

Computational Linguistics

Hidden Markov Models and POS-Tagging

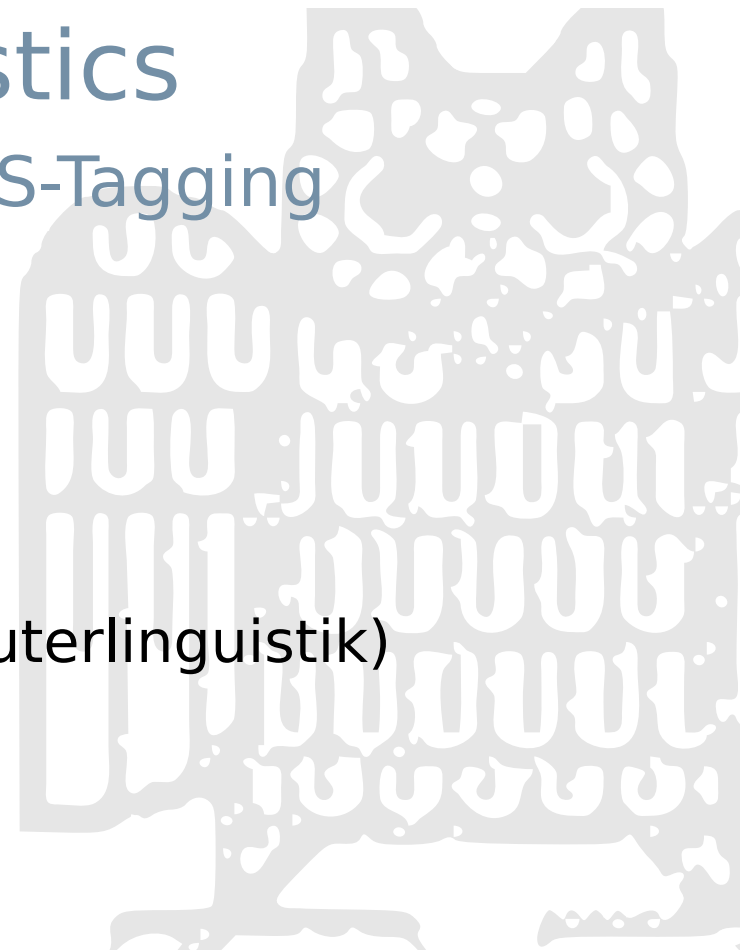
Clayton Greenberg

Stefan Thater

FR 4.7 Allgemeine Linguistik (Computerlinguistik)

