6.047 / 6.878 Computational Biology: Genomes, Networks, Evolution
Fall 2008

Conditional Random Fields for Computational Gene Prediction

Lecture 18

November 4, 2008

Genome Annotation

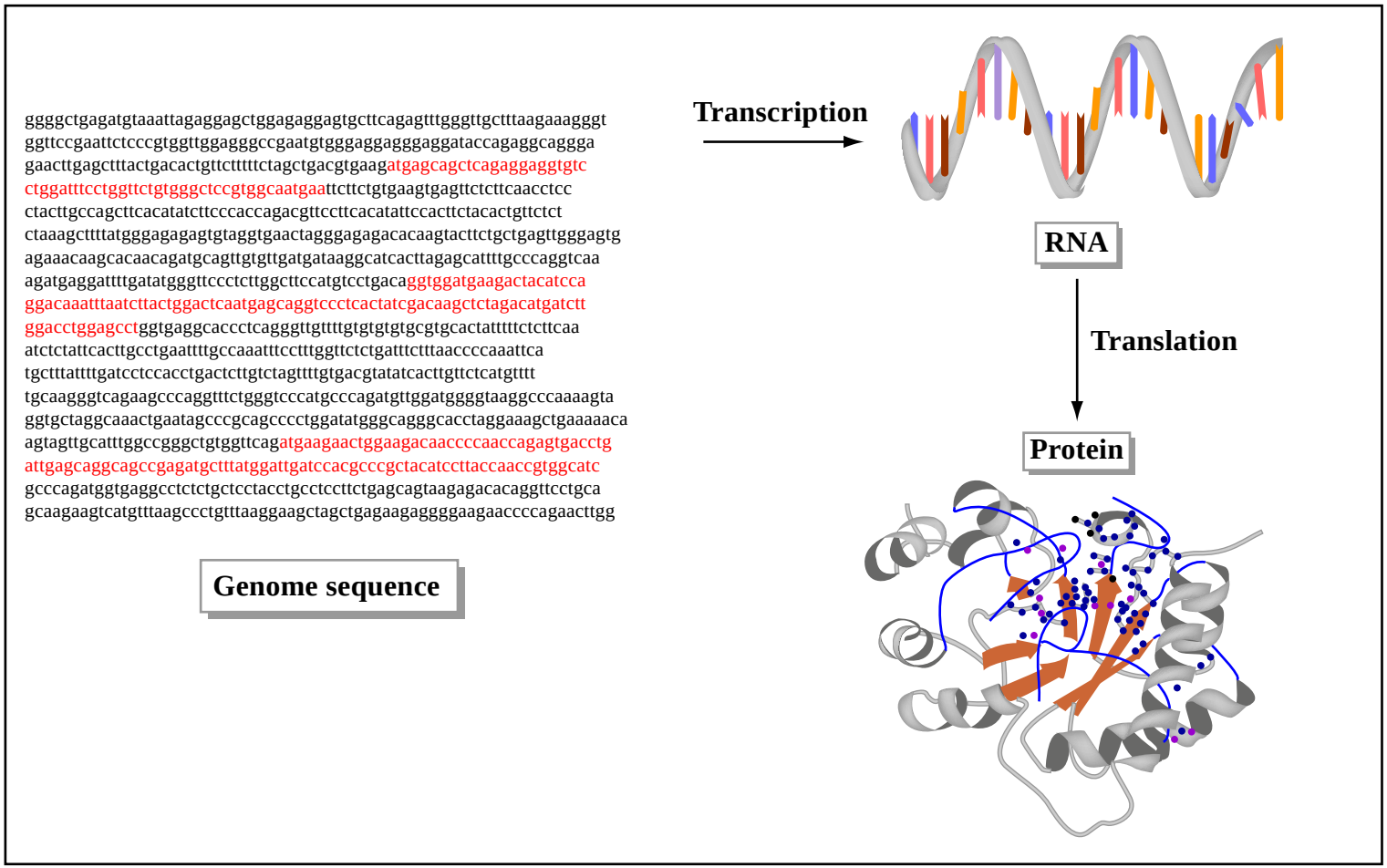
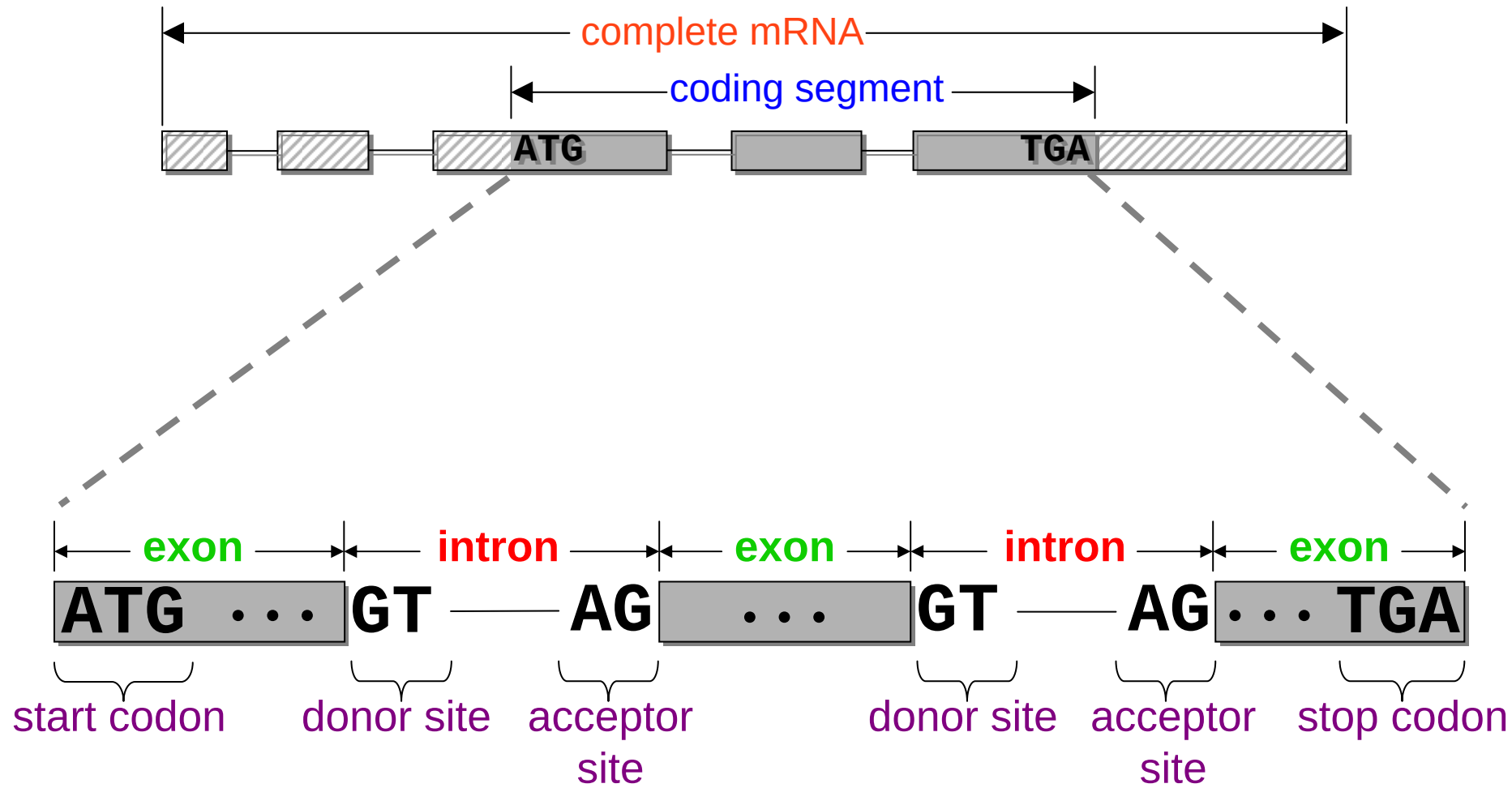


