# Conditional Random Fields

Mike Brodie
CS 778

# Motivation

- Part-Of-Speech Tagger

| | |
|---|---|
| CC | Coordinating conjunction |
| CD | Cardinal number |
| DT | Determiner |
| EX | Existential *there* |
| FW | Foreign word |
| IN | Preposition or subordinating conjunction |
| JJ | Adjective |
| JJR | Adjective, comparative |
| JJS | Adjective, superlative |
| LS | List item marker |
| MD | Modal |
| NN | Noun, singular or mass |
| NNS | Noun, plural |
| NNP | Proper noun, singular |
| NNPS | Proper noun, plural |
| PDT | Predeterminer |
| POS | Possessive ending |
| PRP | Personal pronoun |

| | |
|---|---|
| PRP$ | Possessive pronoun |
| RB | Adverb |
| RBR | Adverb, comparative |
| RBS | Adverb, superlative |
| RP | Particle |
| SYM | Symbol |
| TO | *to* |
| UH | Interjection |
| VB | Verb, base form |
| VBD | Verb, past tense |
| VBG | Verb, gerund or present participle |
| VBN | Verb, past participle |
| VBP | Verb, non-3rd person singular present |
| VBZ | Verb, 3rd person singular present |
| WDT | Wh-determiner |
| WP | Wh-pronoun |
| WP$ | Possessive wh-pronoun |
| WRB | Wh-adverb |

# Motivation

object

# Motivation

I object!

# Motivation

## object

‘Do you see that object?’

text2speech.org

# Motivation

- Part-Of-Speech Tagger - Java CRFTagger

INPUT FILE - Logan.txt

Logan woke up this morning and ate a big bowl of Fruit Loops. On his way to school, a small dog chased after him. Fortunately, Logan's leg had healed and he outran the dog.

# Motivation

- Part-Of-Speech Tagger - Java CRFTagger

OUTPUT FILE - Logan.txt.pos

Logan/NNP woke/VBD up/RP this/DT morning/NN and/CC ate/VB a/DT big/JJ bowl/NN of/IN Fruit/NNP Loops/NNP ./. On/IN his/PRP$ way/NN to/TO school/VB ,/, a/DT small/JJ dog/NN chased/VBN after/IN him/PRP ./. Fortunately/RB ,/, Logan/NNP 's/POS leg/NN had/VBD healed/VBN and/CC he/PRP outran/VB the/DT dog/NN ./.

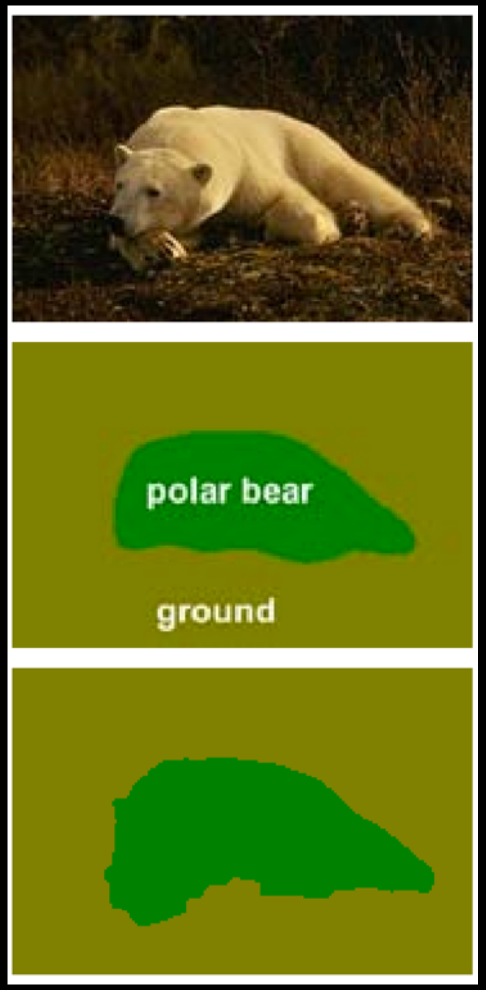
# Motivation

- Stanford Named Entity Recognition
  - Recognizes names, places, organizations
  - http://nlp.stanford.edu:8080/ner/process

# Motivation

- Image Applications

Segmentation



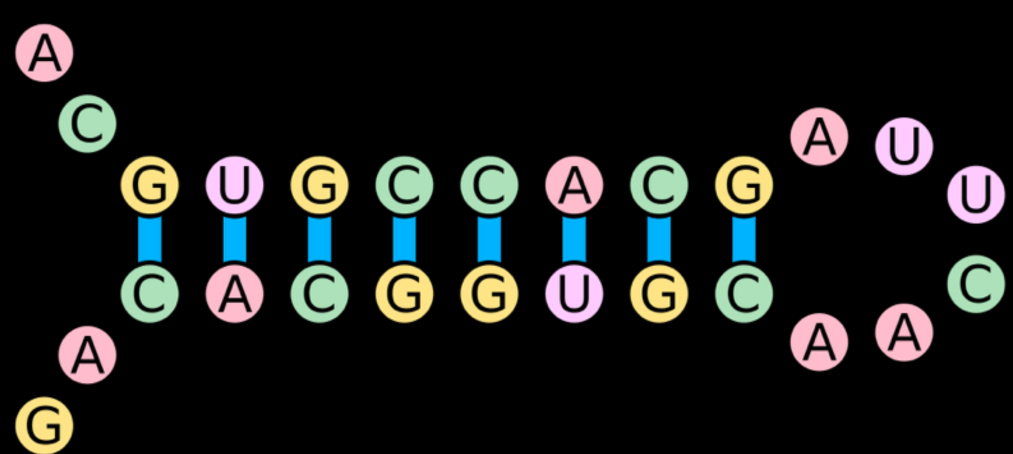

Multi-Label Classification

Labels:
Hat, Shirt
Cup, Stanley

# Motivation

- Bioinformatics Applications

RNA Structural Alignment




Protein Structure Prediction

# Motivation

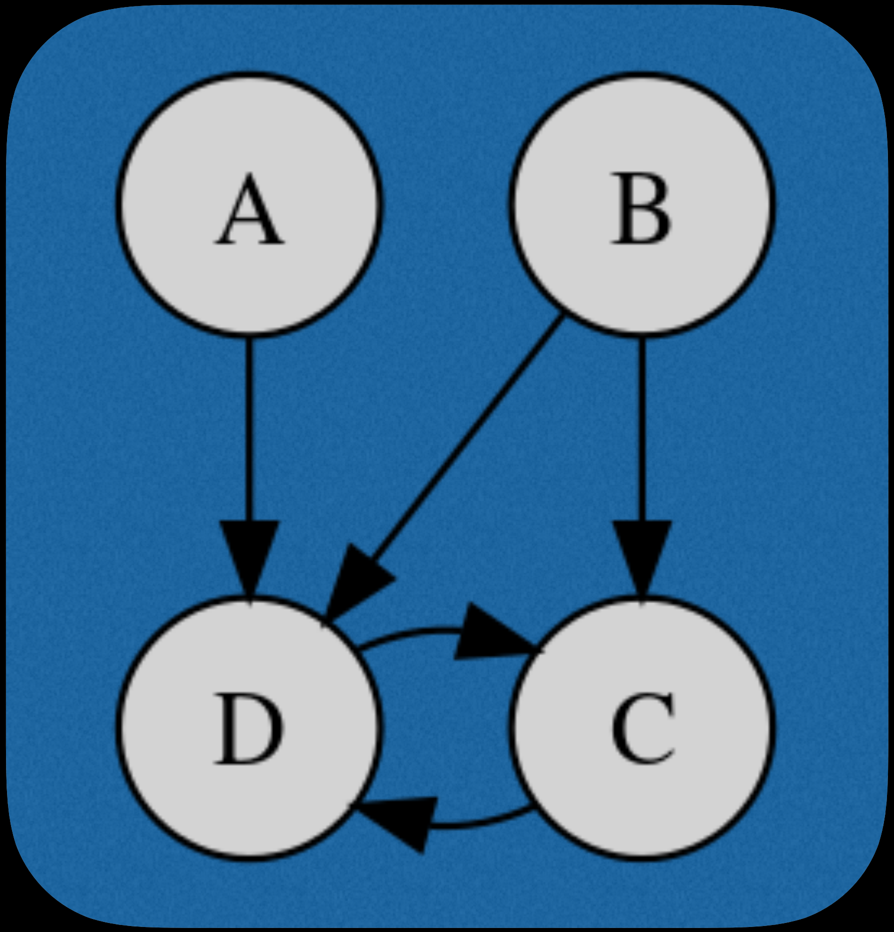
**C**upcakes with **R**ainbow **F**illing

# Background

Isn't it about time…

# Background

Isn't it about time… …for Graphical Models?


Represent distribution as a product of local functions that depend on a smaller subset of variables

# Background

## Generative vs Discriminative

- Generative models describe how a label vector y can probabilistically "generate" a feature vector x.

- Discriminative models describe how to take a feature vector x and assign it a label y.

- Either type of model can be converted to the other type using Bayes' rule

$$P(x|y)P(y) = P(x,y) = P(y|x)P(x)$$

# Background

## Joint Distributions

$$P(x, y)$$

Problems:

?

