Hidden Markov Models

part 1

CBB 261 / CPS 261

B. Majoros

Duke
UNIVERSITY

What is an HMM?

The short answer: it's an opaque, nondeterministic machine that emits variable-length sequences of discrete[1] symbols.

What we mean by "Hidden" in "Hidden Markov Model":
We can take the machine apart and look inside to see how many states there are and how they are connected, but we can't look inside the machine while it's running. (However, we can make inferences about what probably happened inside the machine, based on its output.)

[1] ...usually, though not always...

What is an HMM?

An HMM is a *stochastic machine* $M=(Q, \alpha, P_t, P_e)$ consisting of the following:

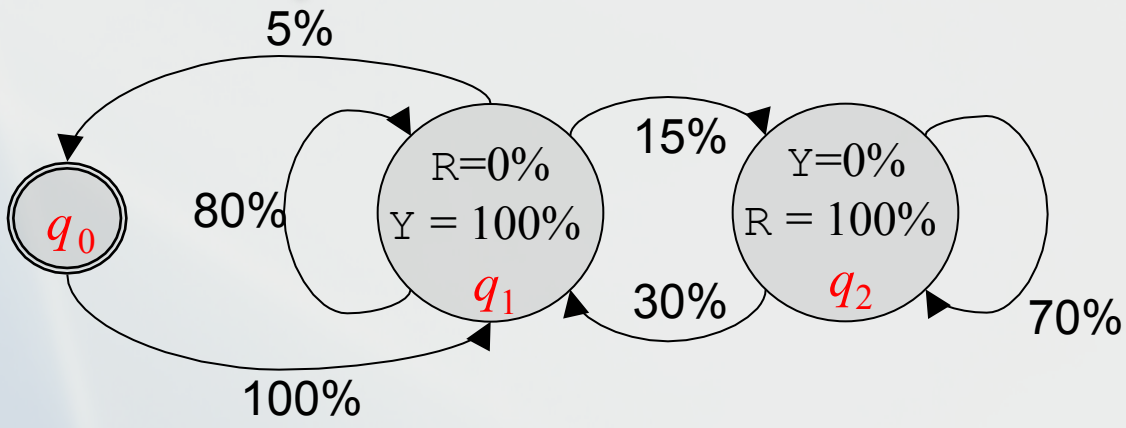
- a finite set of states, $Q=\{q_0, q_1, \ldots, q_m\}$
- a finite alphabet $\alpha=\{s_0, s_1, \ldots, s_n\}$
- a transition distribution $P_t : Q \times Q \mapsto \mathbb{R}$ i.e., $P_t(q_j \mid q_i)$
- an emission distribution $P_e : Q \times \alpha \mapsto \mathbb{R}$ i.e., $P_e(s_j \mid q_i)$

An Example

$M_1=(\{q_0,q_1,q_2\},\{\mathtt{Y},\mathtt{R}\},P_t,P_e)$

$P_t=\{(q_0,q_1,1), (q_1,q_1,0.8), (q_1,q_2,0.15), (q_1,q_0,0.05), (q_2,q_2,0.7), (q_2,q_1,0.3)\}$

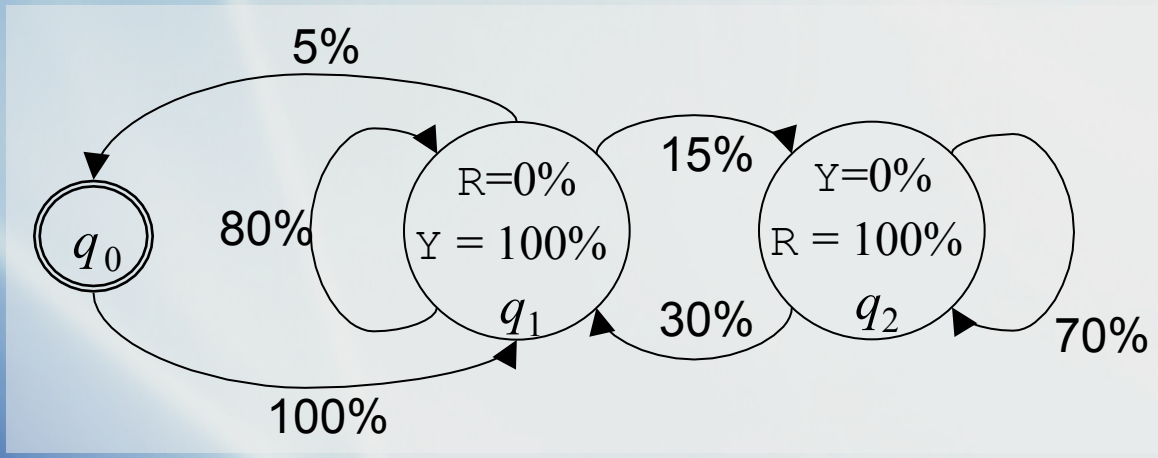
$P_e=\{(q_1,\mathtt{Y},1), (q_1,\mathtt{R},0), (q_2,\mathtt{Y},0), (q_2,\mathtt{R},1)\}$



Probability of a Sequence

S=YRYRY

$$\phi = (q0, q1, q2, q1, q2, q1, q0)$$



$$P(\text{YRYRY}|M_1) =$$

$$a_{0\to1} \times b_{1,\text{Y}} \times a_{1\to2} \times b_{2,\text{R}} \times a_{2\to1} \times b_{1,\text{Y}} \times a_{1\to2} \times b_{2,\text{R}} \times a_{2\to1} \times b_{1,\text{Y}} \times a_{1\to0}$$

$$= 1 \times 1 \times 0.15 \times 1 \times 0.3 \times 1 \times 0.15 \times 1 \times 0.3 \times 1 \times 0.05$$

