NATURAL LANGUAGE PROCESSING

WITH DEEP LEARNING

Lecturer: Javad PourMostafa

NLP981

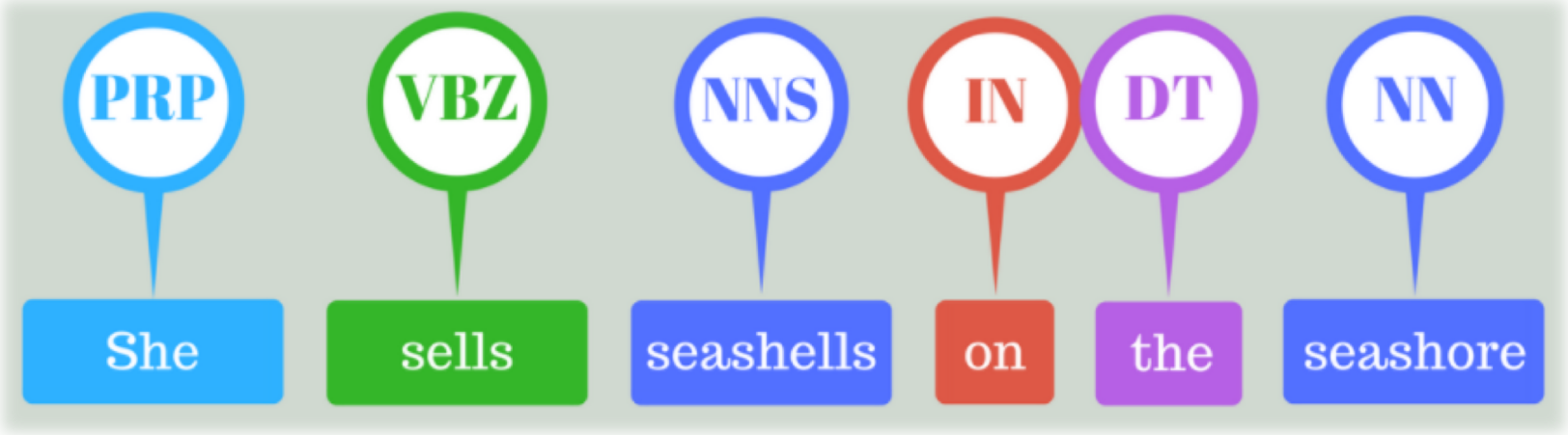
Hidden Markov Model for Sequence Labeling

SEQUENCE LABELING

- Problem:
  - Given a sequence of tokens
  - Find the most probable labels for these tokens
- Example:
  - POS
  - NER
  - Semantic Slot Filling

PART-OF-SPEECH AS A SEQUENCE LABELING TASK

- Tokens: Words
- Labels: POS tags
- Example:



NAME ENTITY RECOGNITION AS A SEQUENCE LABELING TASK

- Tokens: Words
- Labels: Name entities (Persons, locations, amounts, dates and times and so on)
- Example:

Vancouver is a coastal seaport city on the mainland of British Columbia.
The city's mayor is Gregor Robertson.

Vancouver is a coastal seaport city on the mainland of British Columbia.
The city's mayor is Gregor Robertson.

APPROACHES TO SEQUENCE LABELING

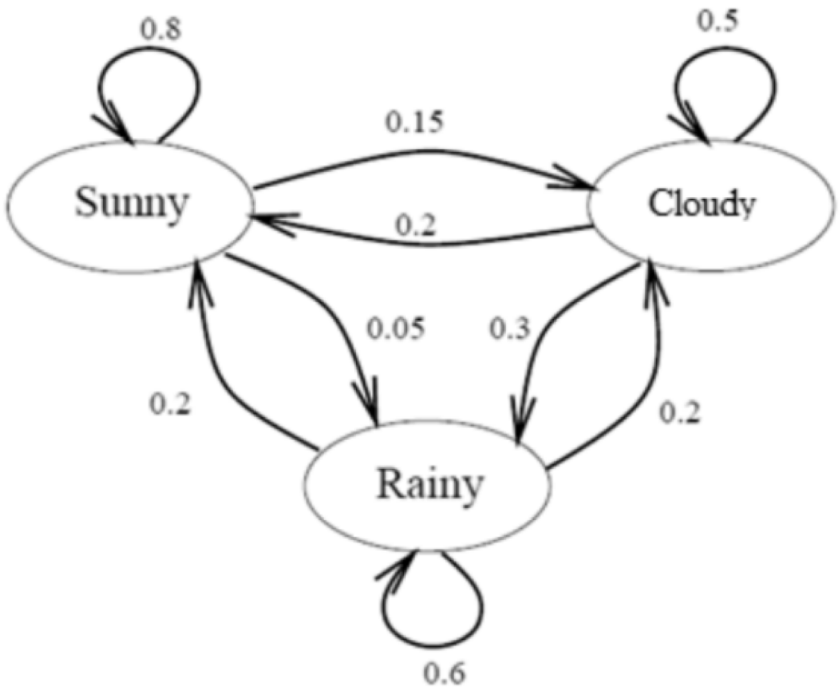
1. Rule-based models
   - EngCG, https://www.researchgate.net/publication/2660191_EngCG_tagger_Version_2
2. Separate label classifiers for each token
   - It's good when a sequence is short!
3. Sequence models (HMM, CRF, MEMM)
4. Neural Networks