# Background

Joint Distributions

$$P(x, y)$$

Problems:

- Modeling inputs => intractable models
- Ignoring inputs => poor performance

# Background

Joint Distributions

$$P(x, y)$$

Naive Bayes

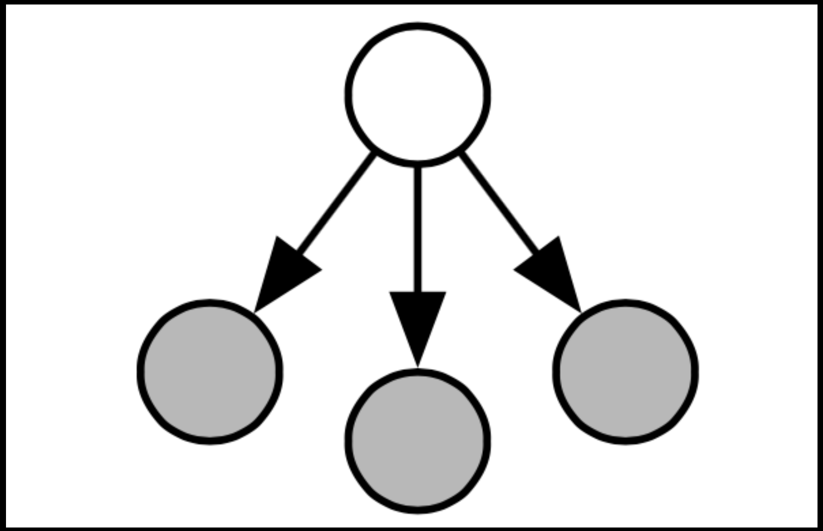
Generative model
Strong assumptions to simplify computation

$$p(y, \mathbf{x}) = p(y) \prod_{k=1}^{K} p(x_k | y)$$

# Background

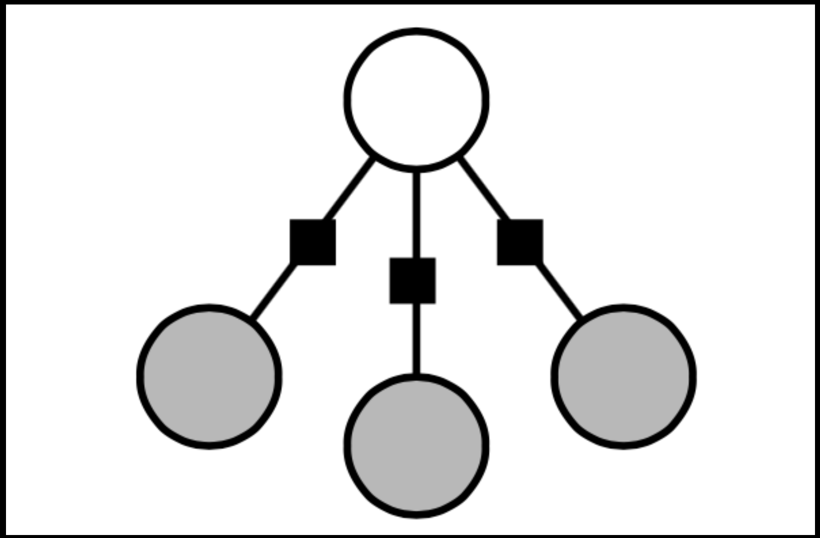
## Generative vs Discriminative

Naive Bayes



Conditional

Logistic Regression



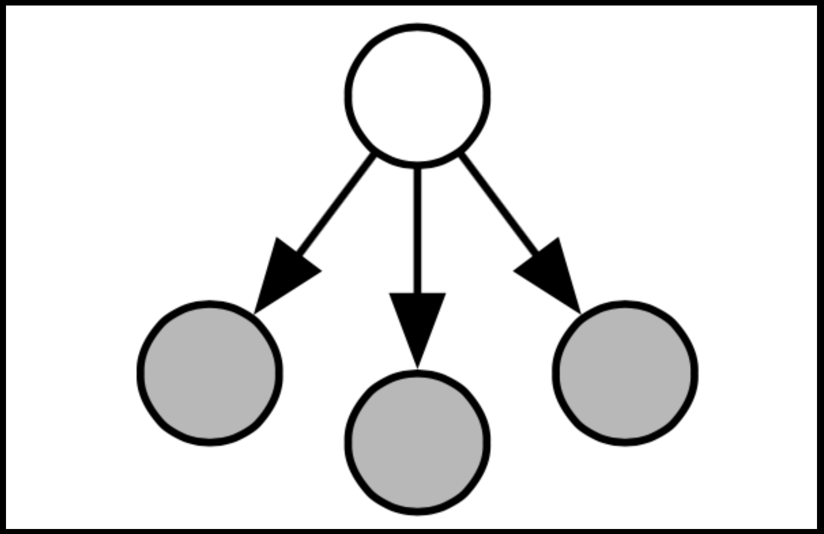
$$p(y, \mathbf{x}) = p(y) \prod_{k=1}^{K} p(x_k | y)$$

$$p(y|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp \left\{ \lambda_y + \sum_{j=1}^{K} \lambda_{y,j} x_j \right\}$$