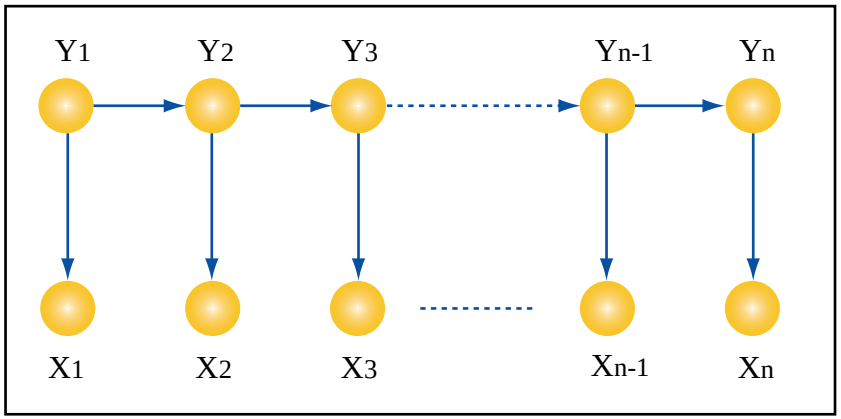
Figure by MIT OpenCourseWare.

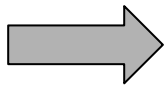
Eukaryotic Gene Structure



Courtesy of William Majoros. Used with permission.

Gene Prediction with HMM





Start
Start
start
Exon
Exon
Exon
Exon
5' Splice
5' Splice
Intron
Intron
Intron
Intron
3' Splice
3' Splice

ATGCCCCAGTTTTTGT

Figure by MIT OpenCourseWare.

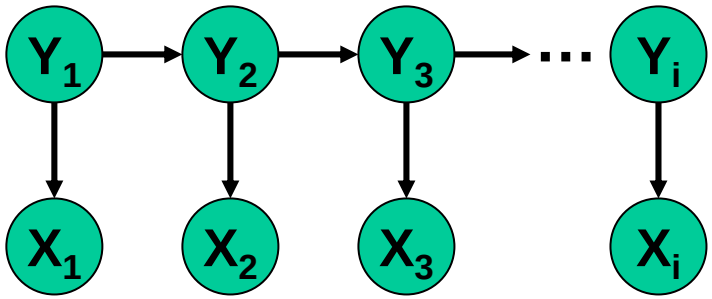
Model of joint distribution P(Y,X) = P(Labels,Seq)

For gene prediction, we are given X…

How do we select a Y efficiently?

Limitations of HMM Approach (1)

All components of HMMs have strict probabilistic semantics



$P(Y_i=\text{exon}|Y_{i-1}=\text{intron})$

$P(X_i=G|Y_i=\text{exon})$

Each sums to 1, etc..

P(HMMer|exon)? P(Blast Hit|exon)?

What about incorporating both Blast and Hmmer?

Dependent Evidence

- HMMer protein domains predictions come from models *based on known protein sequences*
  - Protein sequences for the same domain are aligned
  - Conservation modelled with HMM
- But these are the *same proteins searched by BLAST*
- If we see a HMMer hit, we are already more likely to get a BLAST hit, and vice versa

BLAST and HMMER do not provide independent evidence
- Dependence is the rule for most evidence

Dependent Evidence in HMMs

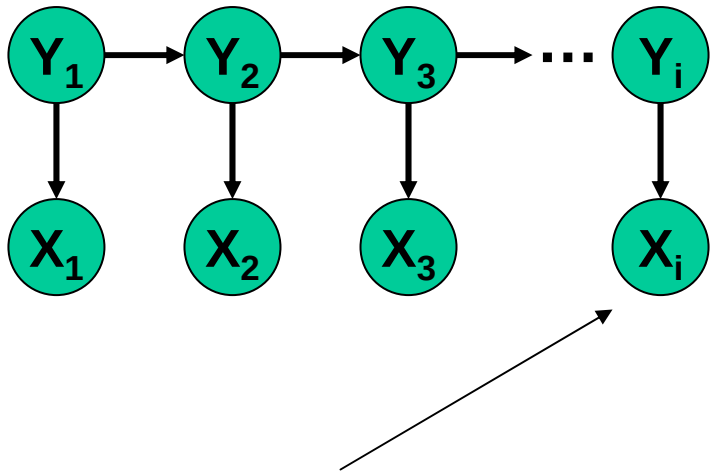
- HMMs explicitly model $P(X_i|Y_i)=P(\text{Blast,Hmmer}|Y_i)$
  - Not enough to know $P(\text{HMMer}|Y_i)$, also need to know $P(\text{HMMer}|Y_i,\text{Blast})$
  - Need to model these dependencies in the input data
- Every time we add new evidence (i.e. ESTs) we need to know about dependence on previous evidence
  - E.g. not just $P(\text{EST}|Y_i)$ but $P(\text{EST}|Y_i,\text{Blast,HMMer})$
- Unpleasant and unnecessary for our task: classification
- A common strategy is to *simply assume independence* (Naïve Bayes assumption)