MARKOV MODEL SCENARIO (A)

- There are **three** kind of weather conditions:
  - **Rainy, Sunny, Cloudy**
- Peter is a small kid
- He loves to play outside.
- He loves when the weather is sunny but hates the rainy weather.
- His mother wants to predict the weather based on her observation in the morning.

MARKOV CHAIN MODEL

- Use Markov chain model
- Construct the state diagram



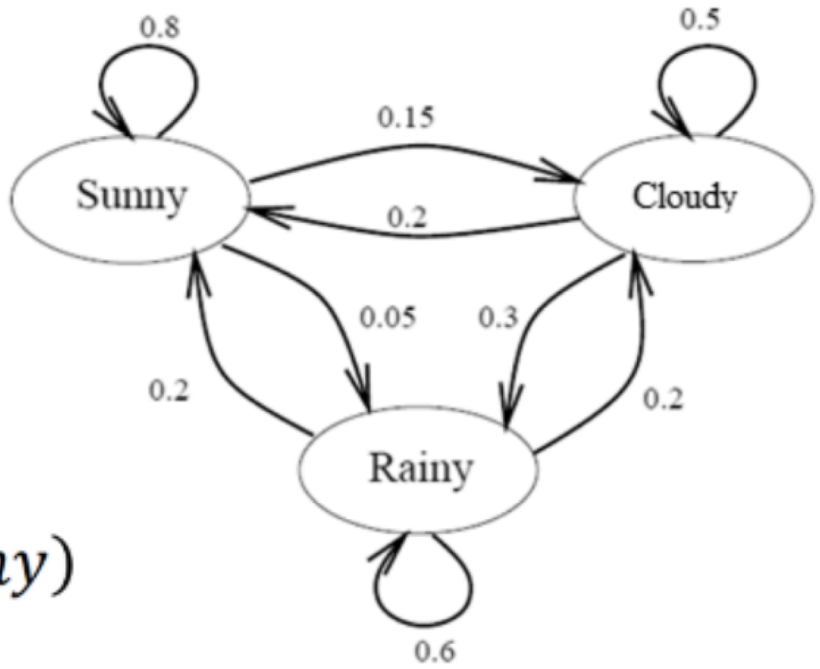
$$P(Sunny|Sunny) = 0.8$$
$$P(Rainy|Sunny) = 0.05$$
$$P(Cloudy|Sunny) = 0.15$$
} 1

$$P(Sunny|Rainy) = 0.2$$
$$P(Rainy|Rainy) = 0.6$$
$$P(Cloudyy|Rainy) = 0.2$$
} 1

$$P(Sunny|Cloudy) = 0.2$$
$$P(Rainy|Cloudy) = 0.3$$
$$P(Cloudy|Cloudy) = 0.5$$
} 1

MARKOV CHAIN MODEL: 1ST EXERCISE

- Given that today is **Sunny**, what's the probability that tomorrow is **Sunny** and the next day **Rainy**?