$$= 0.00010125$$

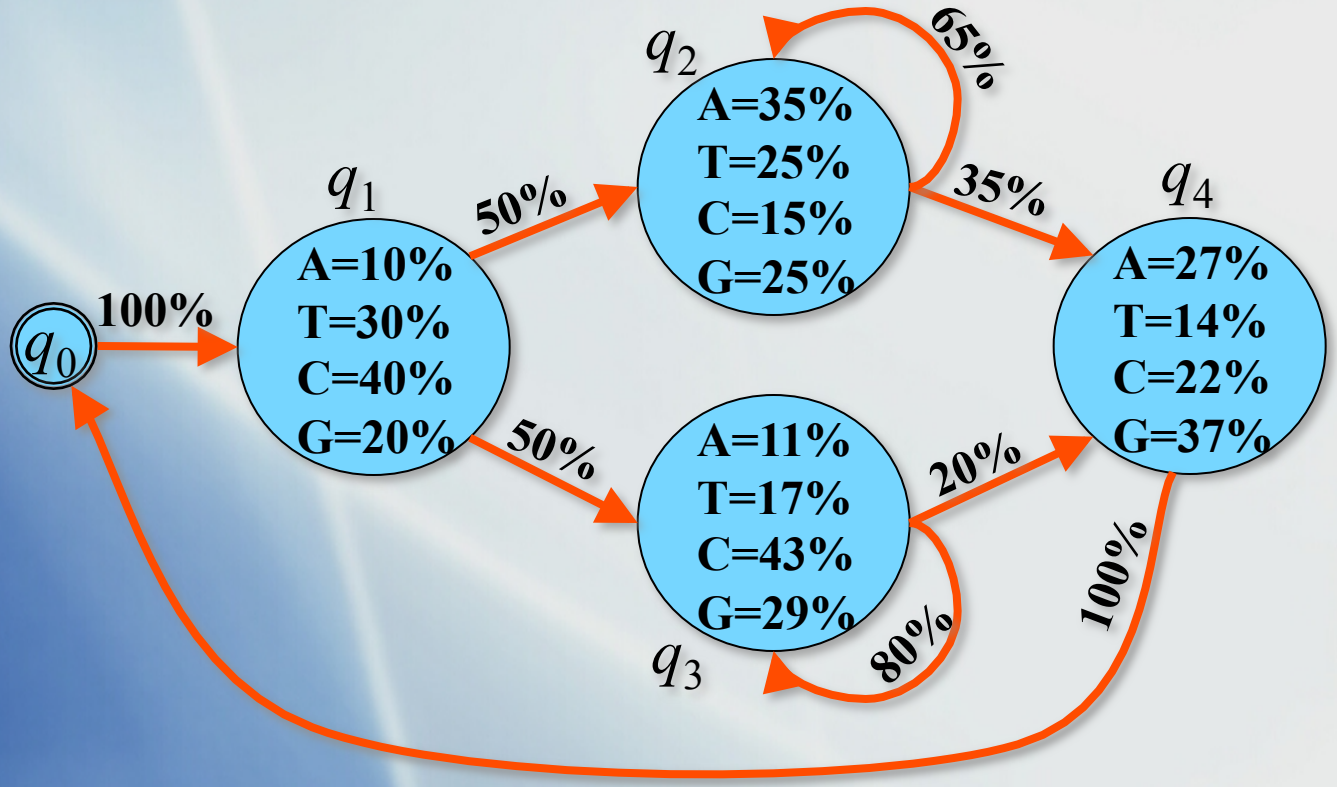
Duke UNIVERSITY

Another Example

$$M_2 = (Q, \alpha, P_t, P_e)$$

$$Q = \{q_0, q_1, q_2, q_3, q_4\}$$

$$\alpha = \{A, C, G, T\}$$

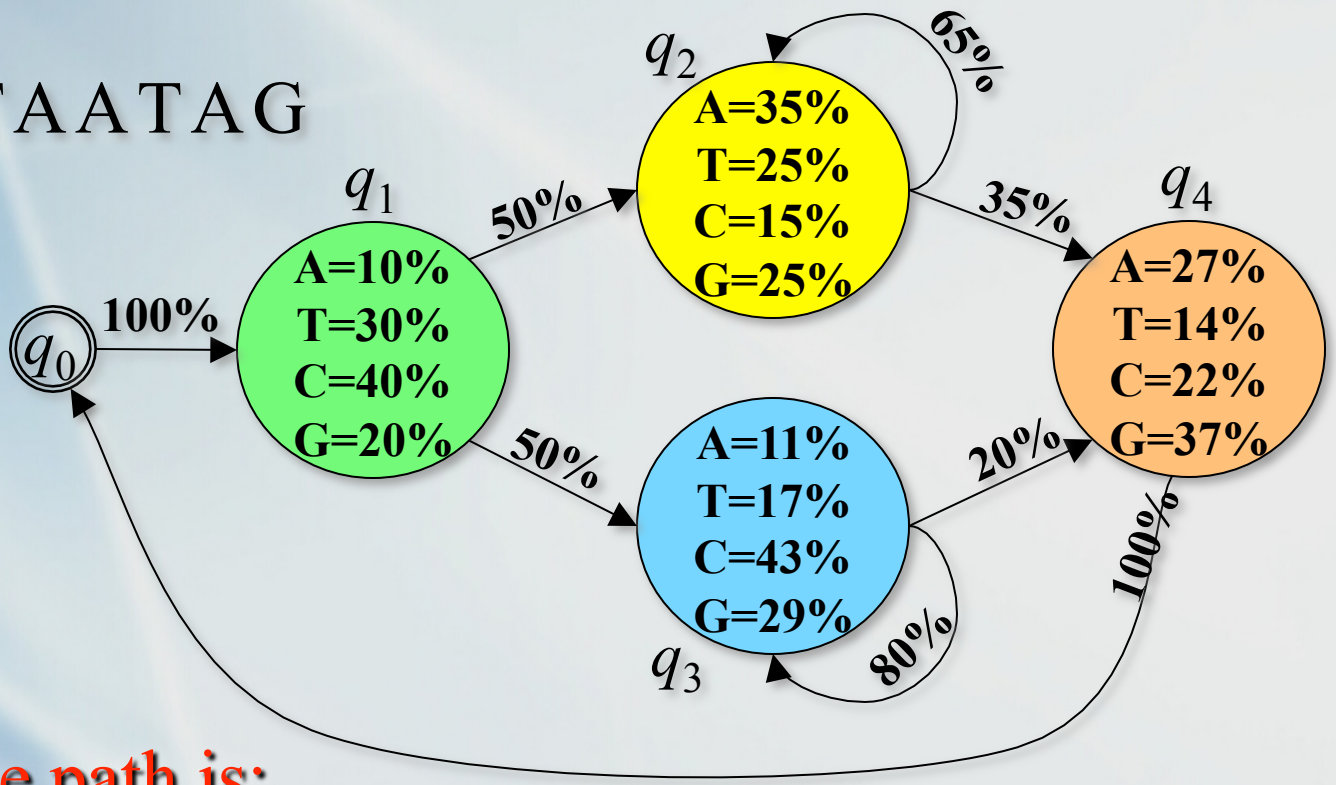


Finding the Most Probable Path

Example: CATTAATAG

top: 7.0×10^{-7}

bottom: 2.8×10^{-9}



The most probable path is:

States: 122222224

Sequence: CATTAATAG

resulting in this parse:

States: 122222224

Sequence: CATTAATAG

feature 1: C

feature 2: ATTAATA

feature 3: G

Duke UNIVERSITY

Decoding with an HMM

$$\phi_{\max} = \underset{\phi}{argmax}\, P(\phi|S) = \underset{\phi}{argmax}\, \frac{P(\phi,S)}{P(S)}$$

$$= \underset{\phi}{argmax}\, P(\phi,S)$$

$$= \underset{\phi}{argmax}\, P(S|\phi)P(\phi)$$

$$S = x_0 x_1 \ldots x_L$$
$$\phi = y_0 y_1 \ldots y_{L+2}$$

$$P(S \mid \phi) = \prod_{i=0}^{L-1} P_e(x_i \mid y_{i+1})$$

emission prob.

$$P(\phi) = \prod_{i=0}^{L} P_t(y_{i+1} \mid y_i)$$

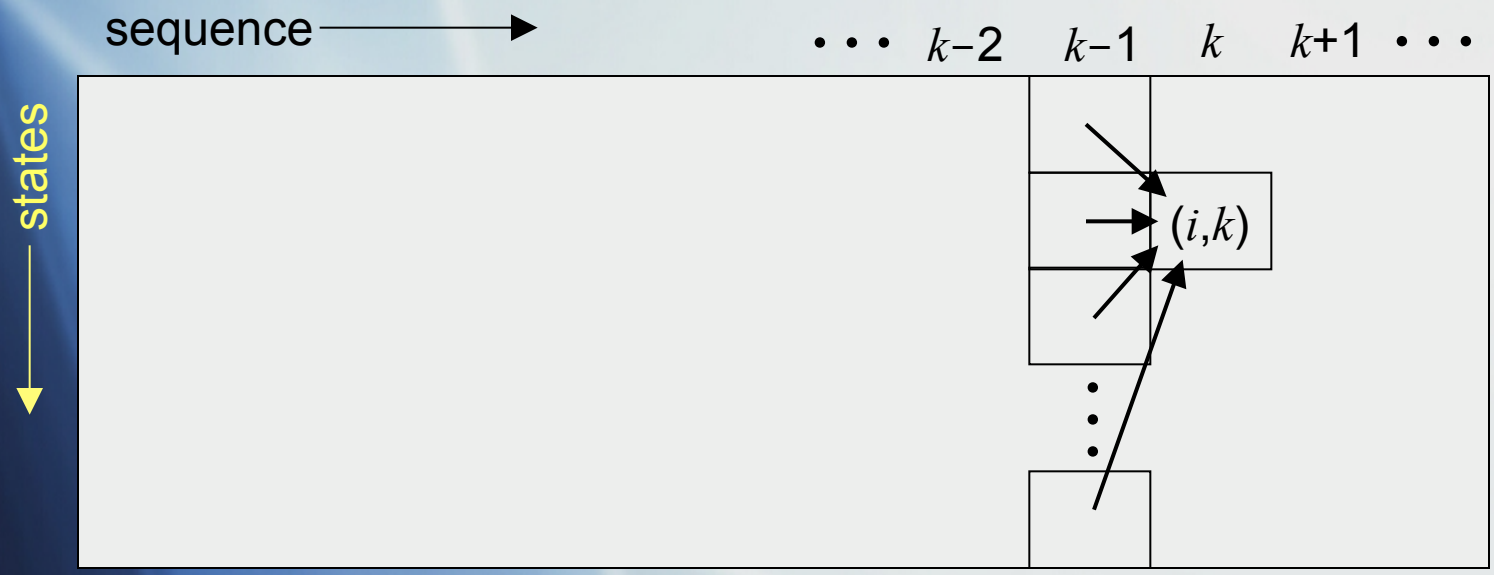
transition prob.

$$\phi_{\max} = \underset{\phi}{argmax}\, P_t(q_0|y_L) \prod_{i=0}^{L-1} P_e(x_i|y_{i+1}) P_t(y_{i+1}|y_i)$$

Duke UNIVERSITY

The Viterbi Algorithm

$$V(i,k) = \begin{cases} \max_j V(j,k-1) P_t(q_i|q_j) P_e(x_k|q_i) & \text{if } k > 0, \\ P_t(q_i|q_0) P_e(x_0|q_i) & \text{if } k = 0. \end{cases}$$



In the final column:

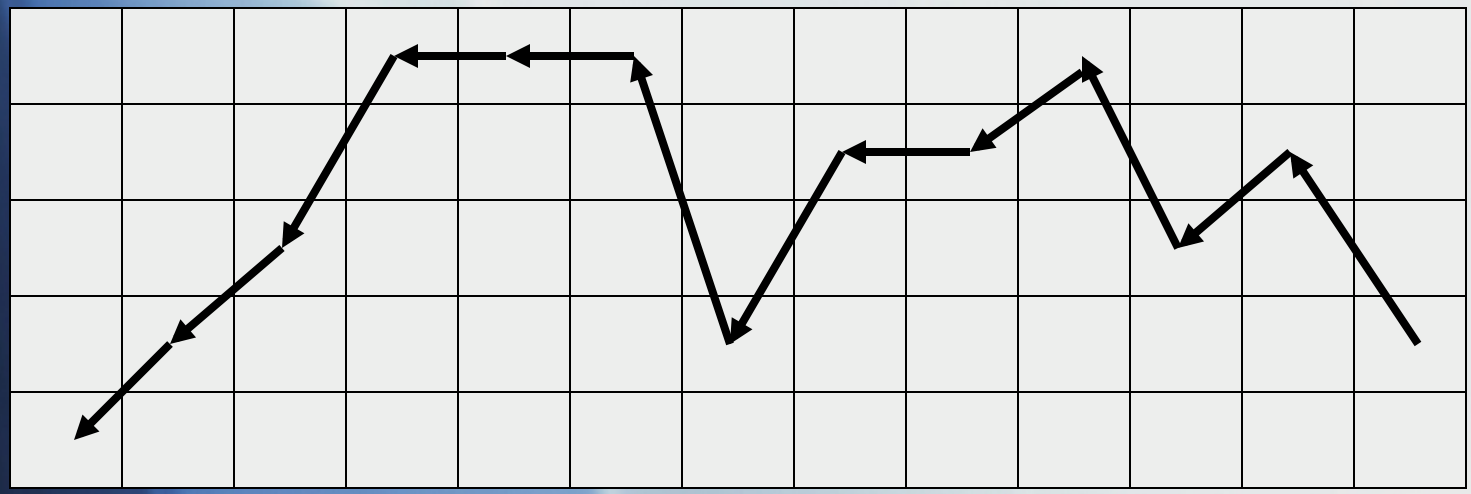
$$P(\phi_{\max}) = \max_i V(i, L-1) P_t(q_0|q_i)$$

