BMI/CS 776
Lecture 7
Prokaryotic Gene Finding

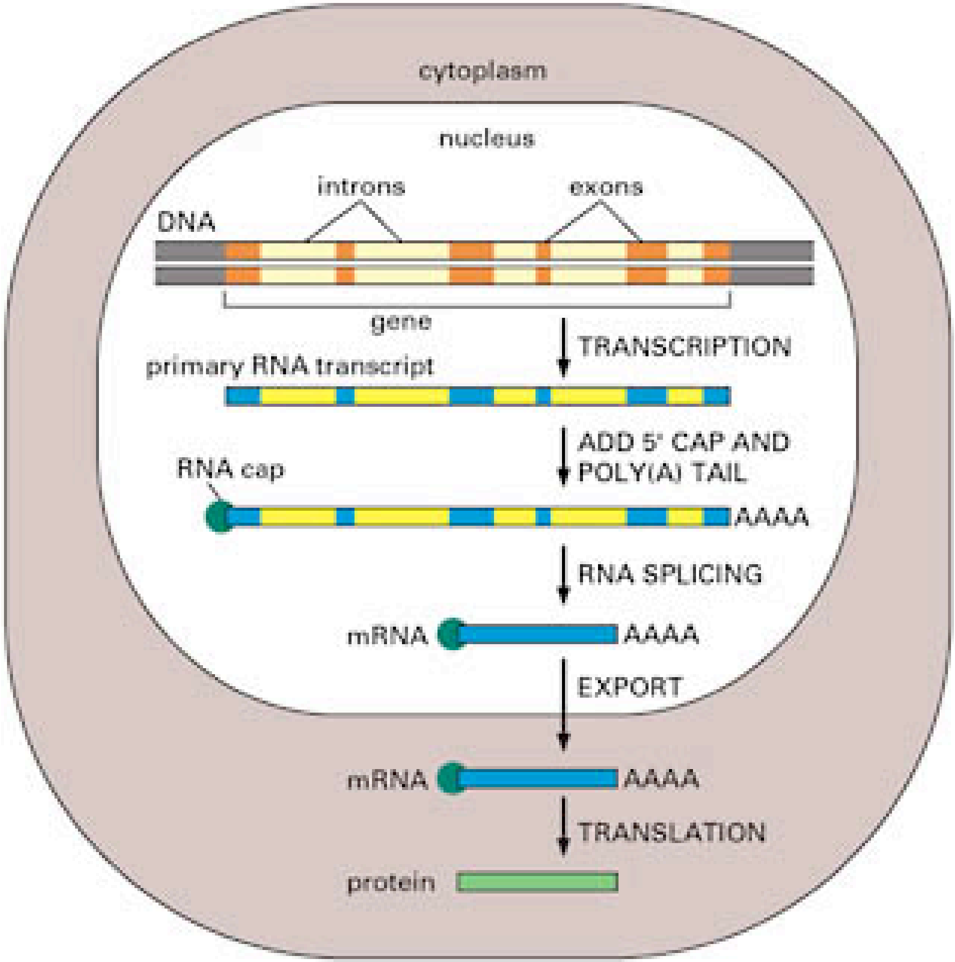
Colin Dewey

(adapted from slides by Mark Craven)

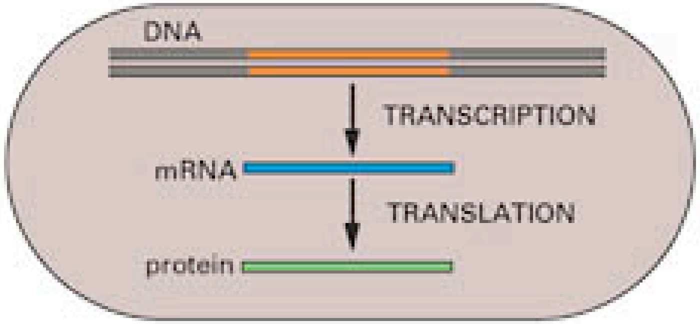
2007.02.13

Gene Expression Revisited

eukaryotes



prokaryotes



Approaches to Finding Genes

- **search by sequence similarity**: find genes by looking for matches to sequences that are known to be related to genes

- **search by signal**: find genes by identifying the sequence *signals* involved in gene expression

- **search by content**: find genes by statistical properties that distinguish protein-coding DNA from non-coding DNA