$$P(q_2, q_3|q_1) = P(q_2|q_1)P(q_3|q_1, q_2)$$

$$= P(q_2|q_1)\ P(q_3|q_2)$$

$$= P(Sunny|Sunny)\ P(Rainy|Sunny)$$

$$= (0.8)(0.05)$$

$$= 0.04$$

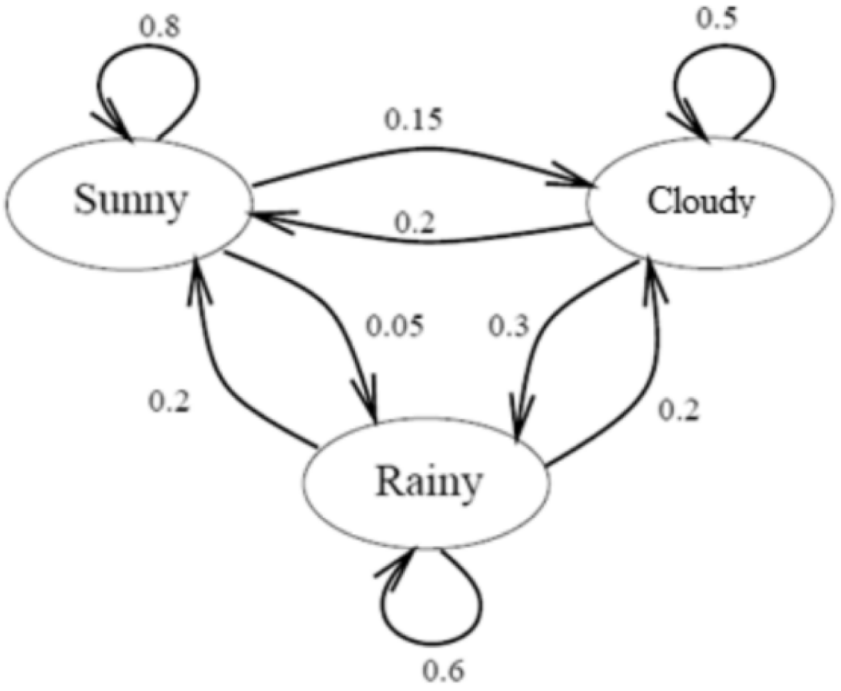
MARKOV CHAIN MODEL: 2ND EXERCISE

- Assume that yesterday's weather was **Rainy**, and today is **Cloudy**, what is the probabilities that tomorrow Will be **Sunny?**

$$P(q_3|q_1, q_2) = P(q_3|q_2)$$

$$= P(Sunny|Cloudy)$$

$$= 0.2$$

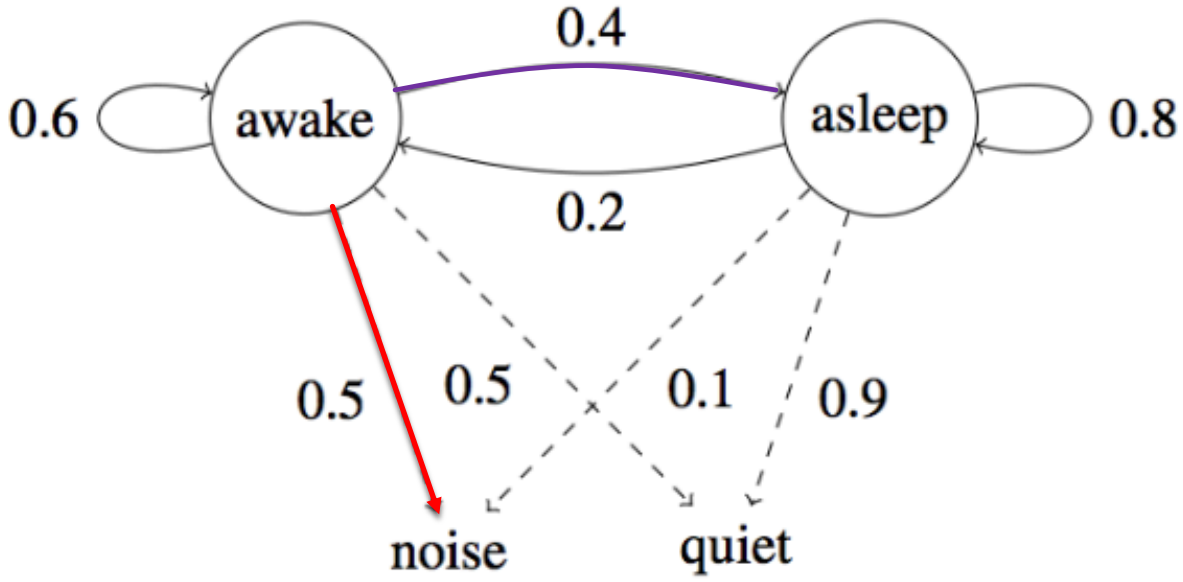


HIDDEN MARKOV MODEL SCENARIO (B)

- Small kid Peter again!
- As a caretaker, one of the most important tasks for you is to tuck Peter into bed and make sure he is sound asleep.
- Once you've tucked him in, you want to make sure he's actually asleep and not up to some mischief.
- You cannot, however, enter the room again
- How?
  - All you have to decide are the noises might come from the room.

HIDDEN MARKOV MODEL SCENARIO (CONT'D)

- Either the room is **quiet** or there is **noise**
- Peter's mother said: "May the sound be with you"
- His mother has given you the following state diagram:
- There are two probabilities (matrices)
  - Emission probabilities
  - Transition probabilities



Hello Caretaker, this might help. ~Peters mother. Have fun !

POS TAGGING WITH HMM

- $X = x_1, x_2, \ldots\ldots, x_T$ (input)
- $Y = y_1, y_2, \ldots\ldots, y_T$ (labels)
- **T** is the length of sequence
- We need to find the most probable sequence of tags given the sentence:

$$y = argmax_y\, p(y|x) = argmax_y\, p(x, y)$$

- Denominator has been skipped (why?)