Viterbi: Traceback

$$V(i,k) = \begin{cases} \max_j V(j,k-1) P_t(q_i|q_j) P_e(x_k|q_i) & \text{if } k > 0, \\ P_t(q_i|q_0) P_e(x_0|q_i) & \text{if } k = 0. \end{cases}$$

$$T(i,k) = \begin{cases} \operatorname{argmax}_j V(j,k-1) P_t(q_i|q_j) P_e(x_k|q_i) & \text{if } k > 0, \\ 0 & \text{if } k = 0. \end{cases}$$

$$T(\,T(\,T(\,\ldots\,T(\,T(i,\,L-1),\,L-2)\,\ldots,\,2),\,1),\,0) = 0$$



(Log) Viterbi Algorithm in Pseudocode

```
procedure viterbi(Q, α, P_t, P_e, S, λ_trans, λ_emit)
1.    for k←0 up to |S|-1 do
2.      for i←0 up to |Q|-1 do
3.        V[i][k]←-∞;
4.        T[i][k]←NIL;
5.    for i←1 up to |Q|-1 do
6.      V[i][0]←log(P_t(q_i|q_0))+log(P_e(S[0]|q_i));
7.      if V[i][0]>-∞ then T[i][0]←0;
8.    for k←1 up to |S|-1 do
9.      foreach q_i∈λ_emit[S[k]] do
10.       foreach q_j∈λ_trans[ q_i] do
11.         v←V[j][k-1]+log(P_t(q_i|q_j))+
12.                    log(P_e(S[k]|q_i));
13.         if v>V[i][k] then
14.           V[i][k]←v;
15.           T[i][k]←j;
16.   y←1;
17.   push φ,0;
18.   for i←2 up to |Q|-1 do
19.     if V[i][|S|-1]+log(P_t(q_0|q_i)) >
20.        V[y][|S|-1]+log(P_t(q_0|q_y)) then y←i;
21.   for k←|S|-1 down to 0 do
22.     push φ,y;
23.     y←T[y][k];
24.   push φ,0;
25.   return φ;
```

$$\lambda_{trans}[q_i]=\{q_j \mid P_t(q_i|q_j)>0\}$$

$$\lambda_{emit}[s] = \{q_i \mid P_e(s|q_i)>0\}$$

initialization

fill out main part of DP matrix

choose best state from last column in DP matrix

traceback

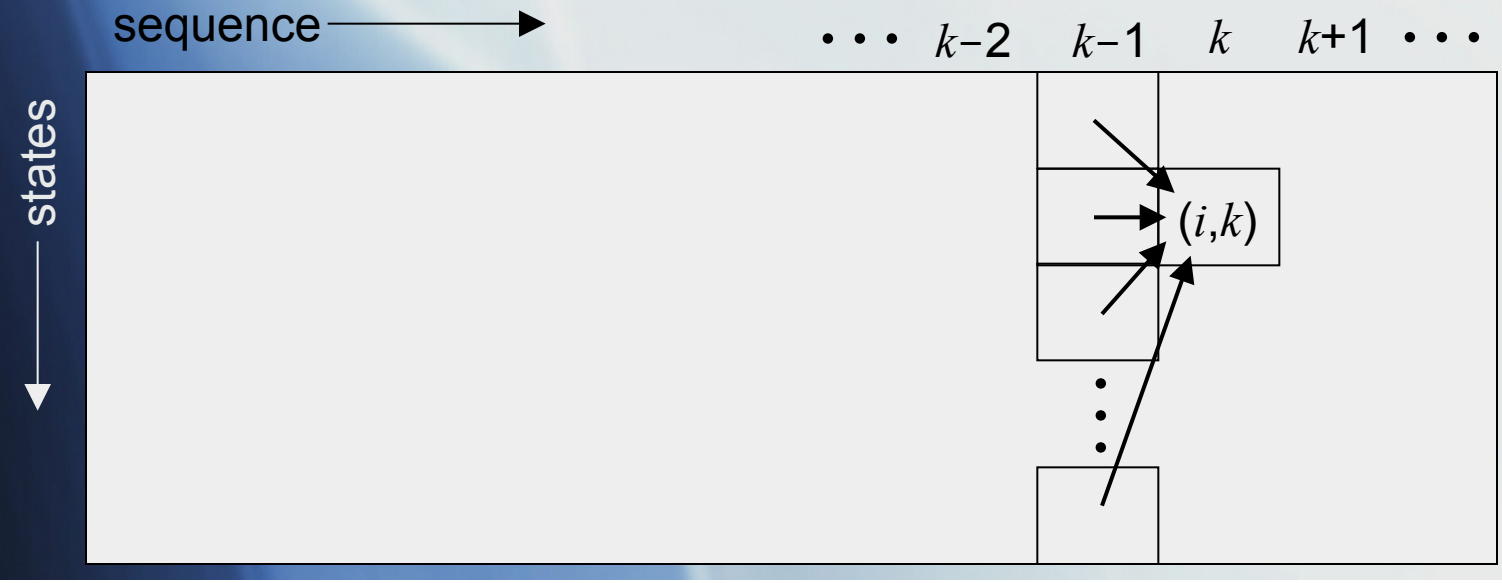
Duke UNIVERSITY

The Forward Algorithm : Probability of a Sequence

Viterbi: $\max_j V(j,k-1)P_t(q_i \mid q_j)P_e(x_k \mid q_i)$ ➡ the single most probable path

Forward: $\sum_{j=0}^{|Q|-1} F(j,k-1)P_t(q_i \mid q_j)P_e(x_{k-1} \mid q_i)$ ➡ sum over all paths

$$\left(i.e., \sum_{\substack{all \\ paths \\ \phi}} P(S,\phi) \right)$$



The Forward Algorithm : Probability of a Sequence

$$F(i,k) = \begin{cases} 1 & \text{for } k = 0, i = 0 \\ 0 & \text{for } k > 0, i = 0 \\ 0 & \text{for } k = 0, i > 0 \\ \sum_{j=0}^{|Q|-1} F(j,k-1) P_t(q_i \mid q_j) P_e(x_{k-1} \mid q_i) & \text{for } 1 \le k \le |S|, \\ & \quad 1 \le i < |Q| \end{cases}$$

$F(i,k)$ represents the probability $P(S_{0..k-1} \mid q_i)$ that the machine emits the subsequence $x_0 \ldots x_{k-1}$ by any path ending in state q_i—i.e., so that symbol x_{k-1} is emitted by state q_i.

$$P(S \mid M) = \sum_{i=0}^{|Q|-1} F(i, |S|) P_t(q_0 \mid q_i)$$

Duke UNIVERSITY

The Forward Algorithm in Pseudocode

```
procedure unscaledForwardAlg(Q, P_t, P_e, S, λ_trans, λ_emit)
1.    ∀_{i∈[0,|Q|-1]} ∀_{k∈[0,|S|]} F[i][k]←0;
2.    F[0][0]←1;
3.    for k←1 up to |S| do
4.      s←S[k-1];
5.      foreach q_i∈λ_emit[s] do
6.        sum←0;
7.        foreach q_j∈λ_trans[i] do
8.          sum←sum+P_t(q_i|q_j)*F[j][k-1];
9.        F[i][k]←sum*P_e(s|q_i);
10.   return F;

procedure unscaledForwardProb(Q, P_t, P_e, S, λ_trans, λ_emit)
1.    F←unscaledForwardAlg(Q, P_t, P_e, S, λ_trans, λ_emit);
2.    sum←0;
3.    len←|S|;
4.    s←S[len-1]
5.    foreach q_i∈λ_emit[s] do
6.      sum←sum+F[i][len]*P_t(q_0|q_i);
7.    return sum;
```

fill out the DP matrix

sum over the final column to get P(S)

Duke UNIVERSITY

Training an HMM from Labeled Sequences

CGATATTCGATTCTACGCGCGTATACTAGCTTATCTGATC

0111111122222211111122221111111222211111 0

transitions

		to state		
		0	**1**	**2**
from state	**0**	0 (0%)	1 (100%)	0 (0%)
	1	1 (4%)	21 (84%)	3 (12%)
	2	0 (0%)	3 (20%)	12 (80%)