$$P(X_1,X_2,X_3,...,X_N|Y_i)=\prod_i P(X_i|Y_i)$$

- Almost always a false assumption

Independencies in X

HMMs make assumptions about dependencies in X

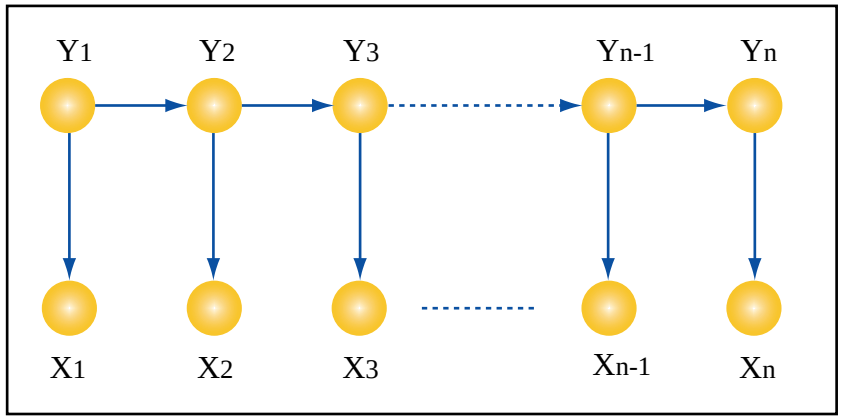


$$P(X_i|Y_i,Y_{i-1},Y_{i-2},Y_{i-3},\ldots,Y_1) = P(X_i|Y_i)$$

Effectively each Y_i "looks" at only a contiguous subset of X given the previous Y_{i-1}

Limitations Stem from Generative Modeling

HMMs are models of full joint probability distribution P(X,Y)



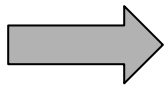


Figure by MIT OpenCourseWare.

$$P(X,Y) = P(Y|X)\,P(X)$$

But this is all we need for gene prediction!

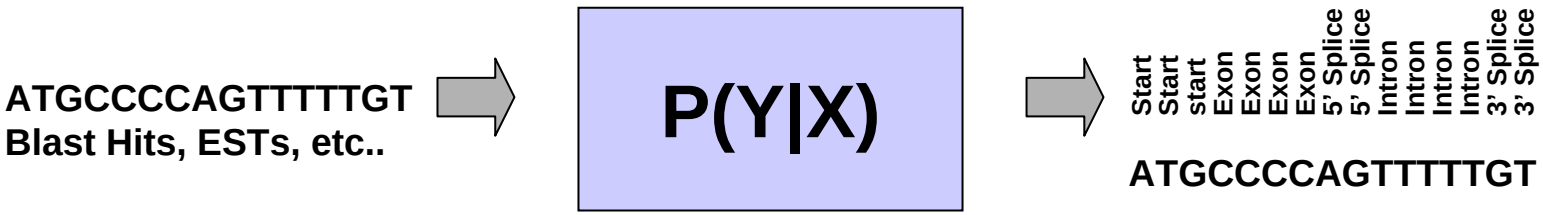
Generative Modeling of P(X)

- HMMs expend unnecessary effort to model P(X) which is never needed for gene prediction
  - Must model dependencies in X

- During learning, we might trade accuracy in modeling P(Y|X) in order to model P(X) accurately
  - Less accurate gene prediction

Discriminative Models

Model conditional distribution P(Y|X) *directly*



Discriminative models outperform generative models in several natural language processing tasks

Discriminative Model

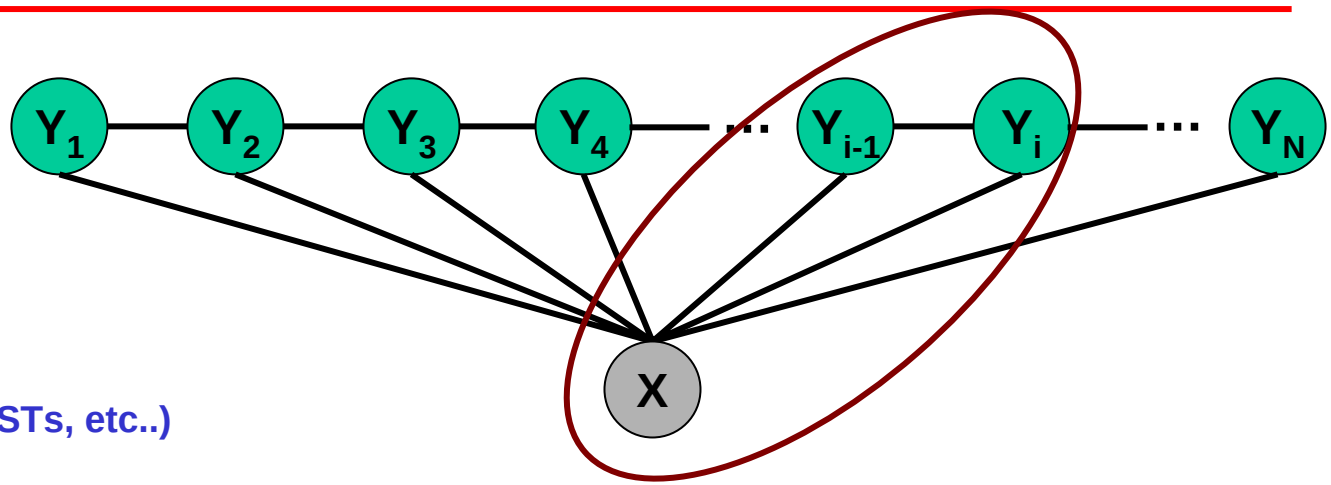
Desirable characteristics

1. **Efficient learning & inference algorithms**
2. **Easily incorporate diverse evidence**
3. **Build on best existing HMM models for gene calling**