# Background

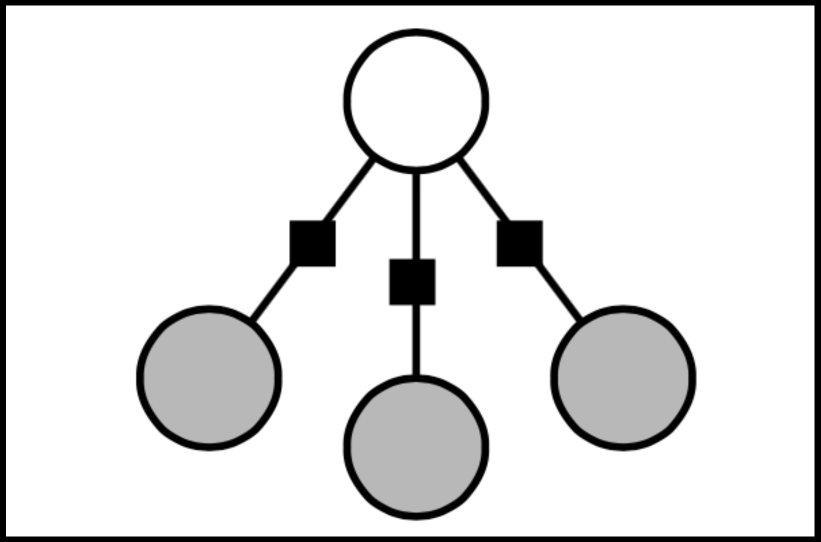
## Generative vs Discriminative

Naive Bayes



Conditional

Logistic Regression

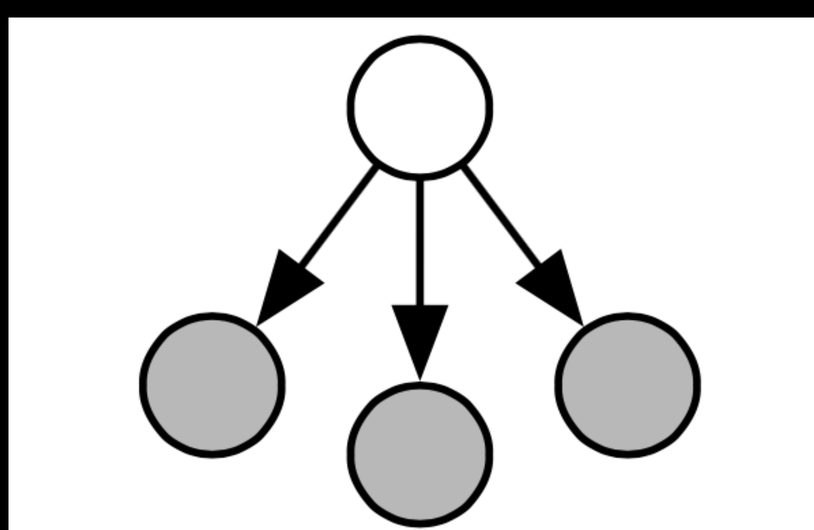


If trained to maximize conditional likelihood

# Background

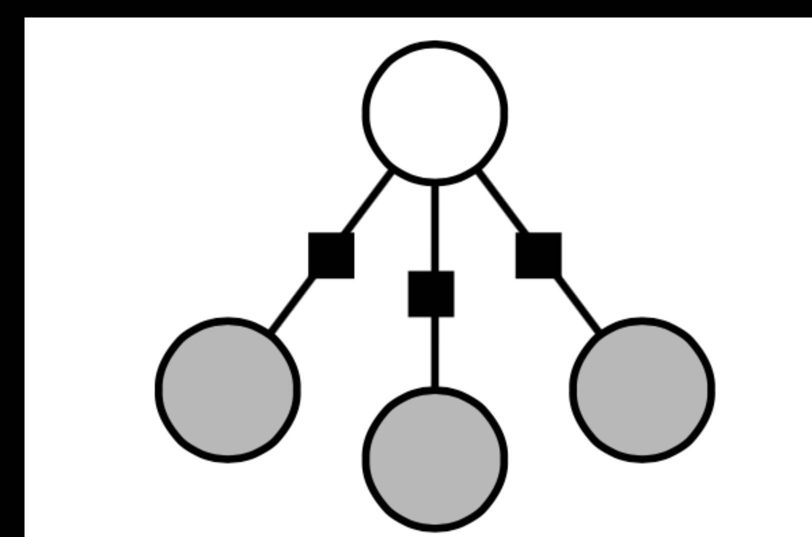
## Generative vs Discriminative

Naive Bayes



Conditional

Logistic Regression

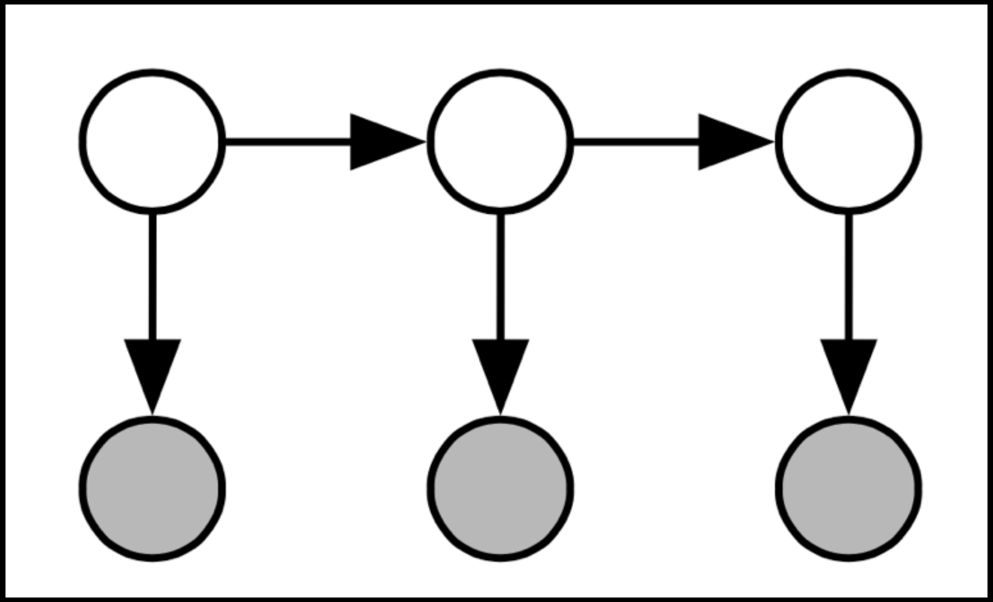


If trained to maximize joint likelihood

# Background

## Sequence Labeling

Hidden Markov Model (HMM)

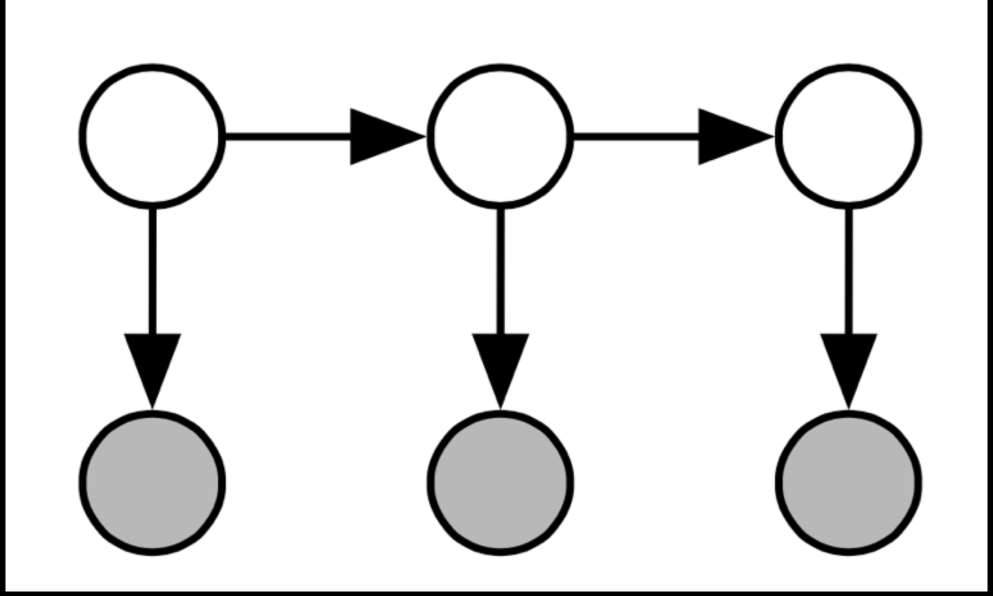


$$p(\mathbf{y}, \mathbf{x}) = \prod_{t=1}^{\mathrm{T}} p(y_t | y_{t-1}) p(x_t | y_t)$$

# Background

## Sequence Labeling

Hidden Markov Model (HMM)



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

ASSUMPTIONS:

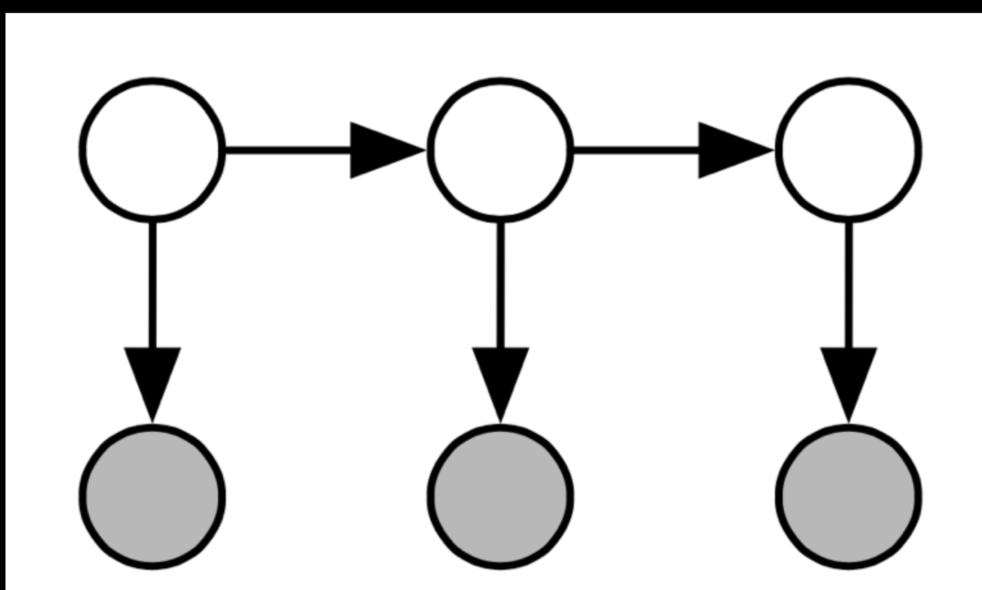
$$y_t \perp\!\!\!\perp y_1, \ldots, y_{t-2} | y_{t-1}$$

$$x_t \perp\!\!\!\perp Y \setminus \{y_t\}, X \setminus \{x_t\} \; | y_t$$

# Background

## Sequence Labeling

Hidden Markov Model (HMM)



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

PROBLEMS:

- Later labels cannot influence previous labels
- Cannot represent overlapping features

# Background

## Improvements to HMMs

### Maximum Entropy Markov Model

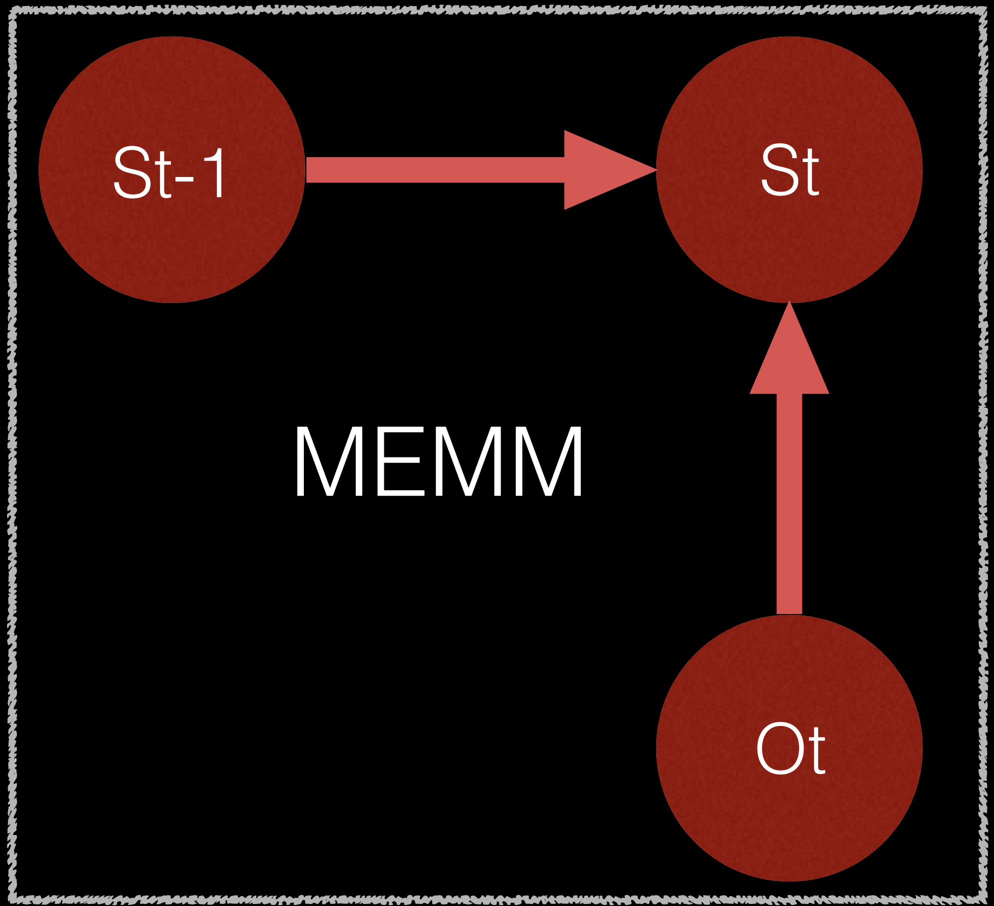
Consider the *Principle of Maximum Entropy* [Jaynes, 1957, Good, 1963], which states that the correct distribution $p(a, b)$ is that which maximizes entropy, or "uncertainty", subject to the constraints, which represent "evidence", i.e., the facts known to the experimenter. [Jaynes, 1957] discusses its advantages:

> ...in making inferences on the basis of partial information we must use that probability distribution which has maximum entropy subject to whatever is known. This is the only unbiased assignment we can make; to use any other would amount to arbitrary assumption of information which by hypothesis we do not have.

# Background

## Improvements to HMMs

### Maximum Entropy Markov Model




$$P(S_1, \ldots, S_n | O_1, \ldots, O_n) = \sum_{t=1}^{n} P(S_t | S_{t-1}, O_t)$$

# Background

## Improvements to HMMs

### Conditional Markov Model

PROS:

More flexible form of context dependence

Independence assumptions among labels, not observations

CONS:

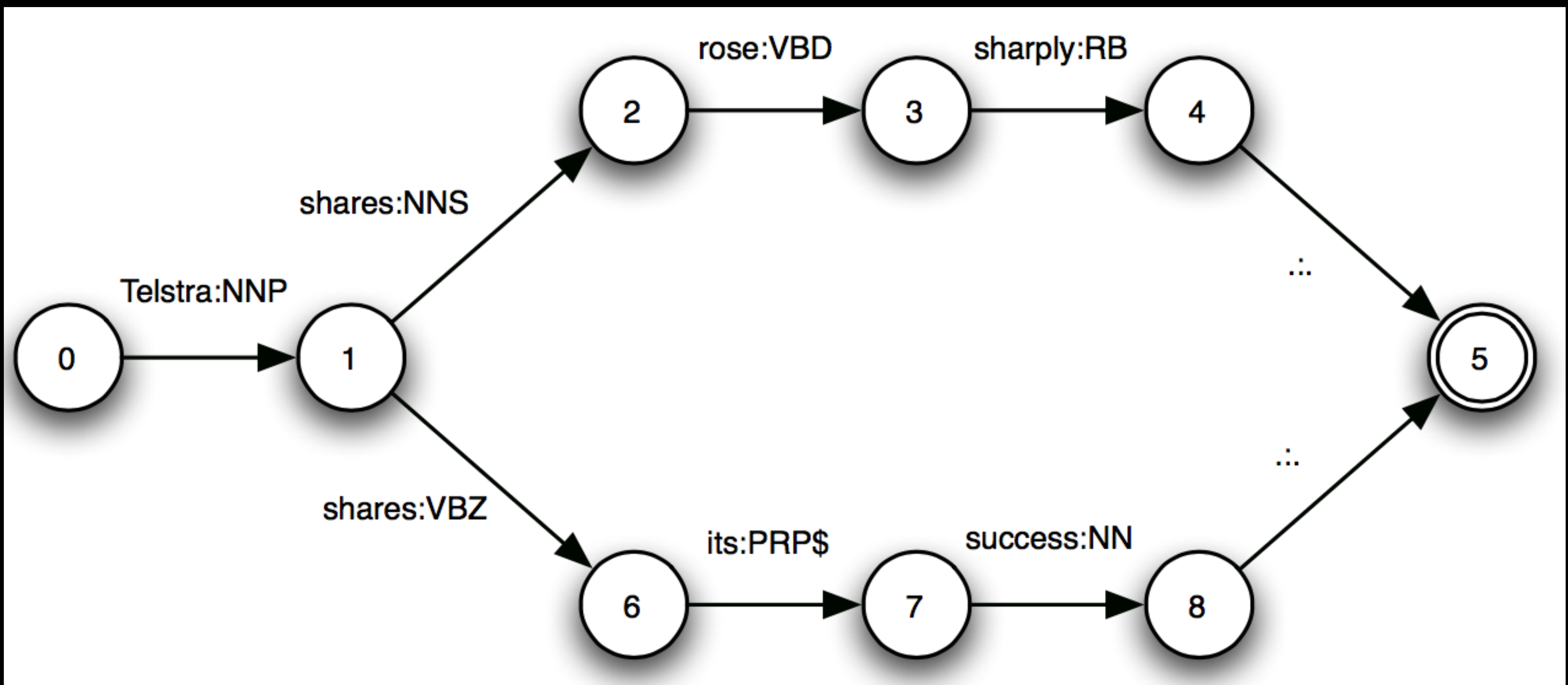
Only models log-linear distribution over component transition distributions

$$p(\mathbf{s}|\mathbf{o}) = \prod_{t=1}^{N} p(s_t|s_{t-1}, \mathbf{o})$$

# Background

## Improvements to HMMs

### Label Bias Problem



# Solution

## Conditional Random Fields

- Drop Local Normalization Constant
- Normalize GLOBALLY Over Full Graph

→ Avoids Label Bias Problem

# Solution

## Conditional Random Fields

Calculate path probability by normalizing the path score by the sum of the scores of all possible paths

Poorly matching path -> low score (probability)
Well-matching path -> high score

# Solution

## Conditional Random Fields

Similar to CMM

- Discriminative
- Model Conditional Distribution p(**y**|**x**)
- Allow arbitrary, overlapping features

# Solution

## Conditional Random Fields

Similar to CMM

- Discriminative
- Model Conditional Distribution p(**y**|**x**)
- Allow arbitrary, overlapping features

TAKEAWAY:
Retains advantages of CMM over HMM
Overcomes label bias problem of CMM and MEMM

# Linear-Chain CRFs

## Conditional Random Fields

Remember the HMM

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

$$p(\mathbf{y}, \mathbf{x}) = \frac{1}{Z} \exp\left\{ \sum_t \sum_{i,j \in S} \lambda_{ij} \mathbf{1}_{\{y_t = i\}} \mathbf{1}_{\{y_{t-1} = j\}} + \sum_t \sum_{i \in S} \sum_{o \in O} \mu_{oi} \mathbf{1}_{\{y_t = i\}} \mathbf{1}_{\{x_t = o\}} \right\}, \tag{1.13}$$

# Linear-Chain CRFs

## Conditional Random Fields

Remember the HMM

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

$$p(\mathbf{y}, \mathbf{x}) = \frac{1}{Z} \exp \left\{ \sum_t \sum_{i,j \in S} \lambda_{ij} \mathbf{1}_{\{y_t = i\}} \mathbf{1}_{\{y_{t-1} = j\}} + \sum_t \sum_{i \in S} \sum_{o \in O} \mu_{oi} \mathbf{1}_{\{y_t = i\}} \mathbf{1}_{\{x_t = o\}} \right\}, \tag{1.13}$$

$$p(\mathbf{y}, \mathbf{x}) = \frac{1}{Z} \exp \left\{ \sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t) \right\}$$

# Linear-Chain CRFs

## Conditional Random Fields

$$p(\mathbf{y}, \mathbf{x}) = \frac{1}{Z} \exp\left\{ \sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t) \right\}$$

$$p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{y}, \mathbf{x})}{\sum_{\mathbf{y}'} p(\mathbf{y}', \mathbf{x})} = \frac{\exp\left\{ \sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t) \right\}}{\sum_{\mathbf{y}'} \exp\left\{ \sum_{k=1}^{K} \lambda_k f_k(y'_t, y'_{t-1}, x_t) \right\}}$$

By expanding Z, we have the conditional distribution p(**y**|**x**)