$$a_{i,j} = \frac{A_{i,j}}{\sum_{h=0}^{|Q|-1} A_{i,h}}$$

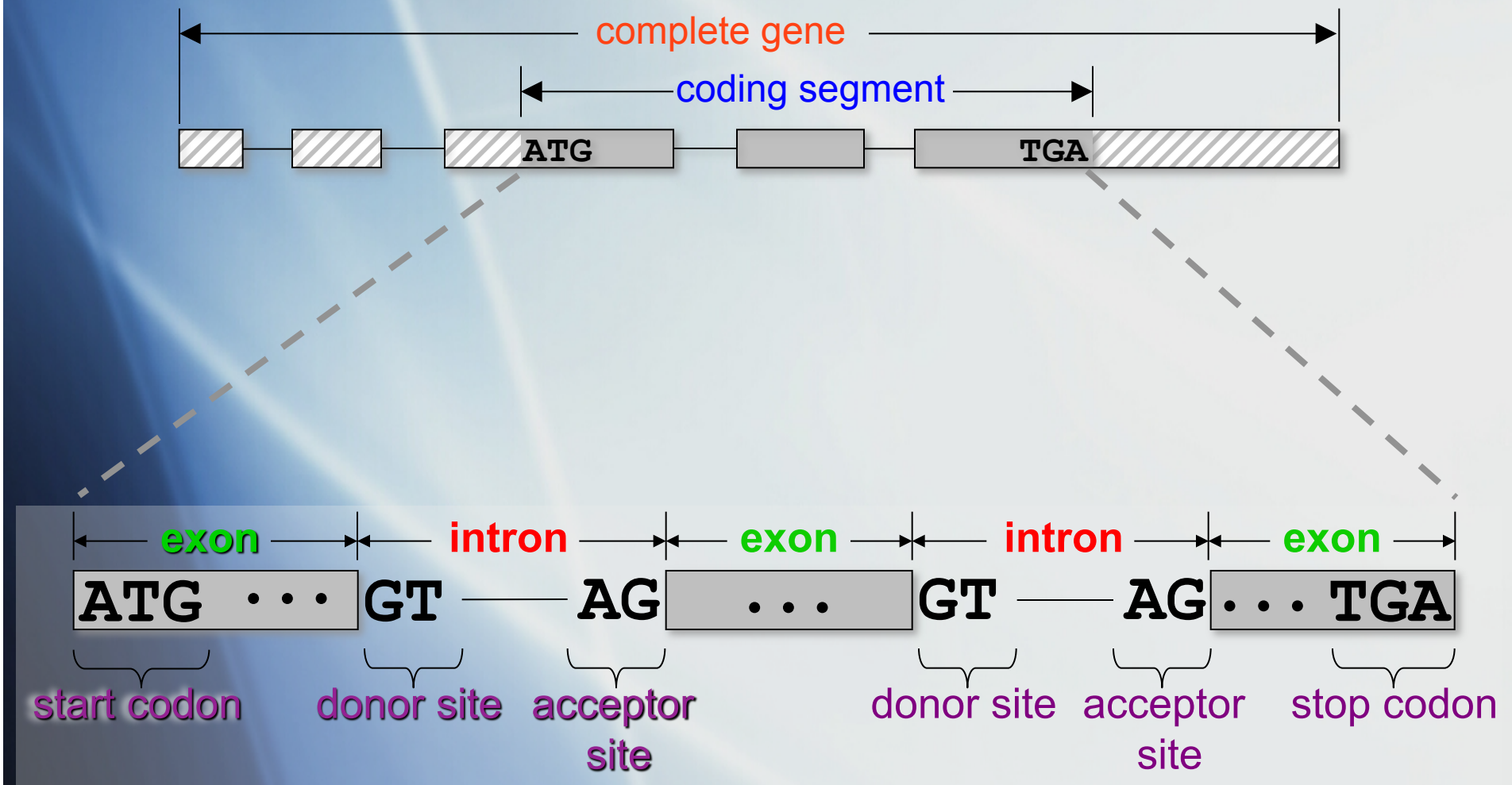
emissions

		symbol			
		A	**C**	**G**	**T**
in state	**1**	6 (24%)	7 (28%)	5 (20%)	7 (28%)
	2	3 (20%)	3 (20%)	2 (13%)	7 (47%)

$$e_{i,k} = \frac{E_{i,k}}{\sum_{h=0}^{|\Sigma|-1} E_{i,h}}$$

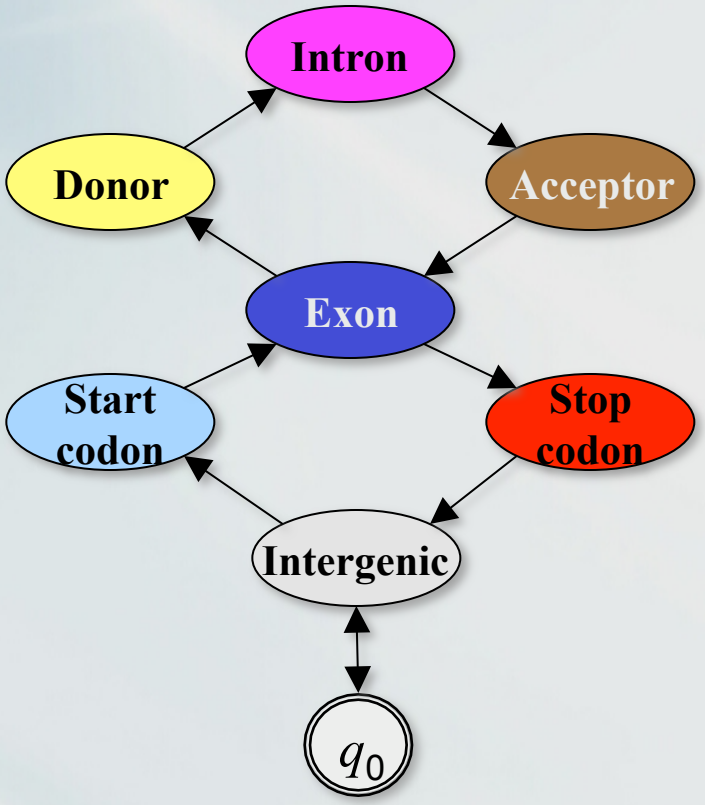
Duke UNIVERSITY

Recall: Eukaryotic Gene Structure



Using an HMM for Gene Prediction

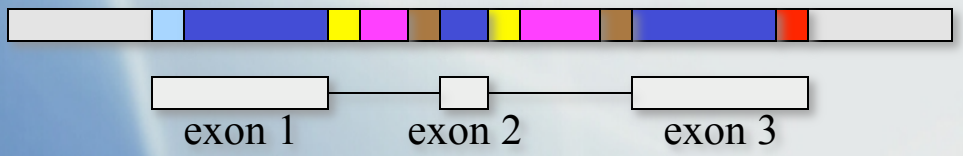
the Markov (meta) model:



the input sequence: AGCTAGCAGT**ATG**TCATGGCAT**GT**TCGG**AG**GTA**GT**ACGTAG**AG**GTAGCTAGTAT**AG**GTCGATAGTA

the most probable path:

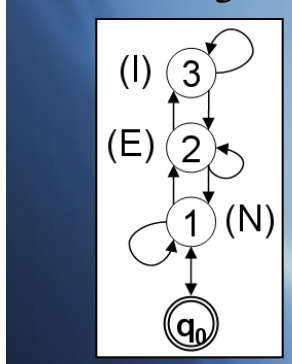
the gene prediction:

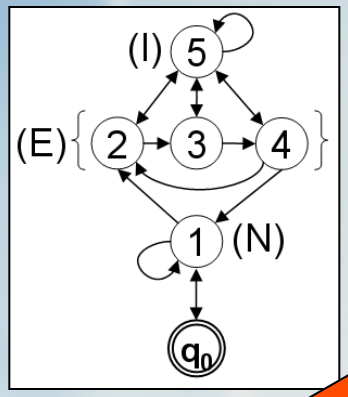


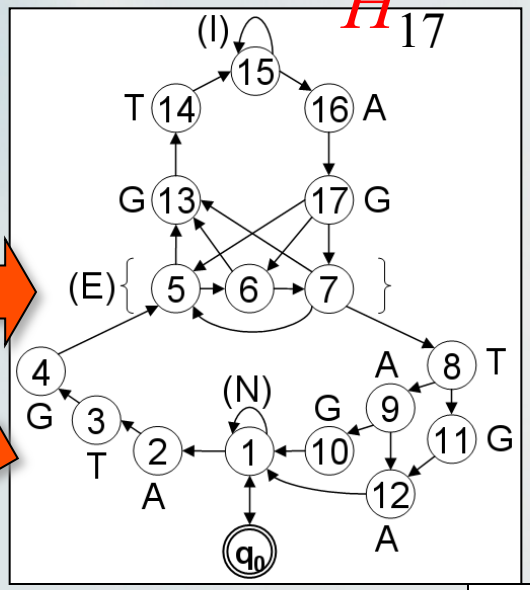
Duke UNIVERSITY

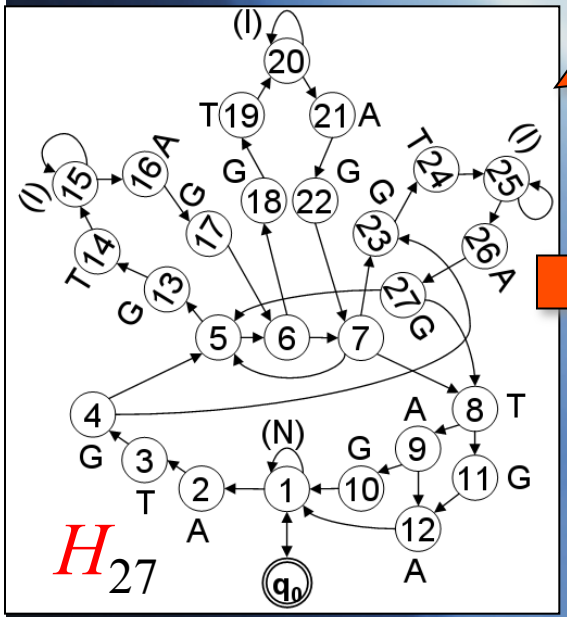
Higher Order Markovian Eukaryotic Recognizer

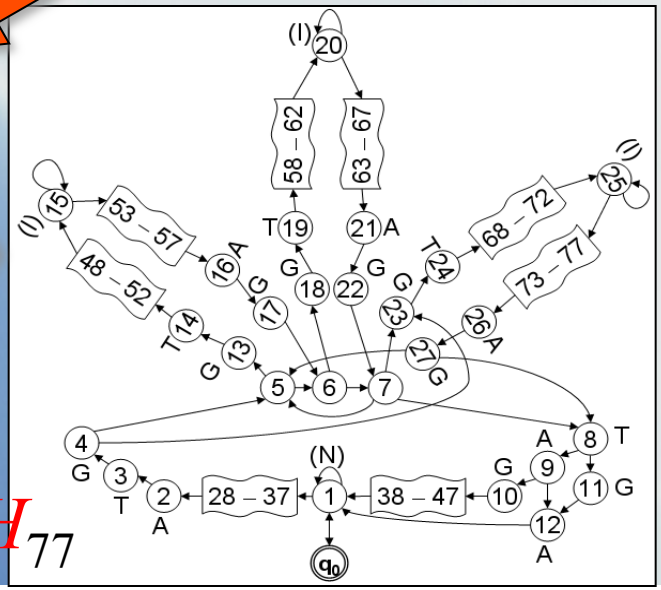
(HOMER)

