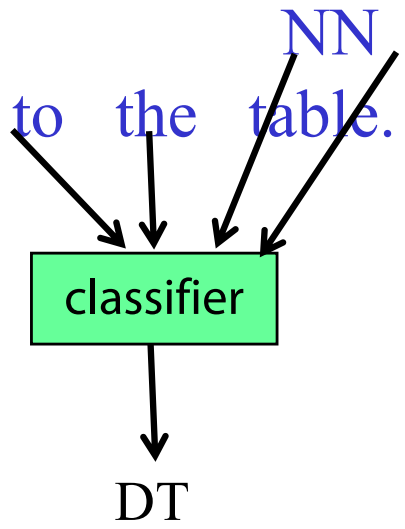
John saw the saw and decided to take it to the table.



Backward Classification

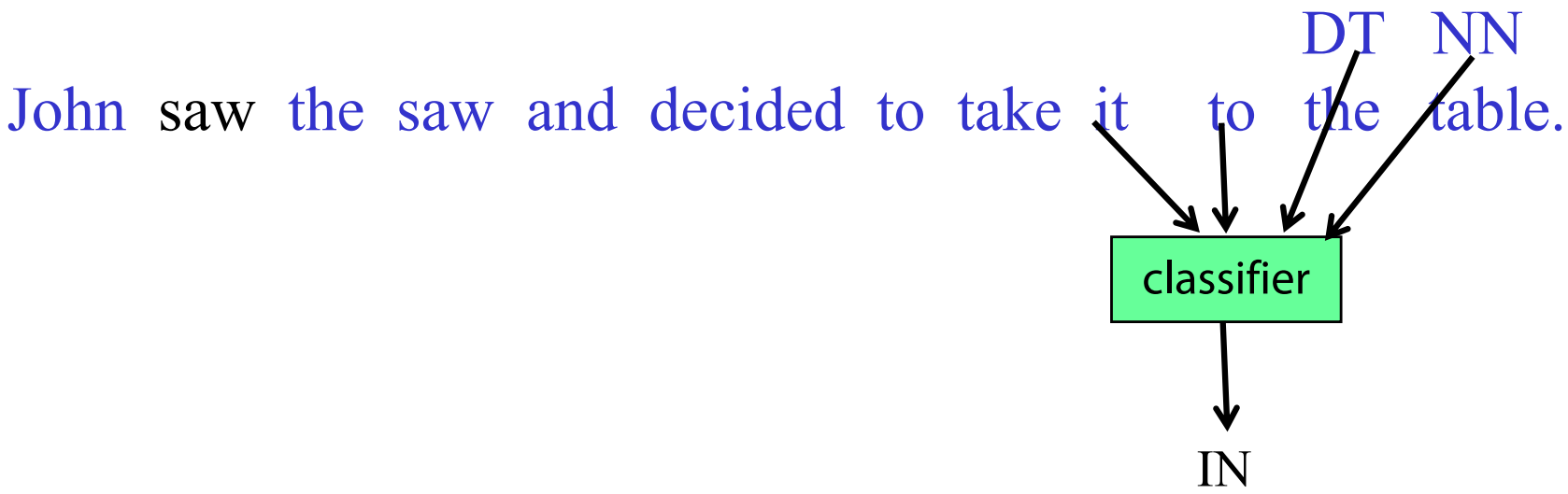
- Disambiguating “to” in this case would be even easier backward.

John saw the saw and decided to take it to the table.



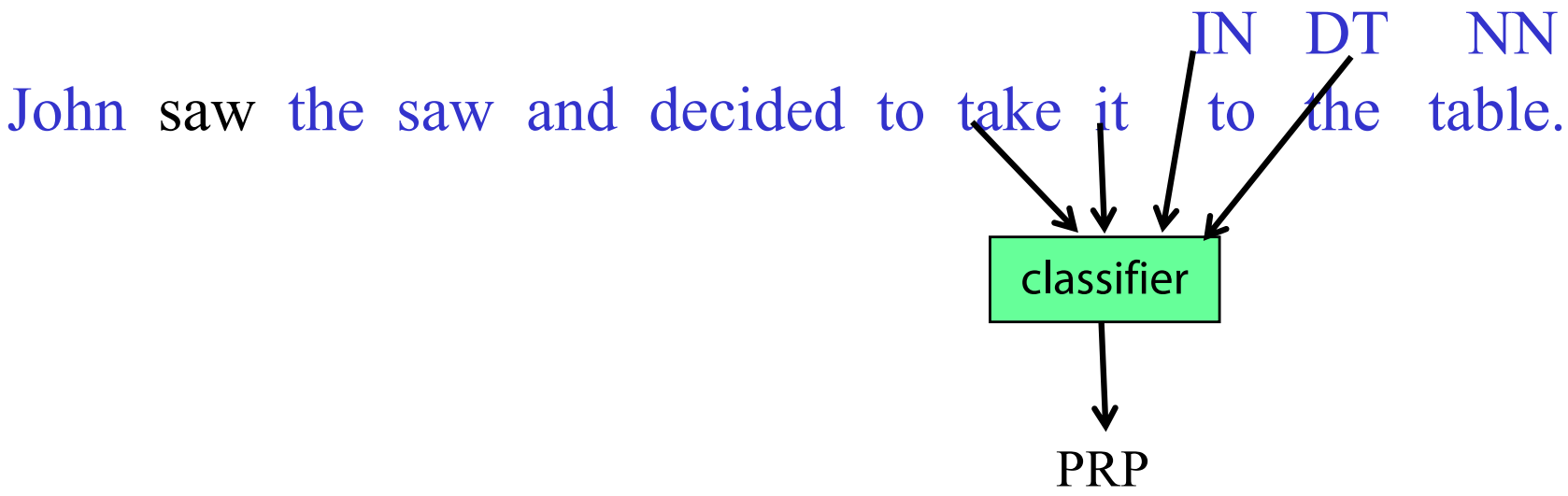
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



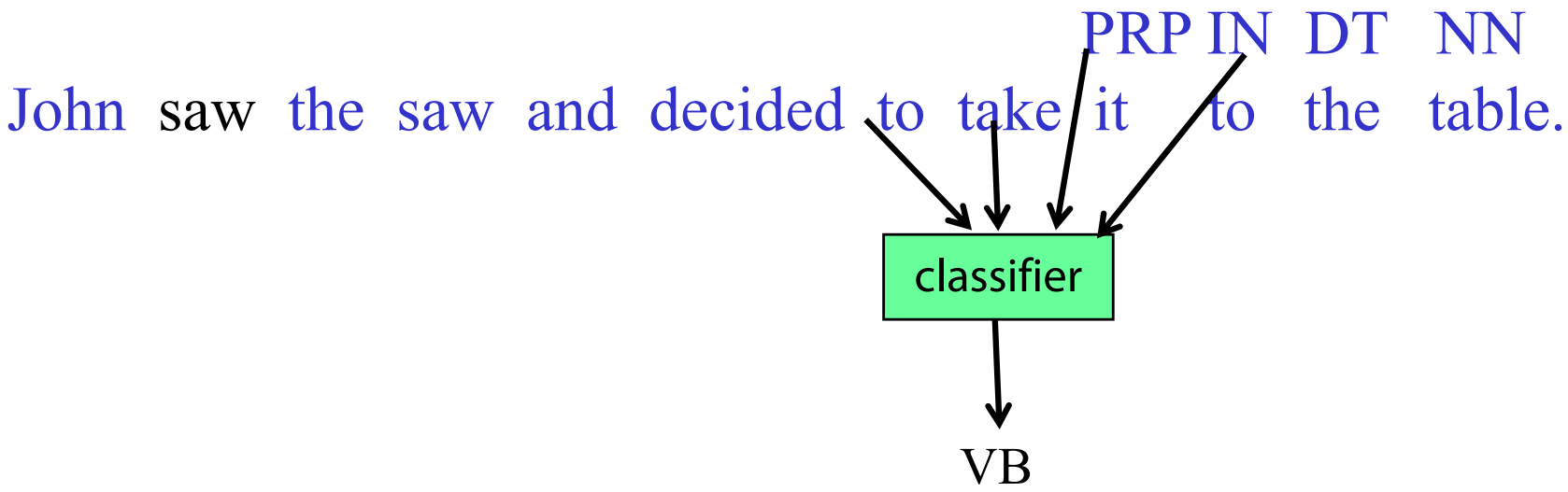
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



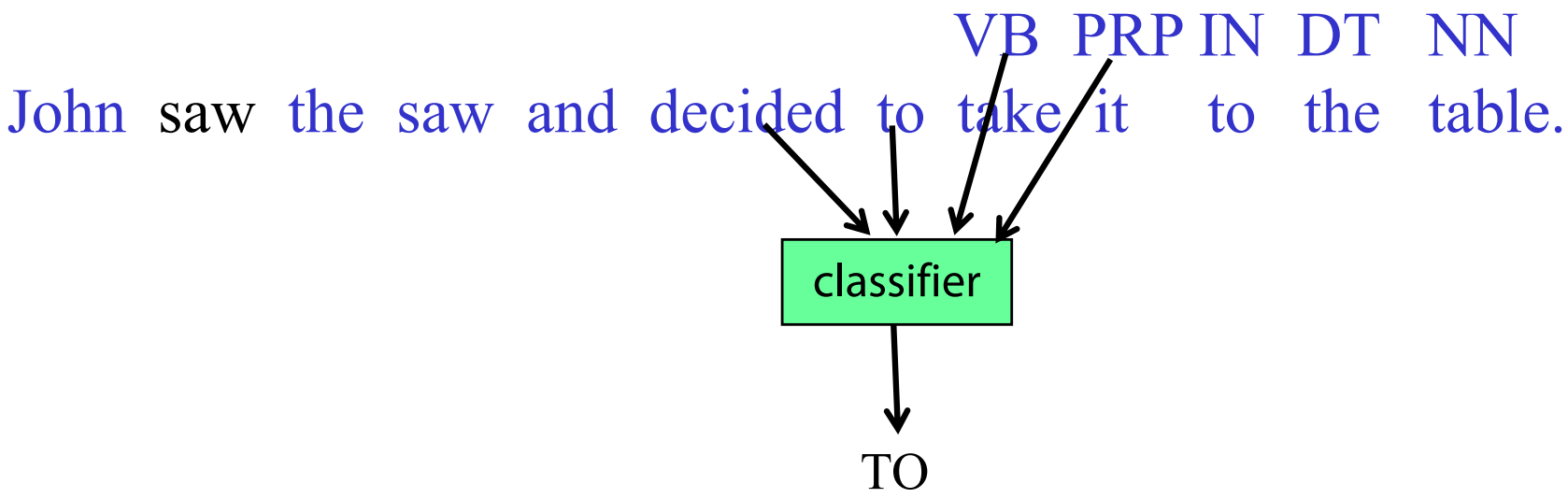
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



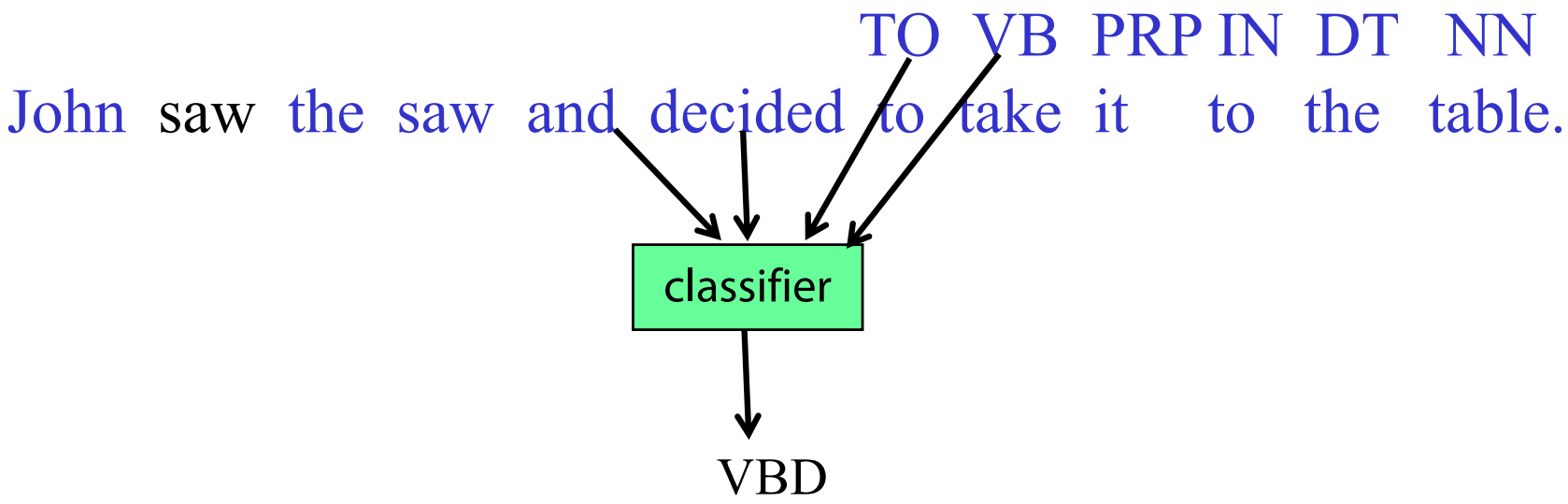
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



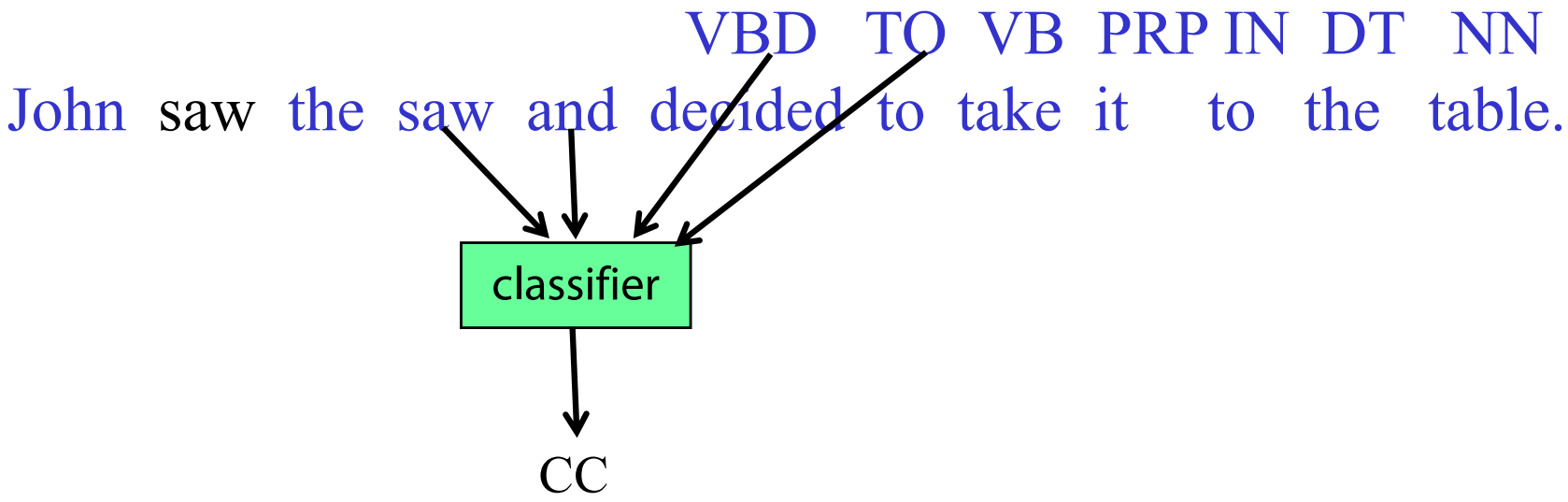
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



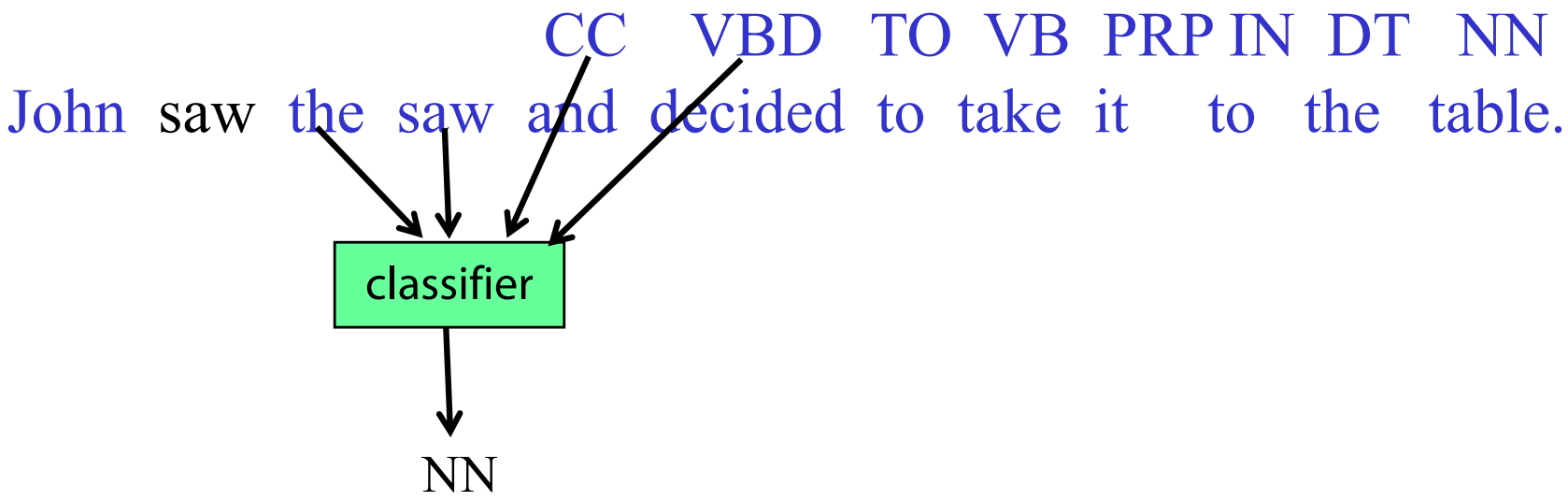
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



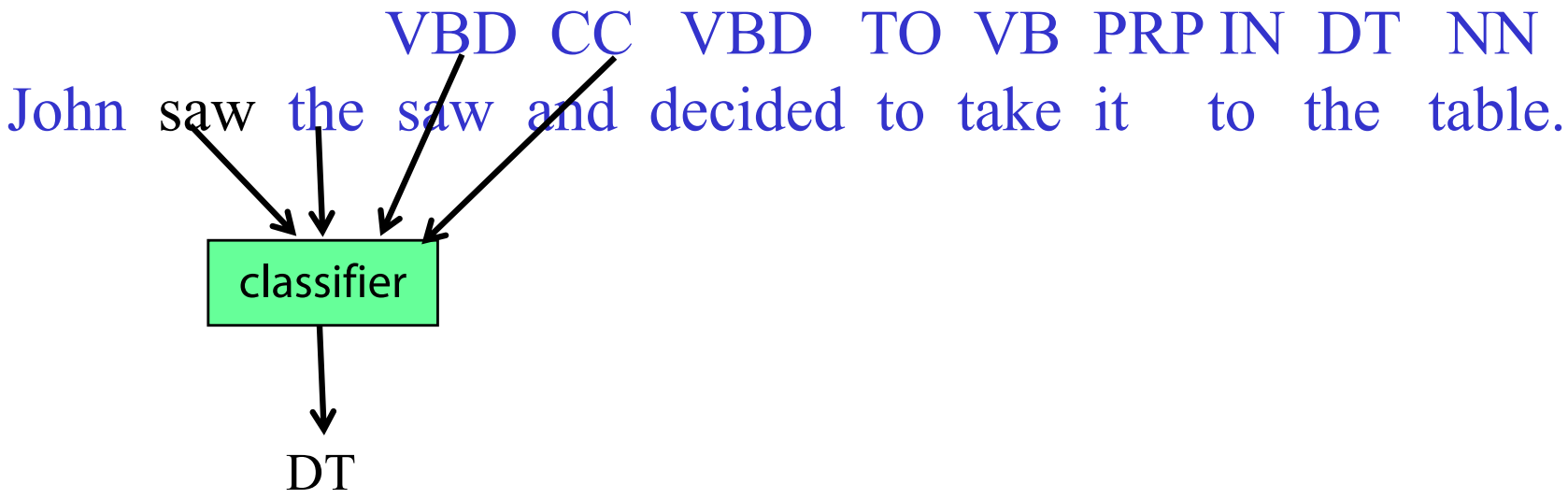
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



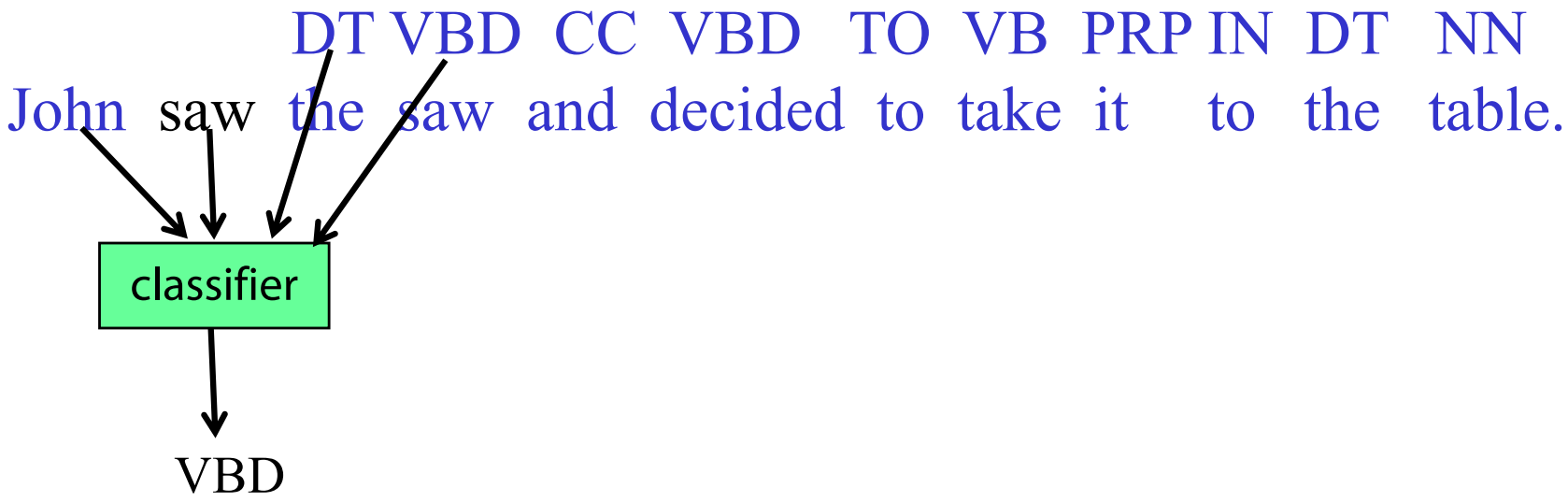
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



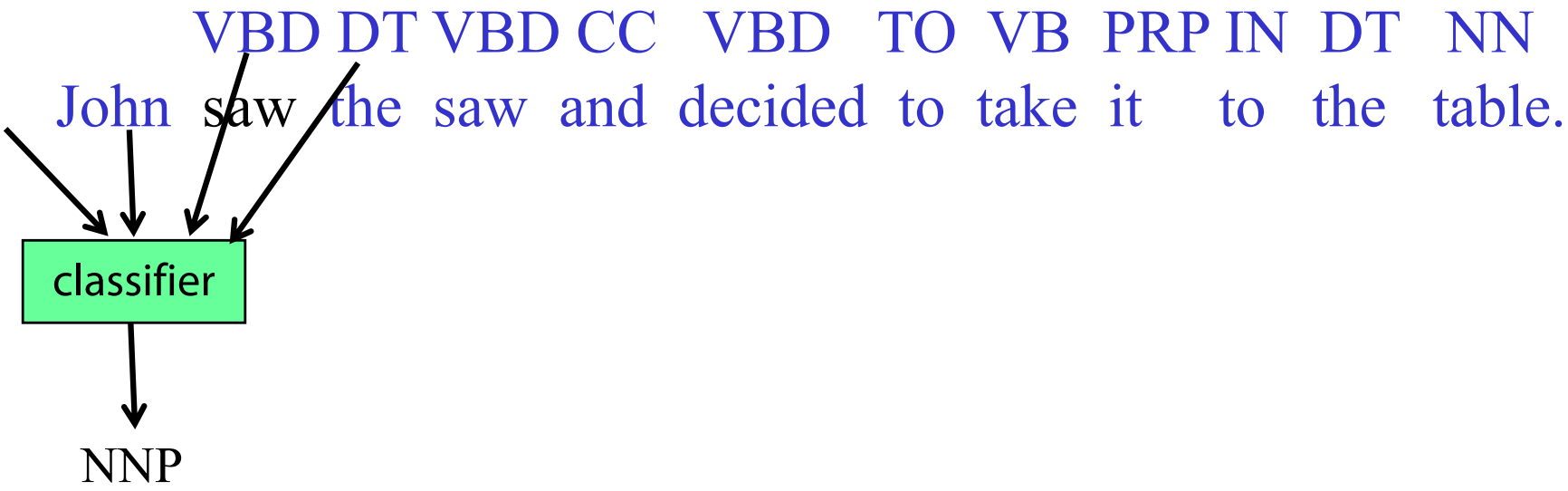
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



Backward Classification

- Disambiguating “to” in this case would be even easier backward.