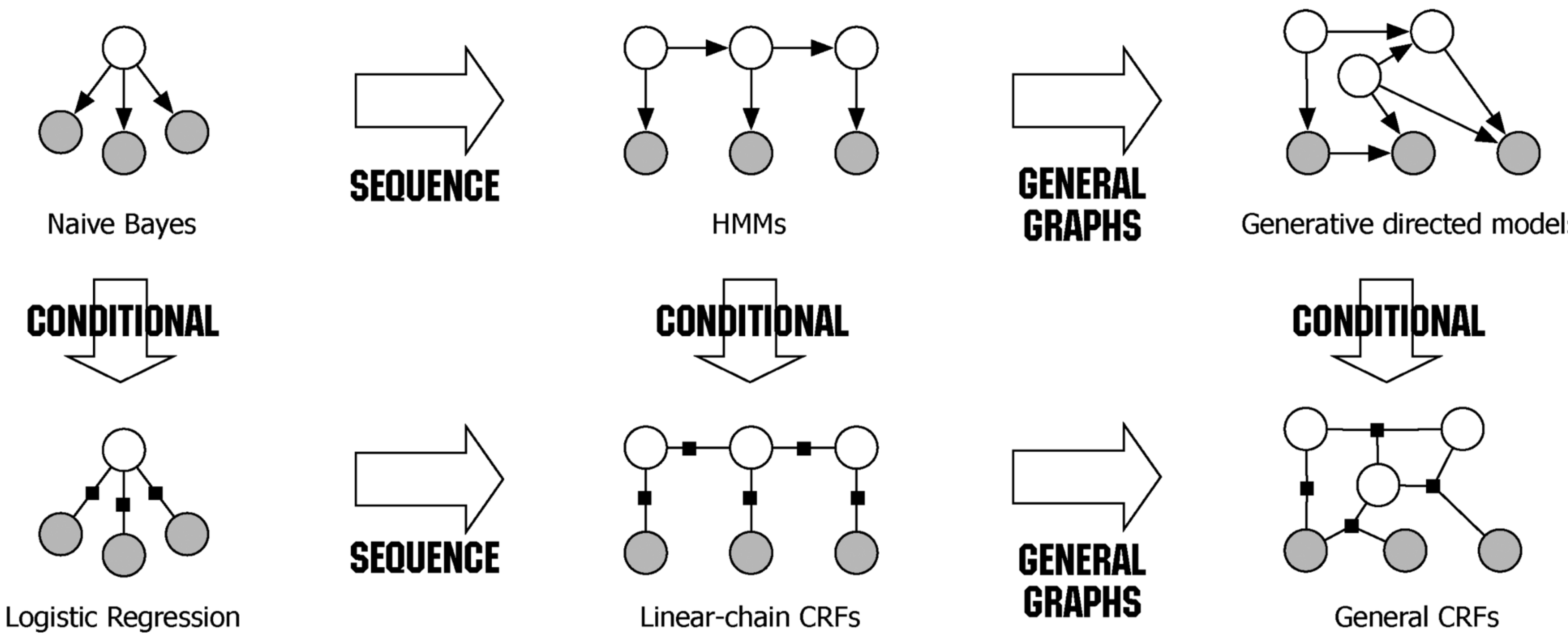
# Feature Functions

students.cs.byu.edu/~mbrodie/778

# 5:00

# Linear-Chain CRFs

## Conditional Random Fields



# Feature Engineering

- Larger feature set benefits:
  - Greater prediction accuracy
  - More flexible decision boundary
- However, may lead to overfitting

# Feature Engineering

<u>Label-Observation Features</u>

$$f_{pk}(\mathbf{y}_c, \mathbf{x}_c) = \mathbf{1}_{\{\mathbf{y}_c = \tilde{\mathbf{y}}_c\}} q_{pk}(\mathbf{x}_c)$$

Where $q_{pk}$ is an observation function rather than a specific word value

For example:
"word $x_t$ is capitalized"
"word $x_t$ ends in *ing*"

# Feature Engineering

Unsupported Features

For example:
word $x_t$ is 'with' and label $y_t$ is 'NOUN'

Greatly increases the number of parameters.

# Feature Engineering

Unsupported Features

For example:
word $x_t$ is 'with' and label $y_t$ is 'NOUN'

Greatly increases the number of parameters.
However, usually gives a slight boost in accuracy

Can be useful ->
Assign negative weights to prevent spurious labels from receiving high probability

# Feature Engineering

Boundary Labels

Boundary labels (start/end of sentence, edge of image) can have different characteristics from other labels.

For example:

Capitalization in the middle of the sentence generally indicates a proper noun (but not necessarily at the beginning of a sentence).

# Feature Engineering

## Other Methods

- Feature Induction
  - Begin with a number of base features.
  - Gradually add conjunctions to those features
- Normalize Features
  - Convert categorical to binary features
- Features from Different Time Steps

# Feature Engineering

Other Methods

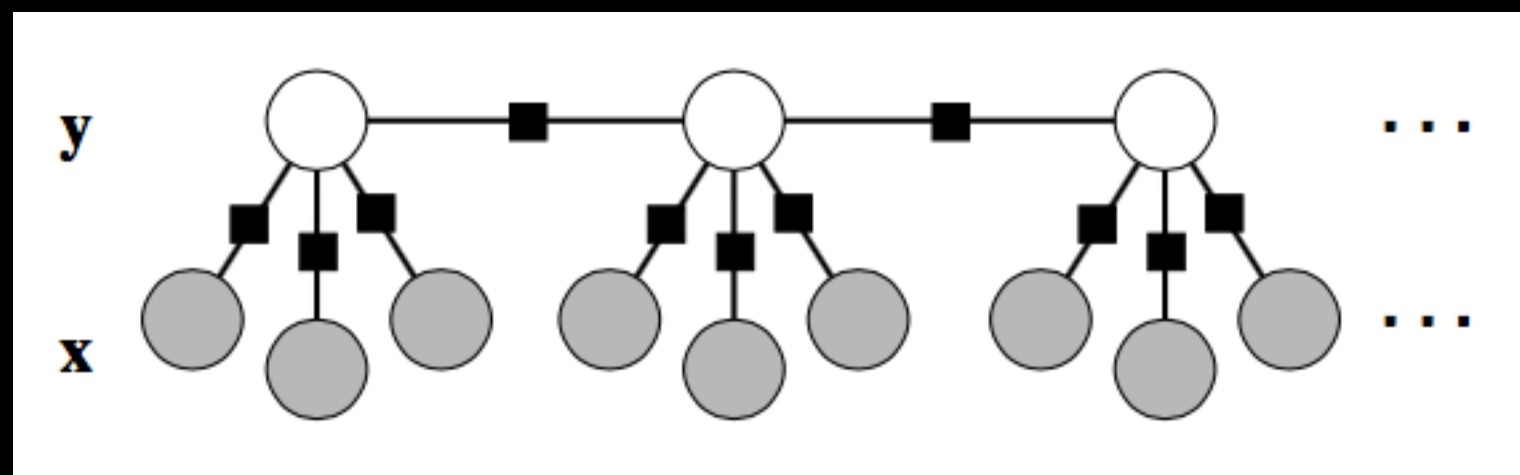
- Features as Model Combination

$$f_t(y, \mathbf{x}) = p_{\text{HMM}}(y_t = y | \mathbf{x})$$

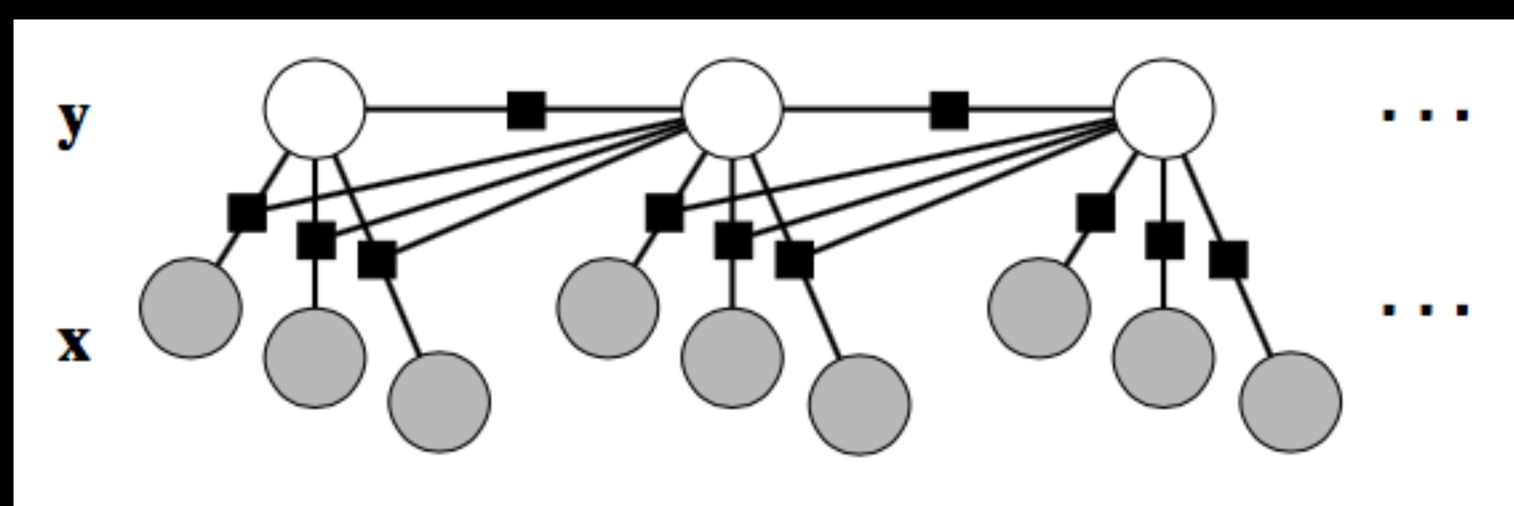
- Input Dependent Structure
  - e.g. Skip-Chain CRFs
    - Encourage identical words in a sentence to have the same label
    - Do this by adding an edge in the graph

