Viterbi Algorithm

POS tagging using Generative Model

Viterbi Algorithm

- Make an inference based on a trained model and some observed data
- Answer the question
- What is the choice of states such that the joint probability reaches maximum?

$$X^*_{0:T} = \underset{X_{0:T}}{\operatorname{argmax}} P[X_{0:T}|Y_{0:T}]$$

Viterbi Algorithm

- To find best set of states we use following recursive formula

$$\mu(X_k) = \max_{X_{0:k-1}} P[X_{0:k}, Y_{0:k}] = \max_{X_{k-1}} \mu(X_{k-1}) P[X_k|X_{k-1}] P[Y_k|X_k]$$

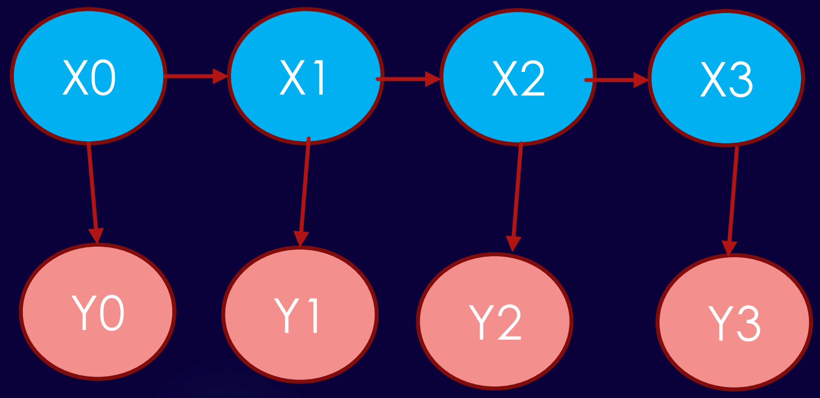
- Let us substitute k=1,2,3

$$\mu(X_0) = P[Y_0|X_0]P[X_0]$$

$$\mu(X_1) = \max_{X_0} \mu(X_0)P[X_1|X_0]P[Y_1|X_1]$$

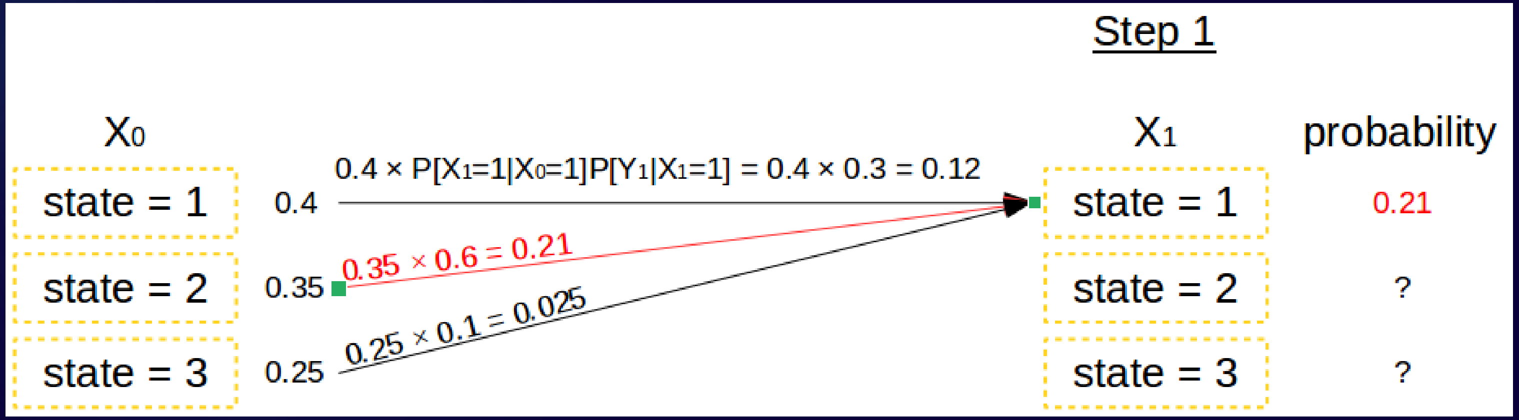
$$\mu(X_2) = \max_{X_1} \mu(X_1)P[X_2|X_1]P[Y_2|X_2]$$