POS TAGGING WITH HMM (CONT'D)

$$p(x, y) = \mathrm{p(x|y)\ p(y)} = \prod_{t=1}^{T} p(x_t|y_t)\ p(y_t|\ y_{t-1})$$

Observable Hidden

- Output independence:

$$p(x|y) \approx \prod_{t=1}^{T} p(x_t|\ y_t)$$

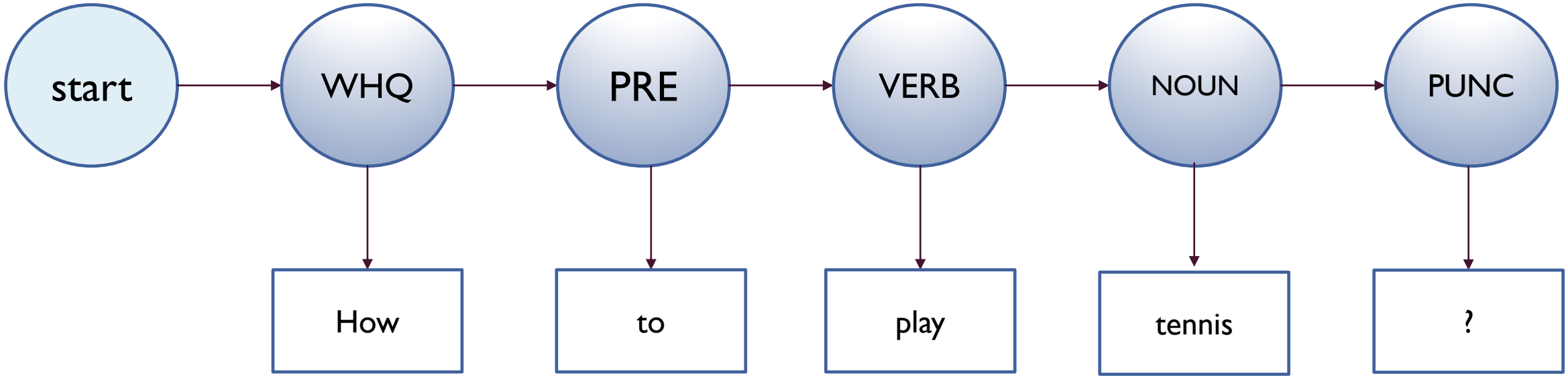
- Markov assumption:

$$p(y) \approx \prod_{t=1}^{T} p(y_t|\ y_{t-1})$$

TEXT GENERATION IN HMM

- Assume that the text is generated is the following manner:
  1. Chooses the next POS tag given the previous tag
  2. Given the current tag, one generates another word
- So, the neighboring words do no depend on each other, but they depend on the underlying tags.

TEXT GENERATION IN HMM: AN EXAMPLE



- A supervised sequence tagging treebank: http://universaldependencies.org/

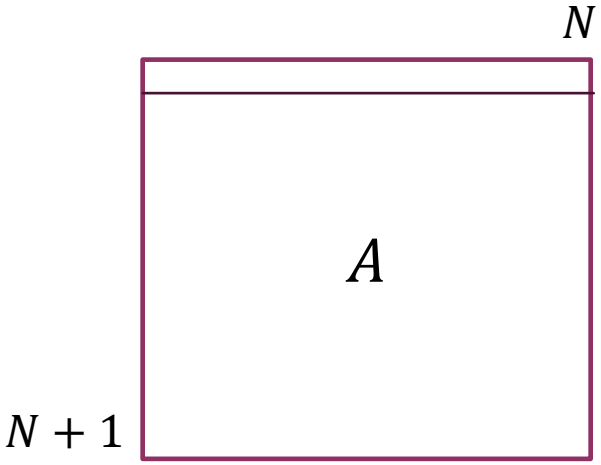
FORMAL DEFINITION OF HMM

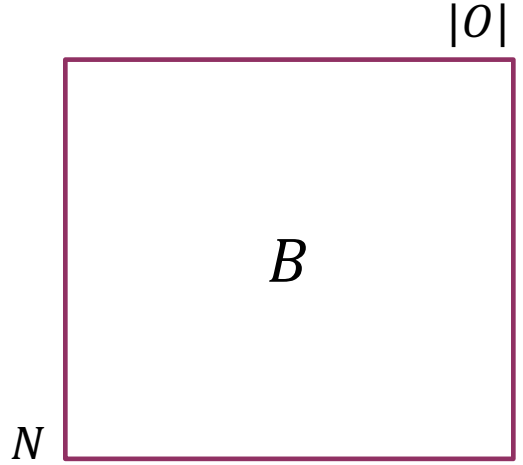
- An HMM is specified by the following components:
  - The set $S = s_1, s_2, \ldots, s_N$ of hidden states
  - The start state s_0
  - The matrix **A** of transition probabilities: $a_{ij} = p(s_j|s_i)$
  - The set of **O** of possible visible outcomes
  - The matrix **B** of output (emission) probabilities: $b_{kj} = p(o_k|s_i)$