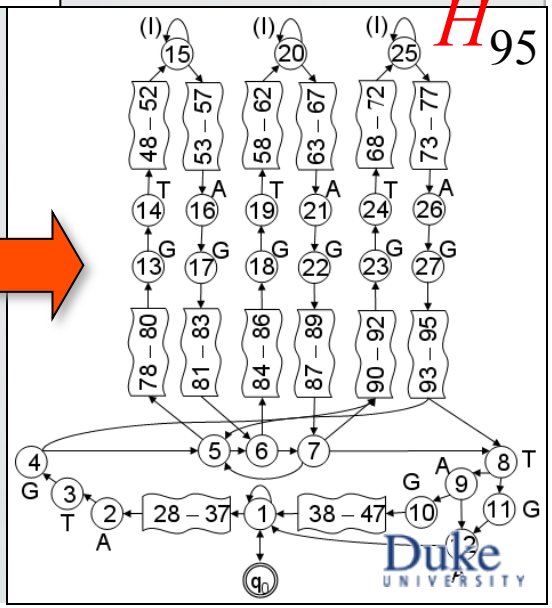












Recall: Sensitivity and Specificity

$$Sn = \frac{TP}{TP + FN}$$

$$Sp = \frac{TP}{TP + FP}$$

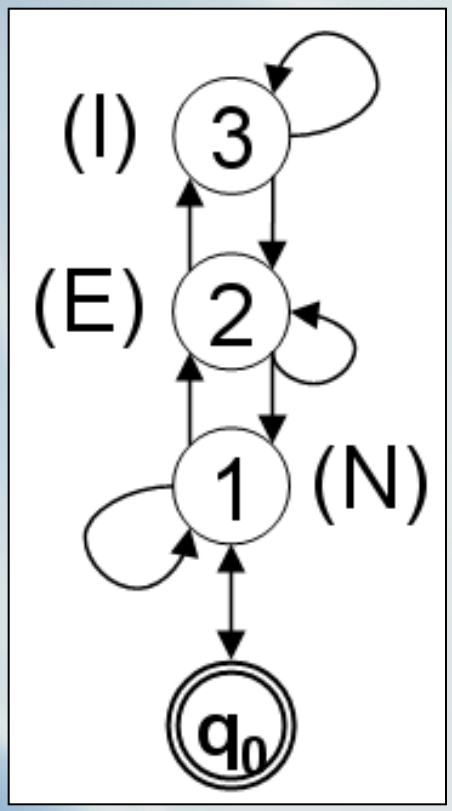
$$F = \frac{2 \times Sn \times Sp}{Sn + Sp}$$

HOMER, version H_3

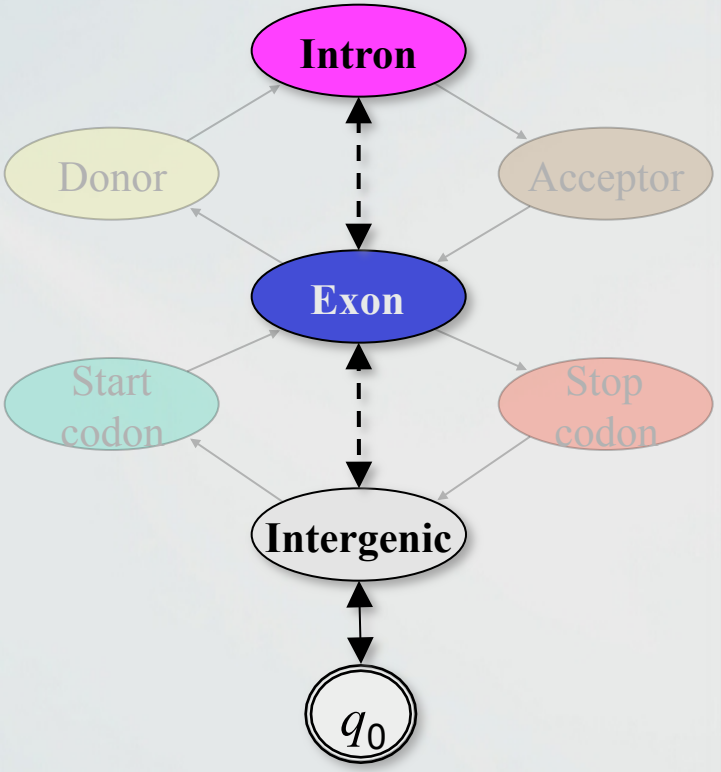
I=intron state

E=exon state

N=intergenic state



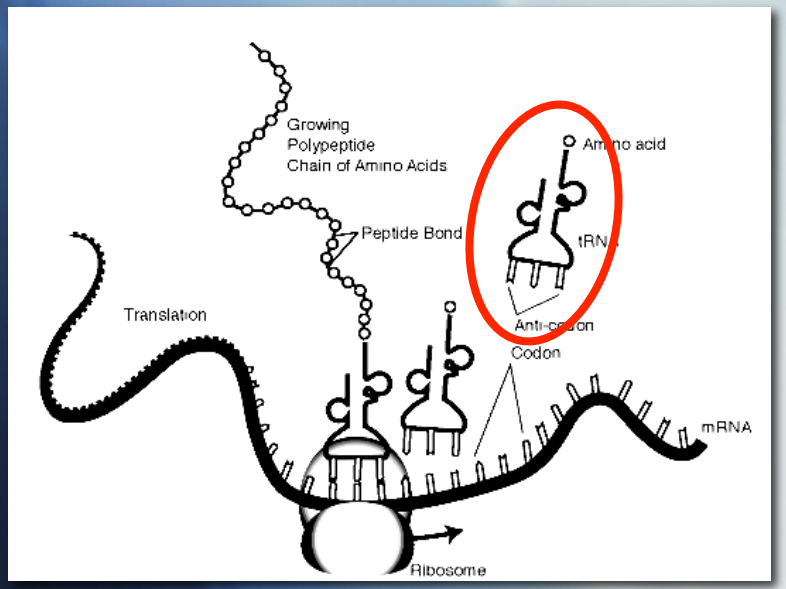
≈

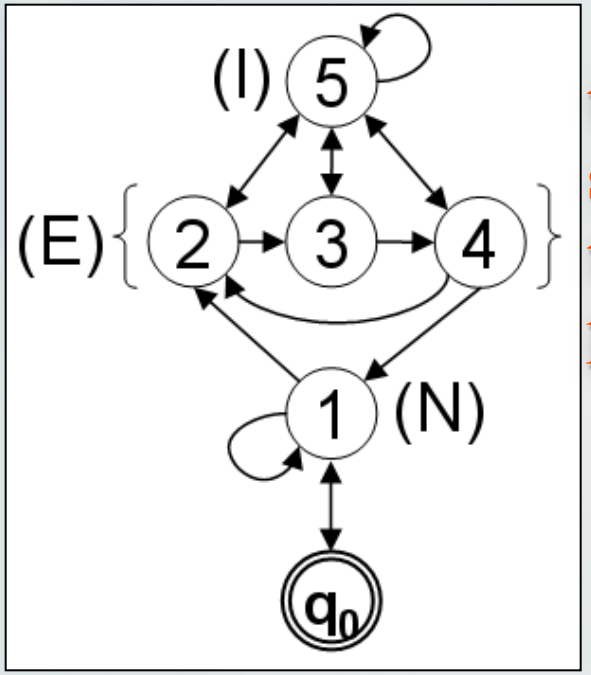


tested on 500 *Arabidopsis* genes:

	nucleotides			splice sites		start/stop codons		exons			genes	
	Sn	*Sp*	*F*	*Sn*	*Sp*	*Sn*	*Sp*	*Sn*	*Sp*	*F*	*Sn*	#
baseline	100	28	44	0	0	0	0	0	0	0	0	0
H_3	53	88	66	0	0	0	0	0	0	0	0	0