Universität des Saarlandes

Summer 2016

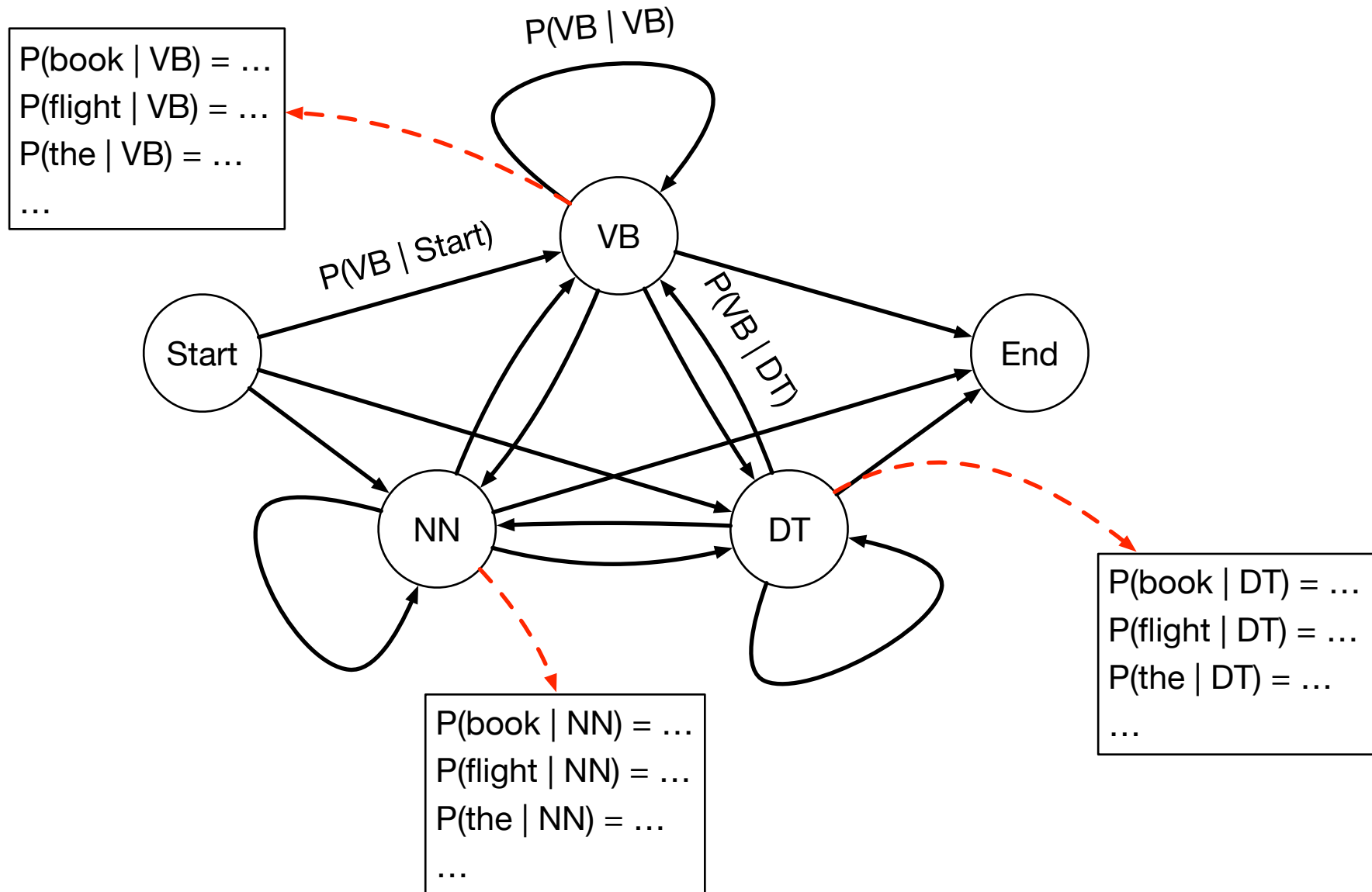


[most stuff on whiteboard]

see also:

<http://www.ling.uni-potsdam.de/tcl/anlp15/lectures/05%20HMMs.pdf>

Hidden Markov Model POS-Tagger (Example)



Hidden Markov Model POS-Tagger

- $Q = q_1 q_2 \dots q_N$ - set of states
 - q_0, q_F - special start and end states
- $A_{i,j}$ - transition probability matrix
 - $a_{i,j}$ represents the probability of moving from q_i to q_j
 - $a_{i,j} = P(q_j | q_i)$
- $O = o_1 o_2 \dots o_t$ - sequence of observations (words)
- $b_i(o_t)$ - emission probabilities
 - represents the probability that observation o_t has been generated by state q_i
 - $b_i(o_t) = P(o_t | q_i)$

Viterbi Decoding

- Task: Find the sequence of POS-tags that best explains the “observations” (words in the input sequence).
- Input:
 - Hidden Markov Model
 - sequence $o_1 \dots o_T$ of words
- Output:
 - sequence $q_1 \dots q_T$ of POS-tags (+ corresponding prob.)
- Algorithm: Dynamic Programming (cf. CYK)

Viterbi Decoding

Matrix: One column per Word, one row per POS-tag

In each cell: compute max. probability + back pointer

	<i>Mary</i>	<i>had</i>	<i>cats</i>	
	DT	DT	DT	
Start 1.0	NN	NN	NN	End
	VBD	VBD	VBD	