Linear Chain CRF

Hidden state labels (exon, intron, etc)

Input data (sequence, blast hits, ESTs, etc..)



feature weights

feature functions

$$P(Y|X)=\frac{1}{Z(X)}\exp\left\{\sum_{j=1}^{J}\lambda_j\sum_{i}^{N}f_j\left(Y_i,Y_{i-1},X\right)\right\}$$

normalization

$$Z(X)=\sum_{Y}\exp\left\{\sum_{j=1}^{J}\lambda_j\sum_{i}^{N}f_j\left(Y_i,Y_{i-1},X\right)\right\}$$

The Basic Idea

$$P(Y|X)=\frac{1}{Z(X)}\exp\left\{\sum_{j=1}^{J}\lambda_j\sum_{i}^{N} f_j\left(Y_i,Y_{i-1},X\right)\right\}$$

- Feature functions, f_j, return real values on pairs of labels and input data that we think are important for determining P(Y|X)
  - e.g. If the last state (y_{i-1}) was intron and we have a blast hit (x), we have a different probability for whether we are in an exon (y_i) now.
- We may not know *how* this probability has changed *or dependence* other evidence
- We *learn* this by selecting weights, λ_j, to maximize the likelihood of training data
- Z(X) is a normalization constant that ensure that P(Y|X) sums to one over all possible Ys

Using CRFs

$$P(Y|X) = \frac{1}{Z(X)} \exp\left\{ \sum_{j=1}^{J} \lambda_j \sum_{i}^{N} f_j\left(Y_i, Y_{i-1}, X\right) \right\}$$

Design

1. Select feature functions on label pairs $\{Y_i, Y_{i-1}\}$ and X.

Inference

2. Given weights and feature functions, find the most probable labeling Y, given an input X

Learning

3. Use a training set of data to select the weights, λ.

What Are Feature Functions?

Core issue in CRF – selecting feature functions

1. **Features are arbitrary functions that return a real value for some pair of labels $\{Y_i, Y_{i-1}\}$, and the input, X**
   - **Indicator function – 1 for certain $\{Y_i, Y_{i-1}, X\}$, 0 otherwise**
   - **Sum, product, etc.. over labels and data**
   - ***Could* return some probability over $\{Y_i, Y_{i-1}, X\}$ – *but this is not required***

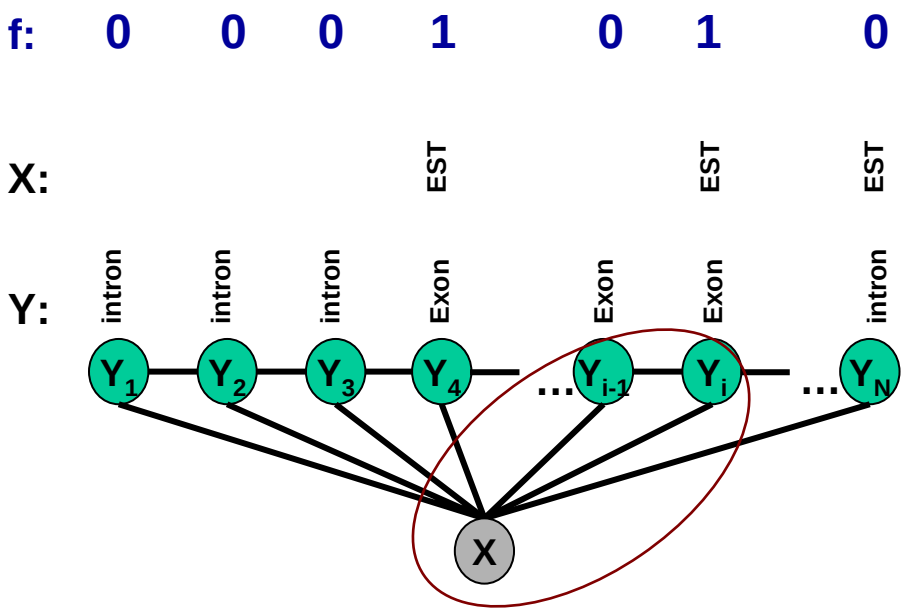
2. **We want to select feature functions that capture constraints or conjunctions of label pairs $\{Y_i, Y_{i-1}\}$, and the input, X that we think are important for P(Y|X)**

3. **Determine characteristics of the training data that must hold in our CRF model**

Example Feature Function

An BLAST hit at position i impacts the probability that Y_i = exon. To capture this, we can define an indicator function:

$$f_{\text{blast,exon}}\left(Y_i, Y_{i-1}, X\right) = \begin{cases} 1 & \text{if } Y_i = \text{exon and } X_i = \text{BLAST} \\ 0 & \text{otherwise} \end{cases}$$



$$P(Y|X) = \frac{\exp\left\{\lambda_{\text{blast,exon}} \sum_{i}^{N} f_{\text{blast,exon}}\left(Y_i, Y_{i-1}, X\right)\right\}}{Z(X)}$$

$$= \frac{\exp\left\{\lambda_{\text{blast,exon}}\left(0+0+0+1+0+1+0\right)\right\}}{Z(X)}$$

$$= \frac{\exp\left\{\lambda_{\text{blast,exon}} 2\right\}}{Z(X)}$$

Adding Evidence

An BLAST hit at position i impacts the probability that Y_i = exon. To capture this, we can define an indicator function:

$$f_{\text{blast,exon}}\left(Y_i, Y_{i-1}, X\right) = \begin{cases} 1 & \text{if } Y_i = \text{exon and } X_i = \text{BLAST} \\ 0 & \text{otherwise} \end{cases}$$

A protein domain predicted by the tool HMMer at position i *also* impacts the probability that Y_i = exon.

$$f_{\text{HMMer,exon}}\left(Y_i, Y_{i-1}, X\right) = \begin{cases} 1 & \text{if } Y_i = \text{exon and } X_i = \text{HMMer} \\ 0 & \text{otherwise} \end{cases}$$

But recall that these two pieces of evidence not independent

Dependent Evidence in CRFs

There is no requirement that evidence represented by feature functions be independent

- Why? CRFs do not model P(X)!
- All that matters is whether evidence constrains P(Y|X)
- The weights determine the extent to which each set of evidence contributes and interacts