- combined: state-of-the-art systems for gene finding combine these strategies

Gene Finding: Search by Content

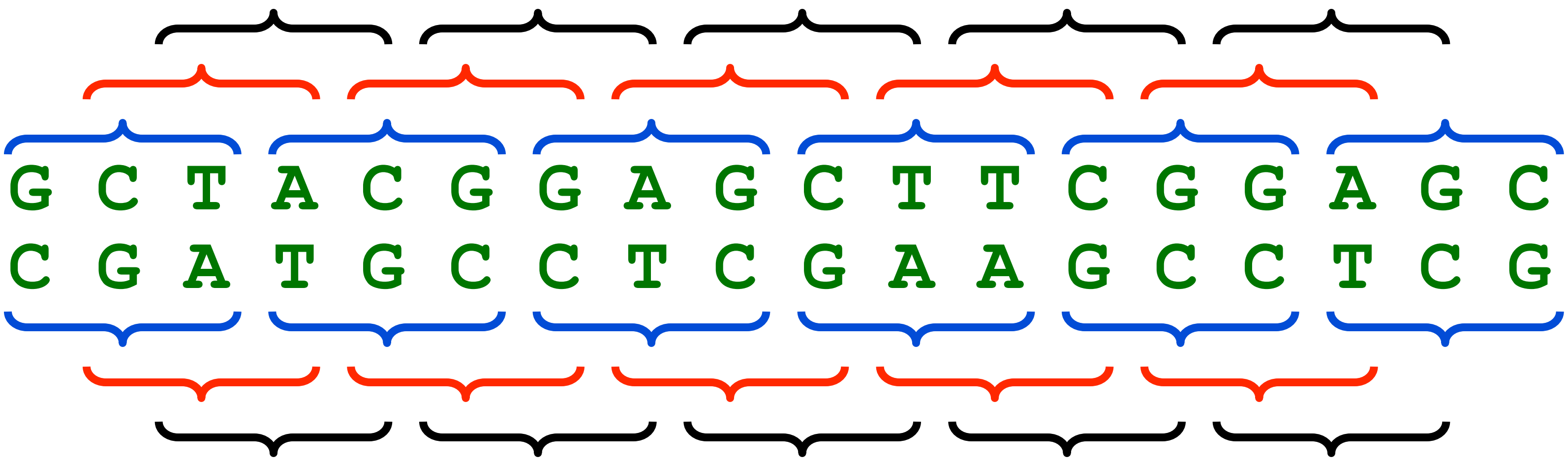
- encoding a protein affects the statistical properties of a DNA sequence
  - some amino acids are used more frequently than others (Leu more popular than Trp)
  - different numbers of codons for different amino acids (Leu has 6, Trp has 1)
  - for a given amino acid, usually one codon is used more frequently than others
    - this is termed *codon preference*
    - these preferences vary by species

Codon Preference in E. Coli

AA	codon	/1000
Gly	GGG	1.89
Gly	GGA	0.44
Gly	GGU	52.99
Gly	GGC	34.55
Glu	GAG	15.68
Glu	GAA	57.20
Asp	GAU	21.63
Asp	GAC	43.26

Reading Frames

- a given sequence may encode a protein in any of the six reading frames



Open Reading Frames (ORFs)

- an ORF is a sequence that
  - starts with a potential start codon
  - ends with a potential stop codon, *in the same reading frame*
  - doesn't contain another stop codon in-frame
  - and is sufficiently long (say > 100 bases)

`G T T A T G G C T ... T C G T G A T T`

- an ORF meets the minimal requirements to be a protein-coding gene in an organism without introns

Markov Models & Reading Frames

- consider modeling a given coding sequence
- for each "word" we evaluate, we'll want to consider its position with respect to the reading frame we're assuming