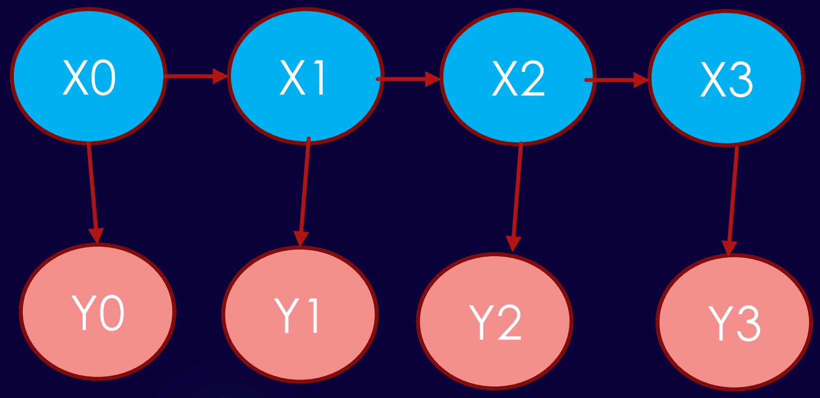
$$\mu(X_3) = \max_{X_2} \mu(X_2)P[X_3|X_2]P[Y_3|X_3]$$



Viterbi Algorithm

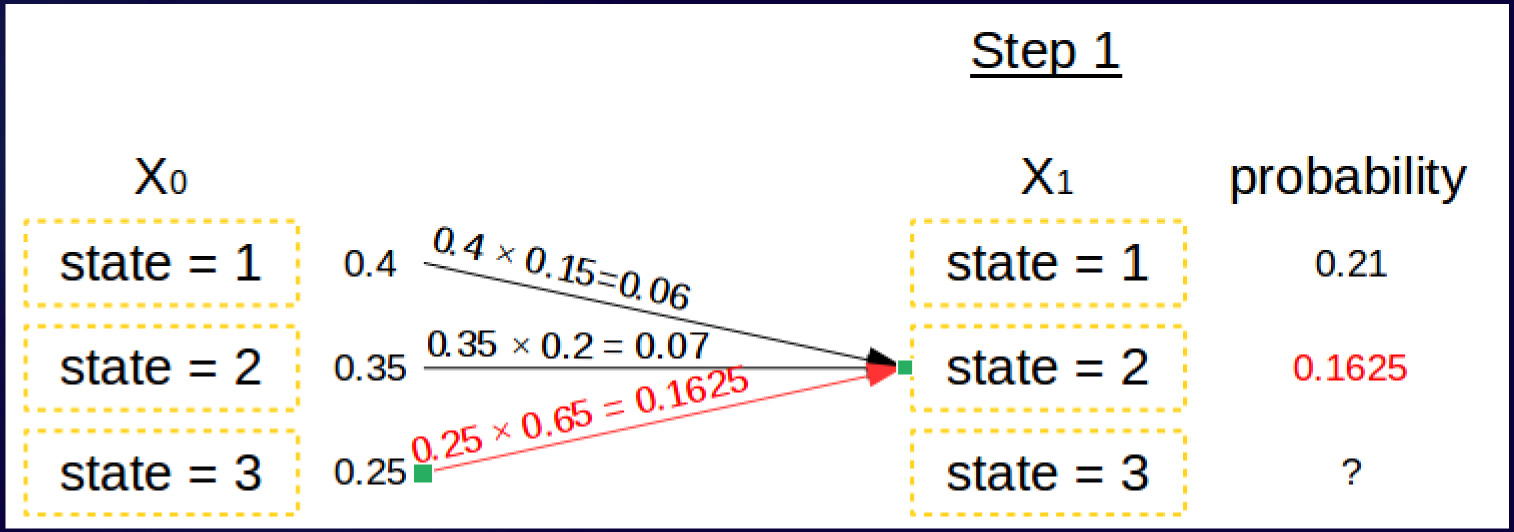
- Assume there are 3 possible states at each step
- At each step maximize the probability
- For X1, state=1, the best possible state at X0, state=2 is chosen