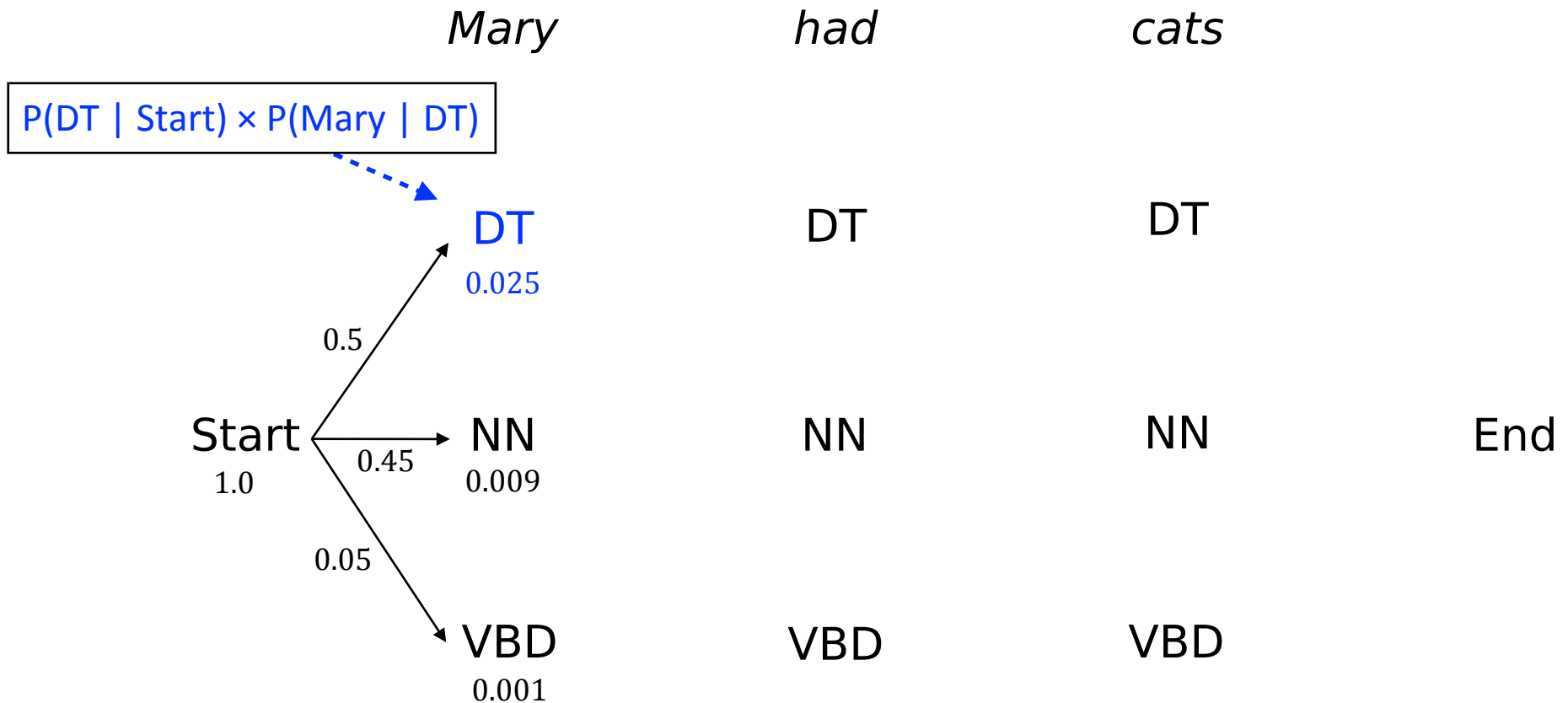
Viterbi Decoding

Matrix: One cell

In each cell: compare max. probability

$P(\text{DT} \mid \text{Start}) = 0.5$
 $P(\text{NN} \mid \text{Start}) = 0.45$
 $P(\text{VBD} \mid \text{Start}) = 0.05$

$P(\text{Mary} \mid \text{DT}) = 0.05$
 $P(\text{Mary} \mid \text{NN}) = 0.2$
 $P(\text{Mary} \mid \text{VBD}) = 0.02$