DBN – Special Cases

- ## Hidden Markov Model (HMMs):

  Temporal probabilistic model in which the state of the process
  is described by a single discrete random variable. (The simplest kind of DBN)

- ## Kalman Filter Models (KFMs):

  Estimate the state of a physical system from noisy observations over time.
  Also known as linear dynamical systems (LDSs).

DBN – Basic Inference

- Filtering

$$P(X_{t+1} / e_{1:t+1}) = \alpha P(e_{t+1} / X_{t+1}) \sum_{X_t} P(X_{t+1} / x_t) P(x_t / e_{1:t})$$

- Smoothing

$$\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:t}) = \alpha \mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:k}) \mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k)$$

$$\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k) = \sum_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1}|\mathbf{x}_{k+1}) P(\mathbf{e}_{k+2:t}|\mathbf{x}_{k+1}) \mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k)$$

- Most likely sequence

$$\max_{\mathbf{x}_1 \ldots \mathbf{x}_t} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{X}_{t+1}|\mathbf{e}_{1:t+1})$$

$$= \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \max_{\mathbf{x}_t} \left(\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t) \max_{\mathbf{x}_1 \ldots \mathbf{x}_{t-1}} P(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{e}_{1:t}) \right)$$

Hidden Markov Models

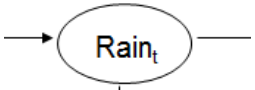
$\mathbf{X}_t$ is a single, discrete variable (usually $\mathbf{E}_t$ is too)
Domain of X_t is $\{1, \ldots, S\}$

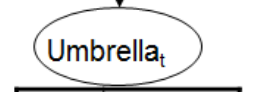
Transition matrix $\mathbf{T}_{ij} = P(X_t = j | X_{t-1} = i)$, e.g., $\begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{pmatrix}$ *new state* / *old state*

Sensor matrix $\mathbf{O}_t$ for each time step, diagonal elements $P(e_t | X_t = i)$

e.g., with $U_1 = true$, $\mathbf{O}_1 = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.2 \end{pmatrix}$ $U_3 = false$ $\mathbf{O}_3 = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.8 \end{pmatrix}$

R_{t-1}	$P(R_t \mid R_{t-1})$
T	0.7
F	0.3





R_t	$P(U_t \mid R_t)$
T	0.9
F	0.2

Forward and backward messages as column vectors:

$$\mathbf{f}_{1:t+1} = \alpha \mathbf{O}_{t+1} \mathbf{T}^\top \mathbf{f}_{1:t}$$
$$\mathbf{b}_{k+1:t} = \mathbf{T} \mathbf{O}_{k+1} \mathbf{b}_{k+2:t}$$

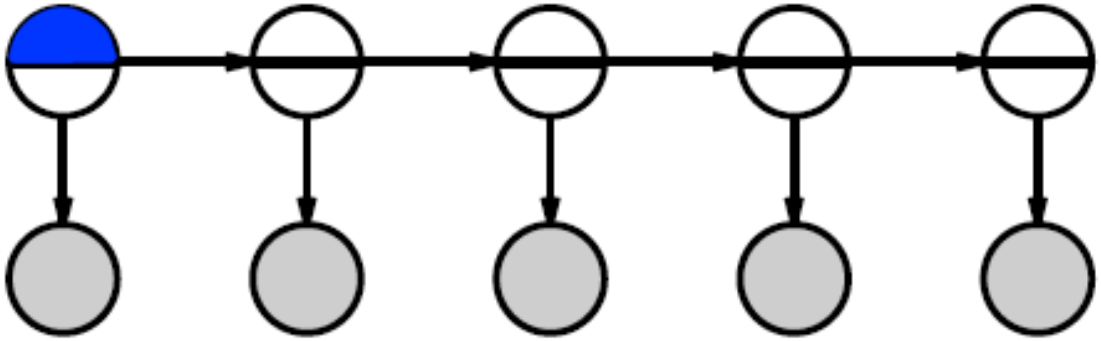
Forward-backward algorithm needs time $O(S^2 t)$ and space $O(St)$

Country Dance Algorithm

Can avoid storing all forward messages in smoothing by running forward algorithm backwards:

$$\begin{aligned} \mathbf{f}_{1:t+1} &= \alpha \mathbf{O}_{t+1} \mathbf{T}^\top \mathbf{f}_{1:t} \\ \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \alpha \mathbf{T}^\top \mathbf{f}_{1:t} \\ \alpha'(\mathbf{T}^\top)^{-1} \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \mathbf{f}_{1:t} \end{aligned}$$

Algorithm: forward pass computes $\mathbf{f}_t$, backward pass does $\mathbf{f}_i$, $\mathbf{b}_i$

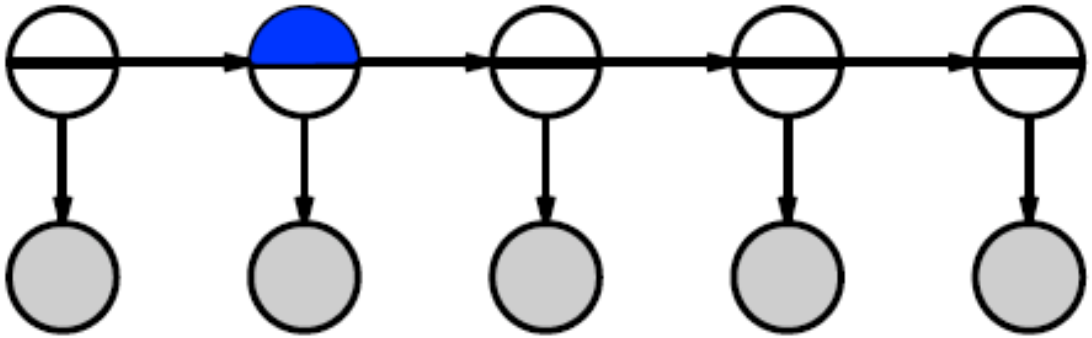


Country Dance Algorithm

Can avoid storing all forward messages in smoothing by running forward algorithm backwards:

$$\begin{aligned} \mathbf{f}_{1:t+1} &= \alpha \mathbf{O}_{t+1} \mathbf{T}^\top \mathbf{f}_{1:t} \\ \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \alpha \mathbf{T}^\top \mathbf{f}_{1:t} \\ \alpha' (\mathbf{T}^\top)^{-1} \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \mathbf{f}_{1:t} \end{aligned}$$

Algorithm: forward pass computes $\mathbf{f}_t$, backward pass does $\mathbf{f}_i$, $\mathbf{b}_i$

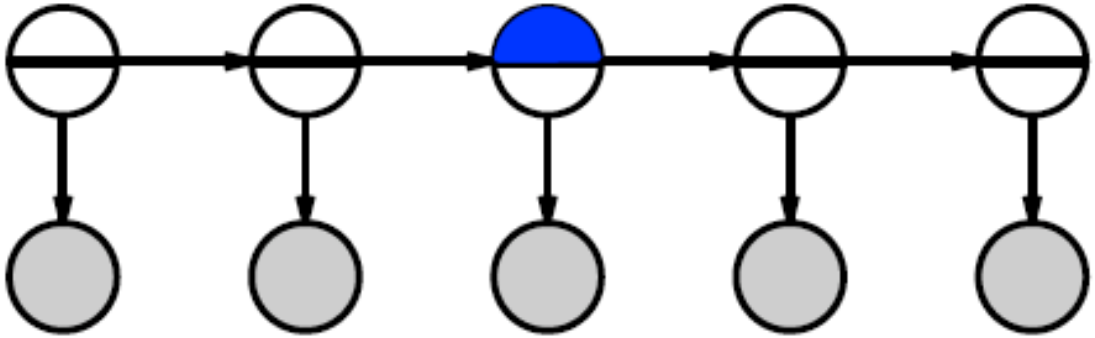


Country Dance Algorithm

Can avoid storing all forward messages in smoothing by running forward algorithm backwards:

$$\begin{aligned} \mathbf{f}_{1:t+1} &= \alpha \mathbf{O}_{t+1} \mathbf{T}^\top \mathbf{f}_{1:t} \\ \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \alpha \mathbf{T}^\top \mathbf{f}_{1:t} \\ \alpha' (\mathbf{T}^\top)^{-1} \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \mathbf{f}_{1:t} \end{aligned}$$

Algorithm: forward pass computes $\mathbf{f}_t$, backward pass does $\mathbf{f}_i$, $\mathbf{b}_i$

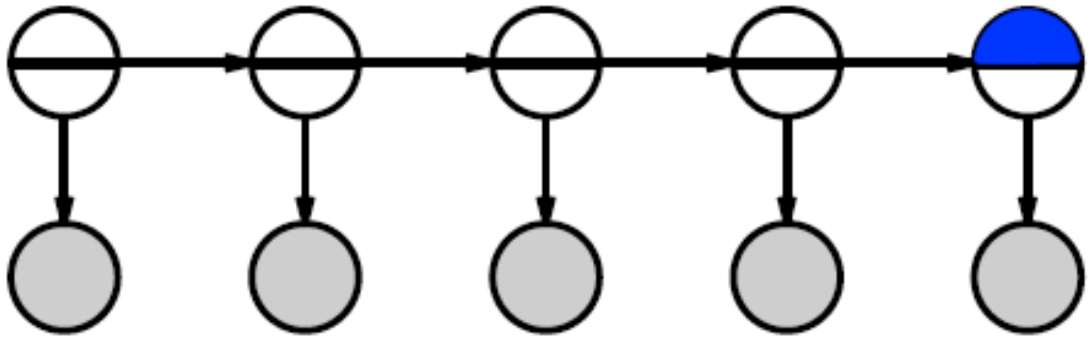


Country Dance Algorithm

Can avoid storing all forward messages in smoothing by running forward algorithm backwards:

$$\begin{aligned} \mathbf{f}_{1:t+1} &= \alpha \mathbf{O}_{t+1} \mathbf{T}^\top \mathbf{f}_{1:t} \\ \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \alpha \mathbf{T}^\top \mathbf{f}_{1:t} \\ \alpha'(\mathbf{T}^\top)^{-1} \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \mathbf{f}_{1:t} \end{aligned}$$

Algorithm: forward pass computes $\mathbf{f}_t$, backward pass does $\mathbf{f}_i$, $\mathbf{b}_i$

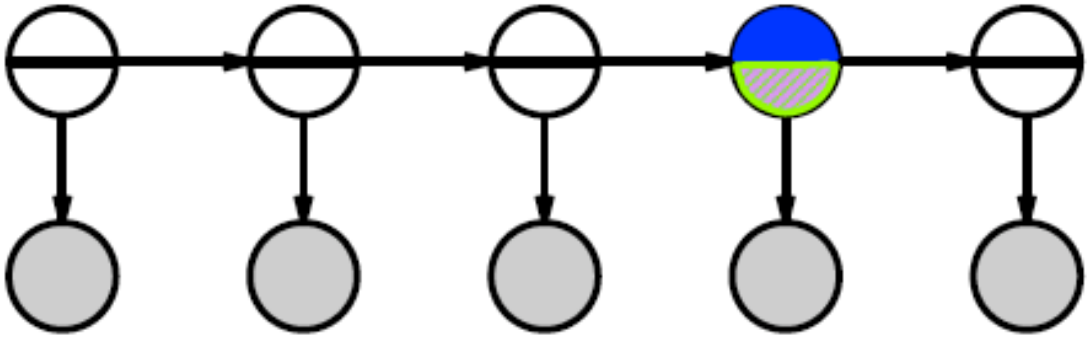


Country Dance Algorithm

Can avoid storing all forward messages in smoothing by running forward algorithm backwards:

$$\begin{aligned} \mathbf{f}_{1:t+1} &= \alpha \mathbf{O}_{t+1} \mathbf{T}^\top \mathbf{f}_{1:t} \\ \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \alpha \mathbf{T}^\top \mathbf{f}_{1:t} \\ \alpha' (\mathbf{T}^\top)^{-1} \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \mathbf{f}_{1:t} \end{aligned}$$

Algorithm: forward pass computes $\mathbf{f}_t$, backward pass does $\mathbf{f}_i$, $\mathbf{b}_i$

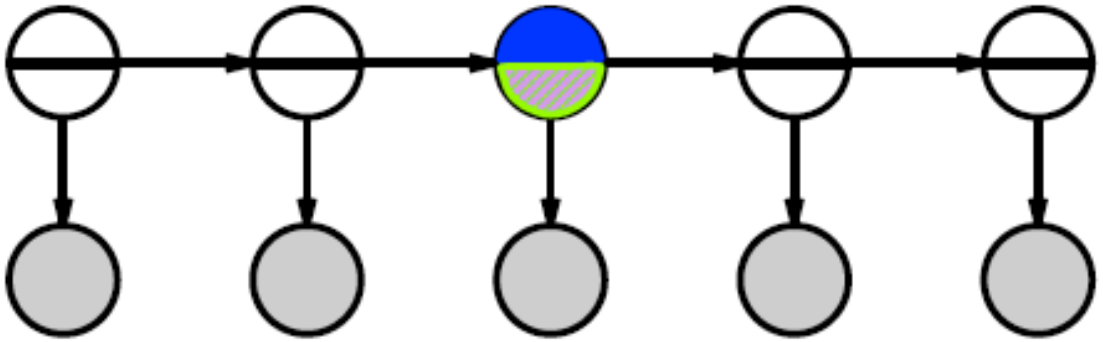


Country Dance Algorithm

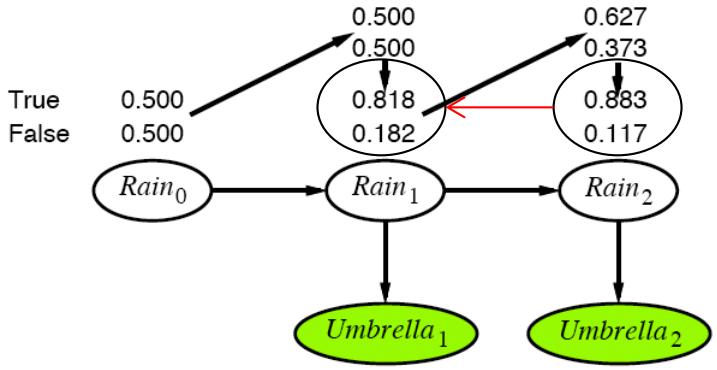
Can avoid storing all forward messages in smoothing by running forward algorithm backwards:

$$\begin{aligned} \mathbf{f}_{1:t+1} &= \alpha \mathbf{O}_{t+1} \mathbf{T}^\top \mathbf{f}_{1:t} \\ \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \alpha \mathbf{T}^\top \mathbf{f}_{1:t} \\ \alpha'(\mathbf{T}^\top)^{-1} \mathbf{O}_{t+1}^{-1} \mathbf{f}_{1:t+1} &= \mathbf{f}_{1:t} \end{aligned}$$

Algorithm: forward pass computes $\mathbf{f}_t$, backward pass does $\mathbf{f}_i$, $\mathbf{b}_i$



Example



$$\alpha'(\mathbf{T}^\top)^{-1}\mathbf{O}_{t+1}^{-1}\mathbf{f}_{1:t+1} = \mathbf{f}_{1:t}$$

$$A^{-1} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{\det(A)}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix} = \frac{1}{ad-bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

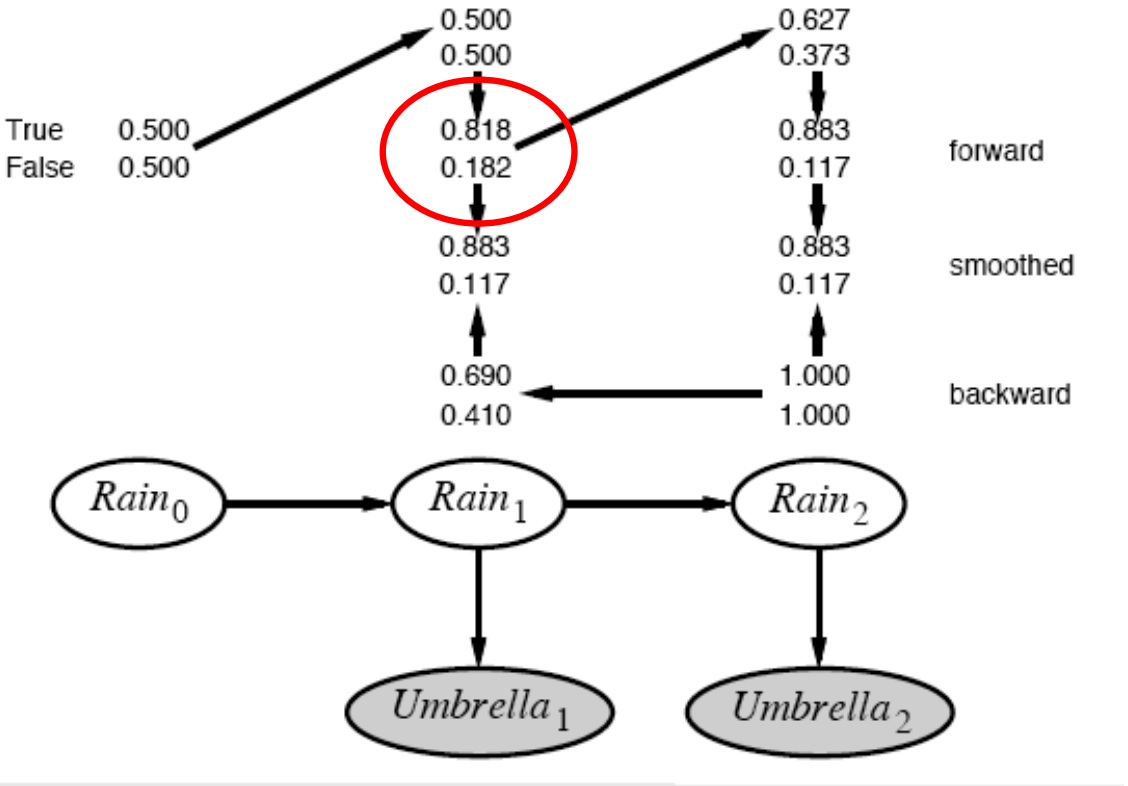
$U_2 = true$ $O_2 = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.2 \end{pmatrix}$

$O_2^{-1} = 5.5 \begin{pmatrix} 0.2 & 0 \\ 0 & 0.9 \end{pmatrix} = \begin{pmatrix} 1.1 & 0 \\ 0 & 4.95 \end{pmatrix}$