FITTING THE MODEL (LABELED DATA)

- **How many parameters does the HMM model have** $(K = |O|)$ **?**

$$N(N+1) + NK$$

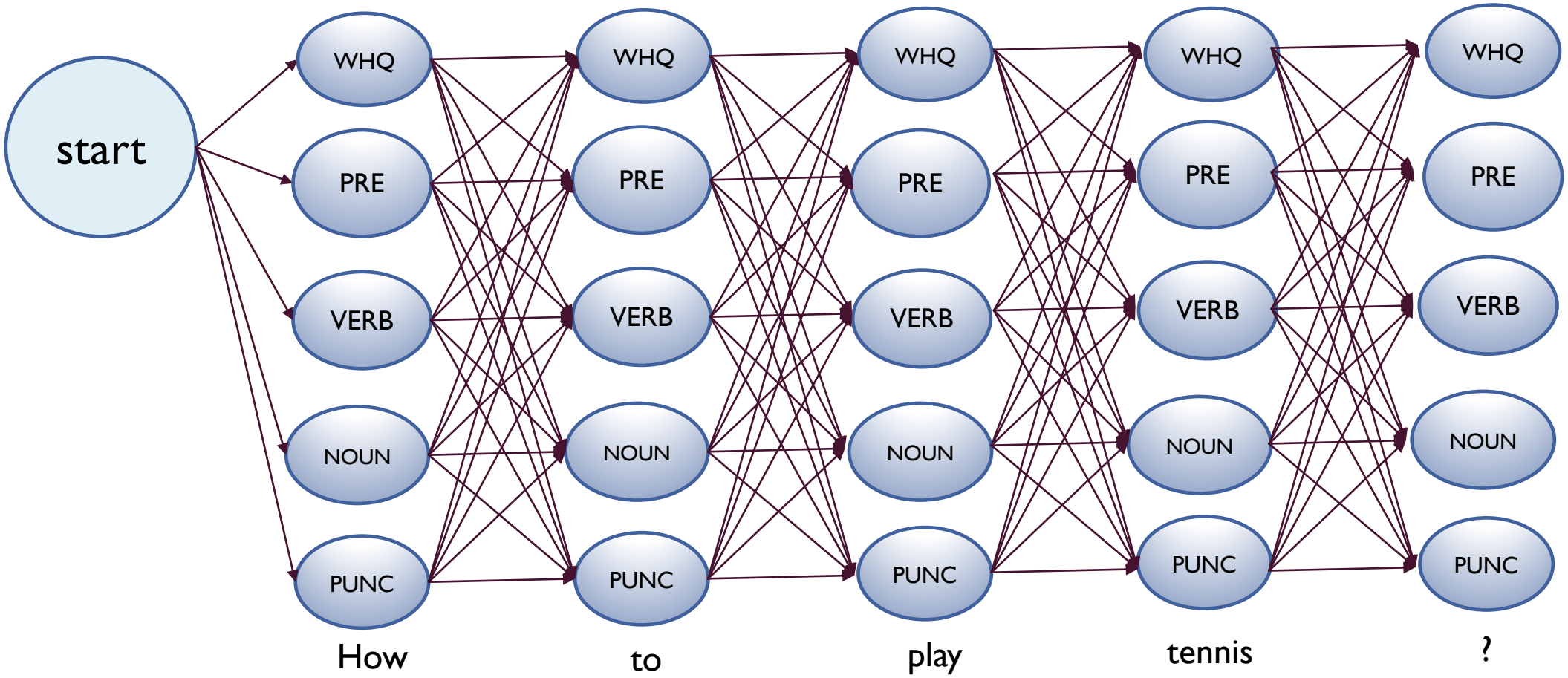




$$a_{ij} = p(s_j|s_i) = \frac{C(s_i \to s_j)}{C(s_i)}$$