HOMER, version H_5



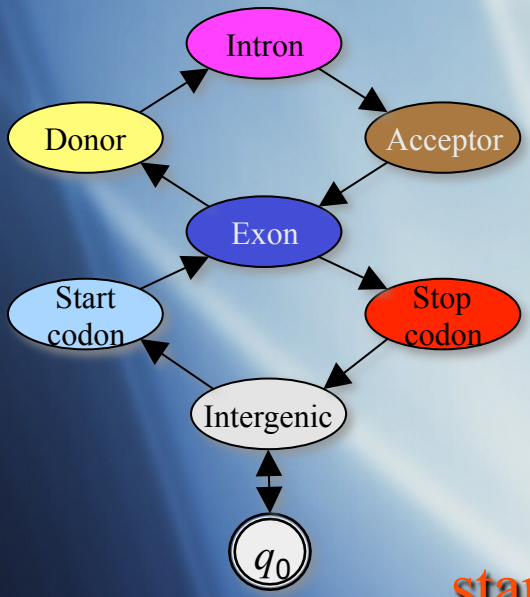


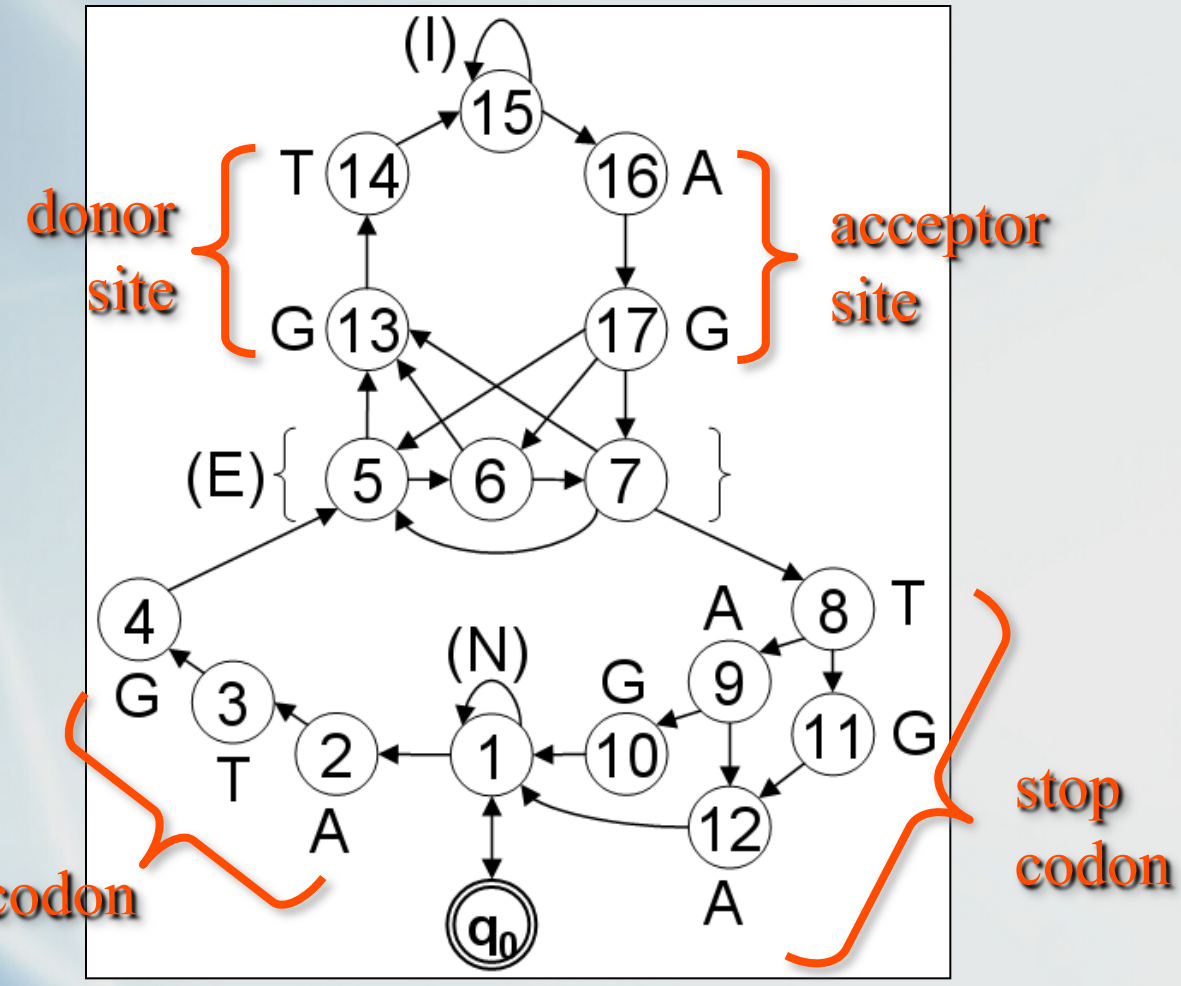
three exon states, for the three codon positions

	nucleotides			splice sites		start/stop codons		exons			genes	
	Sn	*Sp*	*F*	*Sn*	*Sp*	*Sn*	*Sp*	*Sn*	*Sp*	*F*	*Sn*	#
H_3	53	88	66	0	0	0	0	0	0	0	0	0
H_5	65	91	76	1	3	3	3	0	0	0	0	0

Duke UNIVERSITY

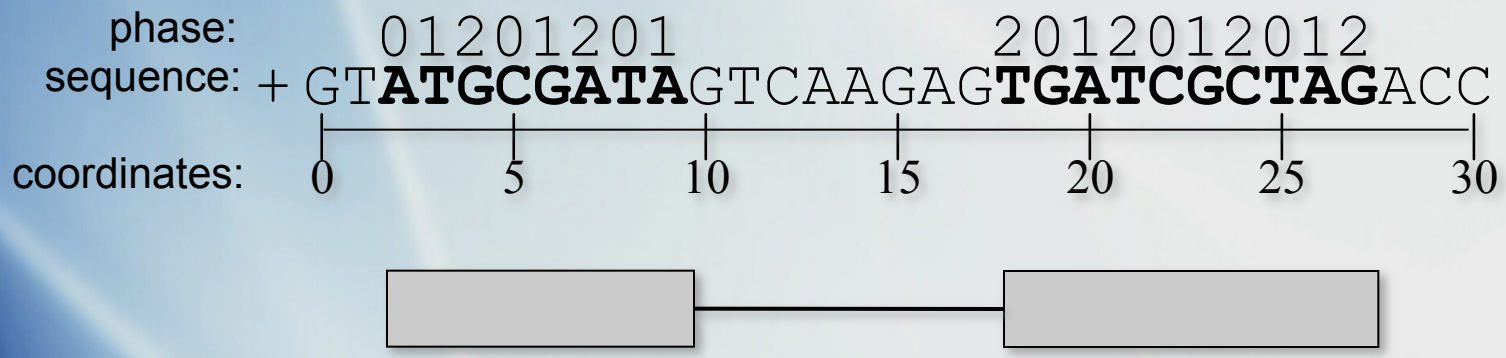
HOMER version H_{17}





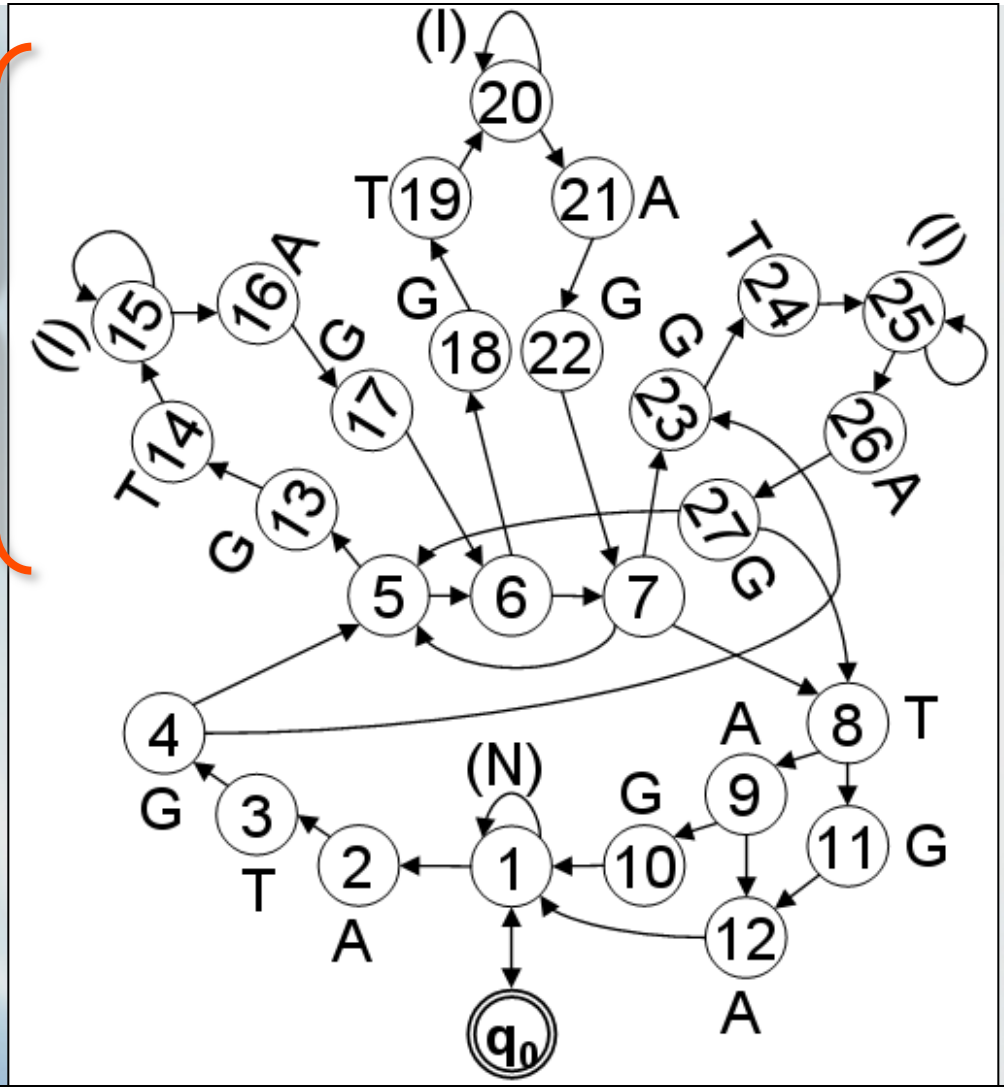
	nucleotides			splice sites		start/stop codons		exons			genes	
	Sn	*Sp*	*F*	*Sn*	*Sp*	*Sn*	*Sp*	*Sn*	*Sp*	*F*	*Sn*	#
H_5	65	91	76	1	3	3	3	0	0	0	0	0
H_{17}	81	93	87	34	48	43	37	19	24	21	7	35

Maintaining Phase Across an Intron



HOMER version H_{27}

three separate intron models



	nucleotides			splice		start/stop		exons			genes	
	Sn	*Sp*	*F*	*Sn*	*Sp*	*Sn*	*Sp*	*Sn*	*Sp*	*F*	*Sn*	#
H_{17}	81	93	87	34	48	43	37	19	24	21	7	35
H_{27}	83	93	88	40	49	41	36	23	27	25	8	38