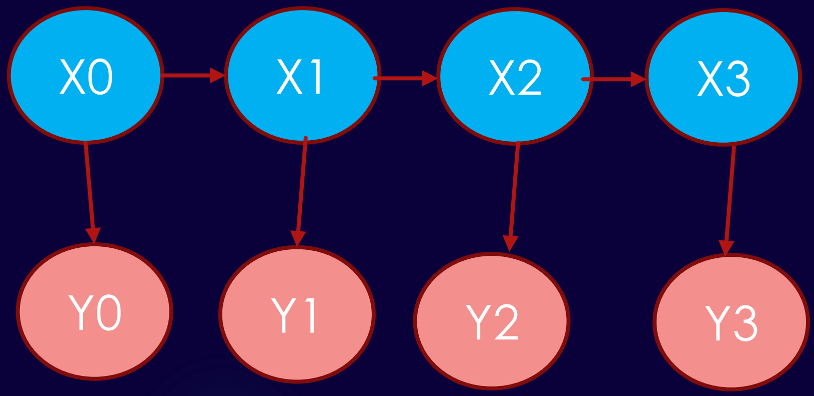




Viterbi Algorithm

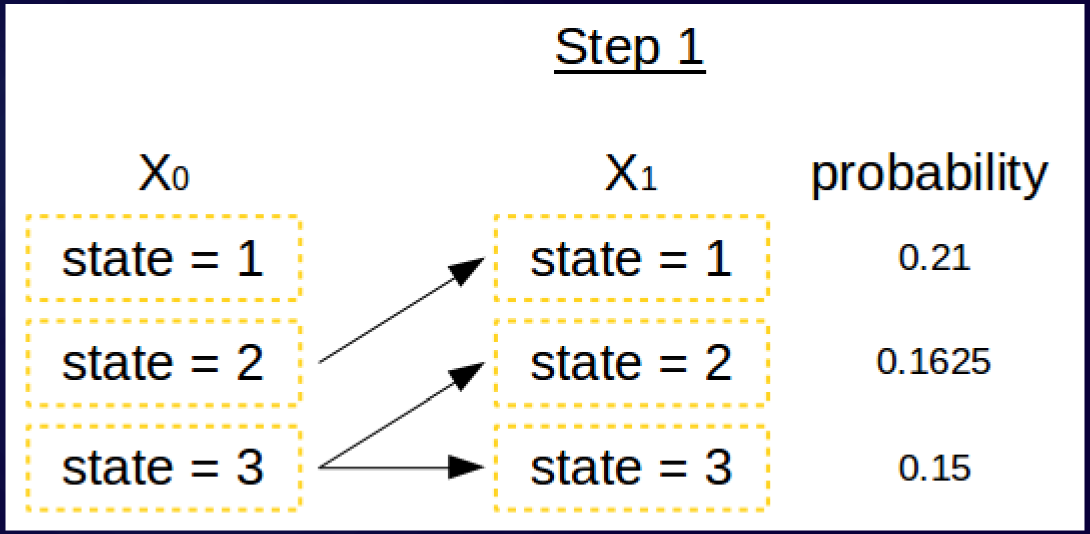
- For X1, state=2, the best possible state at X0, state=3 chosen