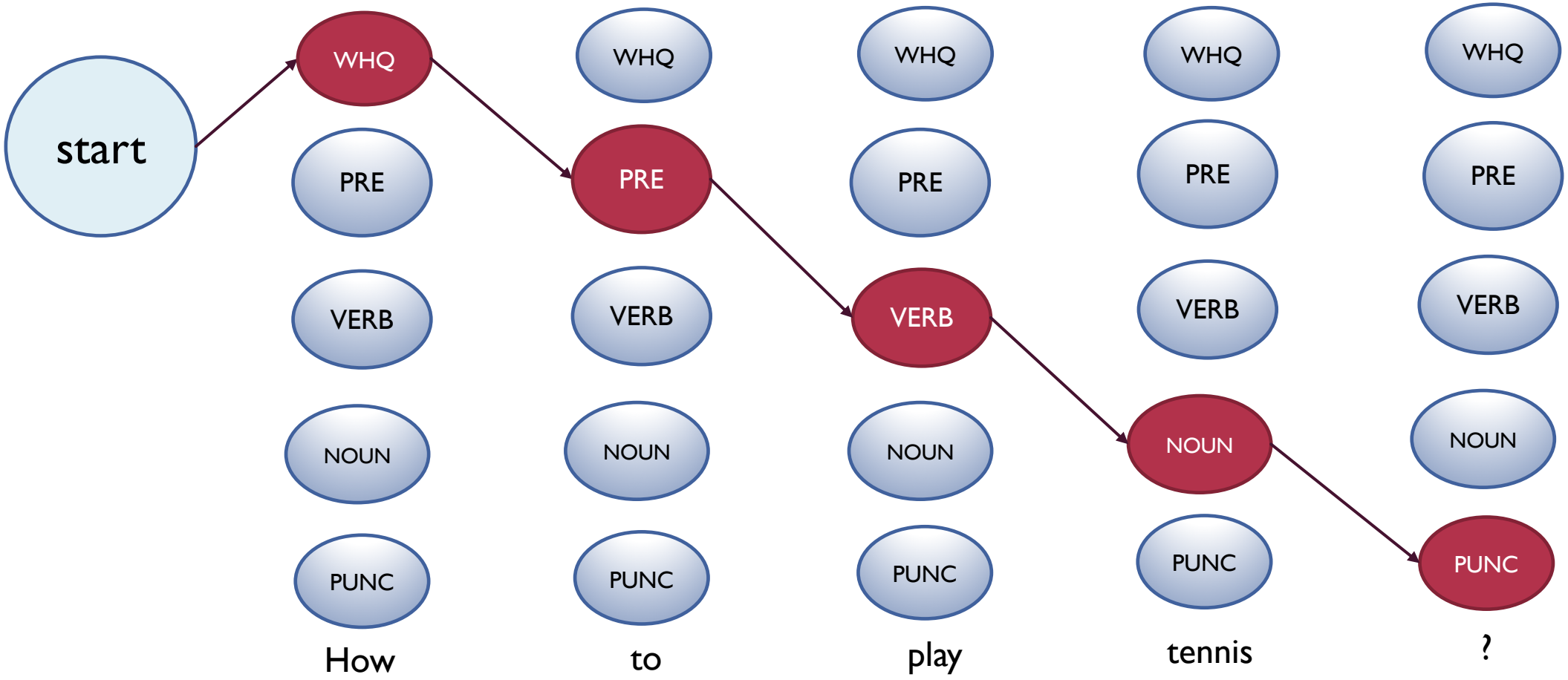
$$b_{ik} = p(o_k|s_i) = \frac{C(s_i \to o_k)}{C(s_i)}$$

WHAT DOES THE GRAPH LOOK LIKE?



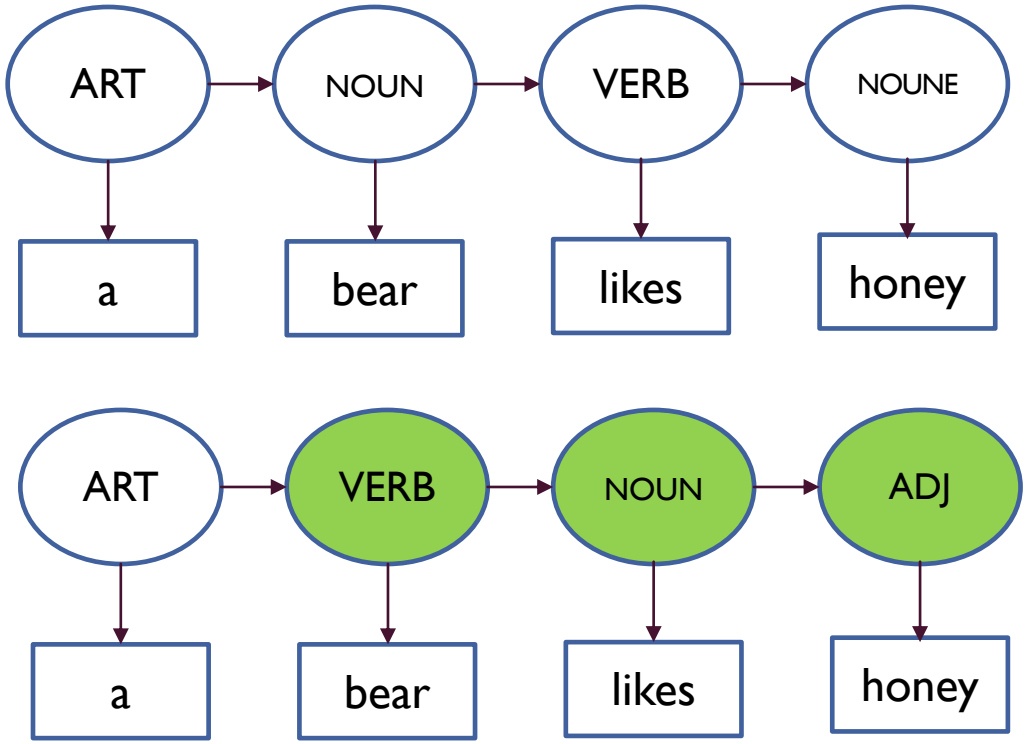
$Note$: $Possible\ tag\ sequences = ?\quad Tag^{Length\ of\ sentence}$