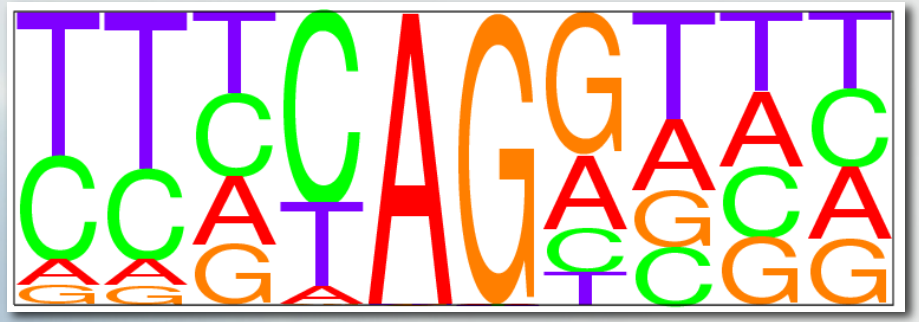
Recall: Weight Matrices

(start codons)

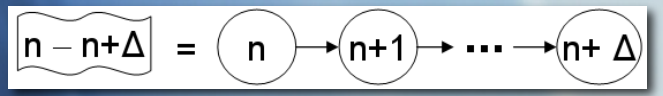
(stop codons)

(donor splice sites)

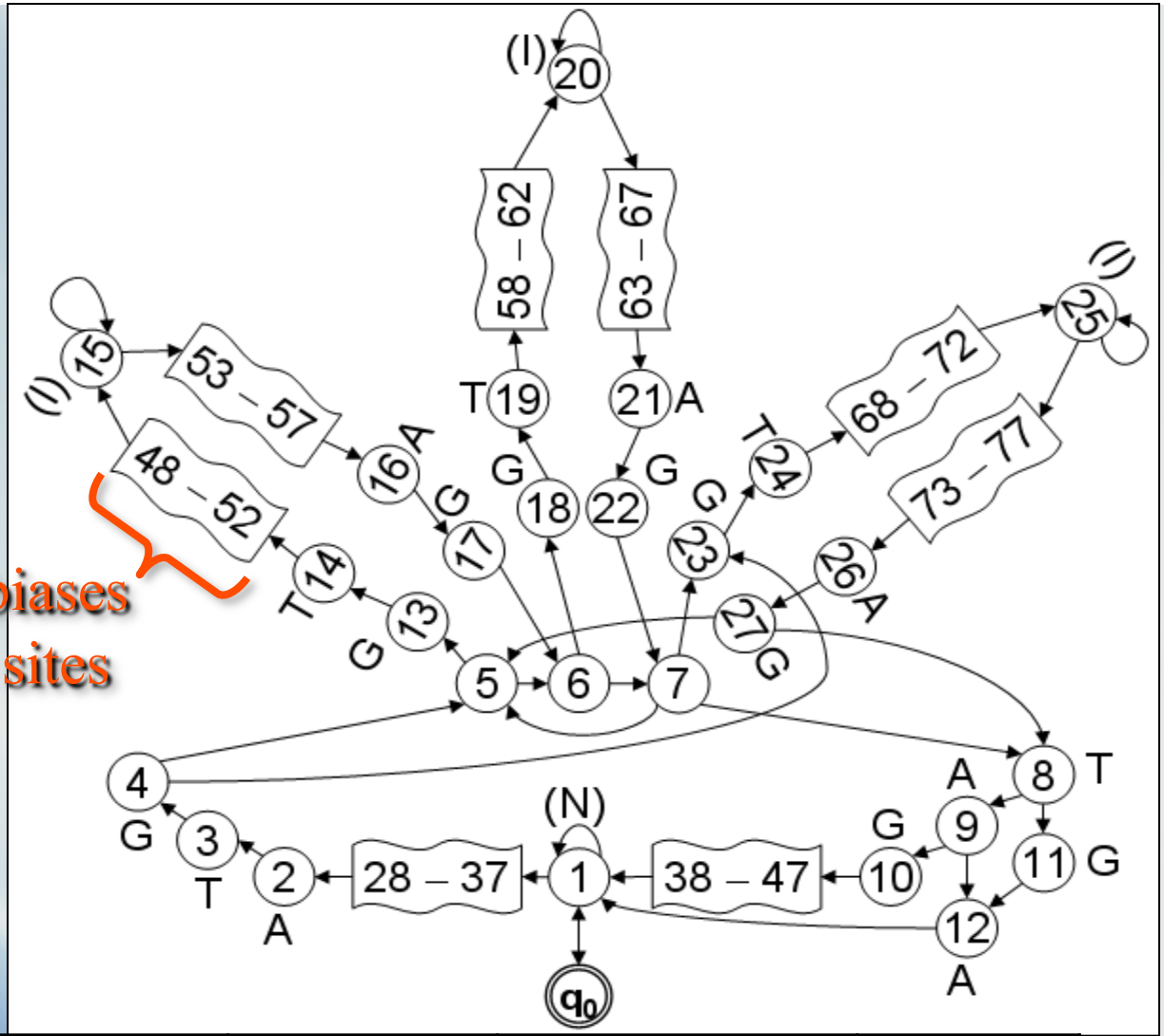
(acceptor splice sites)



HOMER version H_{77}

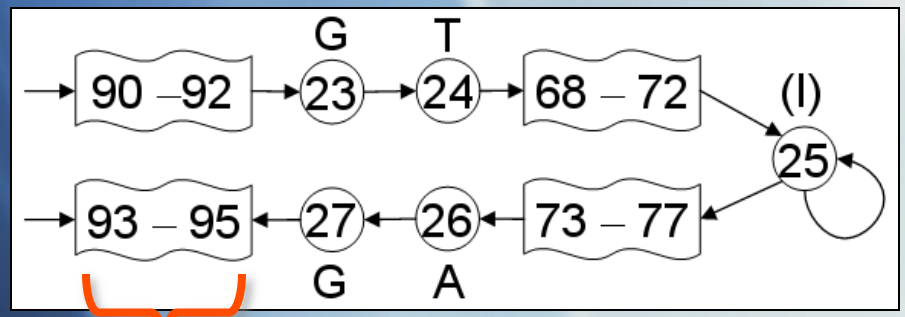


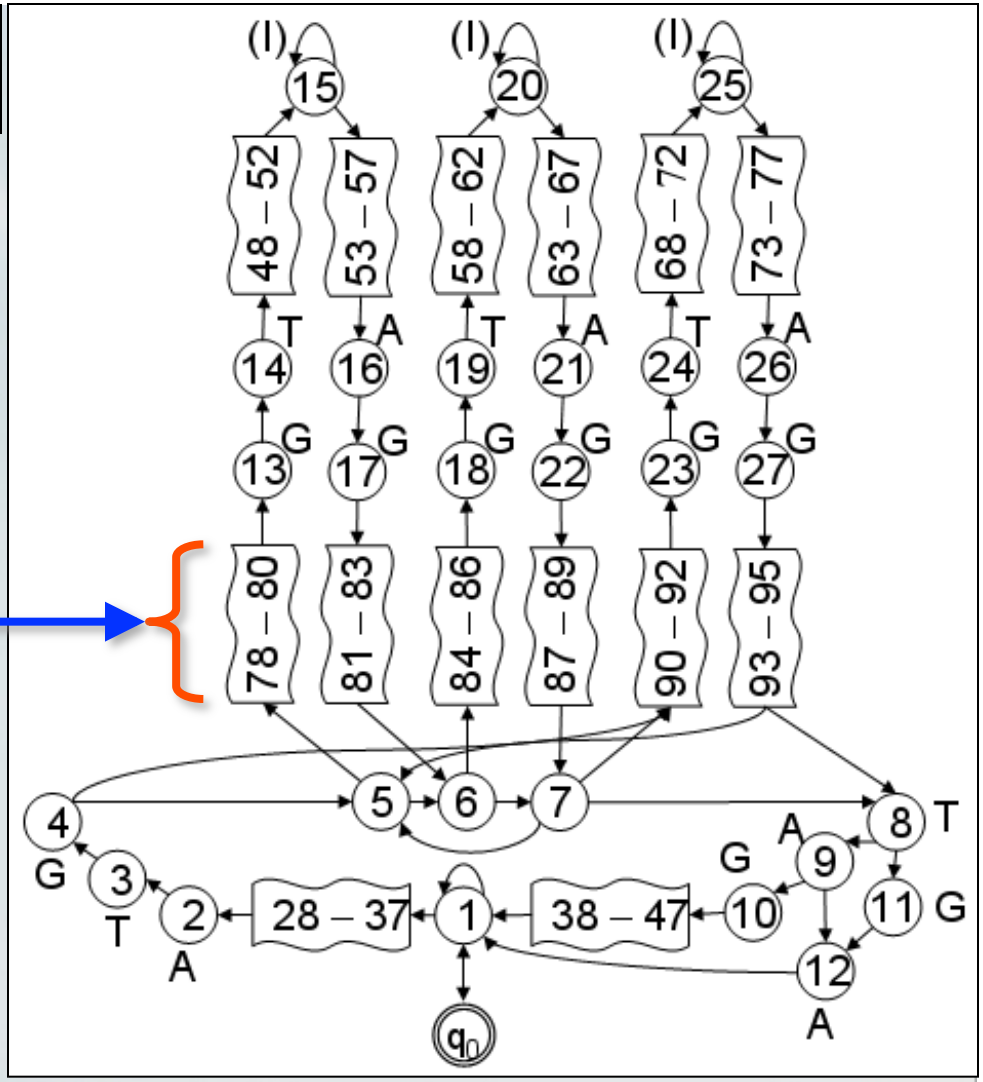
positional biases
near splice sites



	nucleotides			splice		start/stop		exons			genes	
	Sn	*Sp*	*F*	*Sn*	*Sp*	*Sn*	*Sp*	*Sn*	*Sp*	*F*	*Sn*	#
H_{27}	83	93	88	40	49	41	36	23	27	25	8	38
H_{77}	88	96	92	66	67	51	46	47	46	46	13	65

HOMER version H_{95}





	nucleotides			splice		start/stop		exons			genes	
	Sn	*Sp*	*F*	*Sn*	*Sp*	*Sn*	*Sp*	*Sn*	*Sp*	*F*	*Sn*	#
H_{77}	88	96	92	66	67	51	46	47	46	46	13	65
H_{95}	92	97	94	79	76	57	53	62	59	60	19	93