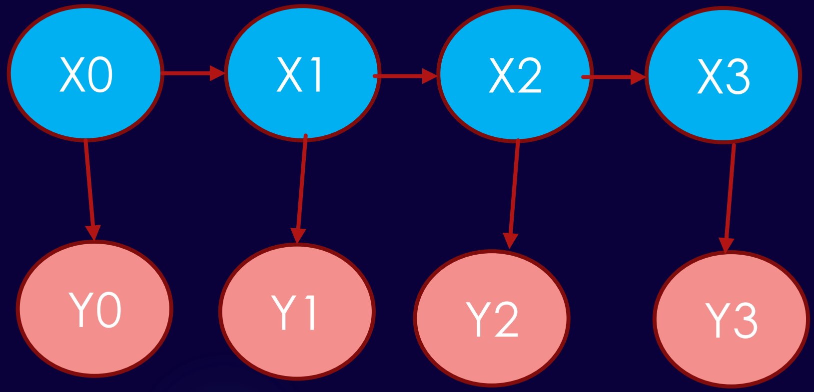




Viterbi Algorithm

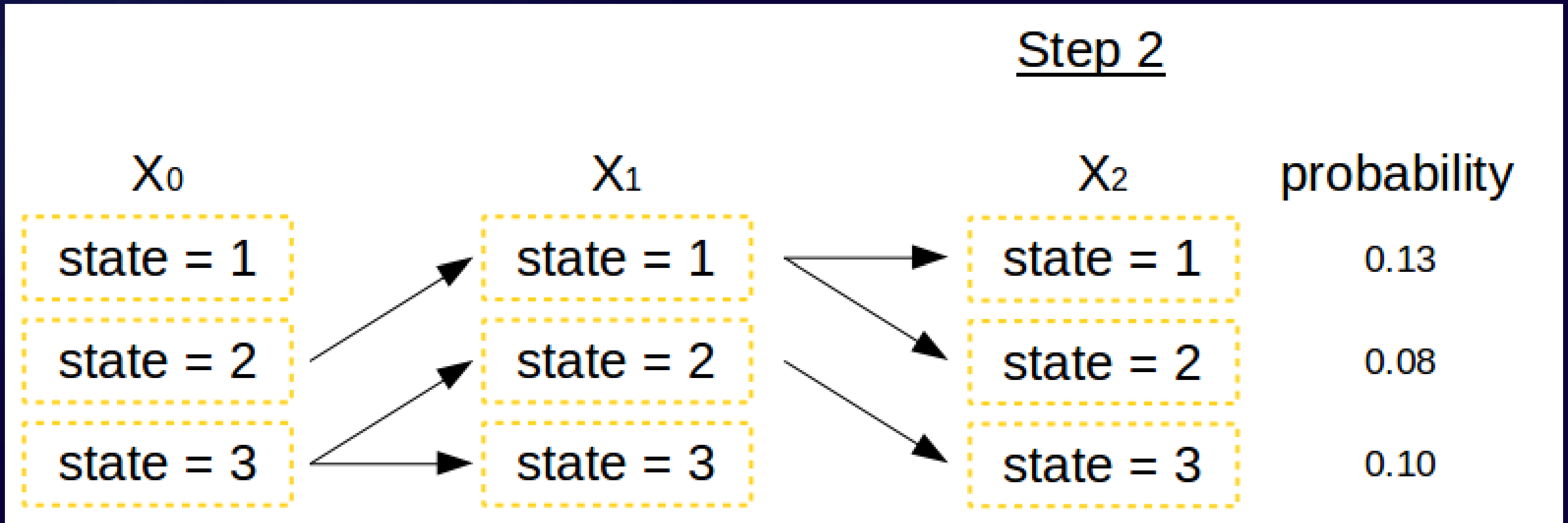
- For X1, state=3 also calculation is done and we get max value 0.15





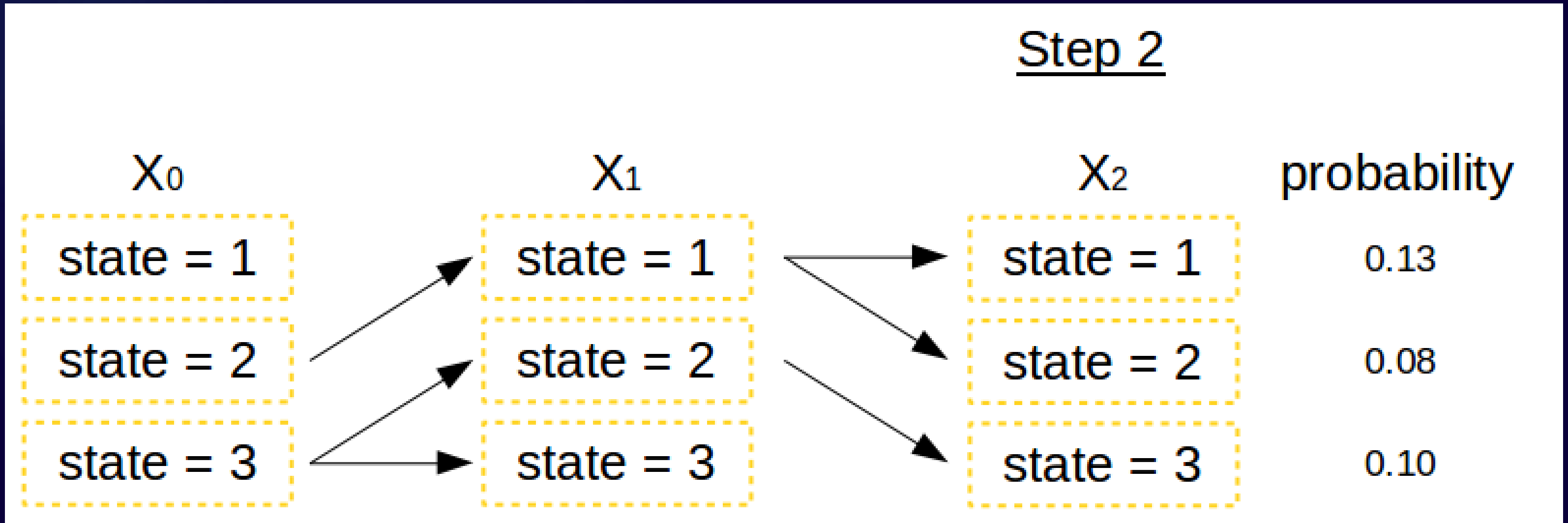
Viterbi Algorithm

- Repeat same steps to get to step 2
- if we end the inference at step 2, then the most likely ending state would be state = 1