# Linear-Chain CRFs

## Conditional Random Fields




HMM-like CRF




Previous observation affects transition

# Feature Effects

Pseudo Weighted Functions on the Brown Corpus:
38 Training, 12 Test Sentences

| Features | Accuracy | # Feature Functions |
|---|---|---|
| yt-1 and yt | 0.0449 | 296 |
| xt and yt | 0.6282 | 486 |
| xt and yt AND<br>xt and yt-1and yt | **0.6603** | 1176 |
| xt and yt AND<br>yt-1 and yt | 0.0929 | 782 |

# Linear-Chain CRFs

## Parameter Estimation

We want to find parameters for feature functions:

$$\theta = \{\lambda_k\}$$

# Linear-Chain CRFs

## Parameter Estimation

We want to find parameters for feature functions:

$$\theta = \{\lambda_k\}$$

Maximize the conditional log likelihood:

$$\ell(\theta) = \sum_{i=1}^{N} \log p(\mathbf{y}^{(i)} | \mathbf{x}^{(i)})$$

# Linear-Chain CRFs

## Parameter Estimation

$$p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{y},\mathbf{x})}{\sum_{\mathbf{y}'} p(\mathbf{y}',\mathbf{x})} = \frac{\exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t)\right\}}{\sum_{\mathbf{y}'} \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y'_t, y'_{t-1}, x_t)\right\}}$$

# Linear-Chain CRFs

## Parameter Estimation

$$p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{y},\mathbf{x})}{\sum_{\mathbf{y}'} p(\mathbf{y}',\mathbf{x})} = \frac{\exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t)\right\}}{\sum_{\mathbf{y}'} \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y'_t, y'_{t-1}, x_t)\right\}}$$

Which becomes…

$$\ell(\theta) = \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)})$$

# Linear-Chain CRFs

## Parameter Estimation

$$p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{y},\mathbf{x})}{\sum_{\mathbf{y}'} p(\mathbf{y}',\mathbf{x})} = \frac{\exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t)\right\}}{\sum_{\mathbf{y}'} \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y'_t, y'_{t-1}, x_t)\right\}}$$

Which becomes…

$$\ell(\theta) = \sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)})$$

Penalize large weight vectors

$$\ell(\theta) = \sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

# Linear-Chain CRFs

## Parameter Estimation

Cannot maximize $\ell(\theta)$ in closed form

$$\ell(\theta) = \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

Instead use numerical optimization
=> Find derivatives w.r.t. $\lambda_k$

$$\frac{\partial \ell}{\partial \lambda_k} =$$

# Linear-Chain CRFs

## Parameter Estimation

Cannot maximize $\ell(\theta)$ in closed form

$$\ell(\theta) = \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

Instead use numerical optimization

=> Find derivatives w.r.t. $\lambda_k$

$$\frac{\partial \ell}{\partial \lambda_k} = \sum_{i=1}^{N} \sum_{t=1}^{T} f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)})$$

# Linear-Chain CRFs

## Parameter Estimation

Cannot maximize $\ell(\theta)$ in closed form

$$\ell(\theta) = \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

Instead use numerical optimization

=> Find derivatives w.r.t. $\lambda_k$

$$\frac{\partial \ell}{\partial \lambda_k} = \sum_{i=1}^{N} \sum_{t=1}^{T} f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{y,y'} f_k(y, y', \mathbf{x}_t^{(i)}) p(y, y' | \mathbf{x}^{(i)})$$

# Linear-Chain CRFs

## Parameter Estimation

Cannot maximize $\ell(\theta)$ in closed form

$$\ell(\theta) = \sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

Instead use numerical optimization

=> Find derivatives w.r.t. $\lambda_k$

$$\frac{\partial \ell}{\partial \lambda_k} = \sum_{i=1}^{N}\sum_{t=1}^{T} f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{y,y'} f_k(y, y', \mathbf{x}_t^{(i)}) p(y, y' | \mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k}{\sigma^2}$$

# Linear-Chain CRFs

## Parameter Estimation

How to optimize $\ell(\theta)$ ?

$$\ell(\theta) = \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

Steepest ascent along Gradient

Newton's Method

(L-) BFGS

# Linear-Chain CRFs

## Parameter Estimation

How to optimize $\ell(\theta)$ ?

$$\ell(\theta) = \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

Steepest ascent along Gradient

Newton's Method

(L-) BFGS

# Linear-Chain CRFs

## Parameter Estimation

How to optimize $\ell(\theta)$ ?

$$\ell(\theta) = \sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

Steepest ascent along Gradient

Newton's Method

(L-) BFGS

# Linear-Chain CRFs

## Parameter Estimation

How to optimize $\ell(\theta)$ ?

$$\ell(\theta) = \sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

Steepest ascent along Gradient

Newton's Method

(L-) BFGS

# Feature Effects

## Pseudo Weighted Functions on the Brown Corpus: 38 Training, 12 Test Sentences

| Features | Accuracy | # Feature Functions | Trained Accuracy 10 Epochs |
|---|---|---|---|
| yt-1 and yt | 0.0449 | 296 | **0.1185** |
| xt and yt | 0.6282 | 486 | **0.6538** |
| xt and yt AND xt and yt-1and yt | 0.6603 | 1176 | **0.6731** |
| xt and yt AND yt-1 and yt | 0.0929 | 782 | **0.2724** |

# Linear-Chain CRFs

## Inference

**During Training**

Marginal Distributions
for each edge

$$p(y_t, y_{t-1}|\mathbf{x})$$

Compute Z(**x**)

**During Testing**

Viterbi Algorithm to compute
the most likely labeling

$$\mathbf{y}^* = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})$$

# Linear-Chain CRFs

## Inference

Use Forward-Backward Approach in HMM:

$$p(\mathbf{y}, \mathbf{x}) = \prod_t \Psi_t(y_t, y_{t-1}, x_t) \text{ where } Z = 1$$

# Linear-Chain CRFs

## Inference

Use Forward-Backward Approach in HMM:

$$p(\mathbf{y}, \mathbf{x}) = \prod_t \Psi_t(y_t, y_{t-1}, x_t) \text{ where } Z = 1$$

Factor Definition:

$$\Psi_t(j, i, x) \stackrel{\text{def}}{=} p(y_t = j | y_{t-1} = i) p(x_t = x | y_t = j)$$

# Linear-Chain CRFs

## Inference

Use Forward-Backward Approach in HMM:

$$p(\mathbf{y}, \mathbf{x}) = \prod_t \Psi_t(y_t, y_{t-1}, x_t) \text{ where } Z = 1$$

Factor Definition:

$$\Psi_t(j, i, x) \stackrel{\text{def}}{=} p(y_t = j | y_{t-1} = i) p(x_t = x | y_t = j)$$

Rewrite Using Distributive Law:

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{y}} \prod_{t=1}^{T} \Psi_t(y_t, y_{t-1}, x_t) \\ &= \sum_{y_\mathrm{T}} \sum_{y_{\mathrm{T}-1}} \Psi_\mathrm{T}(y_\mathrm{T}, y_{\mathrm{T}-1}, x_\mathrm{T}) \sum_{y_{\mathrm{T}-2}} \Psi_{\mathrm{T}-1}(y_{\mathrm{T}-1}, y_{\mathrm{T}-2}, x_{\mathrm{T}-1}) \sum_{y_{\mathrm{T}-3}} \cdots \end{aligned}$$

# Linear-Chain CRFs

## Inference

Use Forward-Backward Approach in HMM:

This leads to a set of **forward variables**

$$\alpha_t(j) \stackrel{\text{def}}{=} p(\mathbf{x}_{\langle 1 \ldots t \rangle}, y_t = j)$$

$$= \sum_{\mathbf{y}_{\langle 1 \ldots t-1 \rangle}} \Psi_t(j, y_{t-1}, x_t) \prod_{t'=1}^{t-1} \Psi_{t'}(y_{t'}, y_{t'-1}, x_{t'}),$$

# Linear-Chain CRFs

## Inference

Use Forward-Backward Approach in HMM:

This leads to a set of **forward variables**

$$\alpha_t(j) \stackrel{\text{def}}{=} p(\mathbf{x}_{\langle 1 \ldots t \rangle}, y_t = j)$$

$$= \sum_{\mathbf{y}_{\langle 1 \ldots t-1 \rangle}} \Psi_t(j, y_{t-1}, x_t) \prod_{t'=1}^{t-1} \Psi_{t'}(y_{t'}, y_{t'-1}, x_{t'}),$$

Compute $\alpha_t$ by recursion

$$\alpha_t(j) = \sum_{i \in S} \Psi_t(j, i, x_t) \alpha_{t-1}(i),$$

# Linear-Chain CRFs

## Inference

Use Forward-Backward Approach in HMM:

Similarly, we computer a set of **backward variables**

$$\beta_t(i) \stackrel{\text{def}}{=} p(\mathbf{x}_{\langle t+1 \ldots \mathrm{T} \rangle} | y_t = i)$$
$$= \sum_{\mathbf{y}_{\langle t+1 \ldots \mathrm{T} \rangle}} \prod_{t'=t+1}^{\mathrm{T}} \Psi_{t'}(y_{t'}, y_{t'-1}, x_{t'}),$$

Compute $\beta_t$ by recursion

$$\beta_t(i) = \sum_{j \in S} \Psi_{t+1}(j, i, x_{t+1}) \beta_{t+1}(j),$$

# Linear-Chain CRFs

## Inference

Compute Marginals Needed for Gradient
Using Forward and Backward Results

$$p(y_{t-1}, y_t|\mathbf{x}) = \Psi_t(y_t, y_{t-1}, x_t) \left( \sum_{\mathbf{y}_{\langle 1 \dots t-2 \rangle}} \prod_{t'=1}^{t-1} \Psi_{t'}(y_{t'}, y_{t'-1}, x_{t'}) \right) \left( \sum_{\mathbf{y}_{\langle t+1 \dots \mathrm{T} \rangle}} \prod_{t'=t+1}^{T} \Psi_{t'}(y_{t'}, y_{t'-1}, x_{t'}) \right)$$

# Linear-Chain CRFs

## Inference

Compute Marginals Needed for Gradient
Using Forward and Backward Results

=

$$p(y_{t-1}, y_t|\mathbf{x}) \propto \alpha_{t-1}(y_{t-1})\Psi_t(y_t, y_{t-1}, x_t)\beta_t(y_t)$$

# Linear-Chain CRFs

## Inference

How does this compare to CRF training/inference?