$T^T = \begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{pmatrix}$

$(T^T)^{-1} = 2.5 \begin{pmatrix} 0.7 & -0.3 \\ -0.3 & 0.7 \end{pmatrix} = \begin{pmatrix} 1.75 & -0,75 \\ -0,75 & 1.75 \end{pmatrix}$

$$\begin{pmatrix} 1,925 & -3.7125 \\ -0,825 & 8.6625 \end{pmatrix}\begin{pmatrix} 0.883 \\ 0.117 \end{pmatrix} = \begin{pmatrix} 1.265 \\ 0,285 \end{pmatrix} \quad \alpha = 0,64497 \quad \rightarrow \begin{pmatrix} 0.817 \\ 0,183 \end{pmatrix}$$



$$\begin{pmatrix} 0.817 \\ 0{,}183 \end{pmatrix}$$

State View: Hidden Markov models

Set of states: $\{s_1, s_2, \ldots, s_N\}$

Process moves from one state to another generating a sequence of states : $s_{i1}, s_{i2}, \ldots, s_{ik}, \ldots$
Markov chain property: probability of each subsequent state depends only on what was the previous state:

$$P(s_{ik} \mid s_{i1}, s_{i2}, \ldots, s_{ik-1}) = P(s_{ik} \mid s_{ik-1})$$

States are not visible, but each state randomly generates one of M observations (or visible states)

$$\{v_1, v_2, \ldots, v_M\}$$

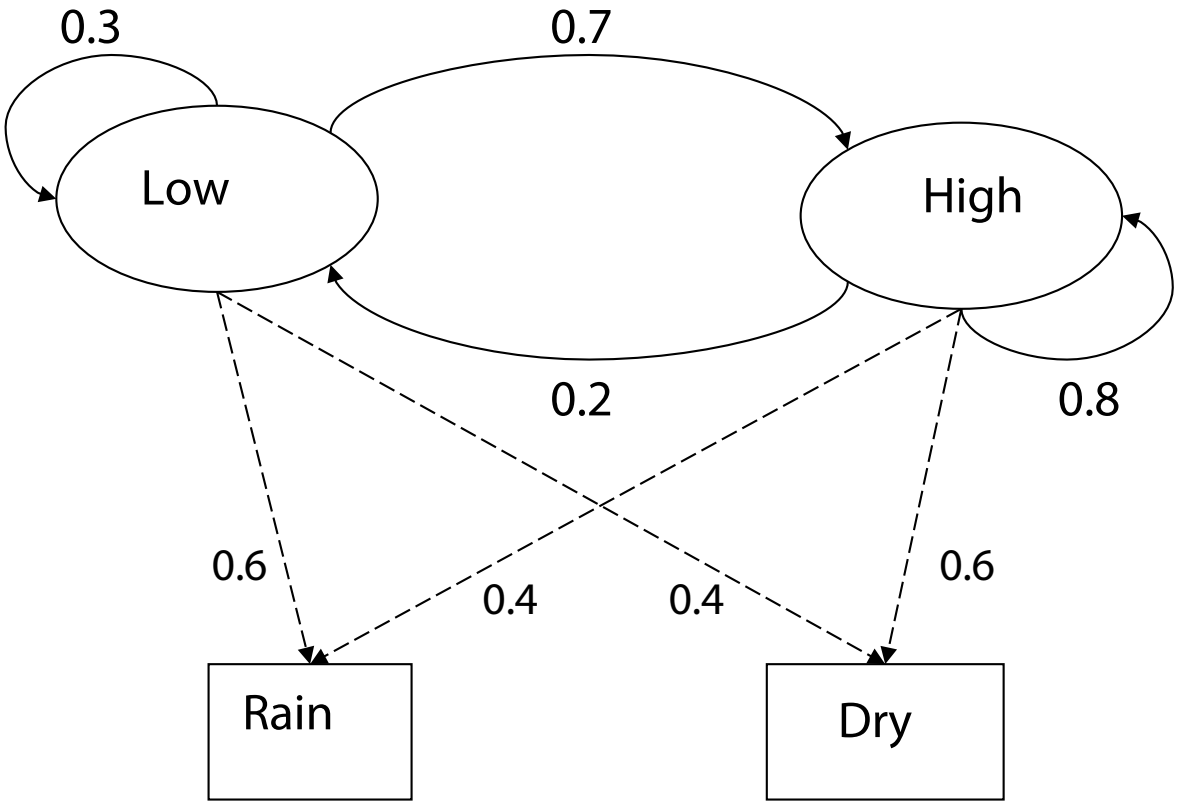
State View: Hidden Markov models

To define a hidden Markov model, the following probabilities have to be specified:

- Matrix of transition probabilities $A=(a_{ij})$, $a_{ij}= P(s_j \mid s_i)$,
- Matrix of observation probabilities $B=(b_i (v_m))$, $b_i(v_m) = P(v_m \mid s_i)$ and a
- Vector of initial probabilities $\pi=(\pi_i)$, $\pi_i = P(s_i)$.

Model is represented by $M=(A, B, \pi)$.

Example of Hidden Markov Model



State View: Learning problem (1)

Given some training observation sequences $O=o_1 o_2 \ldots o_K$ and general structure of HMM (numbers of hidden and visible states), determine HMM parameters $M=(A, B, \pi)$ that best fit training data, that is maximizes $P(O \mid M)$.

State View: Learning problem (2)

If training data has information about sequence of hidden states, then use maximum likelihood estimation of parameters:

$$a_{ij} = P(s_j \mid s_i) = \frac{\text{Number of transitions from state } S_i \text{ to state } S_j}{\text{Number of transitions out of state } S_i}$$

$$b_i(v_m) = P(v_m \mid s_i) = \frac{\text{Number of times observation } V_m \text{ occurs in state } S_i}{\text{Number of times in state } S_i}$$

Otherwise: Use iterative expectation-maximization algorithm to find local maximum of $P(O \mid M)$: **Baum-Welch Algorithm.**

Baum-Welch algorithm

General idea:

$$a_{ij} = P(s_j \mid s_i) = \frac{\text{Expected number of transitions from state } S_i \text{ to state } S_j}{\text{Expected number of transitions out of state } S_i}$$

$$b_i(v_m) = P(v_m \mid s_i) = \frac{\text{Expected number of times observation } V_m \text{ occurs in state } S_i}{\text{Expected number of times in state } S_i}$$

$$\pi_i = P(s_i) = \text{Expected frequency in state } S_i \text{ at time } k=1.$$

Baum-Welch algorithm: Expectation step(1)

Define variable $\xi_k(i,j)$ as the probability of being in state S_i at time k and in state S_j at time $k+1$, given the observation sequence $O_1 O_2 \ldots O_T$ with $k < T$

$$\xi_k(i,j) = P(q_k = s_i , q_{k+1} = s_j \mid o_1 o_2 \ldots o_T)$$

$$\xi_k(i,j) = \frac{P(q_k = s_i , q_{k+1} = s_j , o_1 o_2 \ldots o_T)}{P(o_1 o_2 \ldots o_T)} =$$

$$\frac{P(q_k = s_i , o_1 o_2 \ldots o_k)\, a_{ij}\, b_j(o_{k+1})\, P(o_{k+2} \ldots o_T \mid q_{k+1} = s_j)}{P(o_1 o_2 \ldots o_T)} =$$

$$\alpha\, \text{forward}_k(i)\, a_{ij}\, b_j(o_{k+1})\, \text{backward}_{k+1}(j)$$

Baum-Welch algorithm: Expectation step(2)

Define variable $\gamma_k(i)$ as the probability of being in state S_i at time k, given the observation sequence $O_1 O_2 \ldots O_T$.

$$\gamma_k(i) = P(q_k = s_i \mid O_1 O_2 \ldots O_T)$$

$$\gamma_k(i) = \frac{P(q_k = s_i, O_1 O_2 \ldots O_T)}{P(O_1 O_2 \ldots O_T)} =$$

$$\alpha \; \text{forward}_k(i) \; \text{backward}_k(i)$$

Baum-Welch algorithm: Expectation step(3)

We calculated $\xi_k(i,j) = P(q_k = s_i , q_{k+1} = s_j \mid o_1 o_2 \ldots o_T)$

and $\gamma_k(i) = P(q_k = s_i \mid o_1 o_2 \ldots o_T)$

Expected number of transitions from state s_i to state s_j =

$= \sum_k \xi_k(i,j)$

Expected number of transitions out of state s_i $= \sum_k \gamma_k(i)$

Expected number of times observation v_m occurs in state s_i =

$= \sum_k \gamma_k(i)$, k is such that $o_k = v_m$

Expected frequency in state s_i at time k=1 : $\gamma_1(i)$.

Baum-Welch algorithm: Maximization step

$$a_{ij} = \frac{\text{Expected number of transitions from state } S_j \text{ to state } S_i}{\text{Expected number of transitions out of state } S_j} = \frac{\sum_k \xi_k(i,j)}{\sum_k \gamma_k(i)}$$

$$b_i(v_m) = \frac{\text{Expected number of times observation } v_m \text{ occurs in state } S_i}{\text{Expected number of times in state } S_i} = \frac{\sum_k \xi_k(i,j)}{\sum_{k,o_k = v_m} \gamma_k(i)}$$

$$\pi_i = (\text{Expected frequency in state } S_i \text{ at time } k=1) = \gamma_1(i).$$

DBN – Special Cases

- Hidden Markov Model (HMMs):

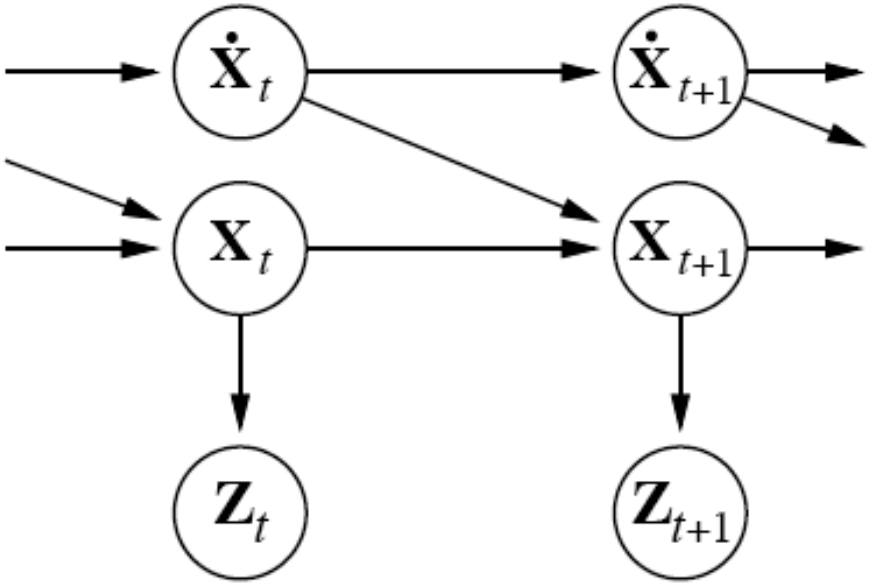
  Temporal probabilistic model in which the state of the process
  is described by a single discrete random variable. (The simplest kind of DBN)

- Kalman Filter Models (KFMs):

  Estimate the state of a physical system from noisy observations over time.
  Also known as linear dynamical systems (LDSs).

Kalman Filters

Modelling systems described by a set of continuous variables,
e.g., tracking a bird flying—$\mathbf{X}_t = X, Y, Z, \dot{X}, \dot{Y}, \dot{Z}$.
Airplanes, robots, ecosystems, economies, chemical plants, planets, . . .



Gaussian prior, linear Gaussian transition model and sensor model

Updating Gaussian Distributions

Prediction step: if $\mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$ is Gaussian, then prediction

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}) = \int_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{e}_{1:t})\, d\mathbf{x}_t$$

is Gaussian. If $\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t})$ is Gaussian, then the updated distribution

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) = \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t})$$

is Gaussian

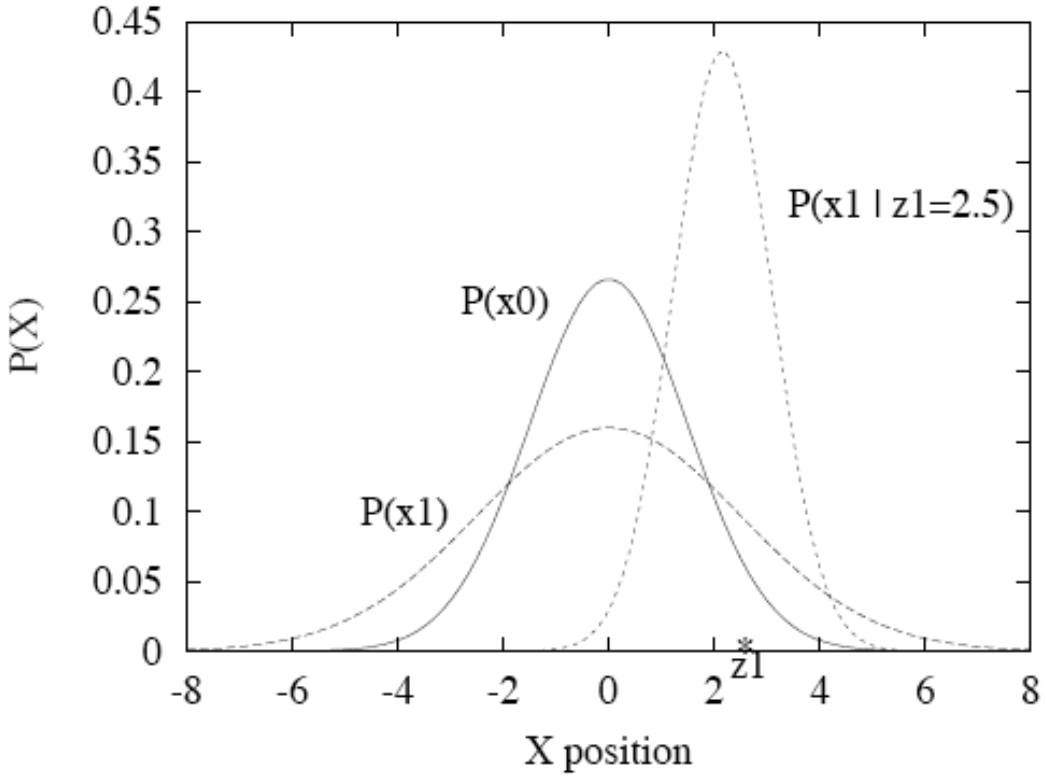
Hence $\mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$ is multivariate Gaussian $N(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ for all t

General (nonlinear, non-Gaussian) process: description of posterior grows **unboundedly** as $t \to \infty$

Simple 1-D Example

Gaussian random walk on X–axis, s.d. σ_x, sensor s.d. σ_z

$$\mu_{t+1} = \frac{(\sigma_t^2 + \sigma_x^2) z_{t+1} + \sigma_z^2 \mu_t}{\sigma_t^2 + \sigma_x^2 + \sigma_z^2} \qquad \sigma_{t+1}^2 = \frac{(\sigma_t^2 + \sigma_x^2)\sigma_z^2}{\sigma_t^2 + \sigma_x^2 + \sigma_z^2}$$



z1: first observation

s.d.= standard deviation

General Kalman Update

Transition and sensor models:

$$P(\mathbf{x}_{t+1}|\mathbf{x}_t) = N(\mathbf{F}\mathbf{x}_t, \boldsymbol{\Sigma}_x)(\mathbf{x}_{t+1})$$
$$P(\mathbf{z}_t|\mathbf{x}_t) = N(\mathbf{H}\mathbf{x}_t, \boldsymbol{\Sigma}_z)(\mathbf{z}_t)$$

Left for your studies

$\mathbf{F}$ is the matrix for the transition; $\boldsymbol{\Sigma}_x$ the transition noise covariance
$\mathbf{H}$ is the matrix for the sensors; $\boldsymbol{\Sigma}_z$ the sensor noise covariance

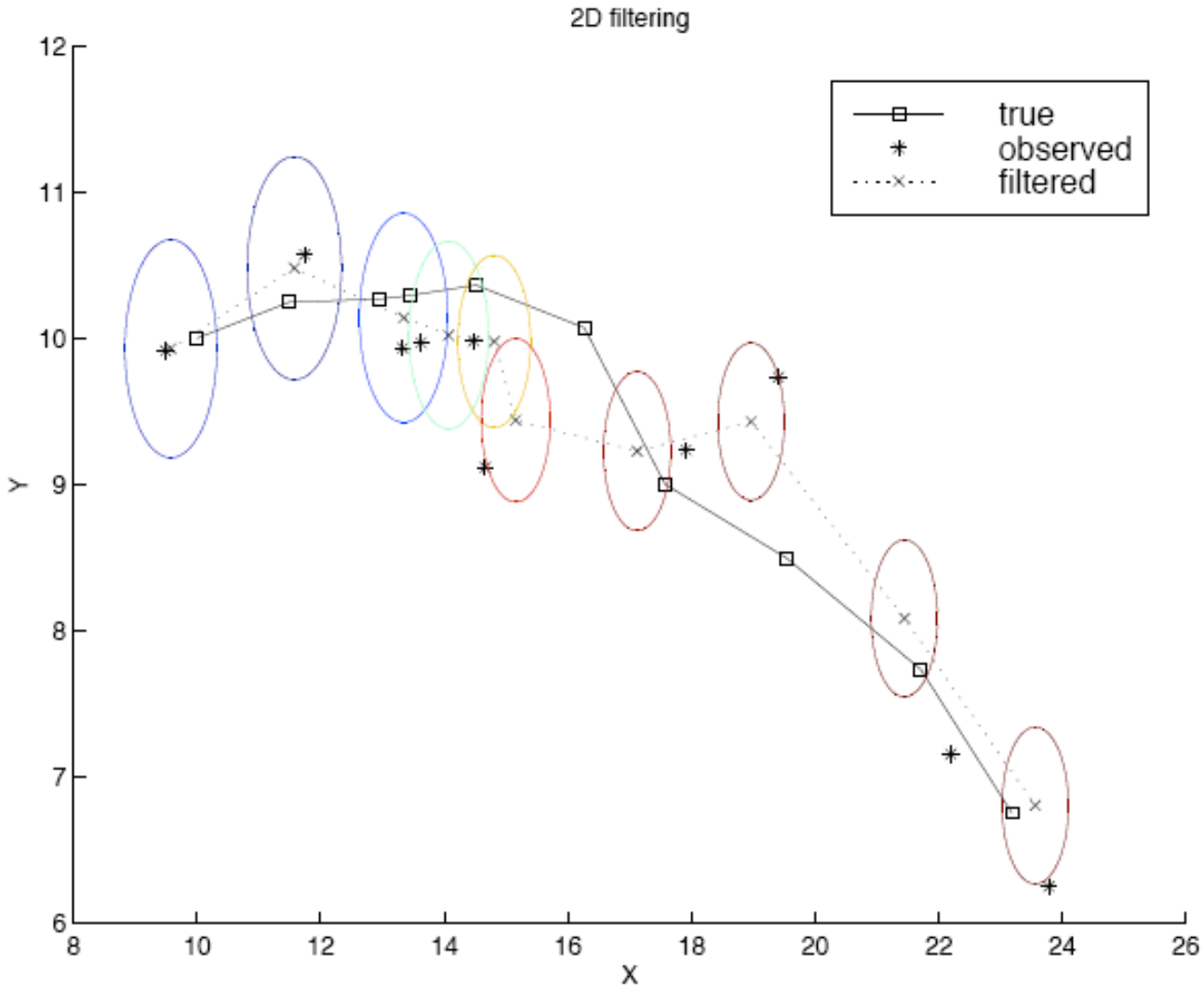
Filter computes the following update:

$$\boldsymbol{\mu}_{t+1} = \mathbf{F}\boldsymbol{\mu}_t + \mathbf{K}_{t+1}(\mathbf{z}_{t+1} - \mathbf{H}\mathbf{F}\boldsymbol{\mu}_t)$$
$$\boldsymbol{\Sigma}_{t+1} = (\mathbf{I} - \mathbf{K}_{t+1})(\mathbf{F}\boldsymbol{\Sigma}_t\mathbf{F}^\top + \boldsymbol{\Sigma}_x)$$

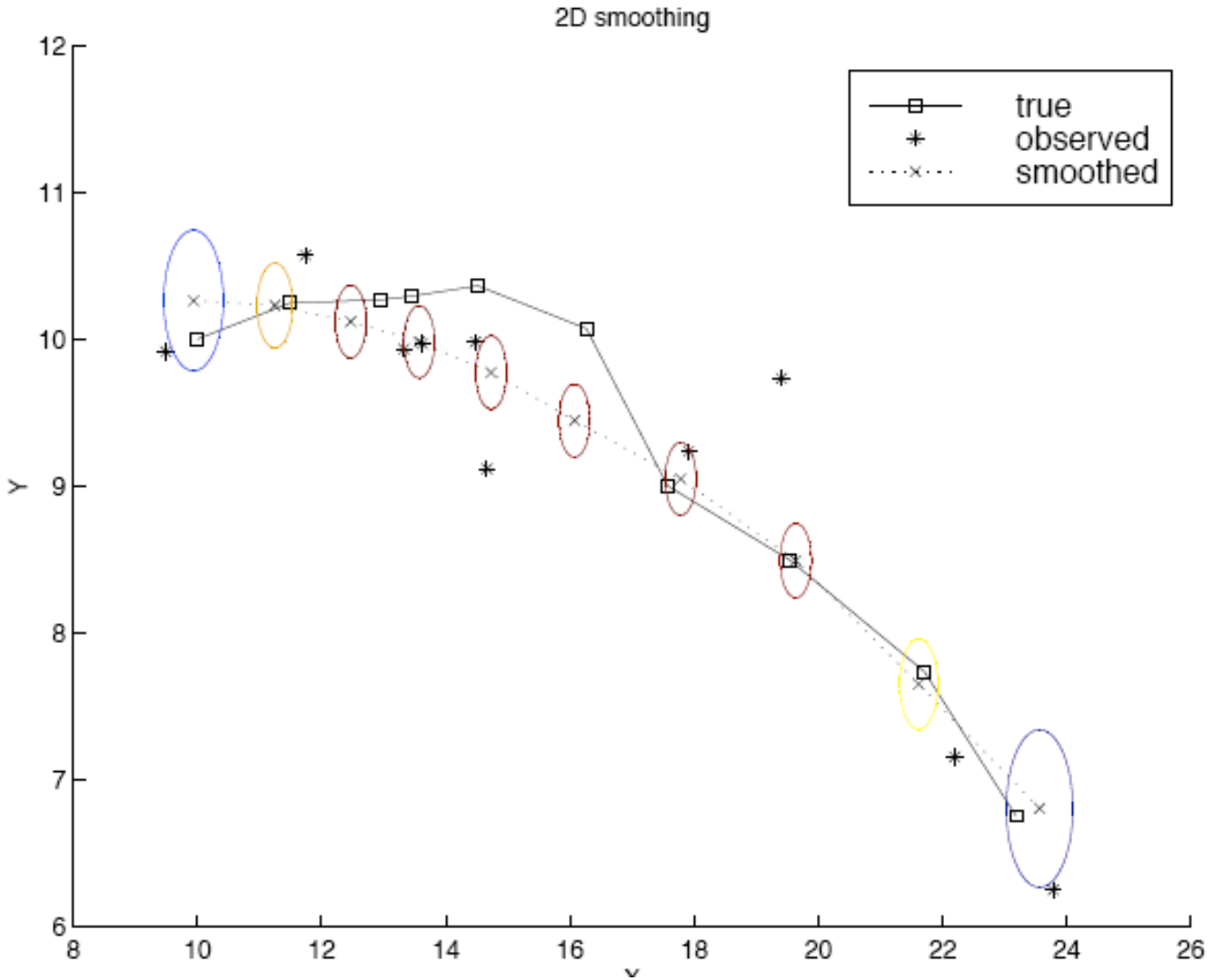
where $\mathbf{K}_{t+1} = (\mathbf{F}\boldsymbol{\Sigma}_t\mathbf{F}^\top + \boldsymbol{\Sigma}_x)\mathbf{H}^\top(\mathbf{H}(\mathbf{F}\boldsymbol{\Sigma}_t\mathbf{F}^\top + \boldsymbol{\Sigma}_x)\mathbf{H}^\top + \boldsymbol{\Sigma}_z)^{-1}$
is the Kalman gain matrix