Viterbi Decoding

Matrix: One column

In each cell: compare max. probability

$$P(\text{DT} \mid \text{DT}) = 0.01$$

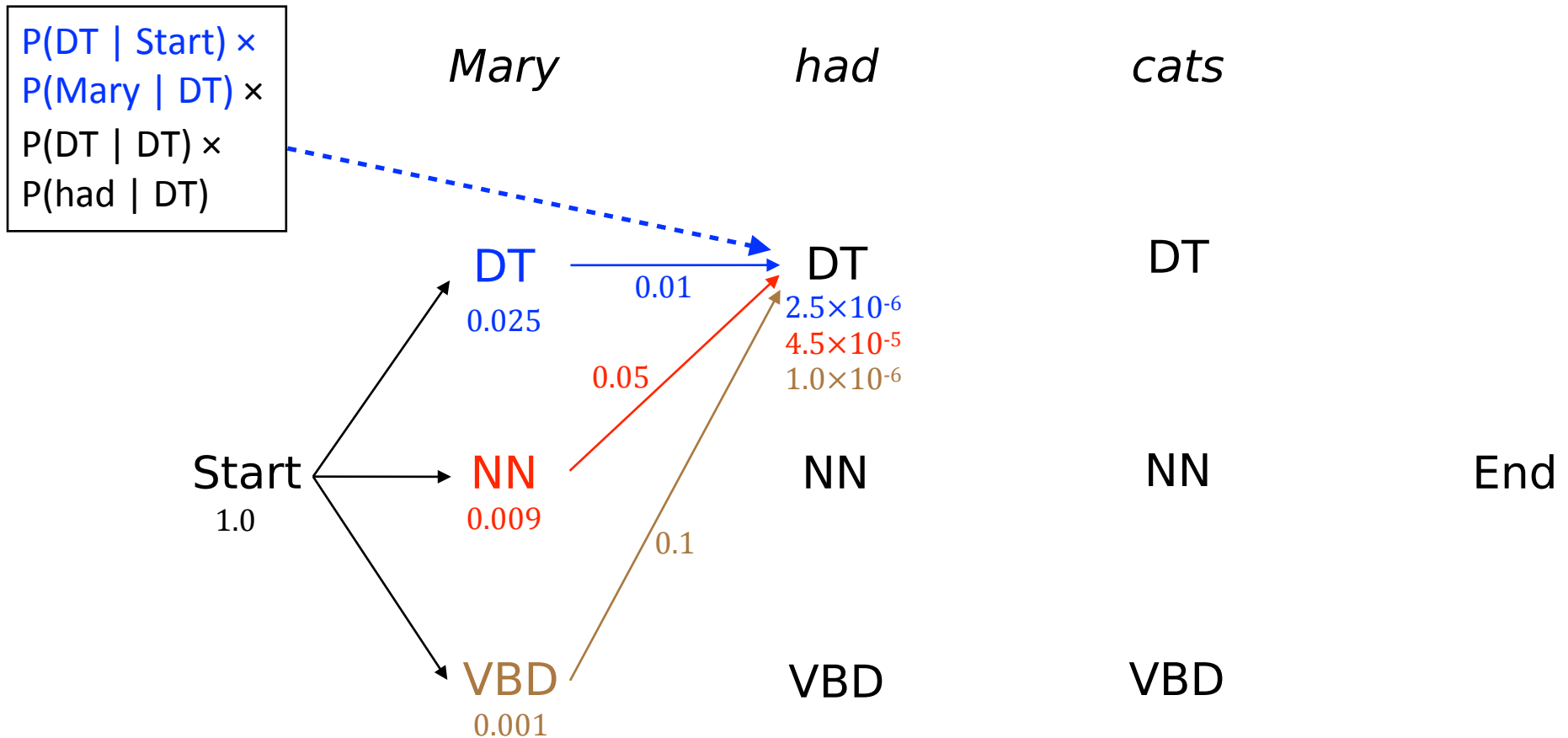
$$P(\text{DT} \mid \text{NN}) = 0.05$$

$$P(\text{DT} \mid \text{VBD}) = 0.1$$

$$P(\text{had} \mid \text{DT}) = 0.01$$

$$P(\text{had} \mid \text{NN}) = 0.05$$

$$P(\text{had} \mid \text{VBD}) = 0.2$$