reading frame

G C T A C G G A G C T T C G G A G C

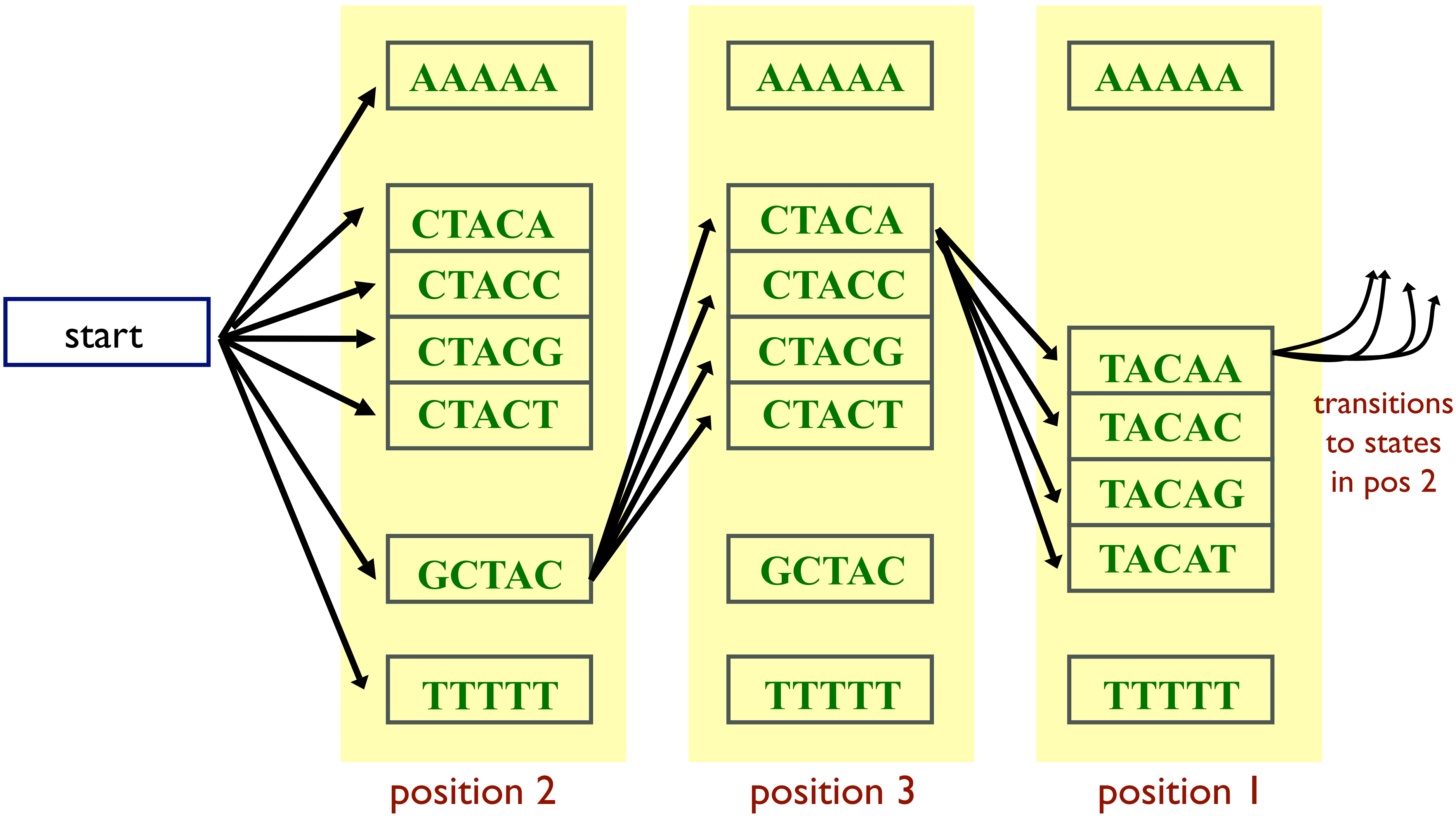
G C T A C **G** — G is in 3rd codon position

C T A C G **G** — G is in 1st position

T A C G G **A** — A is in 2nd position

- can do this using an inhomogenous model

A Fifth Order Inhomogenous Markov Chain



Selecting the Order of a Markov Chain Model

- higher order models remember more "history"
- additional history can have predictive value
- example:
  - predict the next word in this sentence fragment "…ends __"   (up, it, well, of, …?)
  - now predict it given more history

"…that ends ___"

Selecting the Order of a Markov Chain Model

- higher order models remember more "history"
- additional history can have predictive value
- example:
  - predict the next word in this sentence fragment "…ends __" (up, it, well, of, …?)

  – now predict it given more history

"…that ends ___"

"…well that ends ___"

Selecting the Order of a Markov Chain Model

- higher order models remember more "history"
- additional history can have predictive value
- example:
  - predict the next word in this sentence fragment "…ends ___" (up, it, well, of, …?)