$\boldsymbol{\Sigma}_t$ and $\mathbf{K}_t$ are independent of observation sequence, so compute offline

2-D Tracking: Filtering



2-D Tracking: Smoothing

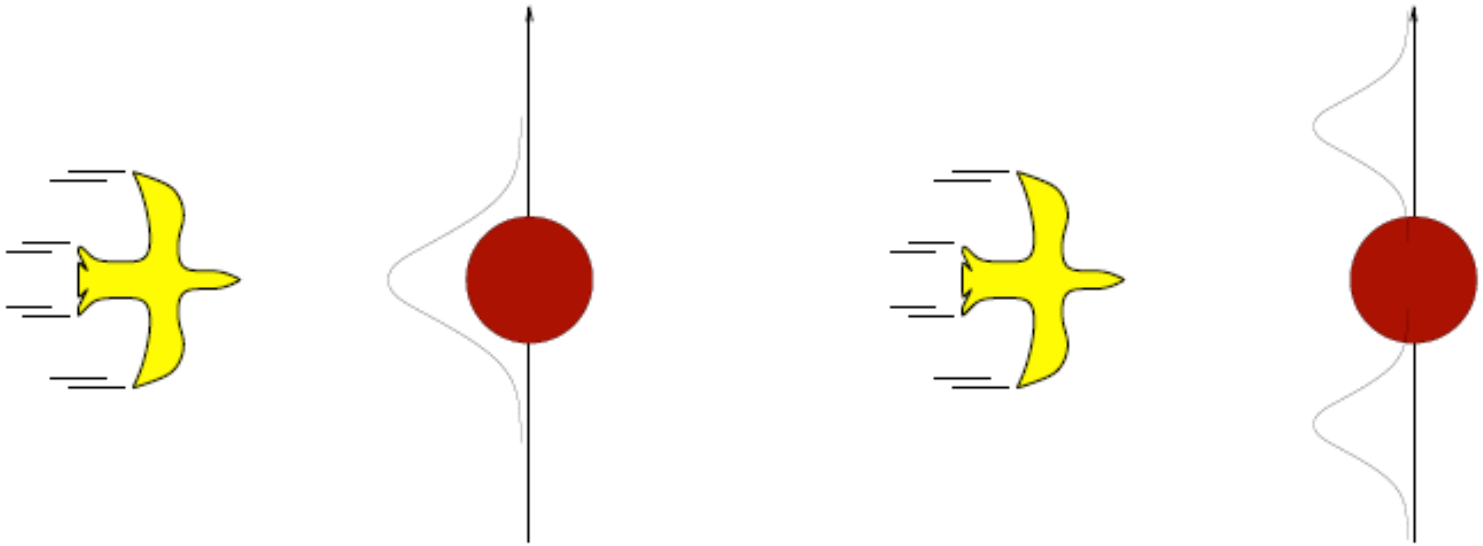


Where it breaks

Cannot be applied if the transition model is nonlinear

Extended Kalman Filter models transition as **locally linear** around $\mathbf{x}_t = \boldsymbol{\mu}_t$
Fails if systems is locally unsmooth

Standard solution: switching Kalman filter



Keeping track of many objects: Identity uncertainty

Web-Mining Agents

Probabilistic Reasoning over Sequential Structures

Prof. Dr. Ralf Möller

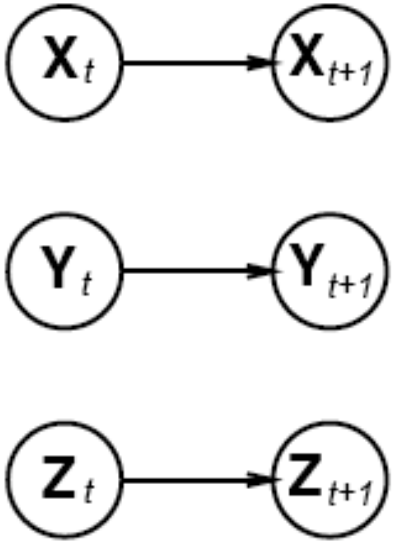
Universität zu Lübeck

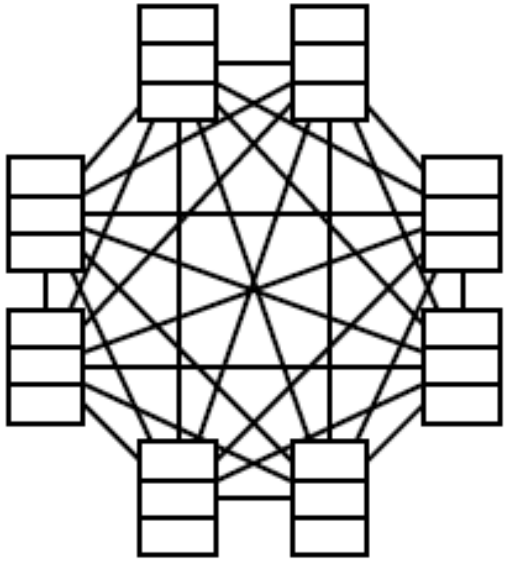
Institut für Informationssysteme

Tanya Braun (Übungen)

DBNs vs. HMMs

Every HMM is a single-variable DBN; every discrete DBN is an HMM





Consider the **transition model**

Sparse dependencies $\Rightarrow$ exponentially fewer parameters;

e.g., 20 state variables, three parents each

DBN has $20 \times 2^3 = 160$ parameters, HMM has $2^{20} \times 2^{20} \approx 10^{12}$

Learning (1)

- The techniques for learning DBN are mostly straightforward extensions of the techniques for learning BNs
- Parameter learning
  - The transition model $P(X_t \mid X_{t-1})$ / The observation model $P(Y_t \mid X_t)$
  - Offline learning
    - Parameters must be tied across time-slices
    - The initial state of the dynamic system can be learned independently of the transition matrix
  - Online learning
    - Add the parameters to the state space and then do online inference (filtering)
  - The usual criterion is maximum-likelihood(ML)

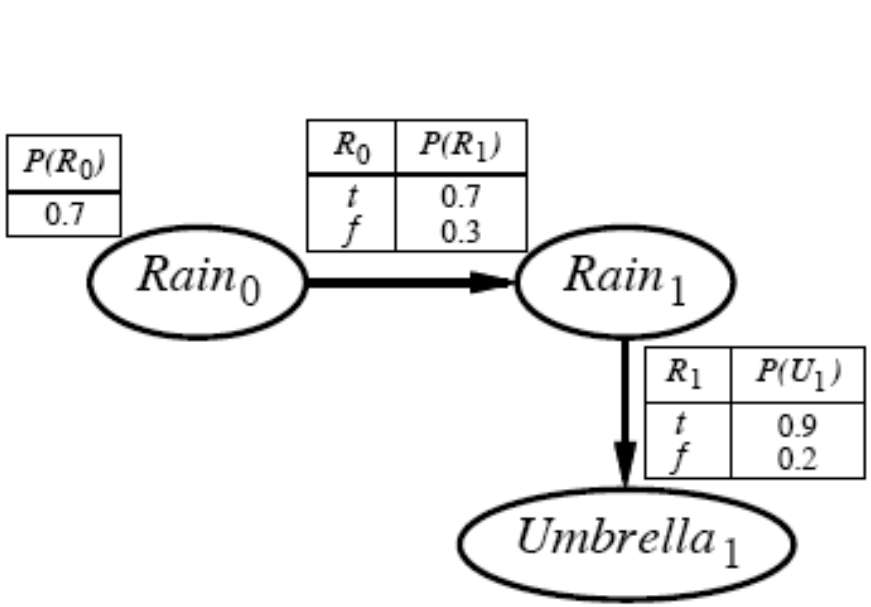
- The goal of parameter learning is to compute
  - $\theta^*_{ML} = \text{argmax}_\theta P(Y \mid \theta) = \text{argmax}_\theta \log P(Y \mid \theta)$
  - $\theta^*_{MAP} = \text{argmax}_\theta \log P(Y \mid \theta) + \log P(\theta)$
  - Two standard approaches: gradient ascent and EM(Expectation Maximization)

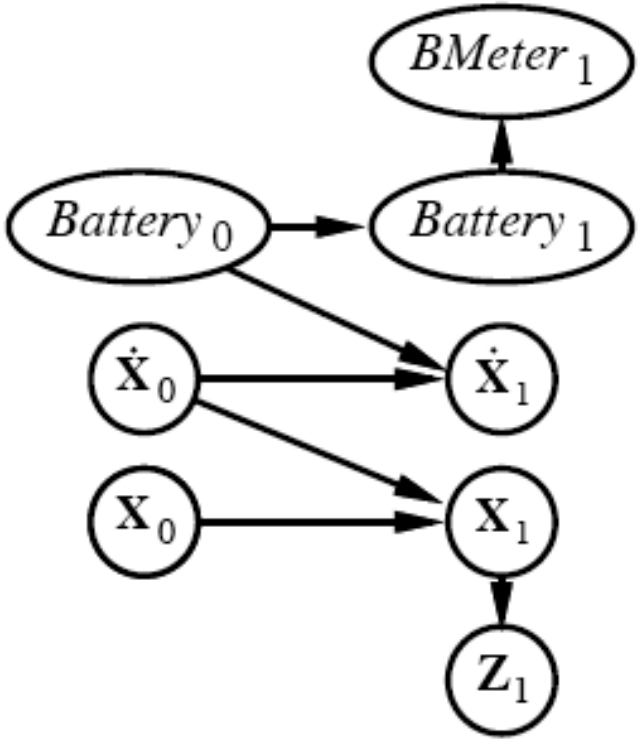
Learning (2)

- Structure learning
  - Intra-slice connectivity: Structural EM
  - Inter-slice connectivity:
    For each node in slice t, we must choose its parents from slice t-1
  - Given structure is unrolled to a certain extent,
    the inter-slice connectivity is identical for all pairs of slices:
    - Constraints on Structural EM

Constructing Dynamic Bayesian Networks

$\mathbf{X}_t$, $\mathbf{E}_t$ contain arbitrarily many variables in a replicated Bayes net



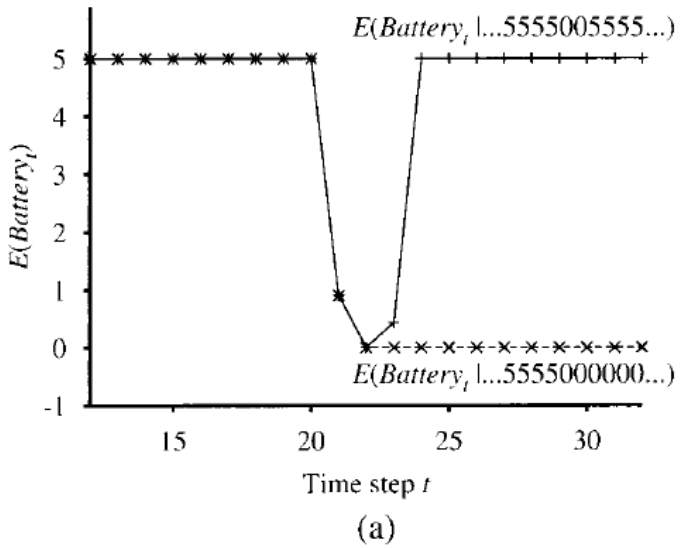


Battery-powered robot on x-y plane

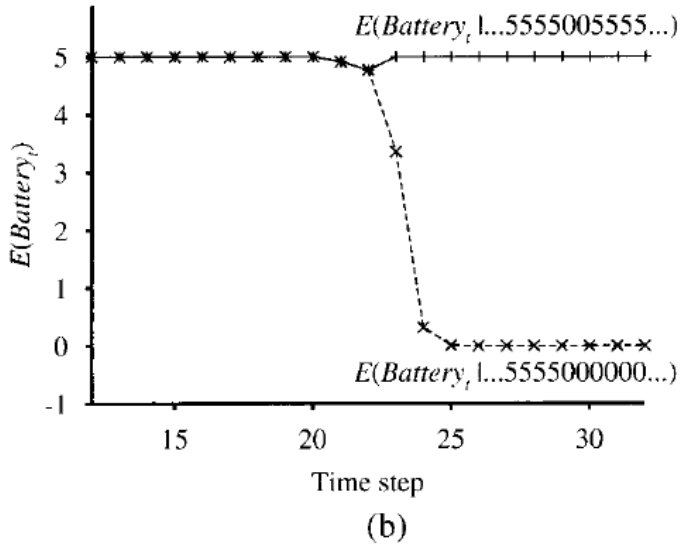
$\mathbf{X}_t = (X_t, Y_t)$, $\mathbf{Z}_t$ = measurements

DBNs transient failure

For simplicity we assume that $BMeter_t$ and $Battery_t$ are taken from 0..5.



(a)



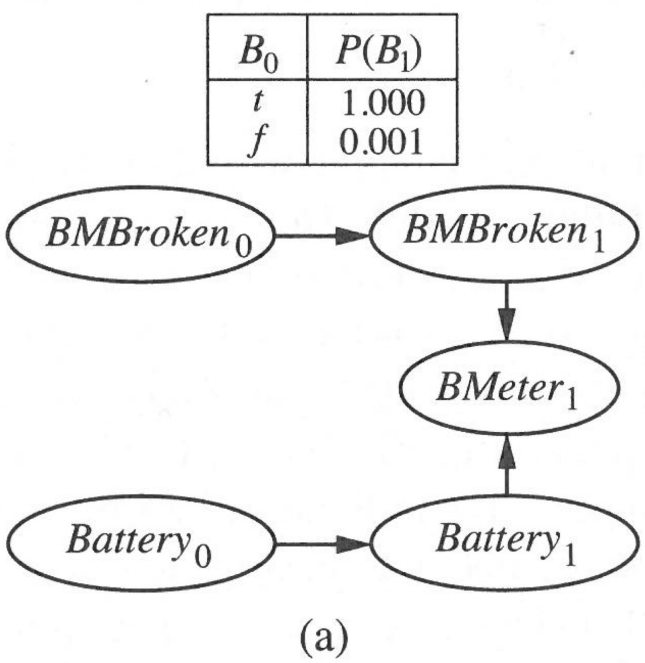
(b)

Generic gaussian error model produces “overreaction“

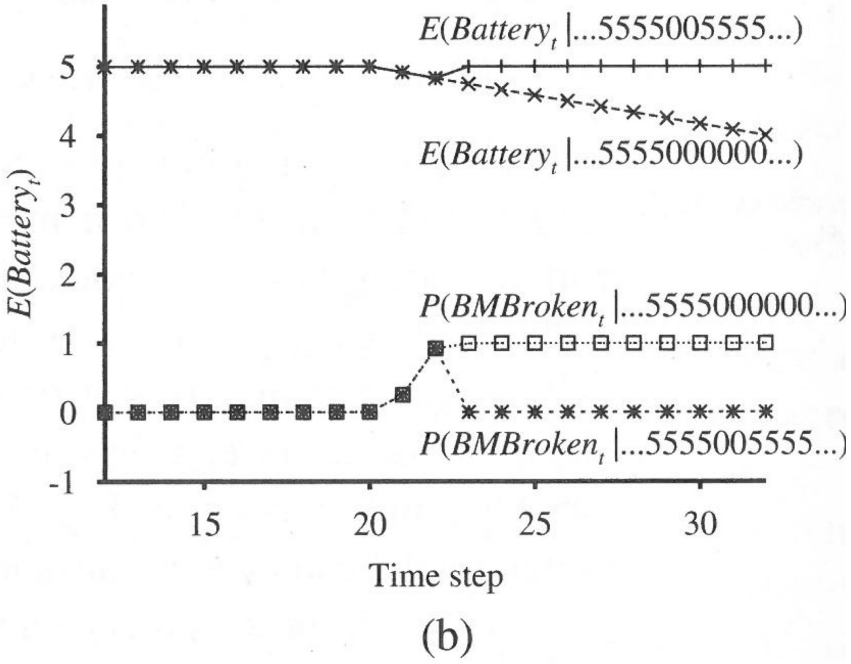
Explicit transient failure model required:

$$P(BMeter_t = 0 | Battery_t = 5) = 0.03$$

DBNs persistent failure



(a)



(b)

Additional variable required: $BMBroken_t$

Upper curve: transient failure
with different observation sequences

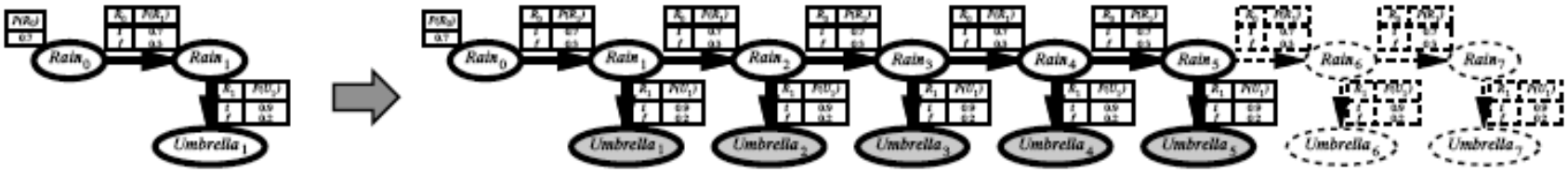
Lower curve: persistent failure
with two different observation sequences

Learning DBN pattern structures?

- Difficult
- Need “deep” domain knowledge

Recap: Exact Inference in DBNs

Naive method: unroll the network and run any exact algorithm for the whole BN



Problem: inference cost for each update grows with t

Rollup filtering: add slice $t+1$, "sum out" slice t using variable elimination

Largest factor is $O(d^{n+1})$, update cost $O(d^{n+2})$
(cf. HMM update cost $O(d^{2n})$)

d = possible values for variables
n = number of states

Example:
20 state variables with 4 values each
means 4^{20+1} =4.398.046.511.104

Employ forward chaining (constant per time update but exponential in the number of variables per state)

Inference: Algorithms

- Exact Inference algorithms
  - Forwards-backwards smoothing algorithm (on any discrete-state DBN)
  - Kalman filtering and smoothing (for continuous variables)
  - The Frontier Algorithm (sweep a Markov blanket, the frontier set F, across the DBN, first forwards and then backwards)
  - The Interface Algorithm (use only the set of nodes with outgoing arcs to the next time slice to d-separate the past from the future)
- Approximate algorithms:
  - The Boyen-Koller (BK) algorithm (approximate the joint distribution over the interface as a product of marginals)
  - Factored Frontier (FF) Algorithm / Loopy propagation algorithm (LBP)
  - Stochastic sampling algorithm:
    - Importance sampling or MCMC (offline inference)
    - Particle filtering (PF) (online)

Approximate inference in DBNs

Central idea:

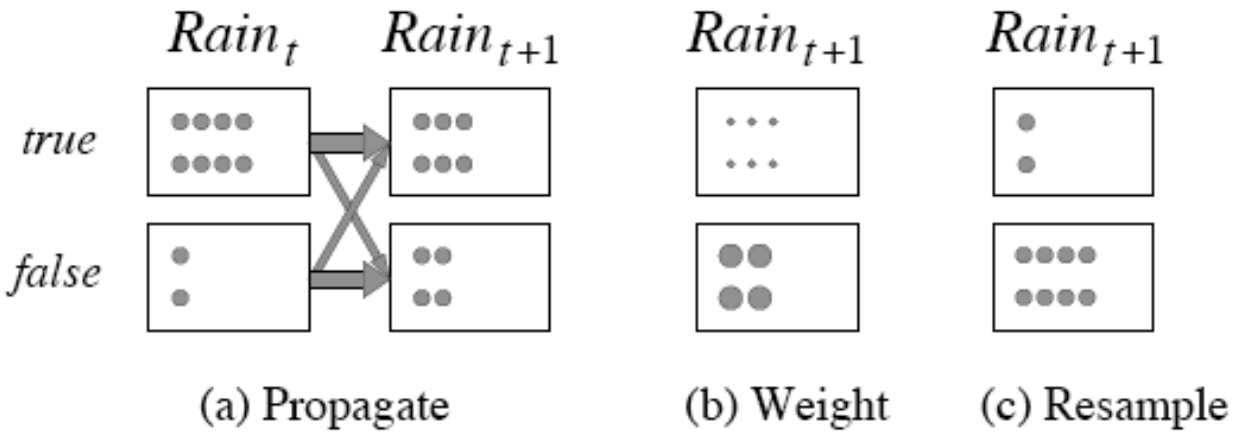
- Create N initial-state examples (from prior dist $P(\mathbf{X}_0)$)
- Based on the transition model, each sample is propagated forward by sampling the next state value $\mathbf{x}_{t+1}$ given the current value $\mathbf{x}_t$
- Each sample is weighted by the likelihood it assigns to the new evidence $P(\mathbf{e}_{t+1} \mid \mathbf{x}_{t+1})$
- Resample to generate new population of N samples
- Select new sample based on its weight

Samples are called particles (→ Particle Filtering)

Particle Filtering

Basic idea: ensure that the population of samples ("particles") tracks the high-likelihood regions of the state-space

Replicate particles proportional to likelihood for $\mathbf{e}_t$

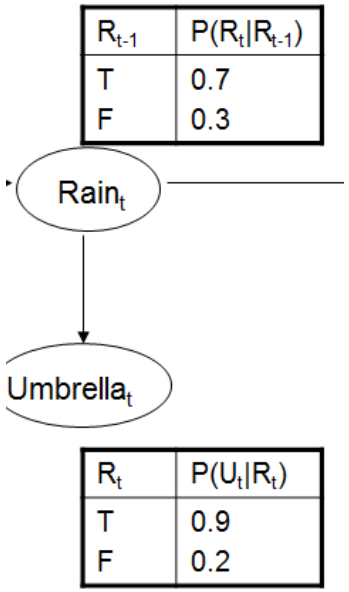


Widely used for tracking nonlinear systems, esp. in vision

Also used for simultaneous localization and mapping in mobile robots

10^5-dimensional state space

Example



$N(r_{t+1}|e) = \Sigma_{x_t} P(x_{t+1}|x_t)\, N(x_t|e)$

For rain = 0.7*8+0.3*2= 6.2 => 6

For not rain = 0.3 *8 + 0.7*2= 3.8 => 4

Suppose no umbrella for t+1

total weight(rain particles) = 0.1 * 6= 0.6

total weight(not rain) = 0.8 * 4= 3.2

Normalized =<0.17, 0.83>

Particle Filtering

function PARTICLE-FILTERING(**e**, N, dbn) **returns** a set of samples for the next time step
  inputs: **e**, the new incoming evidence
    N, the number of samples to be maintained
    dbn, a DBN with prior $\mathbf{P}(\mathbf{X}_0)$, transition model $\mathbf{P}(\mathbf{X}_1|\mathbf{X}_0)$, sensor model $\mathbf{P}(\mathbf{E}_1|\mathbf{X}_1)$
  persistent: S, a vector of samples of size N, initially generated from $\mathbf{P}(\mathbf{X}_0)$
  local variables: W, a vector of weights of size N

  for $i = 1$ to N **do**
    $S[i] \leftarrow$ sample from $\mathbf{P}(\mathbf{X}_1 \mid \mathbf{X}_0 = S[i])$ /* step 1 */
    $W[i] \leftarrow \mathbf{P}(\mathbf{e} \mid \mathbf{X}_1 = S[i])$ /* step 2 */
  $S \leftarrow$ WEIGHTED-SAMPLE-WITH-REPLACEMENT(N, S, W) /* step 3 */
  return S

Particle Filtering (cntd.)