Viterbi Decoding

Matrix: One column

In each cell: compute max. probability

$$P(\text{DT} \mid \text{DT}) = 0.01$$

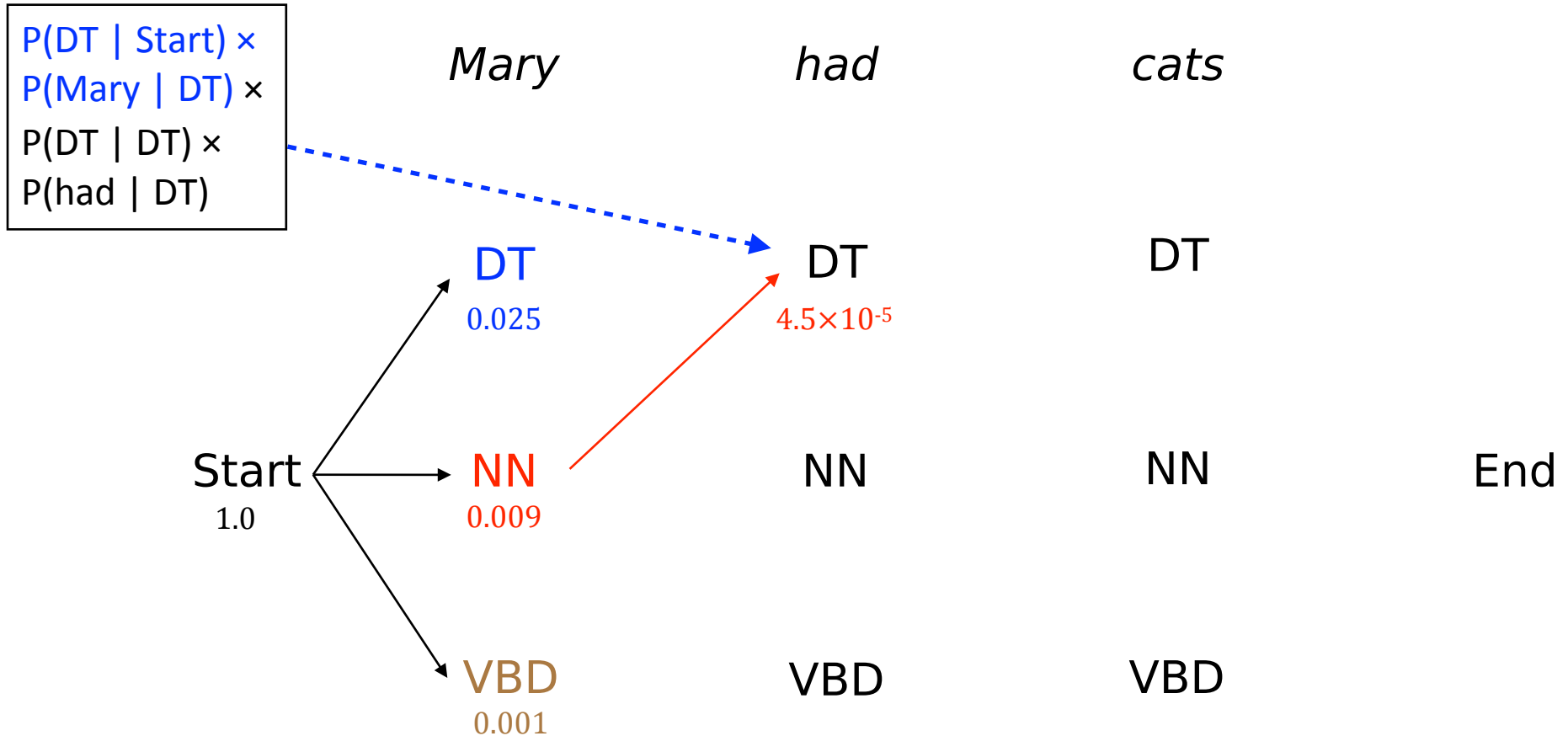
$$P(\text{DT} \mid \text{NN}) = 0.05$$

$$P(\text{DT} \mid \text{VBD}) = 0.1$$

$$P(\text{had} \mid \text{DT}) = 0.01$$

$$P(\text{had} \mid \text{NN}) = 0.05$$

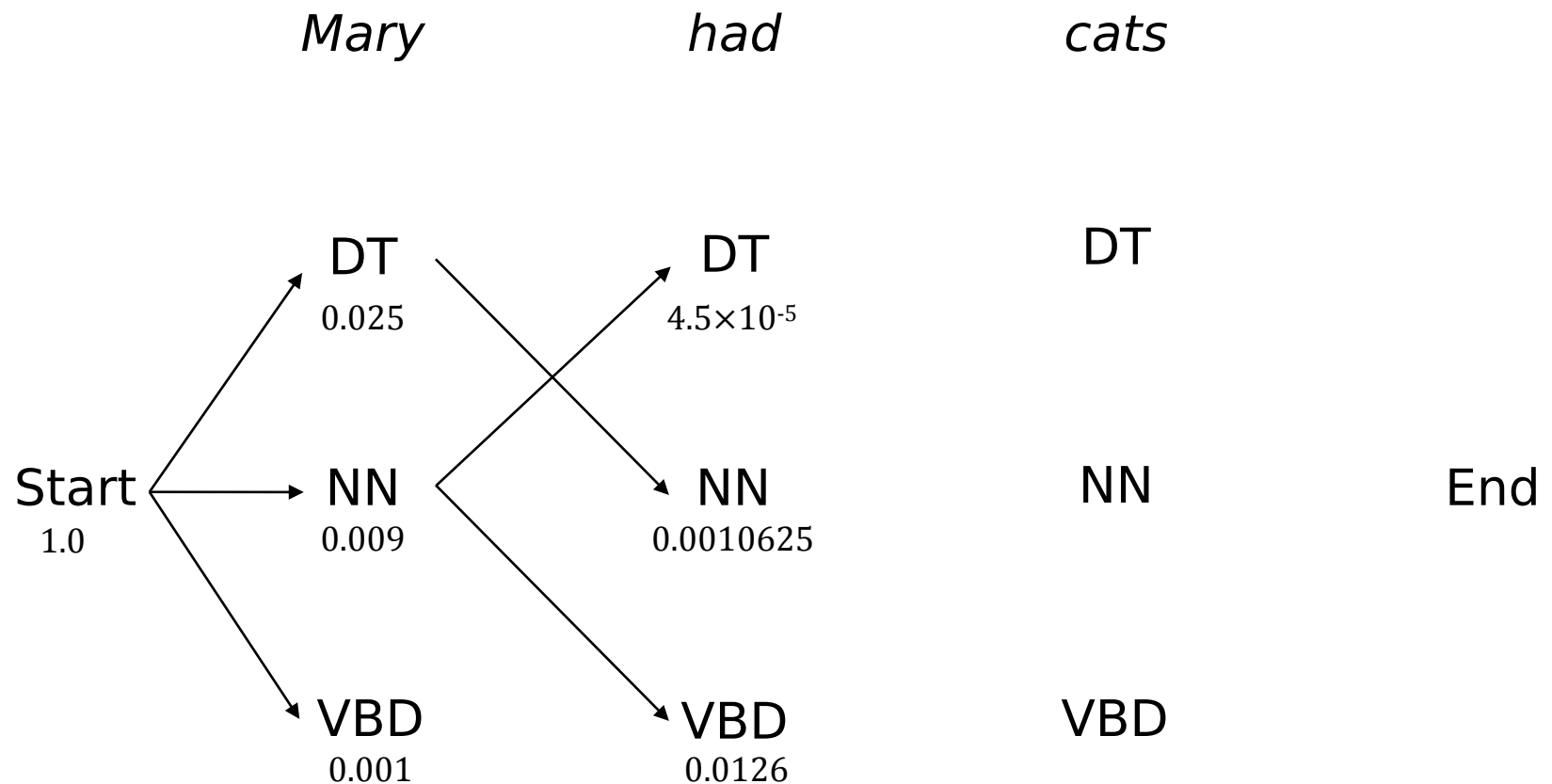