– now predict it given more history

"…that ends ___"

"…well that ends ___"

"All's well that ends ___"

Selecting the Order of a Markov Chain Model

- but the number of parameters we need to estimate grows exponentially with the order
  - for modeling DNA we need $O(4^{n+1})$ parameters for an nth order model

- the higher the order, the less reliable we can expect our parameter estimates to be
  - estimating the parameters of a 2^{nd} order homogenous Markov chain from the complete genome of E. Coli, we'd see each word > 72,000 times on average
  - estimating the parameters of an 8^{th} order chain, we'd see each word ~ 17 times on average

Interpolated Markov Models

- the IMM idea: manage this trade-off by interpolating among models of various orders

- *simple* linear interpolation:

$$\begin{aligned}\Pr_{\mathrm{IMM}}(x_i \mid x_{i-n},\ldots,x_{i-1}) = {} & \lambda_0 \Pr(x_i) \\ & + \lambda_1 \Pr(x_i \mid x_{i-1}) \\ & \ldots \\ & + \lambda_n \Pr(x_i \mid x_{i-n},\ldots,x_{i-1})\end{aligned}$$

- where $\sum_i \lambda_i = 1$

Interpolated Markov Models

- we can make the weights depend on the history
  - for a given order, we may have significantly more data to estimate some words than others
- *general* linear interpolation

$$
\begin{aligned}
\Pr_{\mathrm{IMM}}(x_i \mid x_{i-n},\ldots,x_{i-1}) = {} & \lambda_0 \Pr(x_i) \\
& + \lambda_1(x_{i-1}) \Pr(x_i \mid x_{i-1}) \\
& \ldots \\
& + \lambda_n(x_{i-n},\ldots,x_{i-1}) \Pr(x_i \mid x_{i-n},\ldots,x_{i-1})
\end{aligned}
$$

The GLIMMER System

- Salzberg et al., 1998
- system for identifying genes in bacterial genomes
- uses 8^{th} order, inhomogeneous, interpolated Markov chain models

IMMs in GLIMMER

- how does GLIMMER determine the λ values?
- first, let's express the IMM probability calculation recursively

$$\Pr_{\mathrm{IMM,n}}(x_i \mid x_{i-n},\ldots,x_{i-1}) =$$

$$\lambda_{\mathrm{n}}(x_{i-n},\ldots,x_{i-1})\Pr(x_i \mid x_{i-n},\ldots,x_{i-1}) +$$

$$[1-\lambda_n(x_{i-n},\ldots,x_{i-1})]\Pr_{\mathrm{IMM,n\text{-}1}}(x_i \mid x_{i-n+1},\ldots,x_{i-1})$$

- let $c(x_{i-n},\ldots,x_{i-1})$ be the number of times we see the history $x_{i-n},\ldots,x_{i-1}$ in our training set

$$\lambda_n(x_{i-n},\ldots,x_{i-1}) = 1 \quad \text{if} \quad c(x_{i-n},\ldots,x_{i-1}) > 400$$

IMMs in GLIMMER

- if we haven't seen $x_{i-n},\ldots,x_{i-1}$ more than 400 times, then compare the counts for the following:

*n*th order history + base	(*n-1*)th order history + base
$x_{i-n},\ldots,x_{i-1},a$	$x_{i-n+1},\ldots,x_{i-1},a$
$x_{i-n},\ldots,x_{i-1},c$	$x_{i-n+1},\ldots,x_{i-1},c$
$x_{i-n},\ldots,x_{i-1},g$	$x_{i-n+1},\ldots,x_{i-1},g$
$x_{i-n},\ldots,x_{i-1},t$	$x_{i-n+1},\ldots,x_{i-1},t$

- use a statistical test (χ^2) to get a value d indicating our confidence that the distributions represented by the two sets of counts are different