Assume consistent at time t: $N(\mathbf{x}_t|\mathbf{e}_{1:t})/N = P(\mathbf{x}_t|\mathbf{e}_{1:t})$

Propagate forward: populations of $\mathbf{x}_{t+1}$ are

$$N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t}) = \sum_{\mathbf{x}_t} P(\mathbf{x}_{t+1}|\mathbf{x}_t) N(\mathbf{x}_t|\mathbf{e}_{1:t})$$

Weight samples by their likelihood for $\mathbf{e}_{t+1}$:

$$W(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1}) = P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1}) N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t})$$

Resample to obtain populations proportional to W:

$$\begin{aligned} N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1})/N &= \alpha W(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1}) = \alpha P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1}) N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t}) \\ &= \alpha P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1}) \sum_{\mathbf{x}_t} P(\mathbf{x}_{t+1}|\mathbf{x}_t) N(\mathbf{x}_t|\mathbf{e}_{1:t}) \\ &= \alpha' P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1}) \sum_{\mathbf{x}_t} P(\mathbf{x}_{t+1}|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{e}_{1:t}) \\ &= P(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1}) \end{aligned}$$

Summary

Temporal models use state and sensor variables replicated over time

Markov assumptions and stationarity assumption, so we need
- transition model $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1})$
- sensor model $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)$

Tasks are filtering, prediction, smoothing, most likely sequence;
all done recursively with constant cost per time step

Hidden Markov models have a single discrete state variable; used for speech recognition

Kalman filters allow n state variables, linear Gaussian, $O(n^3)$ update

Dynamic Bayes nets subsume HMMs, Kalman filters; exact update intractable

Particle filtering is a good approximate filtering algorithm for DBNs

HMM-LDA

- In traditional topic modeling, such as LDA, we remove most syntactic words (e.g., stopwords) since we are only interested in *meaning*.

- In doing so, we discard much of the structure, and all of the order the original author intended.

- In topic modeling, we are concerned long-range topic dependencies rather document structure.

Thomas L. Griffiths, Mark Steyvers, David M. Blei, and Joshua B. Tenenbaum. Integrating topics and syntax. In *Proc. of* NIPS'04, pp. 537-544. **2004**.

Introduction

- HMMs are useful for segmenting documents into different types of words, regardless of meaning.
- For example, all nouns will be grouped together because they play the same role in different passages/documents.
- Syntactic dependencies last at most for a sentence.
- The standardized nature of grammar means that it stays fairly constant across different contexts.

Combining syntax and semantics 1

- All words (both syntactic and semantic) exhibit short range dependencies.
- Only content words exhibit long range semantic dependencies.
- This leads to the HMM-LDA.
- HMM-LDA is a composite model, in which an HMM decides the parts of speech, and a topic model (LDA) extracts topics only those words which are deemed semantic.

Generative Process 1

Definitions

Words $\mathbf{w} = \{w_1, \ldots, w_n\}$ form document d where each word w_i is one of **W** words

Topic assignments $\mathbf{z} = \{z_1, \ldots z_n\}$ for each word, where each z_i taking one of **T** topics

Class assignments $\mathbf{c} = \{c_1, \ldots, c_n\}$ for each word, where each c_i taking one of **C** word classes

$\theta^{(d)}$ Multinomial distribution over topics for document *d*

$\phi^{(z)}$ Multinomial distribution over ***semantic*** words for topic indicated by *z*.

$\phi^{(c)}$ Multinomial distribution over ***non-semantic*** words for class indicated by *class c*.

$\pi^{(c_{i-1})}$ Transition probability from c_{i-1} to c_i

Generative Process 2

$\theta^{(d)} \sim \text{Dirichlet}(\alpha)$

$\phi^{(z)} \sim \text{Dirichlet}(\beta)$

$\pi^{(c)} \sim \text{Dirichlet}(\gamma)$ Where $\pi^{(c)}$ is the row of the transition matrix indicated by c.

$\phi^{(c)} \sim \text{Dirichlet}(\delta)$

For document *d*

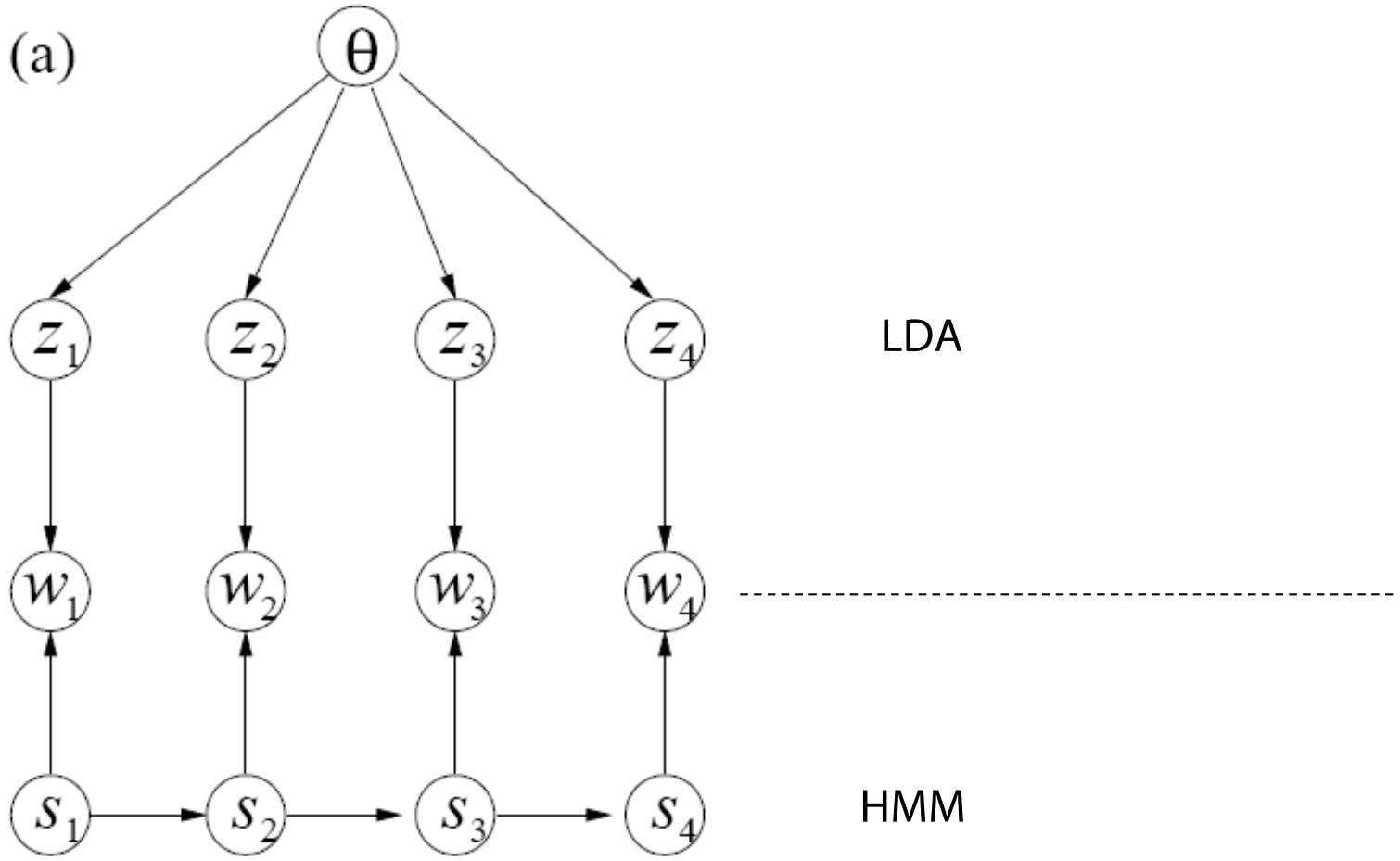
Draw topic distribution →

1. Sample $\theta^{(d)}$ from a Dirichlet(α) prior
2. For each word w_i in document d
   - (a) Draw z_i from $\theta^{(d)}$ ← Draw a topic for word i
   - (b) Draw c_i from $\pi^{(c_{i-1})}$ ← Draw a class for word i from transition matrix
   - (c) If $c_i = 1$, then draw w_i from $\phi^{(z_i)}$, else draw w_i from $\phi^{(c_i)}$

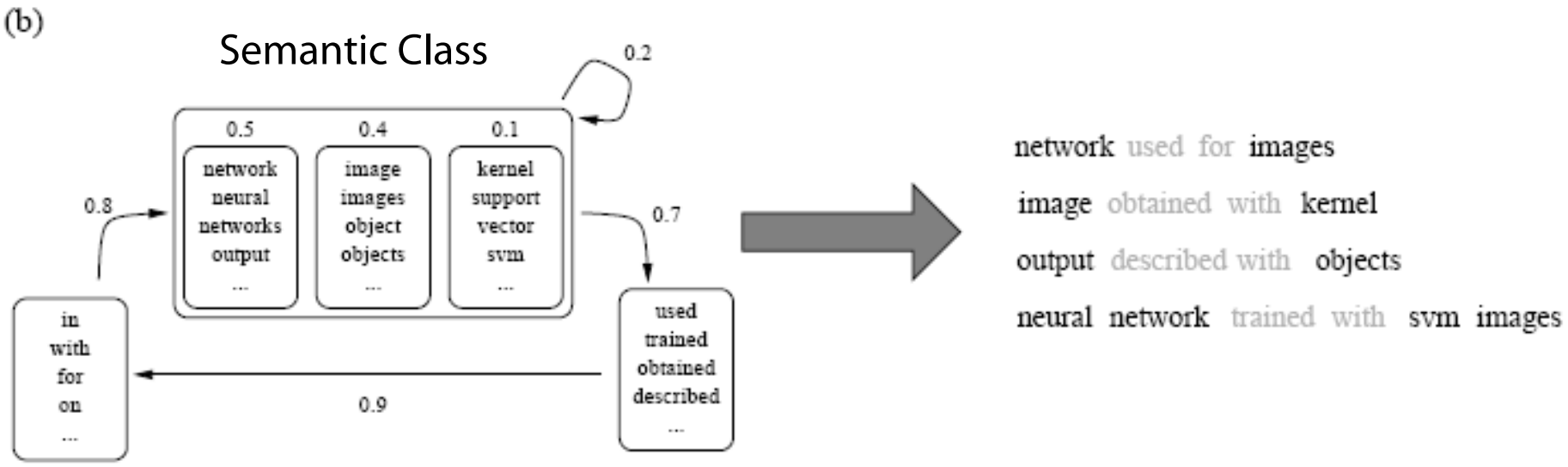
Semantic class

Draw a semantic word **OR** Draw a syntactic word

Graphical Model 1



Simple Example 1



Preposition class

Verb Class

- The HMM allocates words which vary across context to the semantic class, since grammar is fairly standardized but content is not.

Model Inference 1

MCMC inference

Topic indicators

$$P(z_i|\mathbf{z}_{-i},\mathbf{c},\mathbf{w}) \propto P(z_i|\mathbf{z}_{-i}) \quad P(w_i|\mathbf{z},\mathbf{c},\mathbf{w}_{-i})$$

$$\propto \begin{cases} n_{z_i}^{(d_i)} + \alpha & c_i \neq 1 \\ (n_{z_i}^{(d_i)} + \alpha) \quad \frac{n_{w_i}^{(z_i)} + \beta}{n^{(z_i)} + W\beta} & c_i = 1 \end{cases}$$

$n_{z_i}^{(d_i)}$ is the number of words in document d_i signed to topic z_i

$n_{w_i}^{(z_i)}$ is the number of words in topic z_i that are the same as w_i

All counts include only words for which $c_i = 1$ and exclude word i

Model Inference 2

Class indicators

$$P(c_i|\mathbf{c}_{-i},\mathbf{z},\mathbf{w}) \propto P(w_i|\mathbf{c},\mathbf{z},\mathbf{w}_{-i})\, P(c_i|\mathbf{c}_{-i})$$

$$\propto \begin{cases} \frac{n_{w_i}^{(c_i)}+\delta}{n_{\cdot}^{(c_i)}+W\delta} \; \frac{(n_{c_i}^{(c_{i-1})}+\gamma)(n_{c_{i+1}}^{(c_i)}+I(c_{i-1}=c_i)\cdot I(c_i=c_{i+1})+\gamma)}{n_{\cdot}^{(c_i)}+I(c_{i-1}=c_i)+C\gamma} & c_i \neq 1 \\ \frac{n_{w_i}^{(z_i)}+\beta}{n_{\cdot}^{(z_i)}+W\beta} \; \frac{(n_{c_i}^{(c_{i-1})}+\gamma)(n_{c_{i+1}}^{(c_i)}+I(c_{i-1}=c_i)\cdot I(c_i=c_{i+1})+\gamma)}{n_{\cdot}^{(c_i)}+I(c_{i-1}=c_i)+C\gamma} & c_i = 1 \end{cases}$$

$n_{z_i}^{(d_i)}$ is the number of words in document d_i signed to topic z_i

$n_{w_i}^{(z_i)}$ is the number of words in topic z_i that are the same as w_i

$n_{w_i}^{(c_i)}$ is the number of words in class c_i that are the same as w_i

$n_{c_i}^{(c_{i-1})}$ is the number of transitions from class c_{i-1} to class c_i

$I(\cdot)$ is an indicator variable which equals 1 if argument is true

All counts exclude transitions to and from c_i

Extreme Cases 1

- If we set the number of semantic topics to T = 1, then the model reduces to an HMM parts of speech tagger.
- If we set the number of HMM classes to C = 2, where one state is for punctuation, the the model reduces to LDA.

Results 1

Brown + TASA corpus: 38,151 documents; Vocab Size = 37,202;
number of word tokens = 13,328,397 words

the	the	the	the	the	a	the	the	the
blood	,	,	of	a	the	,	,	,
,	and	and	,	of	of	of	a	a
of	of	of	to	,	,	a	of	in
body	a	in	in	in	in	and	and	game
heart	in	land	and	to	water	in	drink	ball
and	trees	to	classes	picture	is	story	alcohol	and
in	tree	farmers	government	film	and	is	to	team
to	with	for	a	image	matter	to	bottle	to
is	on	farm	state	lens	are	as	in	play

LDA only

blood	forest	farmers	government	light	water	story	drugs	ball
heart	trees	land	state	eye	matter	stories	drug	game
pressure	forests	crops	federal	lens	molecules	poem	alcohol	team
body	land	farm	public	image	liquid	characters	people	*
lungs	soil	food	local	mirror	particles	poetry	drinking	baseball
oxygen	areas	people	act	eyes	gas	character	person	players
vessels	park	farming	states	glass	solid	author	effects	football
arteries	wildlife	wheat	national	object	substance	poems	marijuana	player
*	area	farms	laws	objects	temperature	life	body	field
breathing	rain	corn	department	lenses	changes	poet	use	basketball

HMM-LDA Semantic Topics

the	in	he	*	be	said	can	time	,
a	for	it	new	have	made	would	way	;
his	to	you	other	see	used	will	years	(
this	on	they	first	make	came	could	day	:
their	with	i	same	do	went	may	part	)
these	at	she	great	know	found	had	number	
your	by	we	good	get	called	must	kind	
her	from	there	small	go		do	place	
my	as	this	little	take		have		
some	into	who	old	find		did		

HMM-LDA Syntactic Classes

Results 2

NIPS Papers 1713 documents; Vocabulary Size: 17268;
Number of word tokens = 4,321,614

image	data	state	membrane	chip	experts	kernel	network
images	gaussian	policy	synaptic	analog	expert	support	neural
object	mixture	value	cell	neuron	gating	vector	networks
objects	likelihood	function	*	digital	hme	svm	output
feature	posterior	action	current	synapse	architecture	kernels	input
recognition	prior	reinforcement	dendritic	neural	mixture	#	training
views	distribution	learning	potential	hardware	learning	space	inputs
#	em	classes	neuron	weight	mixtures	function	weights
pixel	bayesian	optimal	conductance	#	function	machines	#
visual	parameters	*	channels	vlsi	gate	set	outputs
in	is	see	used	model	networks	however	#
with	was	show	trained	algorithm	values	also	*
for	has	note	obtained	system	results	then	i
on	becomes	consider	described	case	models	thus	x
from	denotes	assume	given	problem	parameters	therefore	t
at	being	present	found	network	units	first	n
using	remains	need	presented	method	data	here	-
into	represents	propose	defined	approach	functions	now	c
over	exists	describe	generated	paper	problems	hence	r
within	seems	suggest	shown	process	algorithms	finally	p

Syntactic Words

Semantic Words

Results 2 (cont'd)

NIPS Papers 1713 documents; Vocabulary Size: 17268;
Number of word tokens = 4,321,614

1. In contrast to this approach, we study here how the overall **network activity** can [control] single **cell** parameters such as **input resistance**, as well as **time** and **space** constants, parameters that are crucial for **excitability** and **spariotemporal (sic) integration**.

   The integrated architecture in this paper combines **feed forward** [control] and **error feedback adaptive** [control] using **neural networks**.

2. In other words, for our **proof** of **convergence**, we require the **softassign algorithm** to [return] a **doubly stochastic matrix** as *sinkhorn theorem guarantees that it will instead of a **matrix** which is merely close to being **doubly stochastic** based on some reasonable **metric**.

   The aim is to construct a **portfolio** with a maximal **expected** [return] for a given **risk level** and **time horizon** while simultaneously obeying *institutional or *legally required constraints.

3. The left [graph] is the standard experiment the right from a **training** with # **samples**.

   The [graph] G is called the *guest [graph] and H is called the **host** [graph].

Black words are semantic, Graylevel words are syntactic. Boxed words are semantic on one passsage and syntactic in another. Asterisked words have low frequency and not considered.

Results 3

Log Marginal probabilities of the data

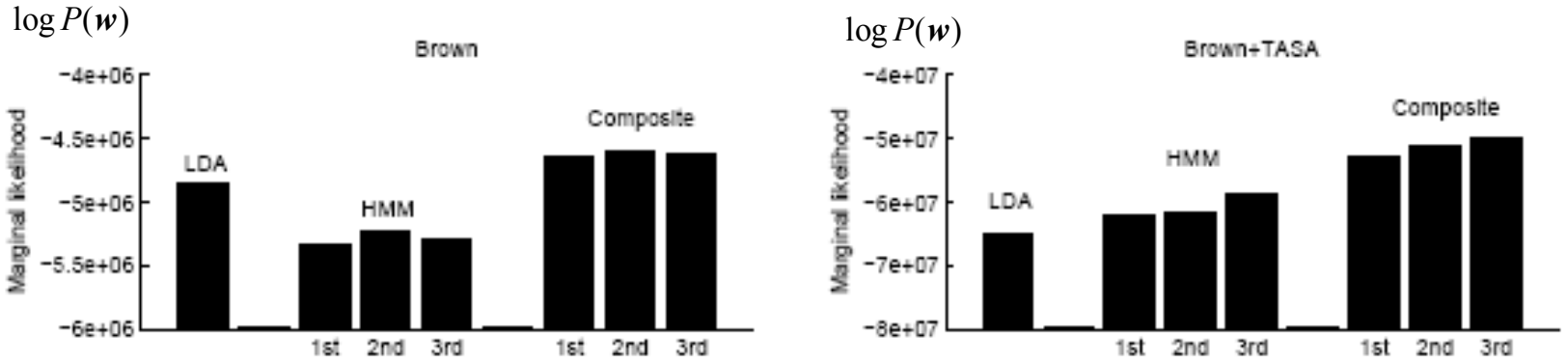
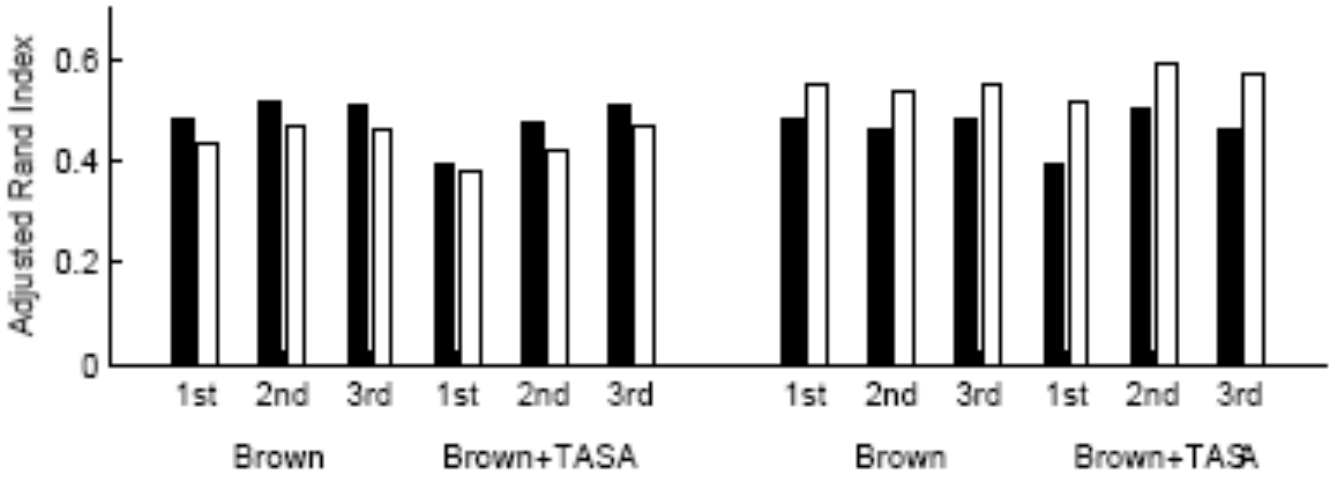


Figure 5: Log marginal probabilities of each corpus under different models. Labels on horizontal axis indicate the order of the HMM.