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\right\}$$

# Linear-Chain CRFs

## Inference

How does this compare to CRF training/inference?

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\right\}$$

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \Psi_t(y_t, y_{t-1}, \mathbf{x}_t)$$

# Linear-Chain CRFs

## Inference

How does this compare to CRF training/inference?

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left\{ \sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t) \right\}$$

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \Psi_t(y_t, y_{t-1}, \mathbf{x}_t)$$

$$\Psi_t(y_t, y_{t-1}, \mathbf{x}_t) = \exp\left\{ \sum_{k} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t) \right\}$$

# Breakthrough

# My Implementation

- 3 Different Implementations

1. Tensorflow

# Tensorflow Tips

Do NOT start with CRFs

$$
\begin{aligned}
\mathcal{L}(\mathcal{T}) &= \sum_{(\vec{x},\vec{y})\in\mathcal{T}} \left[ \log \left( \frac{\exp\left(\sum_{j=1}^{n}\sum_{i=1}^{m} \lambda_i f_i\left(y_{j-1}, y_j, \vec{x}, j\right)\right)}{\sum_{\vec{y}\,'\in\mathcal{Y}} \exp\left(\sum_{j=1}^{n}\sum_{i=1}^{m} \lambda_i f_i\left(y'_{j-1}, y'_j, \vec{x}, j\right)\right)} \right) \right] - \sum_{i=1}^{m} \frac{\lambda_i^2}{2\sigma^2} \\
&= \sum_{(\vec{x},\vec{y})\in\mathcal{T}} \left[ \left( \sum_{j=1}^{n}\sum_{i=1}^{m} \lambda_i f_i\left(y_{j-1}, y_j, \vec{x}, j\right) \right) - \right. \\
&\qquad \left. - \log \left( \sum_{\vec{y}\,'\in\mathcal{Y}} \exp\left( \sum_{j=1}^{n}\sum_{i=1}^{m} \lambda_i f_i\left(y'_{j-1}, y'_j, \vec{x}, j\right) \right) \right) \right] - \sum_{i=1}^{m} \frac{\lambda_i^2}{2\sigma^2} \\
&= \underbrace{\sum_{(\vec{x},\vec{y})\in\mathcal{T}} \sum_{j=1}^{n}\sum_{i=1}^{m} \lambda_i f_i\left(y_{j-1}, y_j, \vec{x}, j\right)}_{\mathcal{A}} - \\
&\qquad \underbrace{- \sum_{(\vec{x},\vec{y})\in\mathcal{T}} \log \underbrace{\left( \sum_{\vec{y}\,'\in\mathcal{Y}} \exp\left( \sum_{j=1}^{n}\sum_{i=1}^{m} \lambda_i f_i\left(y'_{j-1}, y'_j, \vec{x}, j\right) \right) \right)}_{Z_{\vec{\lambda}}(\vec{x})}}_{\mathcal{B}} \underbrace{- \sum_{i=1}^{m} \frac{\lambda_i^2}{2\sigma^2}}_{\mathcal{C}} . \qquad (50)
\end{aligned}
$$

# Tensorflow Tips

Be prepared for awkward, sometimes unintuitive formats

```
import tensorflow as tf

filename_queue = tf.train.string_input_producer(["iris.csv"])
reader = tf.TextLineReader(skip_header_lines=1)
key, value = reader.read(filename_queue)

record_defaults = [[0.0], [0.0], [0.0], [0.0], [""]]
sepal_length, sepal_width, petal_length, petal_width, iris_species = \
    tf.decode_csv(value, record_defaults=record_defaults)
features = tf.pack([
    sepal_length,
    sepal_width,
    petal_length,
    petal_width])

with tf.Session() as sess:
    tf.initialize_all_variables().run()

    coord = tf.train.Coordinator()
    threads = tf.train.start_queue_runners(coord=coord)
    for row in range(0,150):
            example, label = sess.run([features, iris_species])
            print(example, label)
```

# Tensorflow

- Gaaaaah!!
- len(X) => X.get_shape().dims[0].value
- Start out with Theano
  - More widely used
  - More intuitive

# My Implementation

- 3 Different Implementations

1. Tensorflow
2. Scipy Optimization (fmin_l_bfgs_b)

# My Implementation

- 3 Different Implementations
  1. Tensorflow
  2. Scipy Optimization (fmin_l_bfgs_b)
  3. Gradient Descent
     A. Trainable Version
     B. Simplified, Web-based Version

# Experiments

## Data Sets

**Brown Corpus**
News Articles

- 5,000 POS Sentences
- 27,596 Tags
- 75/25 Train-Test Split

**Penn Treebank Corpus**
WSJ

- 3,914 sentence SAMPLE
- $1,700 (or free BYU version)
- 75/25 Train-Test Split

# Experiments

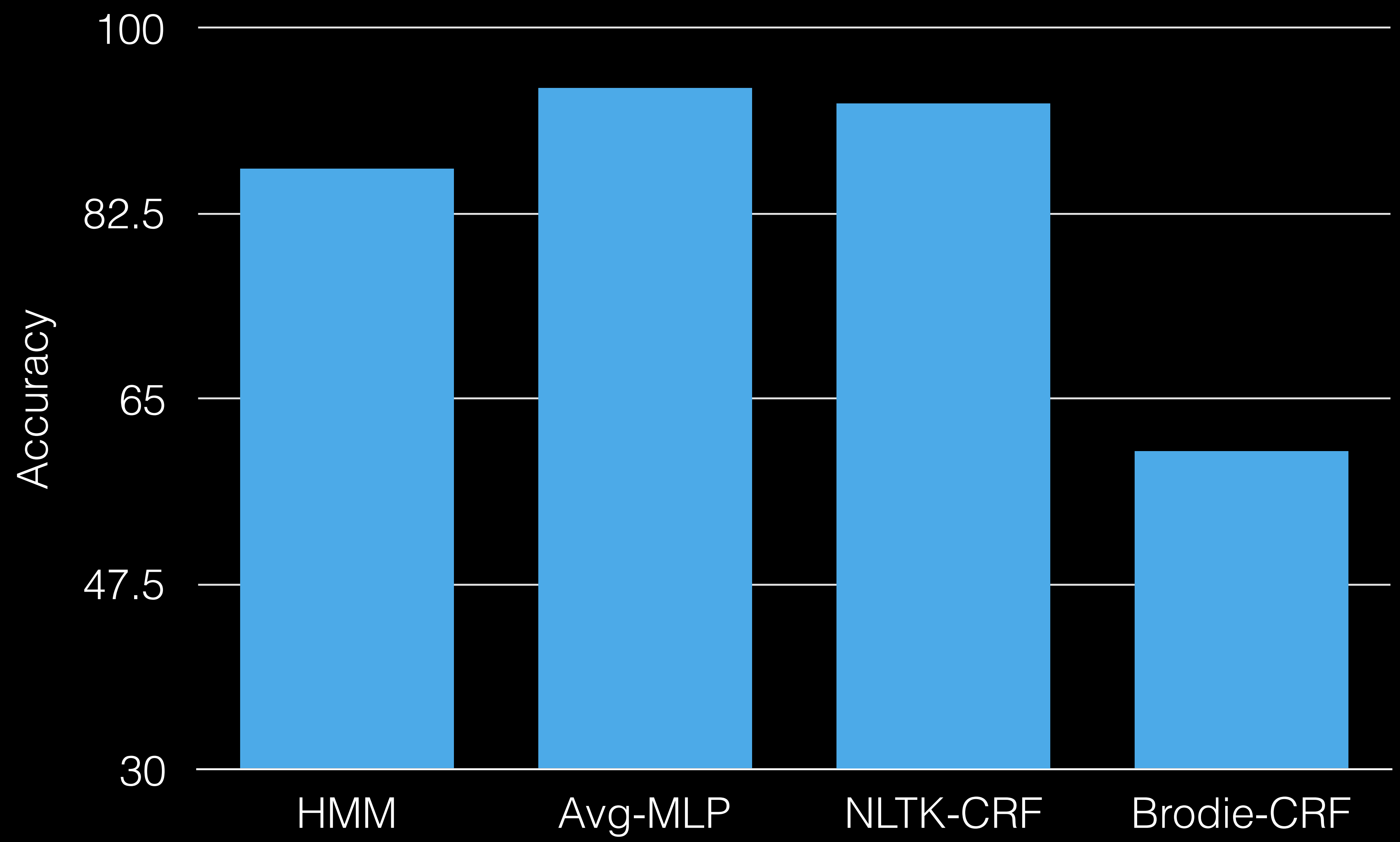
## Comparison Models

Python Natural Language Toolkit (NLTK)