IMMs in GLIMMER

- putting it all together

$$\lambda_n(x_{i-n},\ldots,x_{i-1}) = \begin{cases} 1 & \text{if } c(x_{i-n},\ldots,x_{i-1}) > 400 \\ d \times \dfrac{c(x_{i-n},\ldots,x_{i-1})}{400} & \text{else if } d \geq 0.5 \\ 0 & \text{otherwise} \end{cases}$$

where $d \in (0,1)$

IMM Example

- suppose we have the following counts from our training set

ACGA	25	CGA	100	GA	175
ACGC	40	CGC	90	GC	140
ACGG	15	CGG	35	GG	65
ACGT	20	CGT	75	GT	120
	100		300		500

χ^2 test: $d = 0.857$ χ^2 test: $d = 0.141$

$$\lambda_3(\mathbf{ACG}) = 0.857 \times 100/400$$

$$\lambda_2(\mathbf{CG}) = 0 \quad (d < 0.5,\ c(\mathbf{CG}) < 400)$$

$$\lambda_1(\mathbf{G}) = 1 \quad (c(\mathbf{G}) > 400)$$

IMM Example (Continued)

- now suppose we want to calculate $\Pr_{\mathrm{IMM},3}(T \mid ACG)$

$$\Pr_{\mathrm{IMM},1}(T \mid G) = \lambda_1(G)\Pr(T \mid G) + \big(1 - \lambda_1(G)\big)\Pr_{\mathrm{IMM},0}(T)$$
$$= \Pr(T \mid G)$$

$$\Pr_{\mathrm{IMM},2}(T \mid CG) = \lambda_2(CG)\Pr(T \mid CG) + \big(1 - \lambda_2(CG)\big)\Pr_{\mathrm{IMM},1}(T \mid G)$$
$$= \Pr(T \mid G)$$

$$\Pr_{\mathrm{IMM},3}(T \mid ACG) = \lambda_3(ACG)\Pr(T \mid ACG) + \big(1 - \lambda_3(ACG)\big)\Pr_{\mathrm{IMM},2}(T \mid CG)$$
$$= 0.857 \times \Pr(T \mid ACG) + (1 - 0.857) \times \Pr(T \mid G)$$

Gene Recognition in GLIMMER

- essentially ORF classification
- for each ORF
  - calculate the prob of the ORF sequence in each of the 6 possible reading frames
  - if the highest scoring frame corresponds to the reading frame of the ORF, mark the ORF as a gene
- for overlapping ORFs that look like genes
  - score overlapping region separately
  - predict only one of the ORFs as a gene

GLIMMER Experiment

- 8^{th} order IMM vs. 5^{th} order Markov model
- trained on 1168 genes (ORFs really)
- tested on 1717 annotated (more or less known) genes

Accuracy Metrics

		actual class	
		positive	negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

$$\text{sensitivity (recall)} = \frac{\text{TP}}{\text{actual pos}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{specificity} = \frac{\text{TN}}{\text{actual neg}} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

$$\text{precision} = \frac{\text{TP}}{\text{predicted pos}} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

GLIMMER Results

	TP	FN	FP & TP?
Model	Genes found	Genes missed	Additional genes
GLIMMER IMM	1680 (97.8%	37	209
5^{th}-Order Markov	1574 (91.7%)	143	104

The first column indicates how many of the 1717 annotated genes in *H.influenzae* were found by each algorithm. The 'additional genes' column shows how many extra genes, not included in the 1717 annotated entries, were called genes by each method.

- GLIMMER has greater sensitivity than the baseline
- it's not clear if its precision/specificity is better

An Alternative Approach: Back-off Models

- devised for language modeling
  [Katz, *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1987]

$$\Pr_{BACK}(x_i \mid x_{i-n},\ldots,x_{i-1}) = \begin{cases} (1-\delta)\dfrac{c(x_{i-n},\ldots,x_i)}{c(x_{i-n},\ldots,x_{i-i})}, & \text{if } c(x_{i-n},\ldots,x_i) > k \\ \lambda \Pr_{BACK}(x_i \mid x_{i-n+1},\ldots,x_{i-1}), & \text{otherwise} \end{cases}$$

- use *n*th order probability if we've seen this sequence (*history + current character*) k times
- otherwise back off to lower-order

An Alternative Approach: Back-off Models

$$\Pr_{BACK}(x_i \mid x_{i-n},\ldots,x_{i-1}) = \begin{cases} (1-\delta)\dfrac{c(x_{i-n},\ldots,x_i)}{c(x_{i-n},\ldots,x_{i-1})}, & \text{if } c(x_{i-n},\ldots,x_i) > k \\ \lambda \Pr_{BACK}(x_i \mid x_{i-n+1},\ldots,x_{i-1}), & \text{otherwise} \end{cases}$$