Results 4

Parts of speech tagging

Black bars indicate performance on a fine tagset (297 word types), white bars indicate performance on coarse tagset (10 word types).



HMM

Composite Model

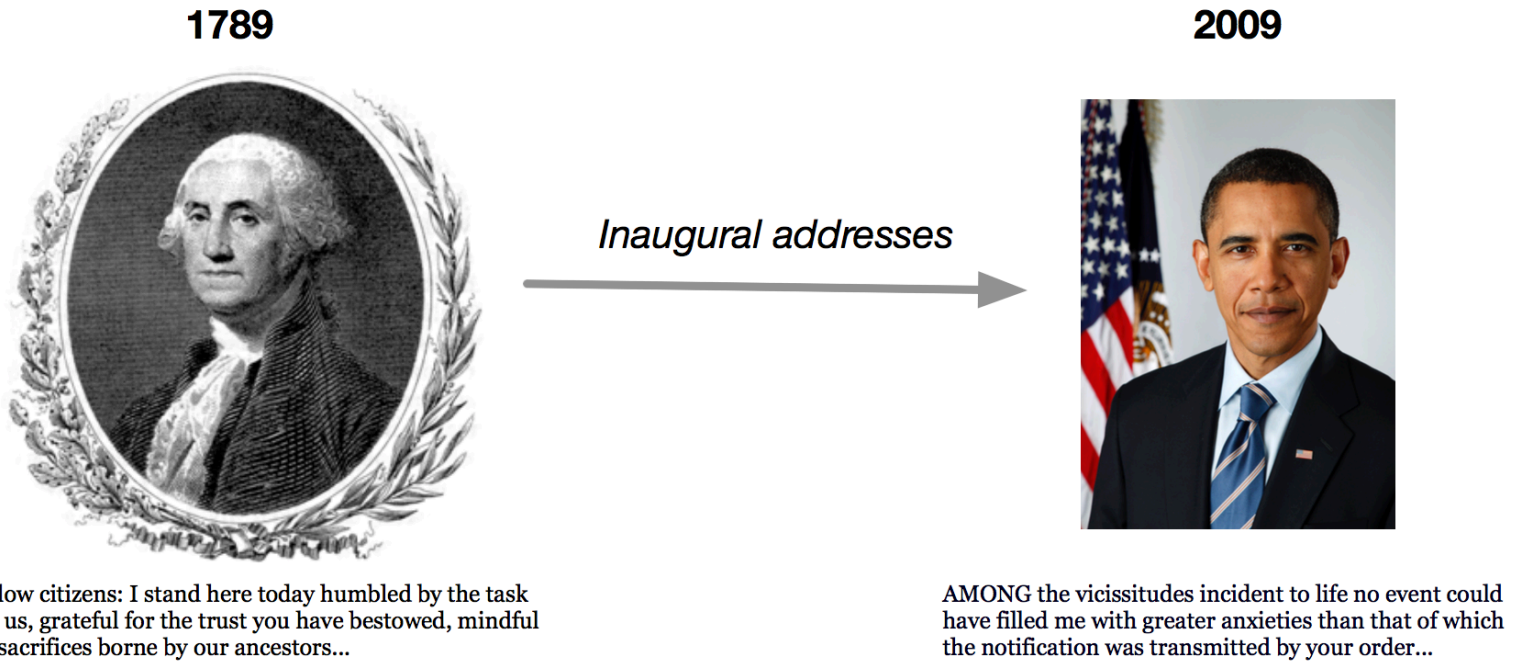
Conclusion

- HMM-LDA is a composite topic model which considers both long range semantic dependencies and short range syntactic dependencies.

- The model is quite competitive with a traditional HMM parts of speech tagger, and outperforms LDA when stopwords and punctuation are not removed.

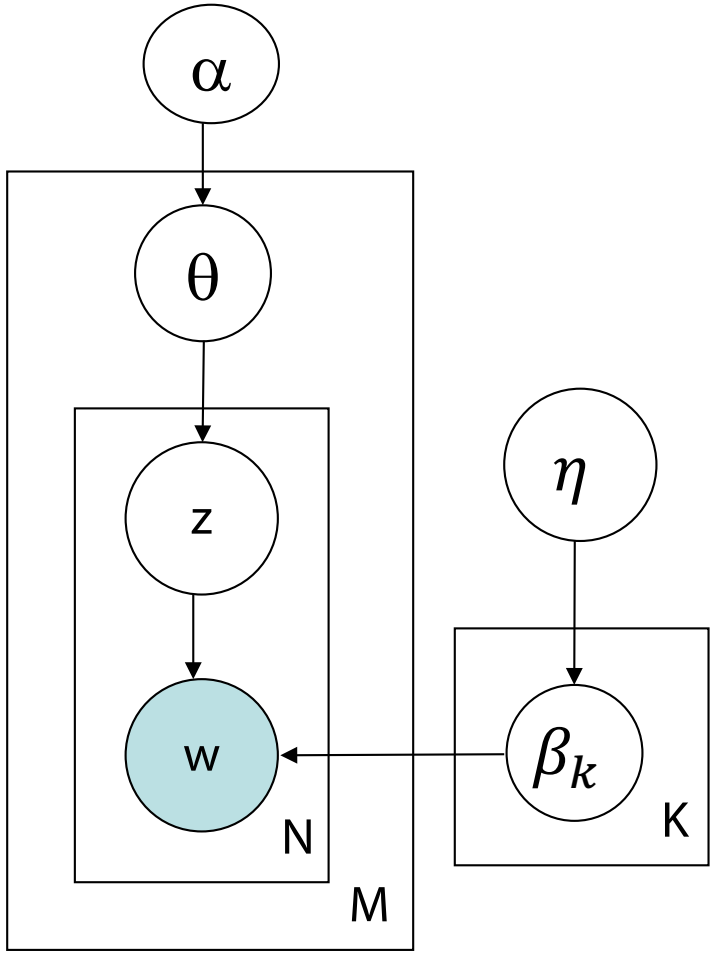
Dynamic Topic Models

- In LDA the order of documents does not matter
- Not appropriate for sequential corpora (e.g., that span hundreds of years)
- Further, we may want to track how language changes over time
- Let the topics *drift* in a sequence.

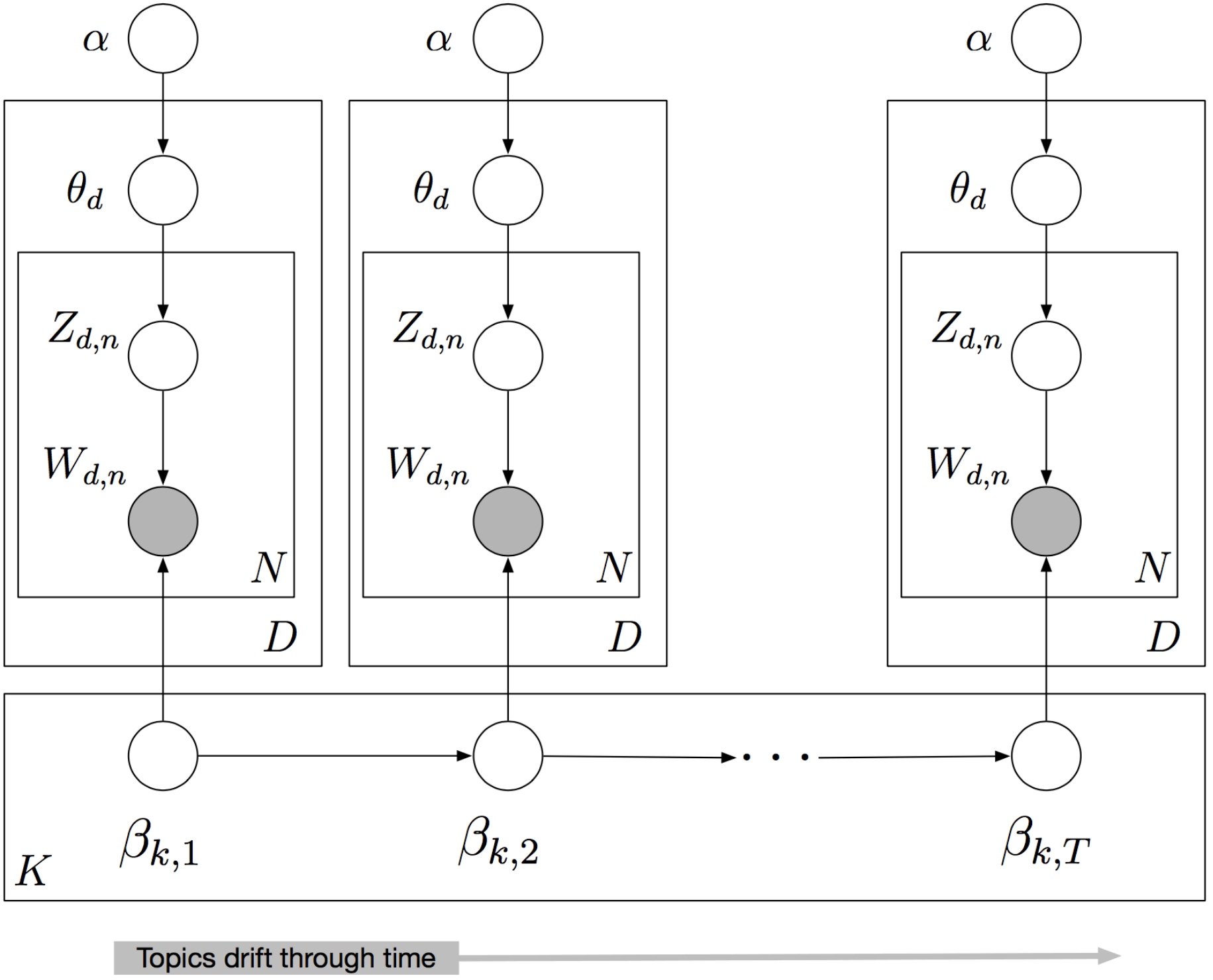


David M. Blei and John D. Lafferty. Dynamic topic models. In Proc. ICML '06. pp. 113-120. **2006**.

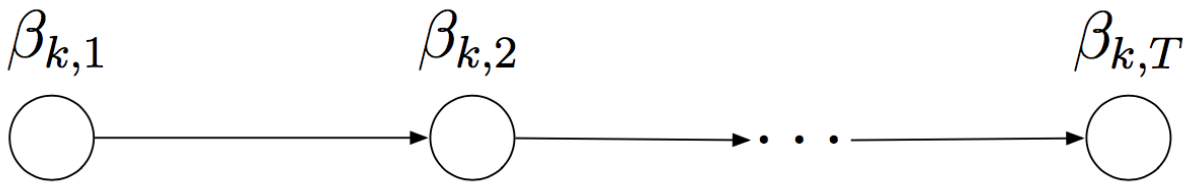
Recap: Smoothed LDA Model



- Give a different word distribution to each topic
  - β is $K \times V$ matrix (V vocabulary size), each row denotes word distribution of a topic
- For each document d
  - Choose $\theta_d \sim$ Dirichlet($\cdot \mid \alpha$)
  - Choose $\beta_k \sim$ Dirichlet($\eta \cdot \mid$)
  - For each position $i = 1, \ldots, N_d$
    - Generate a topic $z_i \sim$ Mult($\cdot \mid \theta_d$)
    - Generate a word $w_i \sim$ Mult($\cdot \mid z_i, \beta_k$)


α
α
α
θ_d
θ_d
θ_d
$Z_{d,n}$
$Z_{d,n}$
$Z_{d,n}$
$W_{d,n}$
$W_{d,n}$
$W_{d,n}$
N
N
N
D
D
D
K
$\beta_{k,1}$
$\beta_{k,2}$
$\beta_{k,T}$
Topics drift through time

Dynamic Topic Models



- Use a logistic normal distribution to model topics evolving over time
- Embed it in a state-space model on the log of the topic distribution

$$\beta_{t,k} \mid \beta_{t-1,k} \sim \mathcal{N}(\beta_{t-1,k}, I\sigma^2)$$
$$p(w \mid \beta_{t,k}) \propto \exp\{\beta_{t,k}\}$$

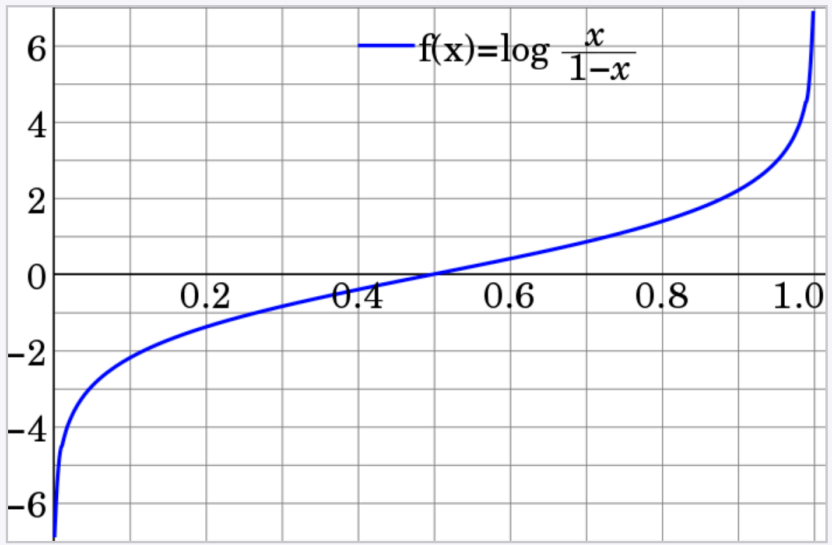
- Lets us make inferences about sequences of documents

Logit Normal Distribution

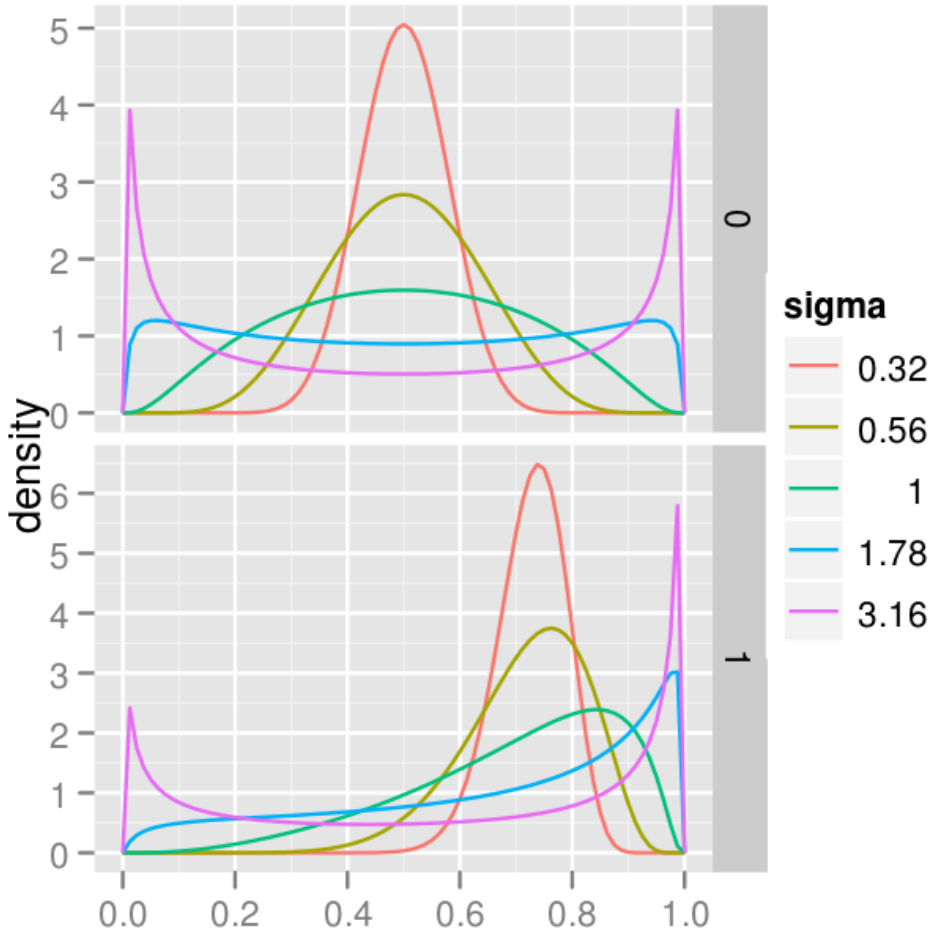
The probability density function (PDF) of a logit-normal distribution, for $0 \le x \le 1$, is:

$$f_X(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}\,\frac{1}{x(1-x)}\,e^{-\frac{(\operatorname{logit}(x)-\mu)^2}{2\sigma^2}}$$

where μ and σ are the mean and standard deviation of the variable's logit (by definition, the variable's logit is normally distributed).



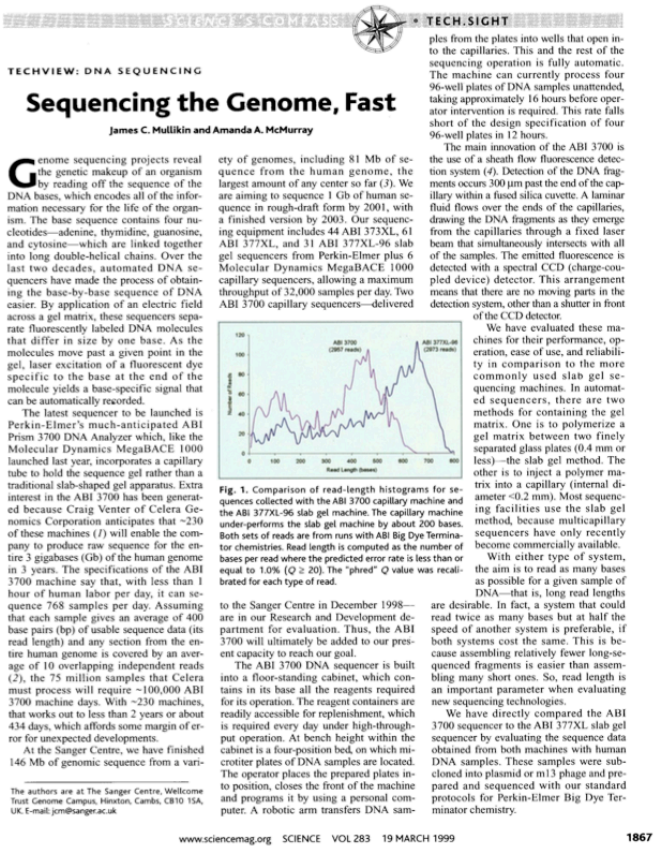
Plot of logit(p) in the domain of 0 to 1, where the base of logarithm is e



[Wikipedia]

Dynamic Topic Models

Original article



SCIENCE'S COMPASS • TECH.SIGHT

TECHVIEW: DNA SEQUENCING

Sequencing the Genome, Fast

James C. Mullikin and Amanda A. McMurray

Genome sequencing projects reveal the genetic makeup of an organism by reading off the sequence of the DNA bases, which encodes all of the information necessary for the life of the organism. The base sequence contains four nucleotides—adenine, thymidine, guanosine, and cytosine—which are linked together into long double-helical chains. Over the last two decades, automated DNA sequencers have made the process of obtaining the base-by-base sequence of DNA easier. By application of an electric field across a gel matrix, these sequencers separate fluorescently labeled DNA molecules that differ in size by one base. As the molecules move past a given point in the gel, laser excitation of a fluorescent dye specific to the base at the end of the molecule yields a base-specific signal that can be automatically recorded.

The latest sequencer to be launched is Perkin-Elmer's much-anticipated ABI Prism 3700 DNA Analyzer which, like the Molecular Dynamics MegaBACE 1000 launched last year, incorporates a capillary tube to hold the sequence gel rather than a traditional slab-shaped gel apparatus. Extra interest in the ABI 3700 has been generated because Craig Venter of Celera Genomics Corporation anticipates that ~230 of these machines (*1*) will enable the company to produce raw sequence for the entire 3 gigabases (Gb) of the human genome in 3 years. The specifications of the ABI 3700 machine say that, with less than 1 hour of human labor per day, it can sequence 768 samples per day. Assuming that each sample gives an average of 400 base pairs (bp) of usable sequence data (its read length) and any section from the entire human genome is covered by an average of 10 overlapping independent reads (*2*), the 75 million samples that Celera must process will require ~100,000 ABI 3700 machine days. With ~230 machines, that works out to less than 2 years or about 434 days, which affords some margin of error for unexpected developments.