A Strategy for Selecting Features

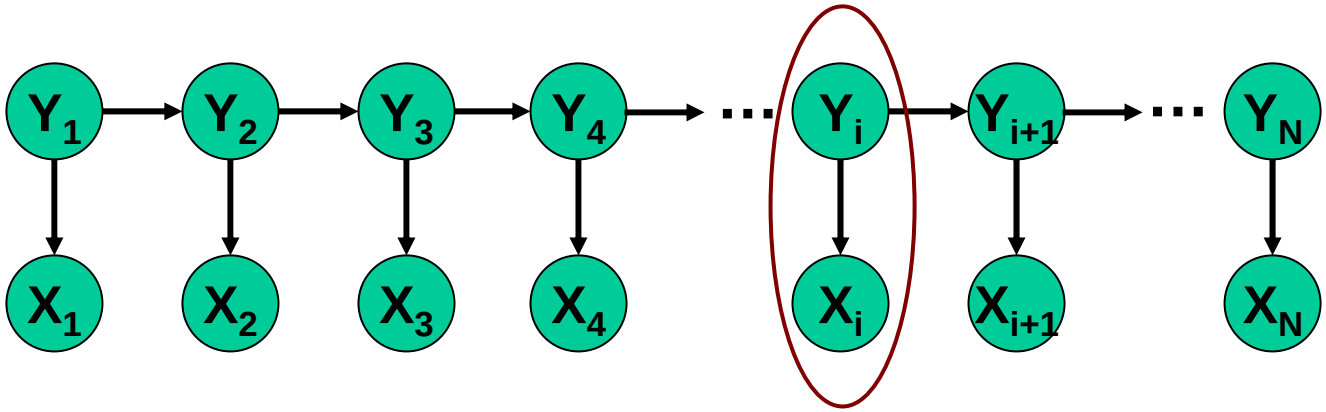
- Typical applications use thousands or millions of arbitrary indicator feature functions – brute force approach
- But we know gene prediction HMMs encode useful information

Strategy

1. Start with feature functions derived from best HMM based gene prediction algorithms

2. Use arbitrary feature functions to capture evidence hard to model probabilistically

Alternative Formulation of HMM

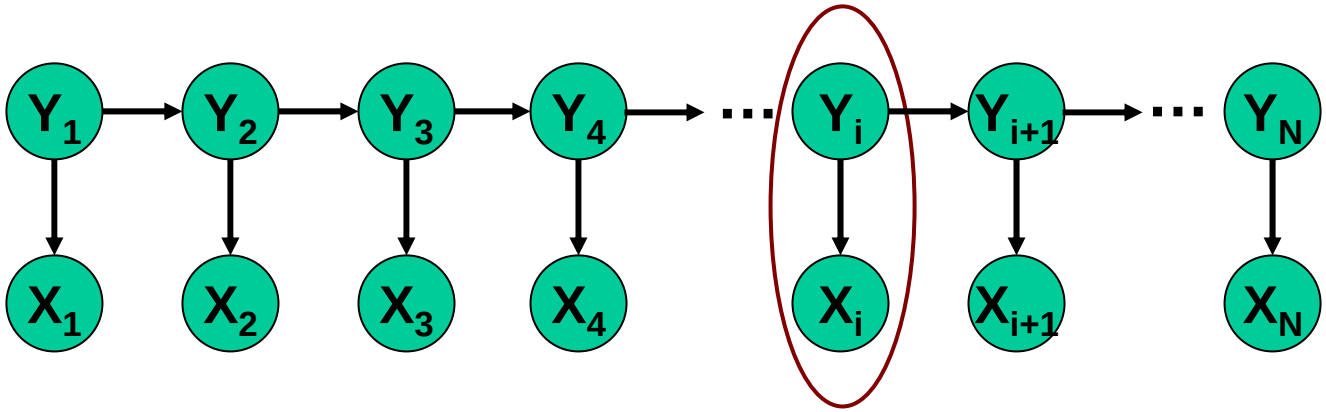


HMM probability factors over pairs of nodes

$$\mathrm{P}\left(y_i \mid y_{i-1}\right) \times \mathrm{P}\left(x_i \mid y_i\right) = \mathrm{P}\left(y_i, x_i \mid y_{i-1}\right)$$

$$\prod_{i=1}^{N} \mathrm{P}\left(y_i, x_i \mid y_{i-1}\right) = \mathrm{P}(\mathrm{Y},\mathrm{X})$$

Alternative Formulation of HMM



We can define a function, f, over each of these pairs

$$\log\left\{\mathrm{P}\left(y_i \mid y_{i-1}\right)\times\mathrm{P}\left(x_i \mid y_i\right)\right\} \equiv \mathrm{f}\left(y_i, y_{i-1}, x_i\right)$$

then,

Conditional Probability from HMM

$$P(Y|X) = \frac{P(Y,X)}{P(X)} = \frac{P(Y,X)}{\sum_{Y} P(Y,X)} = \frac{\exp\left\{\sum_{i=1}^{N} f\left(y_i, y_{i-1}, x_i\right)\right\}}{\sum_{Y} \exp\left\{\sum_{i=1}^{N} f\left(y_i, y_{i-1}, x_i\right)\right\}}$$

$$P(Y|X) = \frac{1}{Z(X)} \exp\left\{\sum_{i=1}^{N} 1\, f\left(y_i, y_{i-1}, x_i\right)\right\}$$

$$\text{where } Z(X) = \sum_{Y} \exp\left\{\sum_{i=1}^{N} 1\, f\left(y_i, y_{i-1}, x_i\right)\right\}$$