- why do we need δ and λ ?
- δ: save some probability mass for sequences we haven't seen
- λ: distribute this saved mass to lower-order sequences (different λ for each history; really $\lambda(x_{i-n+1},\ldots,x_{i-1})$)

Simple Back-off Example

$$\Pr_{BACK}(x_i \mid x_{i-n},\ldots,x_{i-1}) = \begin{cases} (1-\delta)\dfrac{c(x_{i-n},\ldots,x_i)}{c(x_{i-n},\ldots,x_{i-1})}, & \text{if } c(x_{i-n},\ldots,x_i) > k \\ \lambda \Pr_{BACK}(x_i \mid x_{i-n+1},\ldots,x_{i-1}), & \text{otherwise} \end{cases}$$

- given training sequence: **`TAACGACACG`**
- suppose $\delta = 0.2$ and $k = 0$

Simple Back-off Example

$$\Pr_{BACK}(x_i \mid x_{i-n},\ldots,x_{i-1}) = \begin{cases} (1-\delta)\dfrac{c(x_{i-n},\ldots,x_i)}{c(x_{i-n},\ldots,x_{i-1})}, & \text{if } c(x_{i-n},\ldots,x_i) > k \\ \lambda \Pr_{BACK}(x_i \mid x_{i-n+1},\ldots,x_{i-1}), & \text{otherwise} \end{cases}$$

- given training sequence: **TAACGACACG**
- suppose $\delta = 0.2$ and $k = 0$

$$\Pr_{BACK}(A) = \frac{4}{10}$$

$$\Pr_{BACK}(C) = \frac{3}{10}$$

$$\Pr_{BACK}(G) = \frac{2}{10}$$

$$\Pr_{BACK}(T) = \frac{1}{10}$$

Simple Back-off Example

$$\Pr_{BACK}(x_i \mid x_{i-n},\ldots,x_{i-1}) = \begin{cases} (1-\delta)\dfrac{c(x_{i-n},\ldots,x_i)}{c(x_{i-n},\ldots,x_{i-1})}, & \text{if } c(x_{i-n},\ldots,x_i) > k \\ \lambda \Pr_{BACK}(x_i \mid x_{i-n+1},\ldots,x_{i-1}), & \text{otherwise} \end{cases}$$

- given training sequence: **TAACGACACG**
- suppose δ = 0.2 and k = 0

$$\Pr_{BACK}(A) = \frac{4}{10}$$

$$\Pr_{BACK}(C) = \frac{3}{10}$$

$$\Pr_{BACK}(G) = \frac{2}{10}$$

$$\Pr_{BACK}(T) = \frac{1}{10}$$

$$\Pr_{BACK}(A \mid A) = (1-\delta)\frac{1}{4} = 0.2$$

$$\Pr_{BACK}(C \mid A) = (1-\delta)\frac{3}{4} = 0.6$$

Simple Back-off Example

$$\Pr_{BACK}(x_i \mid x_{i-n},\ldots,x_{i-1}) = \begin{cases} (1-\delta)\dfrac{c(x_{i-n},\ldots,x_i)}{c(x_{i-n},\ldots,x_{i-1})}, & \text{if } c(x_{i-n},\ldots,x_i) > k \\ \lambda \Pr_{BACK}(x_i \mid x_{i-n+1},\ldots,x_{i-1}), & \text{otherwise} \end{cases}$$

- given training sequence: **TAACGACACG**
- suppose $\delta = 0.2$ and $k = 0$

$$\Pr_{BACK}(A) = \frac{4}{10} \qquad \Pr_{BACK}(A \mid A) = (1-\delta)\frac{1}{4} = 0.2$$

$$\Pr_{BACK}(C) = \frac{3}{10} \qquad \Pr_{BACK}(C \mid A) = (1-\delta)\frac{3}{4} = 0.6$$

$$\Pr_{BACK}(G) = \frac{2}{10} \qquad \Pr_{BACK}(G \mid A) = \left[\frac{\delta}{\Pr_{BACK}(G) + \Pr_{BACK}(T)}\right] \times \Pr_{BACK}(G) = \frac{0.2}{0.3} \times 0.2$$

$$\Pr_{BACK}(T) = \frac{1}{10} \qquad \Pr_{BACK}(T \mid A) = \left[\frac{\delta}{\Pr_{BACK}(G) + \Pr_{BACK}(T)}\right] \times \Pr_{BACK}(T) = \frac{0.2}{0.3} \times 0.1$$