$$P(\text{had} \mid \text{VBD}) = 0.2$$



Viterbi Decoding

Matrix: One column per Word, one row per POS-tag

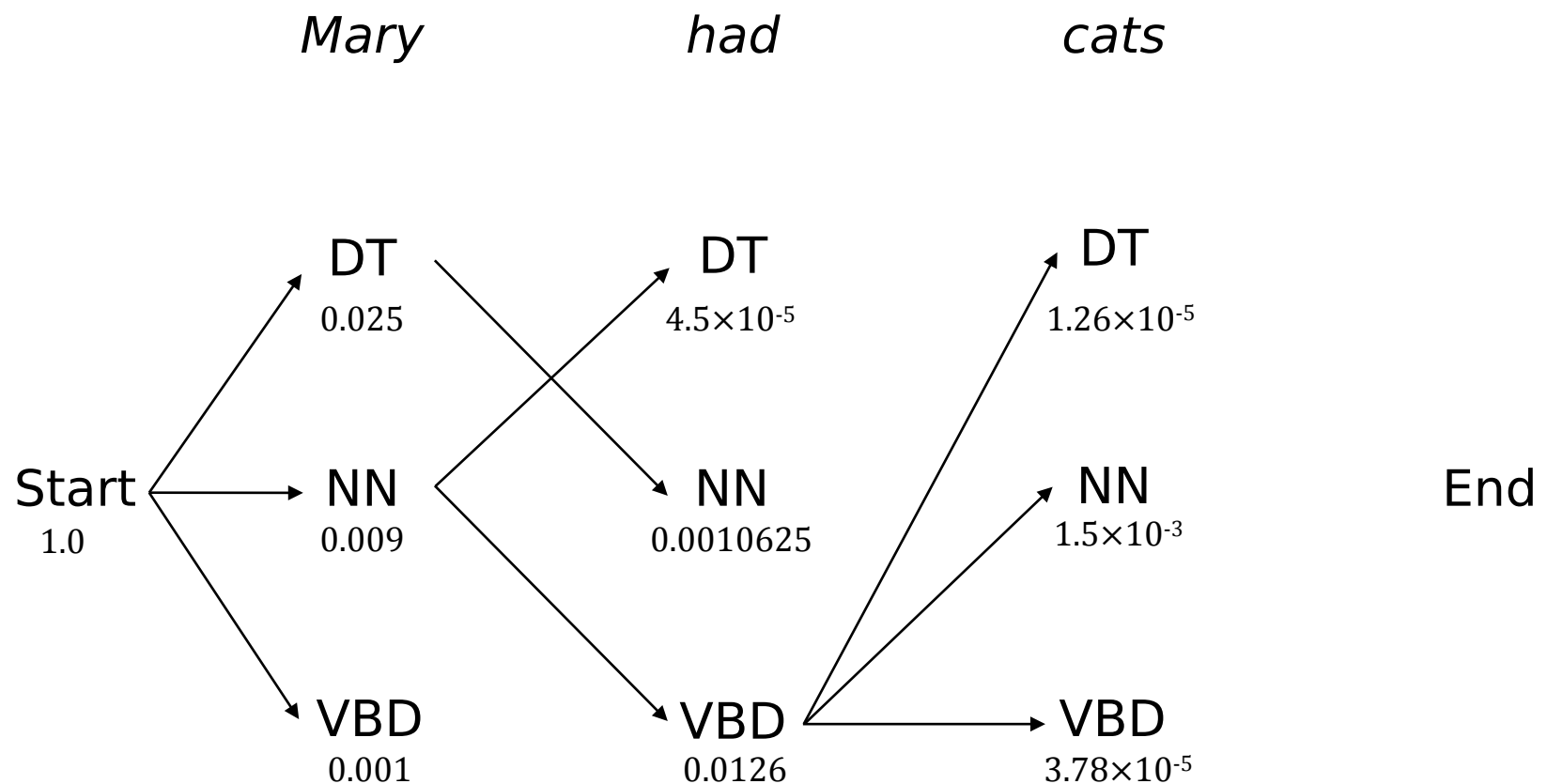
In each cell: compute max. probability + back pointer



Viterbi Decoding

Matrix: One column per Word, one row per POS-tag

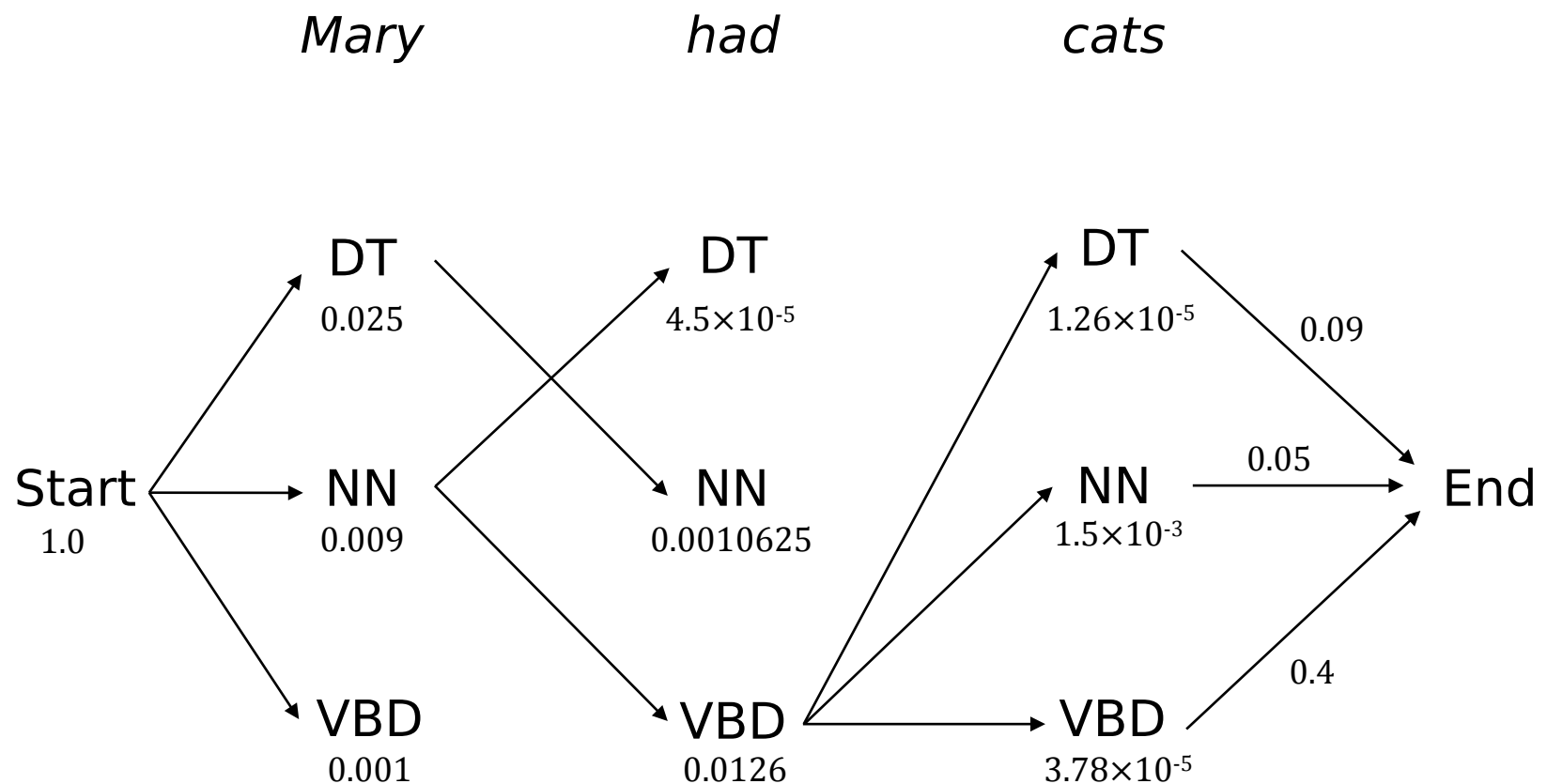
In each cell: compute max. probability + back pointer