THE BEST PATH IN OUR POS SEQUENCE



MOTIVATION

- The same output sentence can be generated by different sequences of hidden states:



VITERBI ALGORITHM: EXAMPLE

Corpus:

eat/VB breakfast/NN at/IN morning/NN time/NN
take/VB time/NN with/IN arrow/NN projects/NN
horse/NN riders/NN like/VB the/DT airport/NN
paper/NN flies/VB on/IN hydrogen/NN gas/NN
bees/NN sting/VB like/IN some/DT flies/NN
beans/NN soil/VB an/DT iron/NN grill/NN
flies/NN smell/VB an/DT arrow/NN drink/NN
people/NN like/VB an/DT army/NN arrow/NN
dinner/NN time/NN flies/VB all/DT day/NN
horse/NN flies/NN time/VB morning/NN rays/NN

Matrix of Transition probabilities

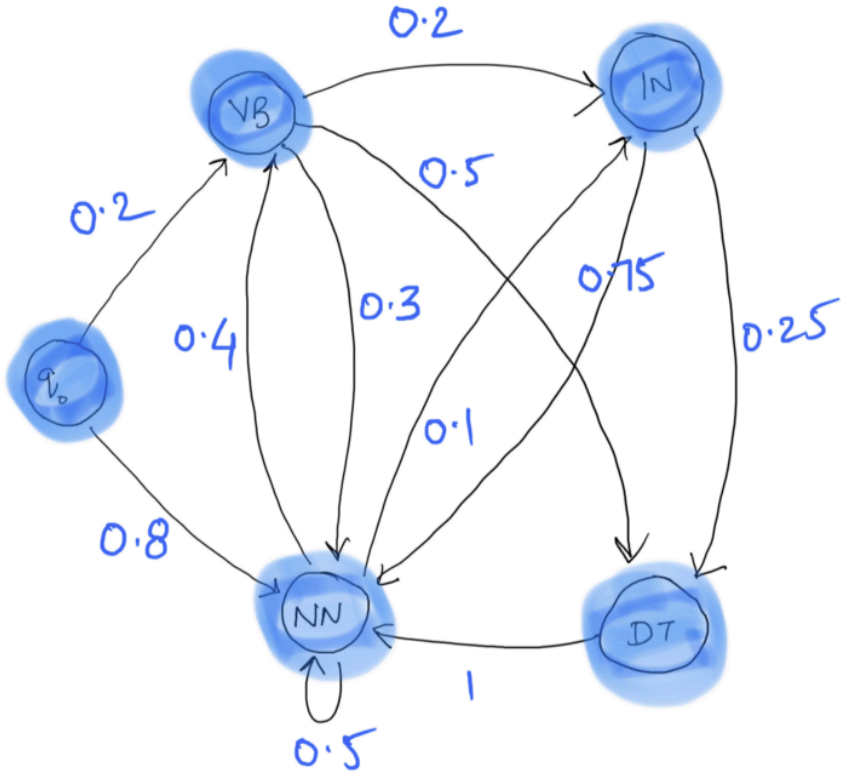
	VB	NN	IN	DT
Start	0.2	0.8	0	0
VB	0	0.3	0.2	0.5
NN	0.4	0.5	0.1	0
IN	0	0.75	0	0.25
DT	0	1	0	0

VITERBI ALGORITHM: EXAMPLE (CONT'D)

Corpus:

eat/VB breakfast/NN at/IN morning/NN time/NN
take/VB time/NN with/IN arrow/NN projects/NN
horse/NN riders/NN like/VB the/DT airport/NN
paper/NN flies/VB on/IN hydrogen/NN gas/NN
bees/NN sting/VB like/IN some/DT flies/NN
beans/NN soil/VB an/DT iron/NN grill/NN
flies/NN smell/VB an/DT arrow/NN drink/NN
people/NN like/VB an/DT army/NN arrow/NN
dinner/NN time/NN flies/VB all/DT day/NN
horse/NN flies/NN time/VB morning/NN rays/NN

Graph of Transition probabilities



VITERBI ALGORITHM: EXAMPLE (CONT'D)