This is the formula for a linear chain CRF with all $\lambda = 1$

Implementing HMMs as CRFs

We can implement an HMM as a CRF by choosing

$$\mathrm{f}_{\mathrm{HMM}}\left(y_i, y_{i-1}, x_i\right) = \log\left\{\mathrm{P}\left(y_i \mid y_{i-1}\right) \times \mathrm{P}\left(x_i \mid y_i\right)\right\}$$

$$\lambda_{\mathrm{HMM}} = 1$$

Or more commonly

$$\mathrm{f}_{\mathrm{HMM\_Transition}}\left(y_i, y_{i-1}, x_i\right) = \log\left\{\mathrm{P}\left(y_i \mid y_{i-1}\right)\right\}$$

$$\mathrm{f}_{\mathrm{HMM\_Emission}}\left(y_i, y_{i-1}, x_i\right) = \log\left\{\mathrm{P}\left(x_i \mid y_i\right)\right\}$$

$$\lambda_{\mathrm{HMM\_Transition}} = \lambda_{\mathrm{HMM\_Emission}} = 1$$

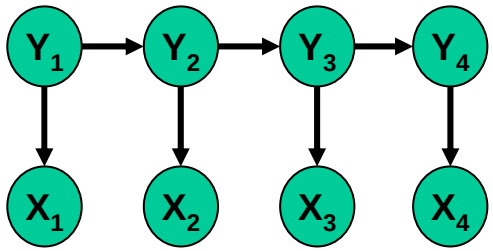
Either formulation creates a CRF that models that same conditional probability P(Y|X) as the original HMM

Adding New Evidence

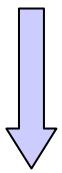
- Additional feature are added with arbitrary feature functions (i.e. $f_{blast,exon}$)
- When features are added, learning of weights *empirically determines the impact of new features relative to existing features* (i.e. relative value of λ_{HMM} vs $\lambda_{blast,exon}$)

CRFs provide a framework for incorporating diverse evidence into the best existing models for gene prediction

Conditional Independence of Y



Directed Graph Semantics

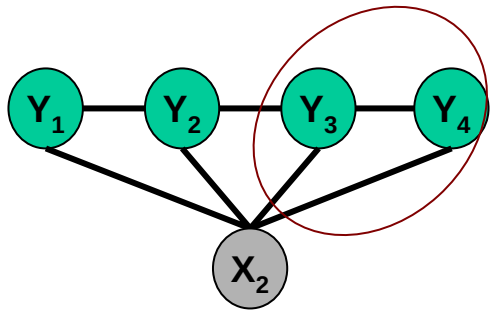


Factorization

$$P(X,Y) = \prod_{\text{all nodes v}} P(v|\text{parents}(v))$$

$$= \prod P(Y_i \mid Y_{i-1}) P(X_i|Y_i)$$

Bayesian Networks



Potential Functions over Cliques (conditioned on X)

Markov Random Field

Factorization

$$P(Y|X) = \prod_{\text{all nodes v}} U(\text{clique}(v),X)$$

$$= \prod P(Y_i \mid Y_{i-1},X)$$

Both cases: Y_i conditionally independent of all other Y given Y_{i-1}

Conditional-Generative Pairs

HMMs and linear chain CRFs explore the same family of conditional distributions P(Y|X)*

Can convert HMM to CRF

- Training an HMM to maximize P(Y|X) yields same decision boundary as CRF

Can convert CRF to HMM

- Training CRF to maximize P(Y,X) yields same classification boundary as HMM

Sutton, McCallum (CRF-Tutorial)

HMMs and CRFs form a *generative-discriminative pair*

Ng, Jordan (2002)

* Assuming P of the form exp(U(Y))/z – exponential family

Conditional-Generative Pairs

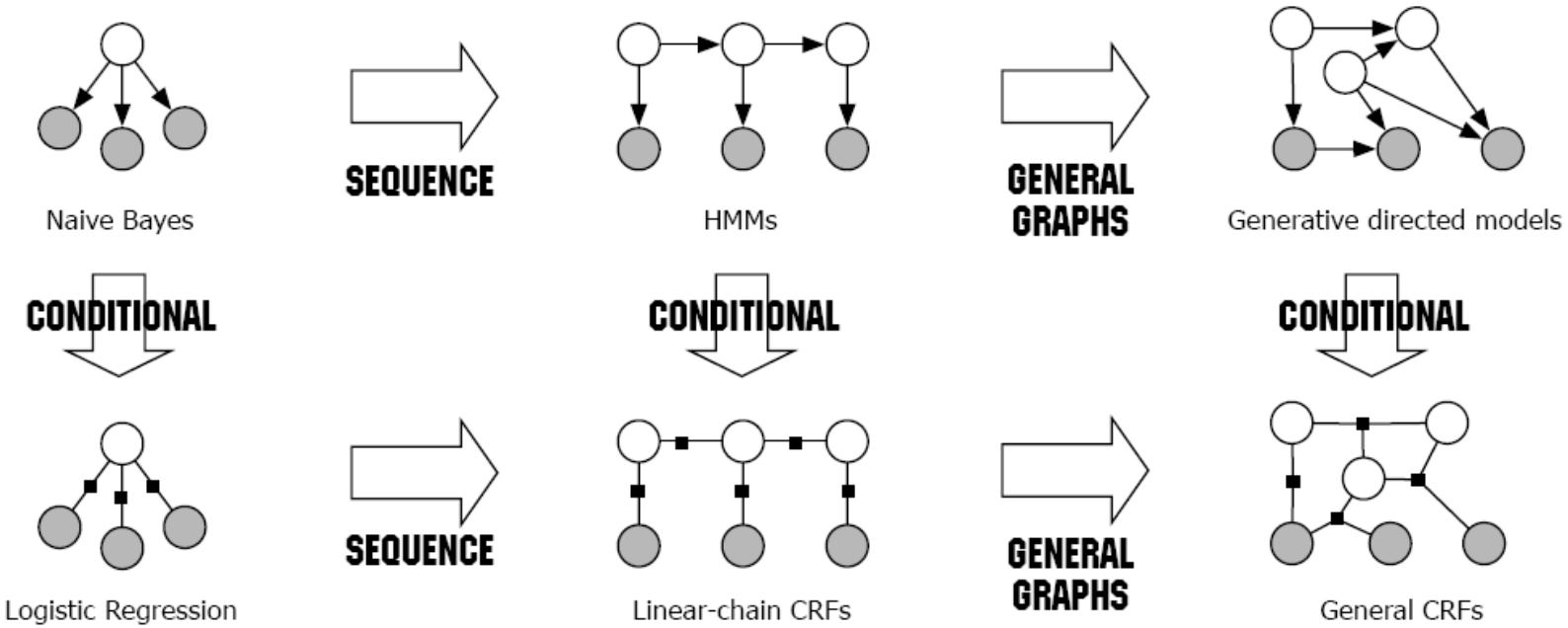


Figure 1.2 from "An Introduction to Conditional Random Fields for Relational Learning," Charles Sutton and Andrew McCallum.
Getoor, Lise, and Ben Taskar, editors. Introduction to Statistical Relational Learning. Cambridge, MA: MIT Press, 2007. ISBN: 978-0-262-07288-5.