Viterbi Algorithm

- Rest of the previous states could be back-traced through the arrows, which are
- state 2 at time 0,
- state 1 at time 1,
- and state 1 at time 2
- The second likely path is 3–2–3, and the least likely path is 2–1–2. It is very unlikely that the path starts with state 1



Example 2

POS Tagging -Viterbi Algorithm

- The fans watch the race

DT	NN VB	NN VB	DT	NN VB

- HMM

$$DT \rightarrow \begin{matrix} NN \\ VB \end{matrix} \rightarrow \begin{matrix} NN \\ VB \end{matrix} \rightarrow DT \rightarrow \begin{matrix} NN \\ VB \end{matrix}$$

- The fans watch the race

$$\underset{P_i}{\mathrm{MAX}} \prod_{k=1}^{5} P(P_k|P_{k-1}) \prod_{k=1}^{5} P(w_k|P_k)$$

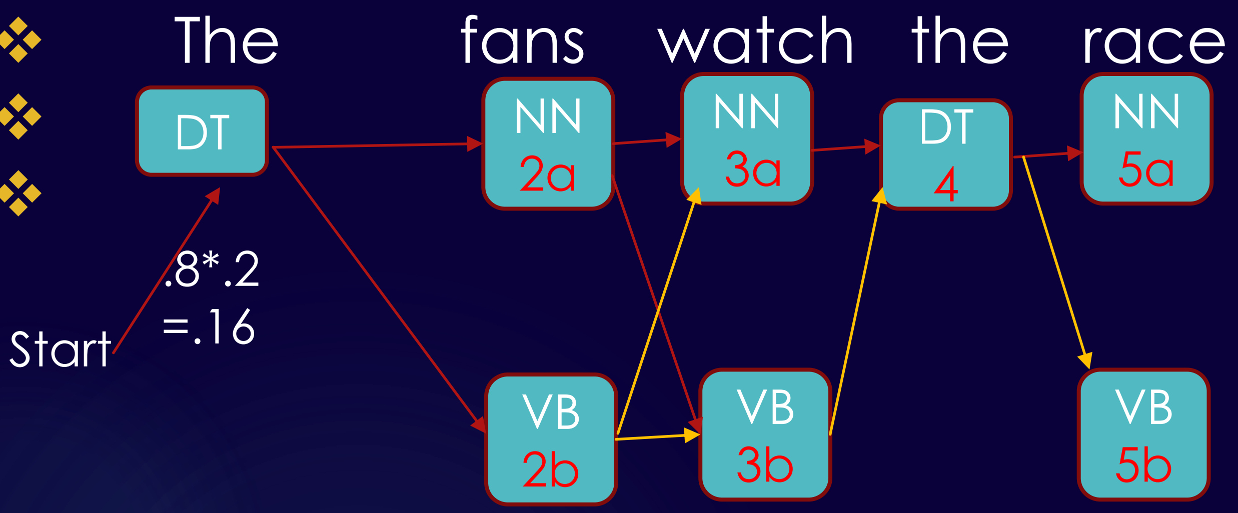
Emission Probability

	The	Fans	Watc h	Race
DT	0.2	0	0	0
NN	0	0.1	0.3	0.1
VB	0	0.2	0.15	0.3

Transition Probability

	DT	NN	VB
Start	0.8	0.2	0
DT	0	0.9	0.1
NN	0	0.5	0.5
VB	0.5	0.5	0

POS Tagging -Viterbi Algorithm



- 1-2a =.9*.1*.16=.0144
- 1-2b =.1*.2*.16 =.0032
- 2a-3a = ?
- 2a-3b = ?
- 2b-3a = ?
- 2b-3b = ?
- 3a-4 = ?
- 3b-4 =.5*.15*.00108=.000081