Viterbi Decoding

Matrix: One column per Word, one row per POS-tag

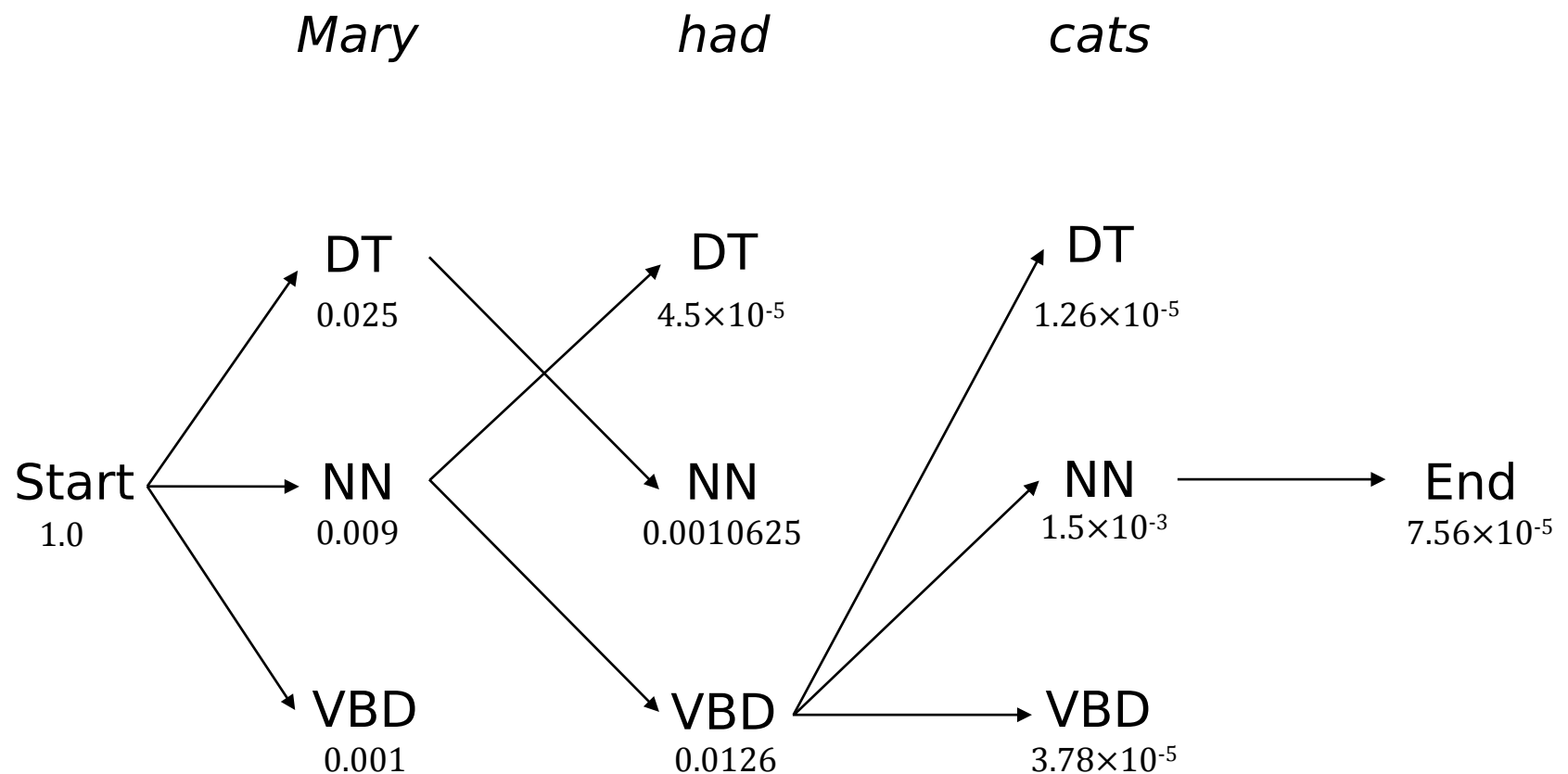
In each cell: compute max. probability + back pointer