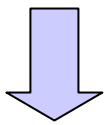
Sutton, C. and A. McCallum. *An Introduction to Conditional Random Fields for Relational Learning*.

Practical Benefit of Factorization

- Allows us to take a very large probability distribution and model it using much smaller distributions over "local" sets of variables

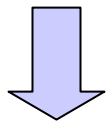
- Example: CRF with N states and 5 labels (ignore X for now)

$P(Y_1, Y_2, Y_3, \ldots, Y_N)$



5^N

$P_i(Y_i, Y_{i-1})$



$5*5*N$

(5*5 if $P_i = P_{i-1}$ for all i)

Using CRFs

$$P(Y|X) = \frac{1}{Z(X)} \exp\left\{ \sum_{j=1}^{J} \lambda_j \sum_{i}^{N} f_j\left(Y_i, Y_{i-1}, X\right) \right\}$$

Design 1. Select feature functions on label pairs $\{Y_i, Y_{i-1}\}$ and X.

Inference 2. Given weights and feature functions, find the most probable labeling Y, given an input X

Learning 3. Use a training set of data to select the weights, λ.

Labeling A Sequence

Given sequence & evidence X, we wish to select a labeling, Y, that is in some sense ‘best” given our model

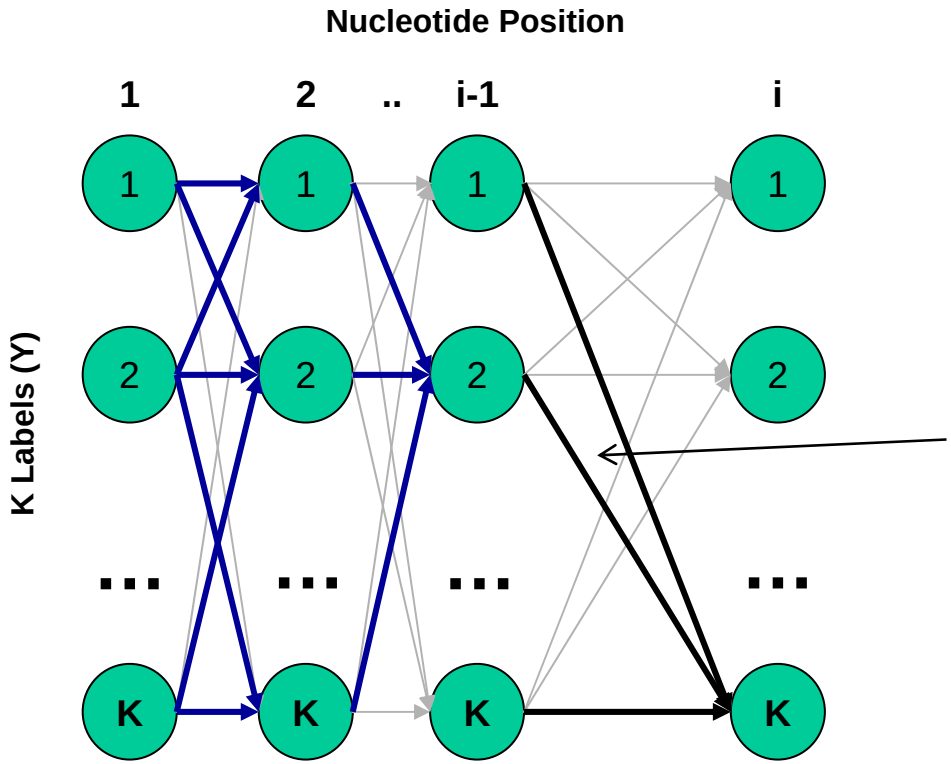
As with HMMs, one sensible choice is the most probable labeling given the data and model:

$$\arg\max_{Y} P(Y|X) = \arg\max_{Y} \left[\frac{1}{Z(X)} \exp\left\{ \sum_{j=1}^{J} \lambda_j \sum_{i}^{N} f_j\left(Y_i, Y_{i-1}, X\right) \right\} \right]$$

But of course, we don’t want to score every possible Y. This is where the chain structure of the linear chain CRF comes in handy…

Why?

Dynamic Programming



$V_k(i-1)$ = probability of most likely path through i-1 ending on K given X

Score derived from feature functions over $Y_{i-1}=2$ and $Y_i=k$

$$\exp\left\{\sum_{j=1}^{J}\lambda_j f_j(K,2,X)\right\}$$

$$V_l(i\text{-}1)\times\exp\left\{\sum_{j=1}^{J}\lambda_j f_j(k,l,X)\right\}$$

By Analogy With HMM

Recall from HMM lectures

$$V_k(i) = e_k(x_i) * \max_j \left(V_k(i) \times a_{jk} \right) = \max_j \left(V_k(i) \times a_{jk} \times e_k(x_i) \right)$$

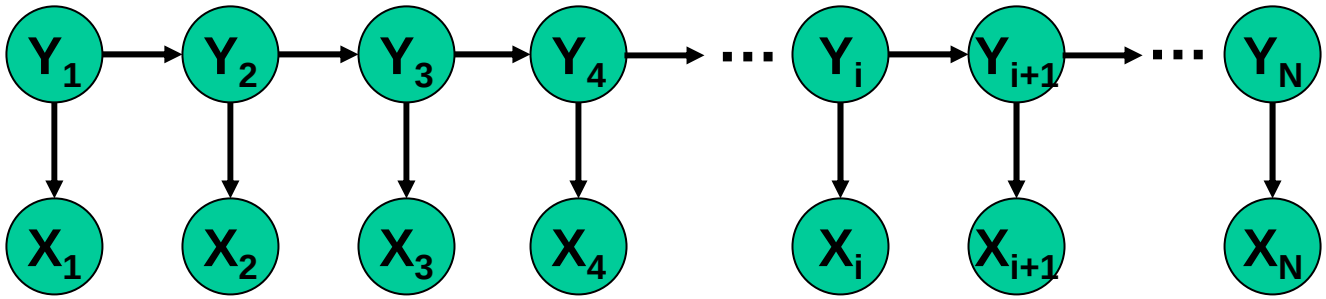
$$= \max_j \left(V_k(i) \times P(Y_i = j | Y_{i-1} = k) P(x_i | Y_i = k) \right)$$