Corpus:

eat/VB breakfast/NN at/IN morning/NN time/NN
take/VB time/NN with/IN arrow/NN projects/NN
horse/NN riders/NN like/VB the/DT airport/NN
paper/NN flies/VB on/IN hydrogen/NN gas/NN
bees/NN sting/VB like/IN some/DT flies/NN
beans/NN soil/VB an/DT iron/NN grill/NN
flies/NN smell/VB an/DT arrow/NN drink/NN
people/NN like/VB an/DT army/NN arrow/NN
dinner/NN time/NN flies/VB all/DT day/NN
horse/NN flies/NN time/VB morning/NN rays/NN

Matrix of Emission probabilities

Note: Ideally we should be looking at all combinations of tags and words in the corpus.

Since that would be too much, we will only consider emission probabilities for the following sentence:

Time flies like an arrow

VITERBI ALGORITHM: EXAMPLE (CONT'D)

Corpus:

eat/VB breakfast/NN at/IN morning/NN time/NN
take/VB time/NN with/IN arrow/NN projects/NN
horse/NN riders/NN like/VB the/DT airport/NN
paper/NN flies/VB on/IN hydrogen/NN gas/NN
bees/NN sting/VB like/IN some/DT flies/NN
beans/NN soil/VB an/DT iron/NN grill/NN
flies/NN smell/VB an/DT arrow/NN drink/NN
people/NN like/VB an/DT army/NN arrow/NN
dinner/NN time/NN flies/VB all/DT day/NN
horse/NN flies/NN time/VB morning/NN rays/NN

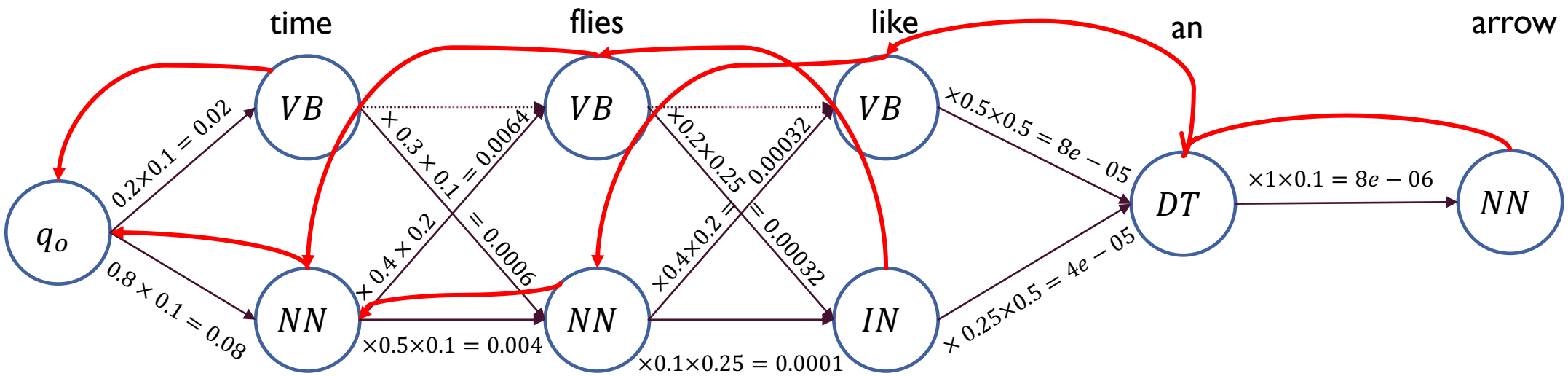
Matrix of Emission probabilities

	times	flies	like	an	arrow
VB	0.1	0.2	0.2	0	0
NN	0.1	0.1	0	0	0.1
IN	0	0	0.25	0	0
DT	0	0	0	0.5	0

VITERBI ALGORITHM: EXAMPLE (CONT'D)

	VB	NN	IN	DT
Start	0.2	0.8	0	0
VB	0	0.3	0.2	0.5
NN	0.4	0.5	0.1	0
IN	0	0.75	0	0.25
DT	0	1	0	0

	time	flies	like	an	arrow
VB	0.1	0.2	0.2	0	0
NN	0.1	0.1	0	0	0.1
IN	0	0	0.25	0	0
DT	0	0	0	0.5	0



VITERBI ALGORITHM

- Find the best path and also the most probable tags.
- Solve this problem by using **Dynamic Programming**
- **Formalize the Viterbi decoding algorithm:**
  - $Q_{t,s}$ **the most probable sequence of hidden states of length T**
  - $Q_{t,s}$ **generates** $o_1, \ldots, o_t$
  - $q_{t,s}$ **is the probability of this sequence**

$$q_{t,s} = max_{\acute{s}}\, q_{t,-1,\acute{s}} \times p(s|\acute{s}) \times p(o_t|s)$$