At the Sanger Centre, we have finished 146 Mb of genomic sequence from a variety of genomes, including 81 Mb of sequence from the human genome, the largest amount of any center so far (*3*). We are aiming to sequence 1 Gb of human sequence in rough-draft form by 2001, with a finished version by 2003. Our sequencing equipment includes 44 ABI 373XL, 61 ABI 377XL, and 31 ABI 377XL-96 slab gel sequencers from Perkin-Elmer plus 6 Molecular Dynamics MegaBACE 1000 capillary sequencers, allowing a maximum throughput of 32,000 samples per day. Two ABI 3700 capillary sequencers—delivered to the Sanger Centre in December 1998—are in our Research and Development department for evaluation. Thus, the ABI 3700 will ultimately be added to our present capacity to reach our goal.

Fig. 1. Comparison of read-length histograms for sequences collected with the ABI 3700 capillary machine and the ABI 377XL-96 slab gel machine. The capillary machine under-performs the slab gel machine by about 200 bases. Both sets of reads are from runs with ABI Big Dye Terminator chemistries. Read length is computed as the number of bases per read where the predicted error rate is less than or equal to 1.0% ($Q \geq 20$). The "phred" Q value was recalibrated for each type of read.

The ABI 3700 DNA sequencer is built into a floor-standing cabinet, which contains in its base all the reagents required for its operation. The reagent containers are readily accessible for replenishment, which is required every day under high-throughput operation. At bench height within the cabinet is a four-position bed, on which microtiter plates of DNA samples are located. The operator places the prepared plates into position, closes the front of the machine and programs it by using a personal computer. A robotic arm transfers DNA samples from the plates into wells that open into the capillaries. This and the rest of the sequencing operation is fully automatic. The machine can currently process four 96-well plates of DNA samples unattended, taking approximately 16 hours before operator intervention is required. This rate falls short of the design specification of four 96-well plates in 12 hours.

The main innovation of the ABI 3700 is the use of a sheath flow fluorescence detection system (*4*). Detection of the DNA fragments occurs 300 μm past the end of the capillary within a fused silica cuvette. A laminar fluid flows over the ends of the capillaries, drawing the DNA fragments as they emerge from the capillaries through a fixed laser beam that simultaneously intersects with all of the samples. The emitted fluorescence is detected with a spectral CCD (charge-coupled device) detector. This arrangement means that there are no moving parts in the detection system, other than a shutter in front of the CCD detector.

We have evaluated these machines for their performance, operation, ease of use, and reliability in comparison to the more commonly used slab gel sequencing machines. In automated sequencers, there are two methods for containing the gel matrix. One is to polymerize a gel matrix between two finely separated glass plates (0.4 mm or less)—the slab gel method. The other is to inject a polymer matrix into a capillary (internal diameter <0.2 mm). Most sequencing facilities use the slab gel method, because multicapillary sequencers have only recently become commercially available.

With either type of system, the aim is to read as many bases as possible for a given sample of DNA—that is, long read lengths are desirable. In fact, a system that could read twice as many bases but at half the speed of another system is preferable, if both systems cost the same. This is because assembling relatively fewer long-sequenced fragments is easier than assembling many short ones. So, read length is an important parameter when evaluating new sequencing technologies.

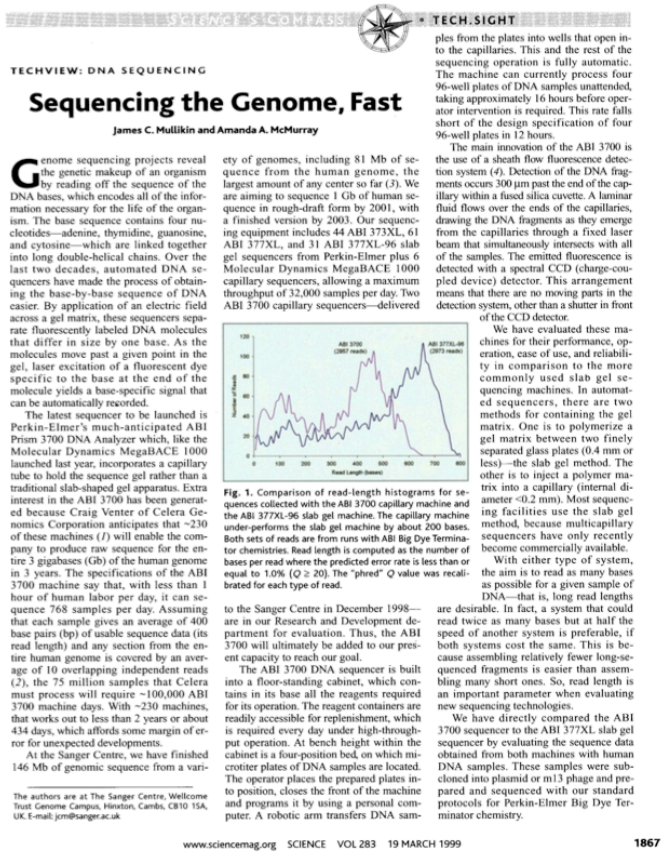
We have directly compared the ABI 3700 sequencer to the ABI 377XL slab gel sequencer by evaluating the sequence data obtained from both machines with human DNA samples. These samples were subcloned into plasmid or m13 phage and prepared and sequenced with our standard protocols for Perkin-Elmer Big Dye Terminator chemistry.

The authors are at The Sanger Centre, Wellcome Trust Genome Campus, Hinxton, Cambs, CB10 1SA, UK. E-mail: jcm@sanger.ac.uk

Topic proportions

Dynamic Topic Models

Original article



TECH.SIGHT

TECHVIEW: DNA SEQUENCING

Sequencing the Genome, Fast

James C. Mullikin and Amanda A. McMurray

Genome sequencing projects reveal the genetic makeup of an organism by reading off the sequence of the DNA bases, which encodes all of the information necessary for the life of the organism. The base sequence contains four nucleotides—adenine, thymidine, guanosine, and cytosine—which are linked together into long double-helical chains. Over the last two decades, automated DNA sequencers have made the process of obtaining the base-by-base sequence of DNA easier. By application of an electric field across a gel matrix, these sequencers separate fluorescently labeled DNA molecules that differ in size by one base. As the molecules move past a given point in the gel, laser excitation of a fluorescent dye specific to the base at the end of the molecule yields a base-specific signal that can be automatically recorded.

The latest sequencer to be launched is Perkin-Elmer's much-anticipated ABI Prism 3700 DNA Analyzer which, like the Molecular Dynamics MegaBACE 1000 launched last year, incorporates a capillary tube to hold the sequence gel rather than a traditional slab-shaped gel apparatus. Extra interest in the ABI 3700 has been generated because Craig Venter of Celera Genomics Corporation anticipates that ~230 of these machines (*1*) will enable the company to produce raw sequence for the entire 3 gigabases (Gb) of the human genome in 3 years. The specifications of the ABI 3700 machine say that, with less than 1 hour of human labor per day, it can sequence 768 samples per day. Assuming that each sample gives an average of 400 base pairs (bp) of usable sequence data (its read length) and any section from the entire human genome is covered by an average of 10 overlapping independent reads (*2*), the 75 million samples that Celera must process will require ~100,000 ABI 3700 machine days. With ~230 machines, that works out to less than 2 years or about 434 days, which affords some margin of error for unexpected developments.

At the Sanger Centre, we have finished 146 Mb of genomic sequence from a variety of genomes, including 81 Mb of sequence from the human genome, the largest amount of any center so far (*3*). We are aiming to sequence 1 Gb of human sequence in rough-draft form by 2001, with a finished version by 2003. Our sequencing equipment includes 44 ABI 373XL, 61 ABI 377XL, and 31 ABI 377XL-96 slab gel sequencers from Perkin-Elmer plus 6 Molecular Dynamics MegaBACE 1000 capillary sequencers, allowing a maximum throughput of 32,000 samples per day. Two ABI 3700 capillary sequencers—delivered to the Sanger Centre in December 1998—are in our Research and Development department for evaluation. Thus, the ABI 3700 will ultimately be added to our present capacity to reach our goal.

Fig. 1. Comparison of read-length histograms for sequences collected with the ABI 3700 capillary machine and the ABI 377XL-96 slab gel machine. The capillary machine under-performs the slab gel machine by about 200 bases. Both sets of reads are from runs with ABI Big Dye Terminator chemistries. Read length is computed as the number of bases per read where the predicted error rate is less than or equal to 1.0% ($Q \geq 20$). The "phred" Q value was recalibrated for each type of read.

The ABI 3700 DNA sequencer is built into a floor-standing cabinet, which contains in its base all the reagents required for its operation. The reagent containers are readily accessible for replenishment, which is required every day under high-throughput operation. At bench height within the cabinet is a four-position bed, on which microtiter plates of DNA samples are located. The operator places the prepared plates into position, closes the front of the machine and programs it by using a personal computer. A robotic arm transfers DNA samples from the plates into wells that open into the capillaries. This and the rest of the sequencing operation is fully automatic. The machine can currently process four 96-well plates of DNA samples unattended, taking approximately 16 hours before operator intervention is required. This rate falls short of the design specification of four 96-well plates in 12 hours.

The main innovation of the ABI 3700 is the use of a sheath flow fluorescence detection system (*4*). Detection of the DNA fragments occurs 300 μm past the end of the capillary within a fused silica cuvette. A laminar fluid flows over the ends of the capillaries, drawing the DNA fragments as they emerge from the capillaries through a fixed laser beam that simultaneously intersects with all of the samples. The emitted fluorescence is detected with a spectral CCD (charge-coupled device) detector. This arrangement means that there are no moving parts in the detection system, other than a shutter in front of the CCD detector.

We have evaluated these machines for their performance, operation, ease of use, and reliability in comparison to the more commonly used slab gel sequencing machines. In automated sequencers, there are two methods for containing the gel matrix. One is to polymerize a gel matrix between two finely separated glass plates (0.4 mm or less)—the slab gel method. The other is to inject a polymer matrix into a capillary (internal diameter <0.2 mm). Most sequencing facilities use the slab gel method, because multicapillary sequencers have only recently become commercially available.

With either type of system, the aim is to read as many bases as possible for a given sample of DNA—that is, long read lengths are desirable. In fact, a system that could read twice as many bases but at half the speed of another system is preferable, if both systems cost the same. This is because assembling relatively fewer long-sequenced fragments is easier than assembling many short ones. So, read length is an important parameter when evaluating new sequencing technologies.

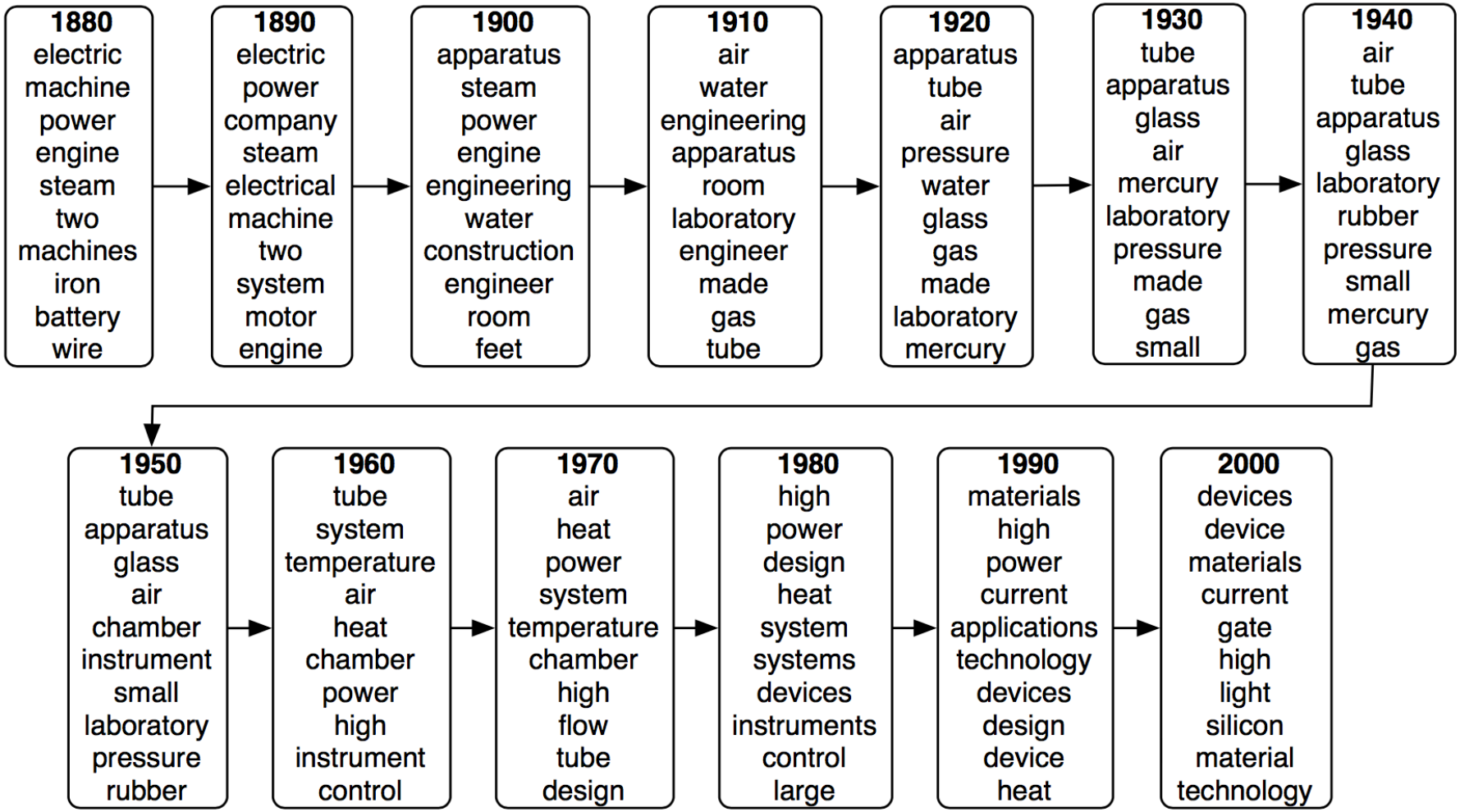
We have directly compared the ABI 3700 sequencer to the ABI 377XL slab gel sequencer by evaluating the sequence data obtained from both machines with human DNA samples. These samples were subcloned into plasmid or m13 phage and prepared and sequenced with our standard protocols for Perkin-Elmer Big Dye Terminator chemistry.

The authors are at The Sanger Centre, Wellcome Trust Genome Campus, Hinxton, Cambs, CB10 1SA, UK. E-mail: jcm@sanger.ac.uk

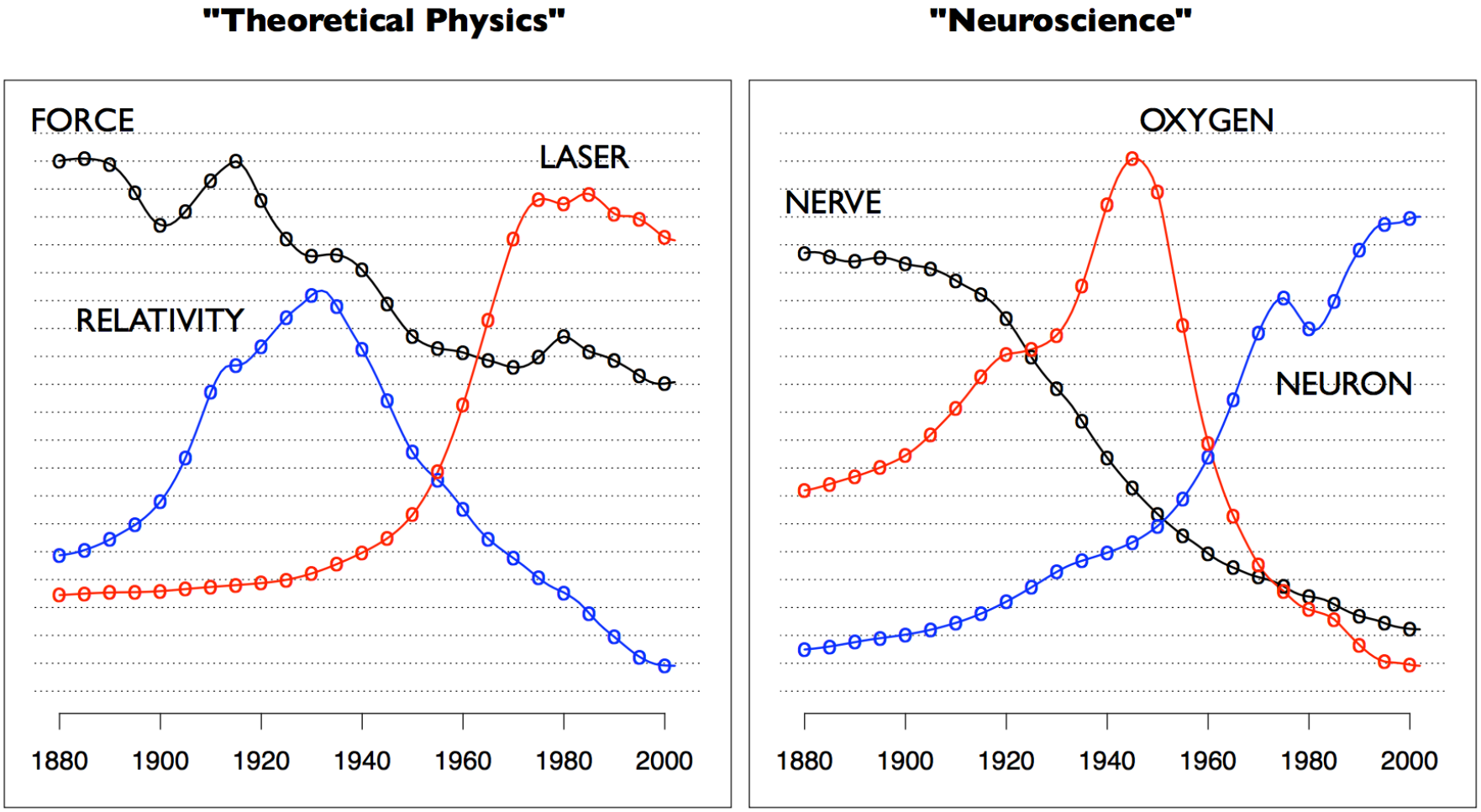
Most likely words from top topics

sequence	devices	data
genome	device	information
genes	materials	network
sequences	current	web
human	high	computer
gene	gate	language
dna	light	networks
sequencing	silicon	time
chromosome	material	software
regions	technology	system
analysis	electrical	words
data	fiber	algorithm
genomic	power	number
number	based	internet

Dynamic Topic Models



Dynamic Topic Models



Wang, Chong; Blei, David; Heckerman, David. "Continuous Time Dynamic Topic Models". *Proceedings of ICML*'08, **2008**.

Dynamic Topic Models

- **Time-corrected similarity** shows a new way of using the posterior.
- Consider the expected Hellinger distance between the topic proportions of two documents,

$$d_{ij} = \mathrm{E}\left[\sum_{k=1}^{K}(\sqrt{\theta_{i,k}} - \sqrt{\theta_{j,k}})^2 \,|\, \mathbf{w}_i, \mathbf{w}_j\right]$$

- Uses the latent structure to define similarity
- Time has been factored out because the topics associated to the components are different from year to year.
- Similarity based only on topic proportions

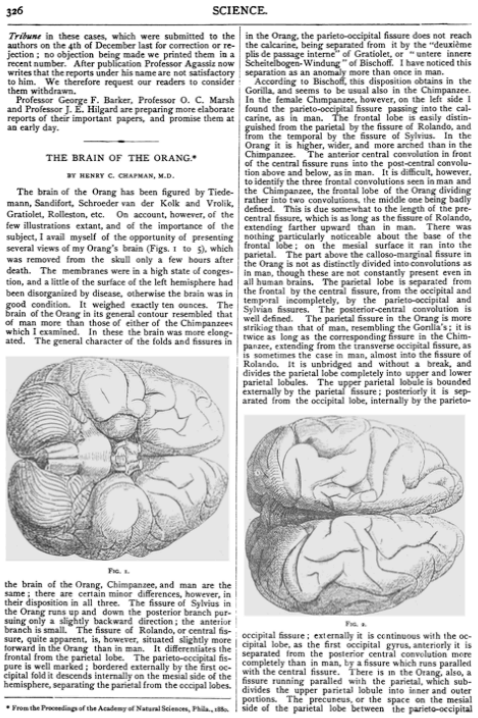
For two discrete probability distributions $P = (p_1, \ldots, p_k)$ and $Q = (q_1, \ldots, q_k)$, their Hellinger distance is defined as

$$H(P, Q) = \frac{1}{\sqrt{2}} \sqrt{\sum_{i=1}^{k} (\sqrt{p_i} - \sqrt{q_i})^2},$$

[Wikipedia]

Dynamic Topic Models

The Brain of the Orang (1880)



326 SCIENCE.

Tribune in these cases, which were submitted to the authors on the 4th of December last for correction or rejection; no objection being made we printed them in a recent number. After publication Professor Agassiz now writes that the reports under his name are not satisfactory to him. We therefore request our readers to consider them withdrawn.

Professor George F. Barker, Professor O. C. Marsh and Professor J. E. Hilgard are preparing more elaborate reports of their important papers, and promise them at an early day.

THE BRAIN OF THE ORANG.*

BY HENRY C. CHAPMAN, M.D.

The brain of the Orang has been figured by Tiedemann, Sandifort, Schroeder van der Kolk and Vrolik, Gratiolet, Rolleston, etc. On account, however, of the few illustrations extant, and of the importance of the subject, I avail myself of the opportunity of presenting several views of my Orang's brain (Figs. 1 to 5), which was removed from the skull only a few hours after death. The membranes were in a high state of congestion, and a little of the surface of the left hemisphere had been disorganized by disease, otherwise the brain was in good condition. It weighed exactly ten ounces. The brain of the Orang in its general contour resembled that of man more than those of either of the Chimpanzees which I examined. In these the brain was more elongated. The general character of the folds and fissures in

FIG. 1.

the brain of the Orang, Chimpanzee, and man are the same; there are certain minor differences, however, in their disposition in all three. The fissure of Sylvius in the Orang runs up and down the posterior branch pursuing only a slightly backward direction; the anterior branch is small. The fissure of Rolando, or central fissure, quite apparent, is, however, situated slightly more forward in the Orang than in man. It differentiates the frontal from the parietal lobe. The parieto-occipital fissure is well marked; bordered externally by the first occipital fold it descends internally on the mesial side of the hemisphere, separating the parietal from the occipal lobes.