Summary of HOMER Results

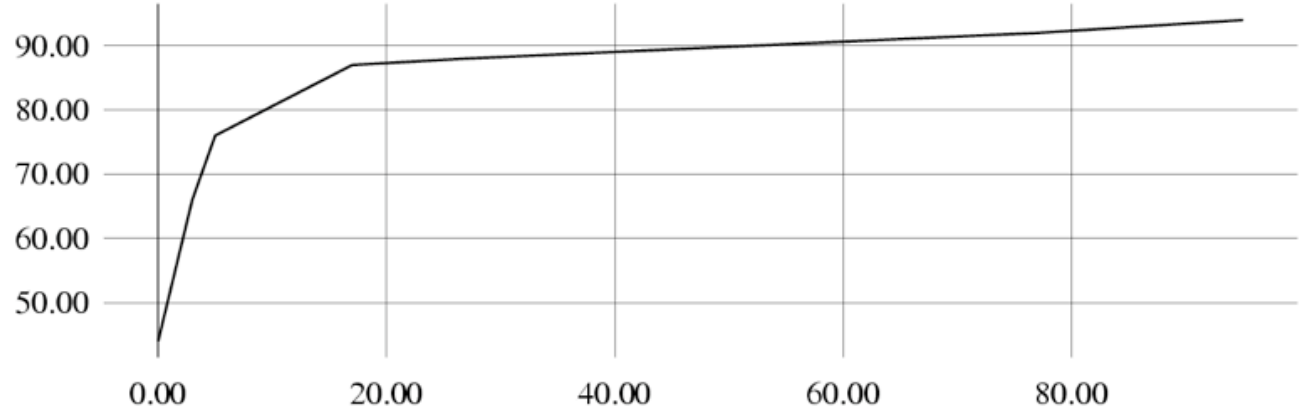


***Figure 6.10**: Nucleotide F-score (y-axis) on a test set of 500 A. thaliana genes, as a function of number of states (x-axis) in an HMM for a simple gene finder.*

	nucleotides			***splice sites***		***start/stop codons***		***exons***			***genes***	
	Sn	Sp	F	Sn	Sp	Sn	Sp	Sn	Sp	F	Sn	#
baseline	*100*	*28*	*44*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
H_3	*53*	*88*	*66*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
H_5	*65*	*91*	*76*	*1*	*3*	*3*	*3*	*0*	*0*	*0*	*0*	*0*
H_{17}	*81*	*93*	*87*	*34*	*48*	*43*	*37*	*19*	*24*	*21*	*7*	*35*
H_{27}	*83*	*93*	*88*	*40*	*49*	*41*	*36*	*23*	*27*	*25*	*8*	*38*
H_{77}	*88*	*96*	*92*	*66*	*67*	*51*	*46*	*47*	*46*	*46*	*13*	*65*
H_{95}	*92*	*97*	*94*	*79*	*76*	*57*	*53*	*62*	*59*	*60*	*19*	*93*

Duke UNIVERSITY

Higher-order Markov Models

0th order: A C G C T A

P(G)

1st order: A C G C T A

P(G|C)

2nd order: A C G C T A

P(G|AC)

$$P_e(g_n \mid g_0 \cdots g_{n-1}, q_j) \approx \frac{C(g_0 \cdots g_n, q_j)}{\sum_{s \in \alpha} C(g_0 \cdots g_{n-1} s, q_j)}$$

Higher-order Markov Models

	order	nucleotides			splice sites		starts/ stops		exons			genes	
		Sn	*Sp*	*F*	*Sn*	*Sp*	*Sn*	*Sp*	*Sn*	*Sp*	*F*	*Sn*	#
H_{95}^{0}	0	92	97	94	79	76	57	53	62	59	60	19	93
H_{95}^{1}	1	95	98	97	87	81	64	61	72	68	70	25	127
H_{95}^{2}	2	98	98	98	91	82	65	62	76	69	72	27	136
H_{95}^{3}	3	98	98	98	91	82	67	63	76	69	72	28	140
H_{95}^{4}	4	98	97	98	90	81	69	64	76	68	72	29	143
H_{95}^{5}	5	98	97	98	90	81	66	62	74	67	70	27	137

Variable-Order Markov Models

$$P_e^{backoff}\left(g_n \middle| g_0 \ldots g_{n-1}, q_j\right) =$$

$$\begin{cases} \dfrac{C(g_0 \ldots g_n, q_j)}{\sum_{s \in \alpha} C(g_0 \ldots g_{n-1} s, q_j)} & \text{if } C(g_0 \ldots g_{n-1}, q_j) \geq K \text{ or } n = 0 \\ P_e^{backoff}\left(g_n \middle| g_1 \ldots g_{n-1}, q_j\right) & \text{otherwise} \end{cases}$$

AGC…TA
0 1 2 ..... n-1 n

$$P_e^{IMM}(g_n | g_0 \ldots g_{n-1}) =$$

$$\begin{cases} \lambda_n^G P_e(g_n | g_0 \ldots g_{n-1}) + (1 - \lambda_n^G) P_e^{IMM}(g_n | g_1 \ldots g_{n-1}) & \text{if } n > 0 \\ P_e(g_n) & \text{if } n = 0 \end{cases}$$

$$\lambda_n^G = \begin{cases} 1 & \text{if } m \geq 400 \\ 0 & \text{if } m < 400 \text{ and } c < 0.5 \\ \dfrac{c}{400} \sum_{x \in \alpha} C(g_0 \ldots g_{n-1} x) & \text{otherwise} \end{cases}$$

$c = 1 - P(\chi^2)$
=confidence that k^{th} order is significantly different

Duke UNIVERSITY

Interpolation Results

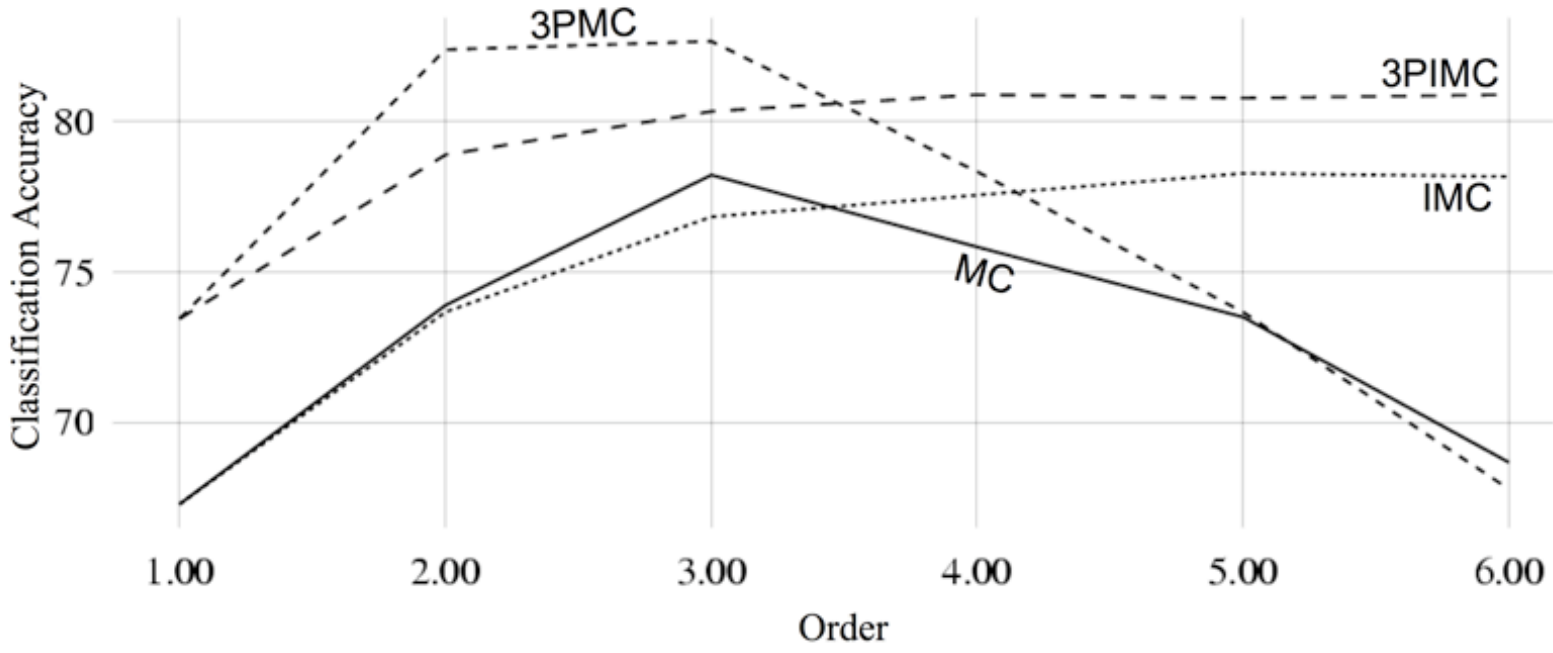


***Figure 7.4**: Relative accuracy of Markov chains (MC), IMC's, three-periodic MC's (3PMC), and three-periodic IMC's (3PIMC) on a particular task involving classification of human DNA sequences as coding versus noncoding.*

Summary

- An HMM is a *stochastic generative model* which emits sequences
- Parsing with an HMM can be accomplished using a *decoding algorithm* (such as *Viterbi*) to find the most probable state-path generating the input sequence
- When state-labeled sequences are available, training of HMMs can be accomplished using *labeled sequence training*
- Otherwise, training of HMMs can be accomplished using *Expectation-Maximization (EM) (next lesson...)*
- *Posterior decoding* can be used to estimate the probability that a given symbol or substring was generate by a particular state *(next lesson...)*