$$= \max_j \left(V_k(i) \times \Psi_{HMM}(Y_i = j, Y_{i-1} = k, X) \right)$$

Where we have defined

$$\Psi_{HMM}(Y_i, Y_{i-1}, X) = P(Y_i | Y_{i-1}) P(X_i | Y_i)$$

Recall From Previous Slides



$$\log\left\{\mathrm{P}\left(y_i \mid y_{i-1}\right)\times \mathrm{P}\left(x_i \mid y_i\right)\right\} = \mathrm{f}\left(y_i, y_{i-1}, x_i\right)$$

$$\mathrm{P}\left(\mathrm{Y}|\mathrm{X}\right) = \frac{1}{\mathrm{Z(X)}}\exp\left\{\sum_{i=1}^{\mathrm{N}} \lambda_{\mathrm{HMM}}\mathrm{f}\left(y_i, y_{i-1}, x_i\right)\right\}$$

linear chain CRF

$$\Psi_{HMM}\left(\mathrm{Y_i}, \mathrm{Y_{i\text{-}1}}, \mathrm{X}\right) = \mathrm{P}(\mathrm{Y_i}|\mathrm{Y_{i\text{-}1}})\mathrm{P}(\mathrm{X_i}|\mathrm{Y_i}) = \exp\left\{\lambda_{\mathrm{HMM}}\mathrm{f}\left(y_i, y_{i-1}, x_i\right)\right\}$$

Combined HMM and CRF Inference

We can define the same quantity for a generic CRFs

$$\Psi_{CRF}(\mathrm{Y_i}, \mathrm{Y_{i\text{-}1}}, \mathrm{X}) = \exp\left\{\lambda_{\mathrm{k}} \mathrm{f_k}\left(y_i, y_{i-1}, x_i\right)\right\}$$

We can rewrite all HMM equations in terms of Ψ_{HMM}
If we then plug Ψ_{CRF} in for Ψ_{HMM} , *they work analogously:*

$$\mathrm{V_k}(\mathrm{i}) = \max_{l}\ \left(\mathrm{V}_l(\mathrm{i\text{-}1}) \times \Psi(\mathrm{k}, l, \mathrm{X})\right)$$ Viterbi

$$\alpha_k(i) = \sum_{l} \Psi(\mathrm{k}, l, \mathrm{X})\ \alpha_l(\mathrm{i\text{-}1})$$ Forward

$$\beta_k(i) = \sum_{l} \Psi(\mathrm{k}, l, \mathrm{X})\ \beta_l(\mathrm{i+1})$$ Backward

Using CRFs

$$P(Y|X) = \frac{1}{Z(X)} \exp\left\{ \sum_{j=1}^{J} \lambda_j \sum_{i}^{N} f_j\left(Y_i, Y_{i-1}, X\right) \right\}$$

Design

1. Select feature functions on label pairs $\{Y_i, Y_{i-1}\}$ and X.

Inference

2. Given weights and feature functions, find the most probable labeling Y, given an input X

Learning

3. Use a training set of data to select the weights, λ.

Maximum Likelihood Learning

- We assume an iid training set $\{(x^{(k)},y^{(k)})\}$ of K labeled sequences of length N
  - A set of manually curated genes sequences for which all nucleotides are labeled
- We then select weights, λ, that maximize the *log-likelihood, L(λ),* of the data

$$L(\lambda) = \sum_{k=1}^{K} \sum_{j=1}^{J} \lambda_j \sum_{i=1}^{N} f_j\left(Y_i^{(k)}, Y_{i-1}^{(k)}, X^{(k)}\right) - \sum_{k=1}^{K} \log Z\left(X^{(k)}\right)$$

Good news

L(λ) is concave - guaranteed global max

Maximum Likelihood Learning

- Maximum where $\partial L(\lambda)/\partial\lambda = 0$
- From homework, at maximum we know

$$\sum_k \sum_{i=1}^{N} f_j\left(Y_i^{(k)}, Y_{i-1}^{(k)}, X^{(k)}\right) = \sum_k \sum_{i=1}^{N} f_j\left(Y_i^{(k)}, Y_{i-1}^{(k)}, X^{(k)}\right) P_{\text{model}}\left(Y' \mid X^{(k)}\right)$$

Count in data **Expected count by model**

Features determine characteristics of the training data that must hold in our CRF model

Maximum entropy solution – no assumptions in CRF distribution other than feature constraints

Gradient Search

Bad news

No closed solution – need gradient method
Need efficient calculation of $\delta L(\lambda)/\delta\lambda$ and Z(X)

Outline

1. Define forward/backward variables akin to HMMs
2. Calculate Z(X) using forward/backward
3. Calculate $\delta L(\lambda)/\delta\lambda_i$ using Z(x) and forward/backward
4. Update each parameter with gradient search (quasi-Newton)
5. Continue until convergence to global maximum

Very slow – many iterations of forward/backward

Using CRFs

$$P(Y|X) = \frac{1}{Z(X)} \exp\left\{ \sum_{j=1}^{J} \lambda_j \sum_{i}^{N} f_j\left(Y_i, Y_{i-1}, X\right) \right\}$$

Design

1. Select feature functions on label pairs $\{Y_i, Y_{i-1}\}$ and X.

Inference

2. Given weights and feature functions, find the most probable labeling Y, given an input X

Learning

3. Use a training set of data to select the weights, λ.

CRF Applications to Gene Prediction

CRF actually work for gene prediction

- Culotta, Kulp, McCallum (2005)
- CRAIG (Bernal et al, 2007)
- Conrad (DeCaprio et al (2007), Vinson et al (2006))
- PhyloCRF (Majores, http://geneprediction.org/book/)

Conrad

- A Semi-Markov CRF
  - Explicit state duration models
  - Features over intervals
- Incorporates diverse information
  - GHMM features
  - Blast
  - EST
- Comparative Data
  - Genome alignments with model of evolution
  - Alignment gaps
- Alternative Objective Function
  - Maximize expected accuracy instead of likelihood during learning