Emission Probability

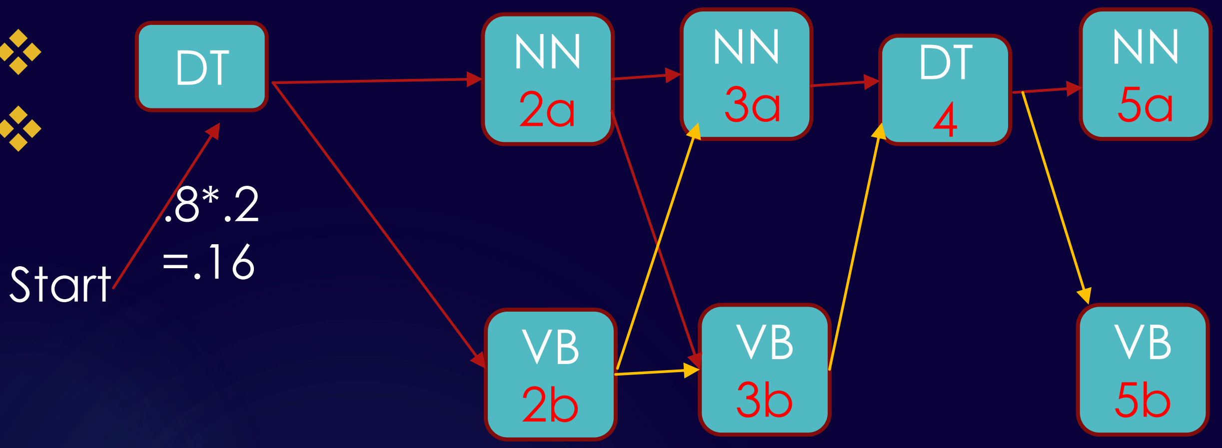
	The	Fans	Watch	Race
DT	0.2	0	0	0
NN	0	0.1	0.3	0.1
VB	0	0.2	0.15	0.3

Transition Probability

	DT	NN	VB
Start	0.8	0.2	0
DT	0	0.9	0.1
NN	0	0.5	0.5
VB	0.5	0.5	0

POS Tagging -Viterbi Algorithm

- The fans watch the race



- 1-2a =.9*.1*.16=.0144
- 1-2b =.1*.2*.16 =.0032
- 2a-3a =.5*.3*.0144 = .00126 (Taken)
- 2b-3a =.5*.3*.0032=.00048
- 2a-3b =.5*.15*.0144=.00108 (Taken)
- 2b-3b = 0*…=0
- 3a-4 =0*…=0
- 3b-4 =.5*.15*.00108=.000081

Emission Probability

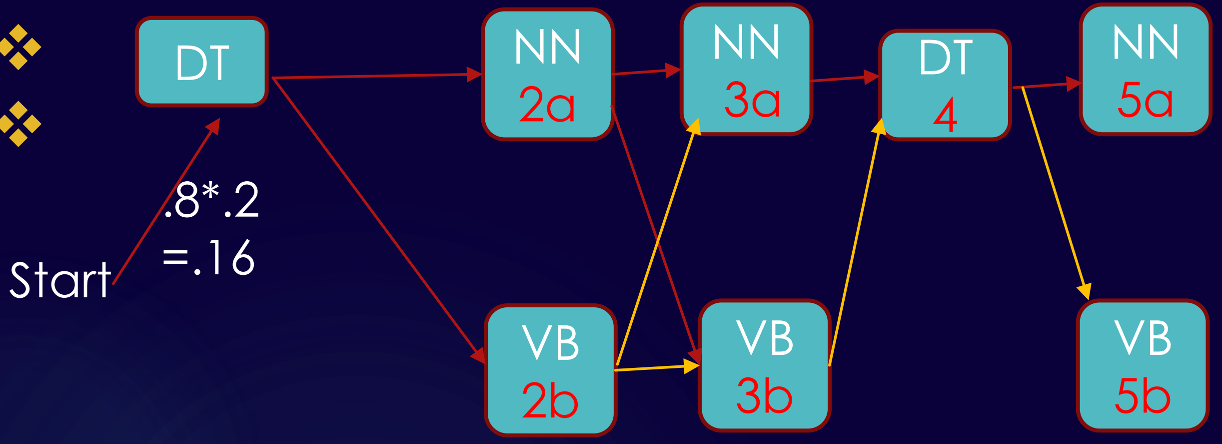
	The	Fans	Watch	Race
DT	0.2	0	0	0
NN	0	0.1	0.3	0.1
VB	0	0.2	0.15	0.3