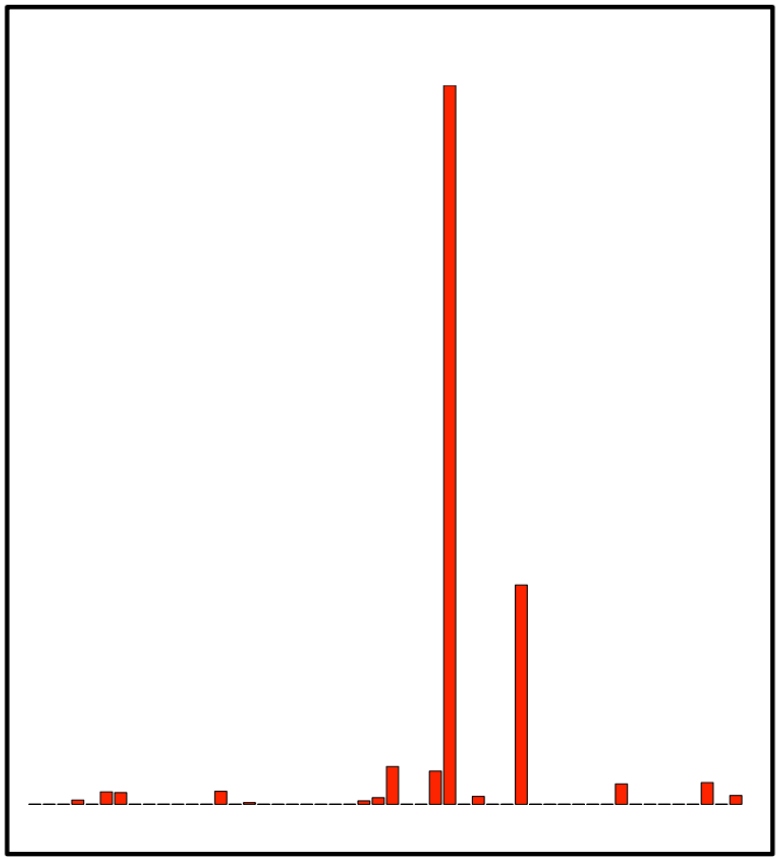
* From the Proceedings of the Academy of Natural Sciences, Phila., 1880.

in the Orang, the parieto-occipital fissure does not reach the calcarine, being separated from it by the "deuxième plis de passage interne" of Gratiolet, or "untere innere Scheitelbogen-Windung" of Bischoff. I have noticed this separation as an anomaly more than once in man.

According to Bischoff, this disposition obtains in the Gorilla, and seems to be usual also in the Chimpanzee. In the female Chimpanzee, however, on the left side I found the parieto-occipital fissure passing into the calcarine, as in man. The frontal lobe is easily distinguished from the parietal by the fissure of Rolando, and from the temporal by the fissure of Sylvius. In the Orang it is higher, wider, and more arched than in the Chimpanzee. The anterior central convolution in front of the central fissure runs into the post-central convolution above and below, as in man. It is difficult, however, to identify the three frontal convolutions seen in man and the Chimpanzee, the frontal lobe of the Orang dividing rather into two convolutions, the middle one being badly defined. This is due somewhat to the length of the precentral fissure, which is as long as the fissure of Rolando, extending farther upward than in man. There was nothing particularly noticable about the base of the frontal lobe; on the mesial surface it ran into the parietal. The part above the calloso-marginal fissure in the Orang is not as distinctly divided into convolutions as in man, though these are not constantly present even in all human brains. The parietal lobe is separated from the frontal by the central fissure, from the occipital and temporal incompletely, by the parieto-occipital and Sylvian fissures. The posterior-central convolution is well defined. The parietal fissure in the Orang is more striking than that of man, resembling the Gorilla's; it is twice as long as the corresponding fissure in the Chimpanzee, extending from the transverse occipital fissure, as is sometimes the case in man, almost into the fissure of Rolando. It is unbridged and without a break, and divides the parietal lobe completely into upper and lower parietal lobules. The upper parietal lobule is bounded externally by the parietal fissure; posteriorly it is separated from the occipital lobe, internally by the parieto-

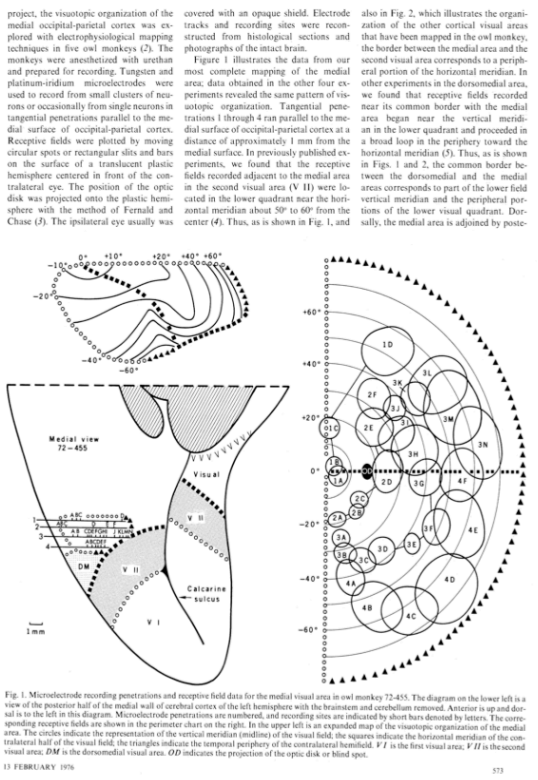
FIG. 2.

occipital fissure; externally it is continuous with the occipital lobe, as the first occipital gyrus, anteriorly it is separated from the posterior central convolution more completely than in man, by a fissure which runs parallel with the central fissure. There is in the Orang, also, a fissure running parallel with the parietal, which subdivides the upper parietal lobule into inner and outer portions. The precuneus, or the space on the mesial side of the parietal lobe between the parieto-occipital



Dynamic Topic Models

Representation of the Visual Field on the Medial Wall of Occipital-Parietal Cortex in the Owl Monkey (1976)



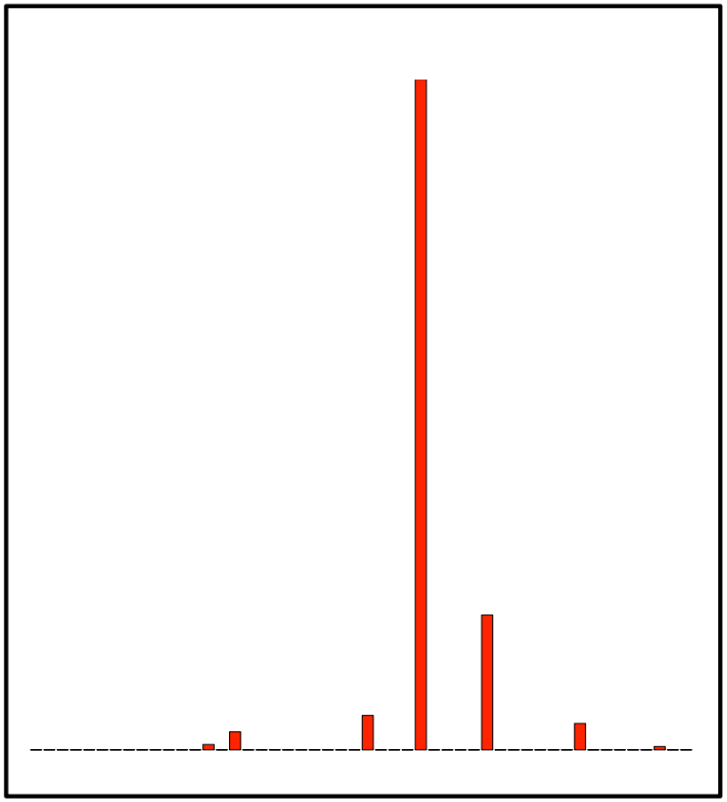
project, the visuotopic organization of the medial occipital-parietal cortex was explored with electrophysiological mapping techniques in five owl monkeys (*2*). The monkeys were anesthetized with urethan and prepared for recording. Tungsten and platinum-iridium microelectrodes were used to record from small clusters of neurons or occasionally from single neurons in tangential penetrations parallel to the medial surface of occipital-parietal cortex. Receptive fields were plotted by moving circular spots or rectangular slits and bars on the surface of a translucent plastic hemisphere centered in front of the contralateral eye. The position of the optic disk was projected onto the plastic hemisphere with the method of Fernald and Chase (*3*). The ipsilateral eye usually was covered with an opaque shield. Electrode tracks and recording sites were reconstructed from histological sections and photographs of the intact brain.

Figure 1 illustrates the data from our most complete mapping of the medial area; data obtained in the other four experiments revealed the same pattern of visuotopic organization. Tangential penetrations 1 through 4 ran parallel to the medial surface of occipital-parietal cortex at a distance of approximately 1 mm from the medial surface. In previously published experiments, we found that the receptive fields recorded adjacent to the medial area in the second visual area (**V II**) were located in the lower quadrant near the horizontal meridian about 50° to 60° from the center (*4*). Thus, as is shown in Fig. 1, and also in Fig. 2, which illustrates the organization of the other cortical visual areas that have been mapped in the owl monkey, the border between the medial area and the second visual area corresponds to a peripheral portion of the horizontal meridian. In other experiments in the dorsomedial area, we found that receptive fields recorded near its common border with the medial area began near the vertical meridian in the lower quadrant and proceeded in a broad loop in the periphery toward the horizontal meridian (*5*). Thus, as is shown in Figs. 1 and 2, the common border between the dorsomedial and the medial areas corresponds to part of the lower field vertical meridian and the peripheral portions of the lower visual quadrant. Dorsally, the medial area is adjoined by poste-

Fig. 1. Microelectrode recording penetrations and receptive field data for the medial visual area in owl monkey 72-455. The diagram on the lower left is a view of the posterior half of the medial wall of cerebral cortex of the left hemisphere with the brainstem and cerebellum removed. Anterior is up and dorsal is to the left in this diagram. Microelectrode penetrations are numbered, and recording sites are indicated by short bars denoted by letters. The corresponding receptive fields are shown in the perimeter chart on the right. In the upper left is an expanded map of the visuotopic organization of the medial area. The circles indicate the representation of the vertical meridian (midline) of the visual field; the squares indicate the horizontal meridian of the contralateral half of the visual field; the triangles indicate the temporal periphery of the contralateral hemifield. *V I* is the first visual area; *V II* is the second visual area; *DM* is the dorsomedial visual area. *OD* indicates the projection of the optic disk or blind spot.

13 FEBRUARY 1976

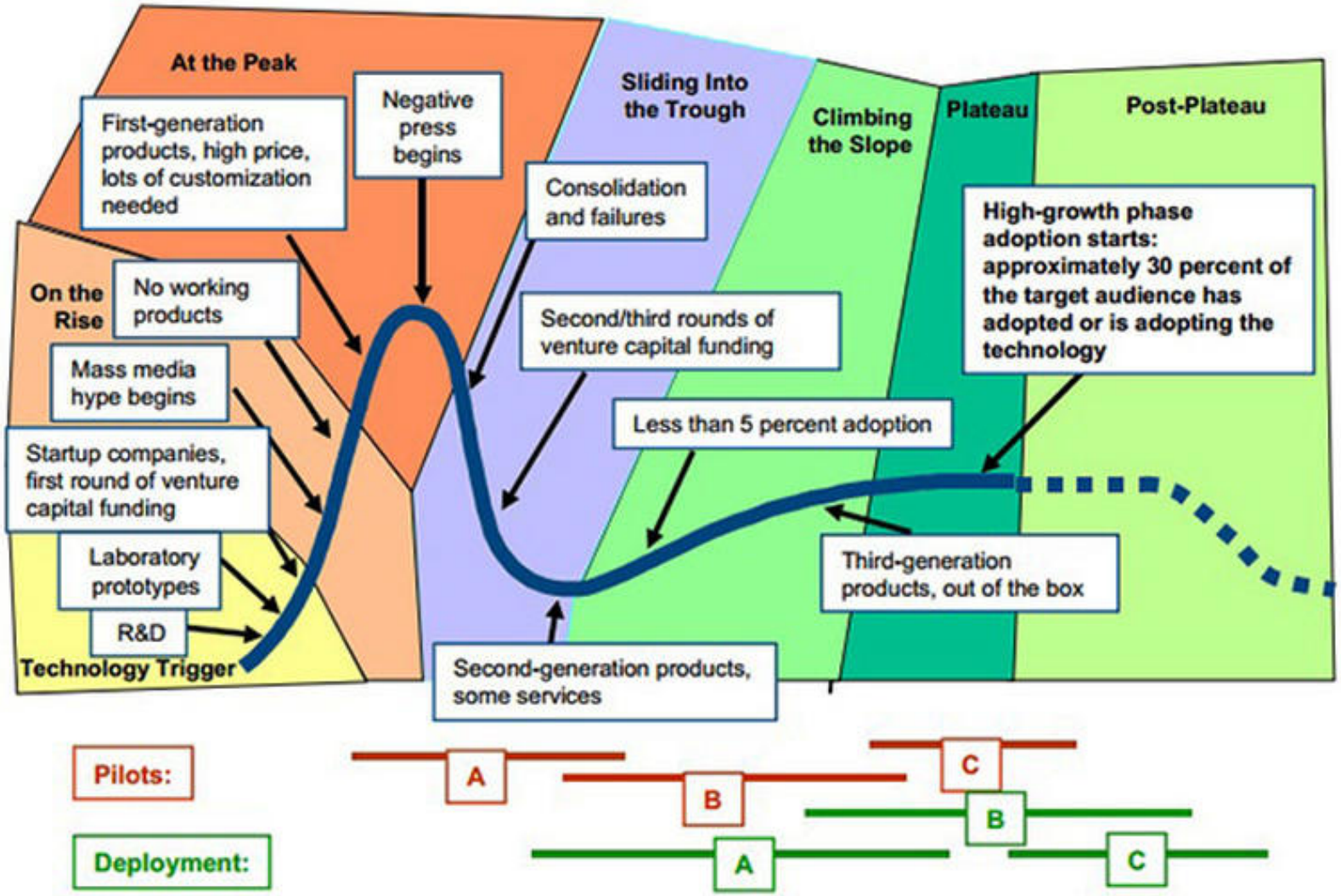
573



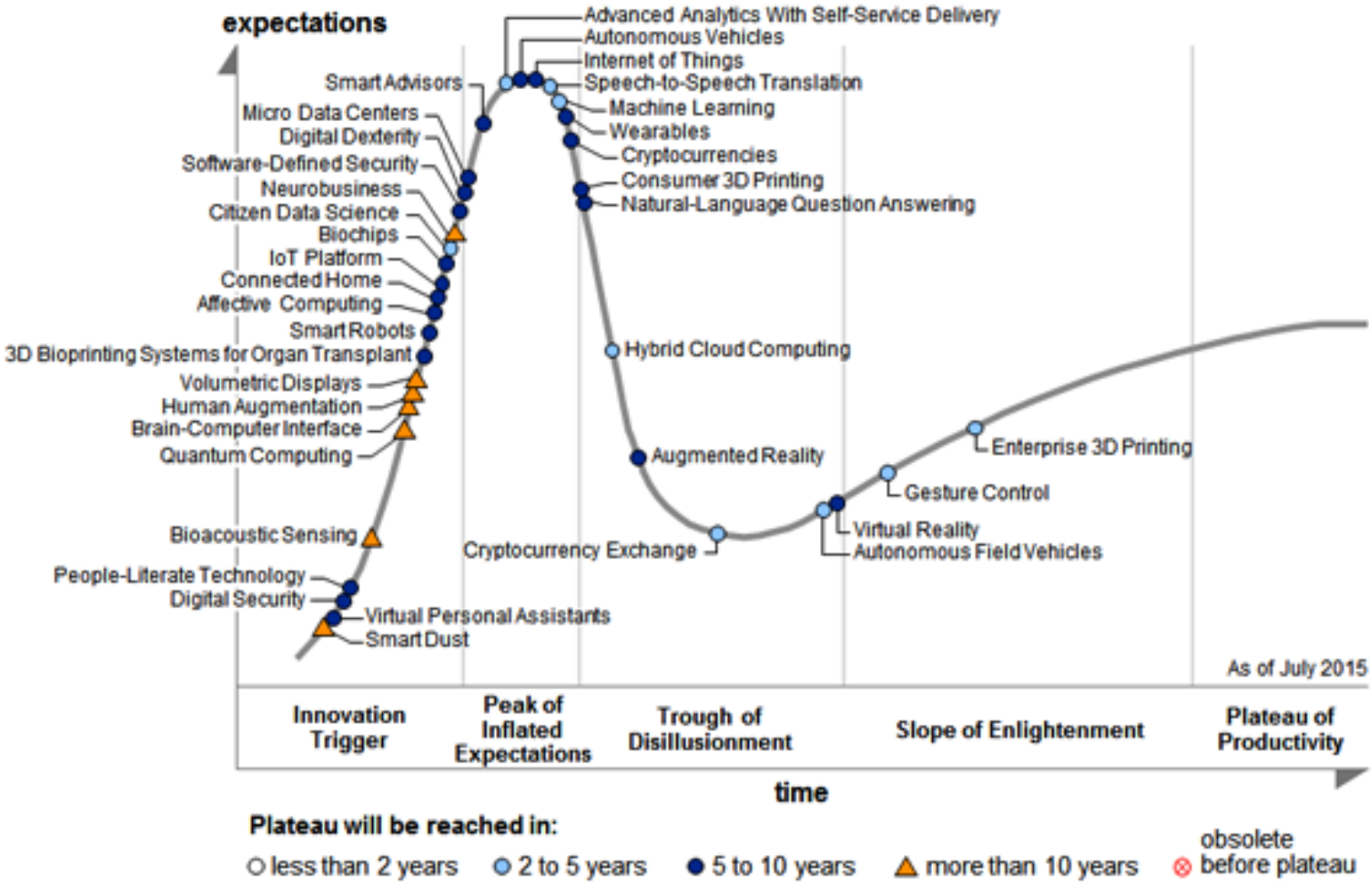
Dynamic Topic Models: Summary

- The Dirichlet assumption on topics and topic proportions makes strong conditional independence assumptions about the data.
- The **dynamic topic model** uses a logistic normal in a linear dynamic model to capture how topics change over time.

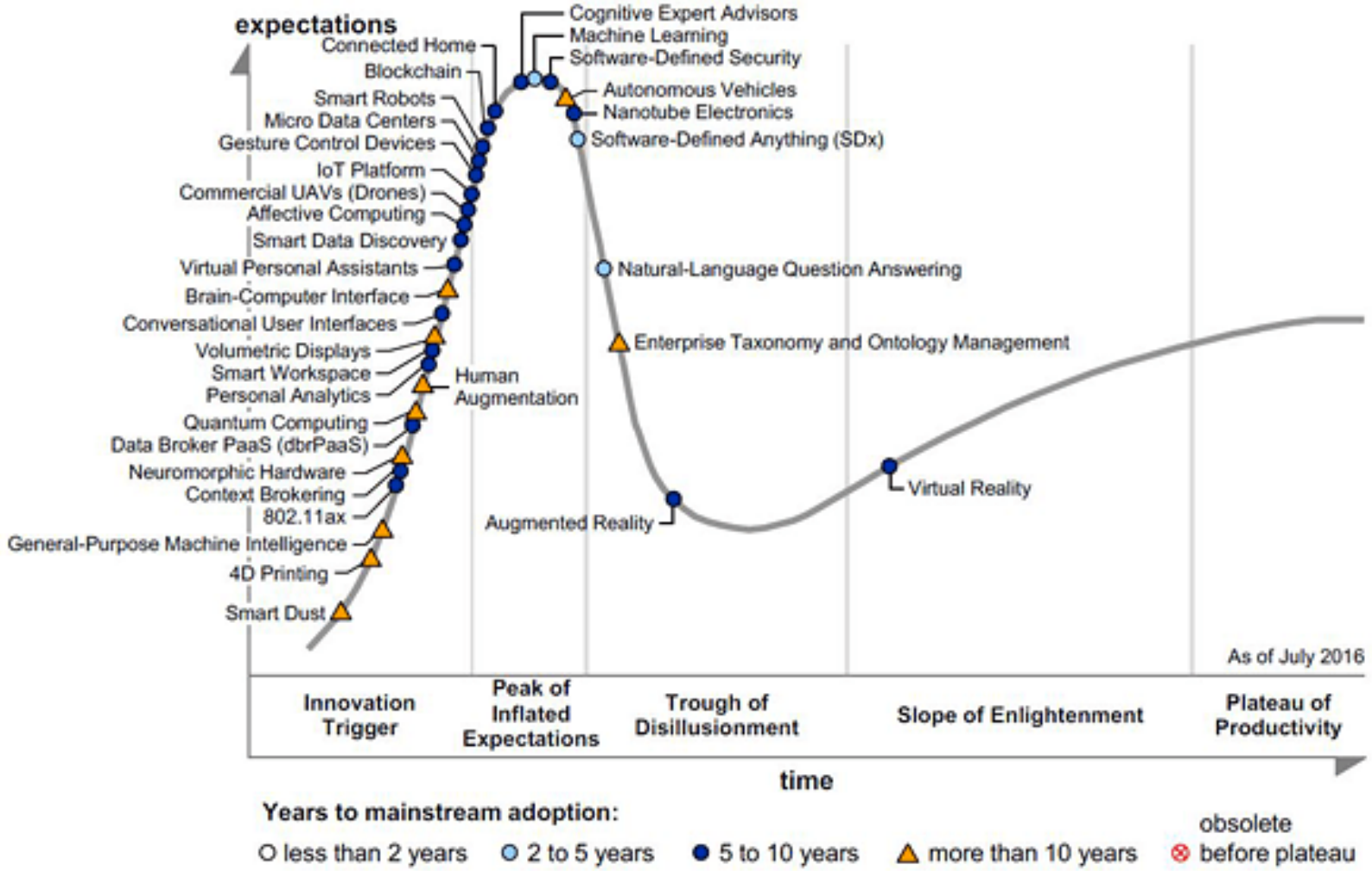
Gartner Hype Cycle



Gartner Hype Cycle 2015



Gartner Hype Cycle



Source: Gartner (July 2016)

Gartner Hype Cycle