Viterbi Decoding

Matrix: One column per Word, one row per POS-tag

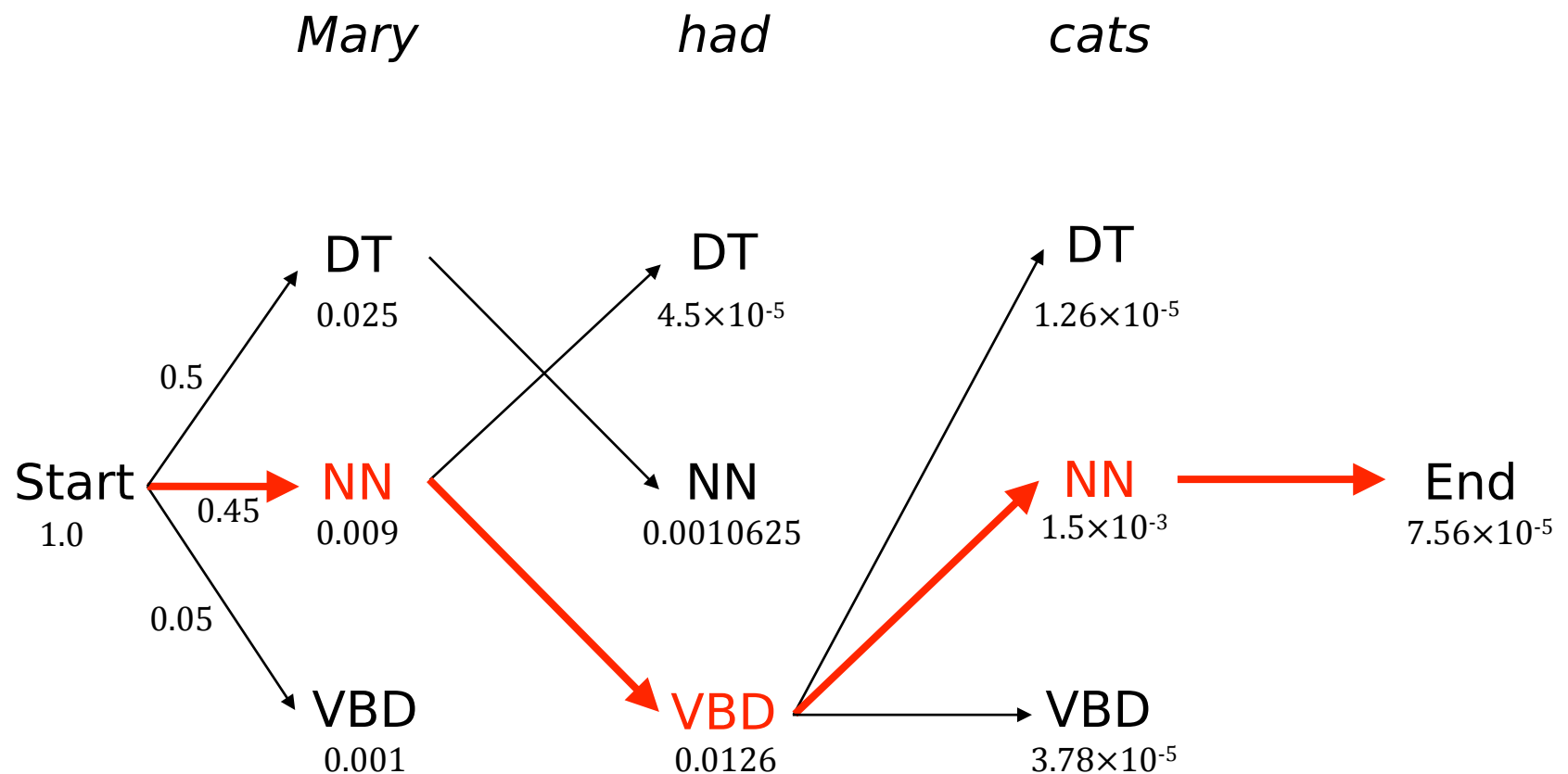
In each cell: compute max. probability + back pointer



Viterbi Decoding

Matrix: One column per Word, one row per POS-tag

In each cell: compute max. probability + back pointer



Pseudo code

```

function decode_viterbi(observations of len T, state-graph of len N):
  create a path probability matrix viterbi[N+2,T]
  for each state s from 1 to N do      # initialization, states = tags
    viterbi[s,1]  $\leftarrow a_{0,s} * b_s(o_1)$ 
    backpointer[s,1]  $\leftarrow 0$ 
  for each t from 2 to T:
    for each state s from 1 to N do:
      viterbi[s,t]  $\leftarrow \max_{s'} \text{viterbi}[s', t-1] * a_{s',s} * b_s(o_t)$ 
      backpointer[s,t]  $\leftarrow \arg \max_{s'} \text{viterbi}[s', t-1] * a_{s',s} * b_s(o_t)$ 
    viterbi[qF,T]  $\leftarrow \max_s \text{viterbi}[s, T] * a_{s,q_F}$       # termination step
    backpointer[qF,T]  $\leftarrow \arg \max_s \text{viterbi}[s, T] * a_{s,q_F}$ 
  return the backtrace path by following backpointers starting at backpointer[qF,T]

```

References

- Jurafsky & Martin: Speech and Language Processing. Pearson. Chapters 5 and 6.