Transition Probability

	DT	NN	VB
Start	0.8	0.2	0
DT	0	0.9	0.1
NN	0	0.5	0.5
VB	0.5	0.5	0

POS Tagging -Viterbi Algorithm

- The fans watch the race



- 3a-4 =0*…=0
- 3b-4 =.5*.2*.00108=.000108(Taken)
- 4-5a =.9*.1*.000108 = 9.72 * 10^-6
- 4-5b = .1*.3*.000108 =3.24 * 10^-6

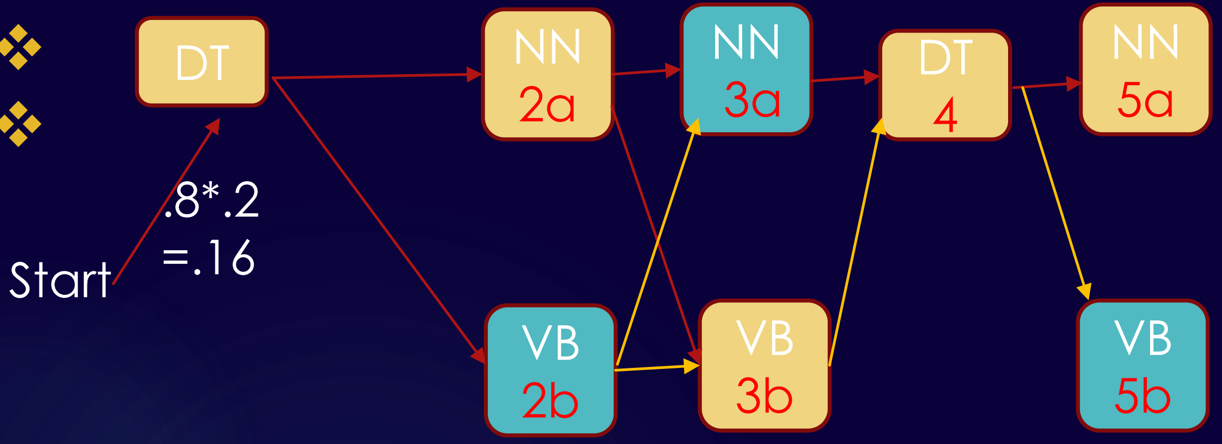
Emission Probability

	The	Fans	Watch	Race
DT	0.2	0	0	0
NN	0	0.1	0.3	0.1
VB	0	0.2	0.15	0.3

	DT	NN	VB
Start	0.8	0.2	0
DT	0	0.9	0.1
NN	0	0.5	0.5
VB	0.5	0.5	0

POS Tagging -Viterbi Algorithm

- The fans watch the race



- 3a-4 =0*…=0
- 3b-4 =.5*.2*.00108=.000108(Taken)
- 4-5a =.9*.1*.000108 = 9.72 * 10^-6
- 4-5b = .1*.3*.000108 =3.24 * 10^-6

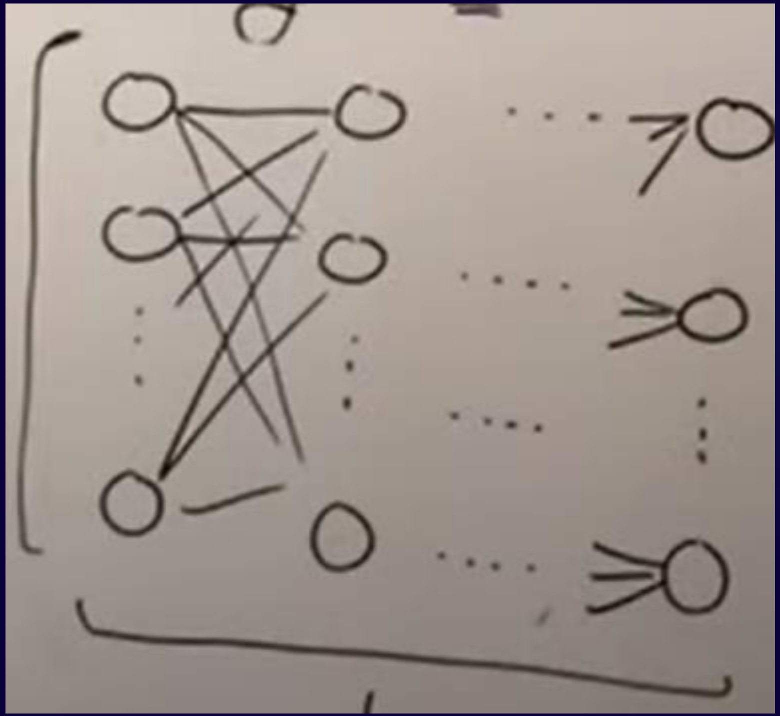
Emission Probability

	The	Fans	Watch	Race
DT	0.2	0	0	0
NN	0	0.1	0.3	0.1
VB	0	0.2	0.15	0.3

	DT	NN	VB
Start	0.8	0.2	0
DT	0	0.9	0.1
NN	0	0.5	0.5
VB	0.5	0.5	0

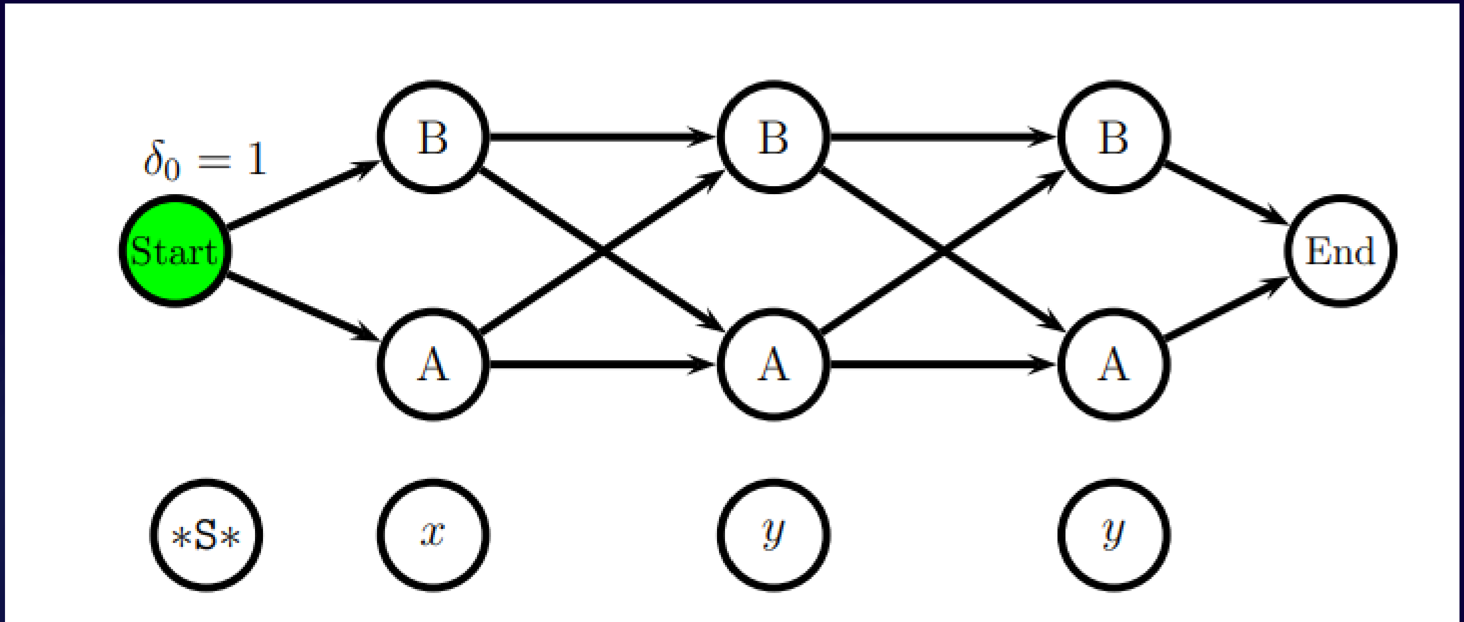
Why Viterbi Algorithm is better?

- Imagine sentence of length L and each word can have one of P POS
- O(P^L) in Brute force
- In Viterbi
- O(P^2 * L)
- So Viterbi is better



Example 3

Use Viterbi Algorithm



	Next		
Current	A	B	End
Start	0.7	0.3	0
A	0.2	0.7	0.1
B	0.7	0.2	0.1

	Word		
State	*S*	x	y
Start	1	0	0
A	0	0.4	0.6
B	0	0.3	0.7

Why Viterbi Algorithm is better?

- Imagine sentence of length L and each word can have one of P POS
- O(P^L) in Brute force
- In Viterbi
- O(P^2 * L)
- So Viterbi is better