- Hidden Markov Model
- Conditional Random Field
- Averaged-Multilayer Perceptron
    - Average the weight updates -> prevent radical changes due to different training batches

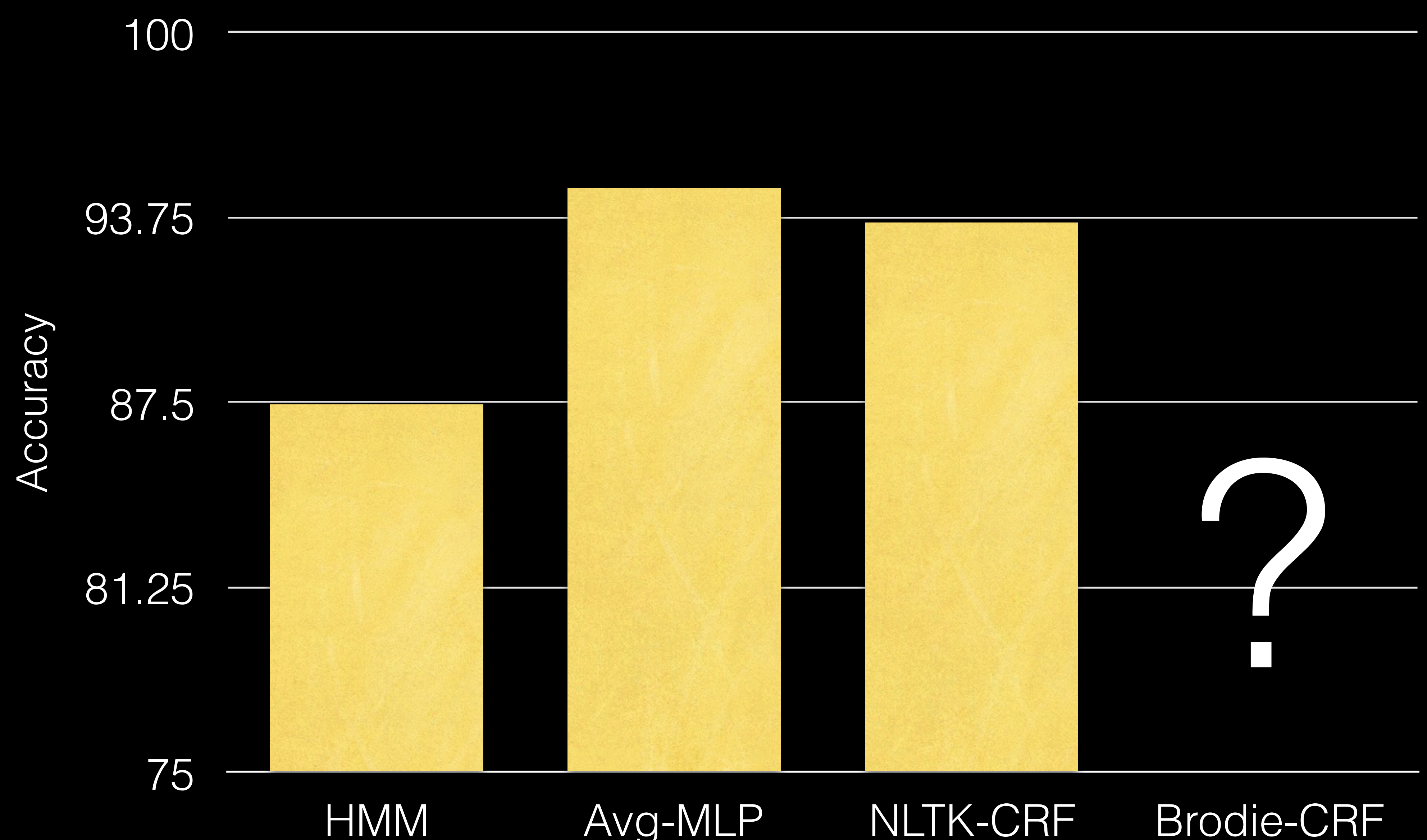
# Experiments

## Brown Corpus Results



# Experiments

## Penn Corpus Results



# Pros/Cons of Implementation

**Pros**

- Can use forward/backward and Viterbi Algorithm
- Learn weights for features

**Cons**

- Slow
- Limited to gradient descent training

# Extensions

- General CRFs
- Skip-chain CRFs
- Hidden Node CRFs

# CRF-RNN + CNN

- Problem: Traditional CNNs produce coarse outputs for pixel-level labels.

- Solution: Apply CRF as a post-processing refinement

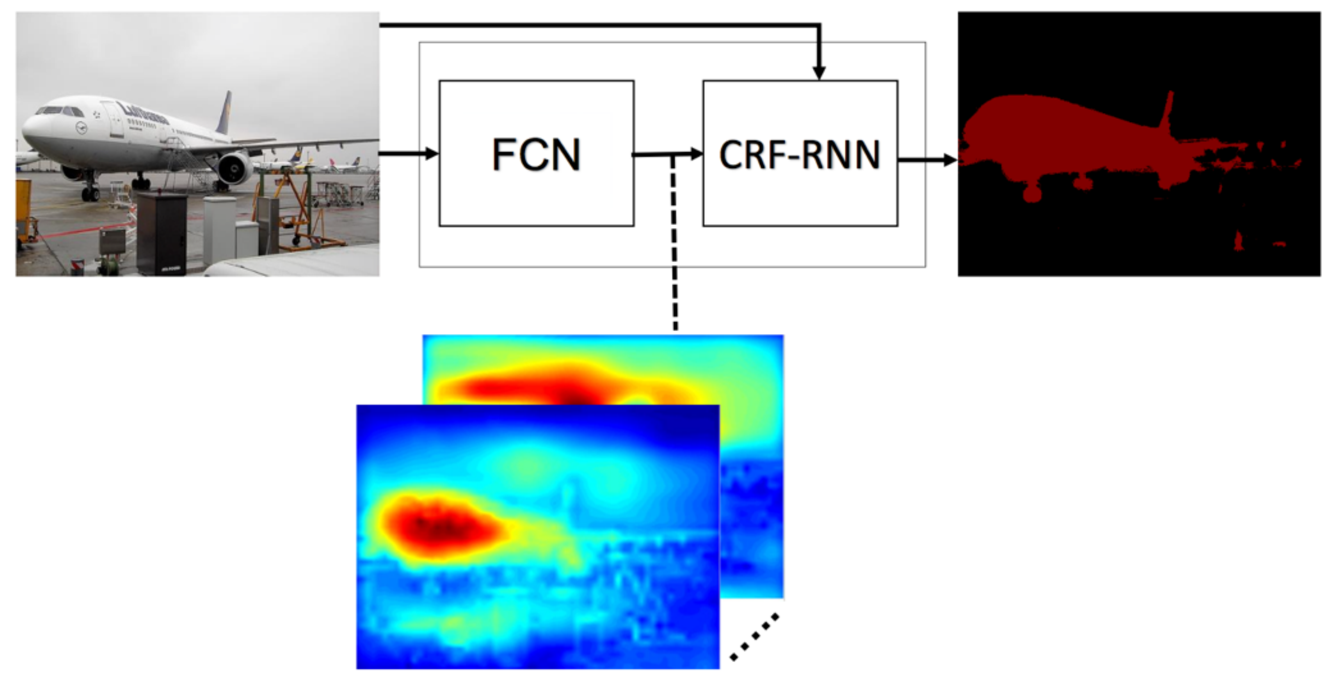



Figure 3. **The End-to-end Trainable Network.** Schematic visualization of our full network which consists of a CNN and the CNN-CRF network. Best viewed in colour.

# BI-LSTM-CRF

Table 2: Comparison of tagging performance on POS, chunking and NER tasks for various models.

| | | POS | CoNLL2000 | CoNLL2003 |
|---|---|---|---|---|
| Random | Conv-CRF (Collobert et al., 2011) | 96.37 | 90.33 | 81.47 |
| | LSTM | 97.10 | 92.88 | 79.82 |
| | BI-LSTM | 97.30 | 93.64 | 81.11 |
| | CRF | 97.30 | 93.69 | 83.02 |
| | LSTM-CRF | **97.45** | 93.80 | 84.10 |
| | BI-LSTM-CRF | 97.43 | **94.13** | **84.26** |
| Senna | Conv-CRF (Collobert et al., 2011) | 97.29 | 94.32 | 88.67 (89.59) |
| | LSTM | 97.29 | 92.99 | 83.74 |
| | BI-LSTM | 97.40 | 93.92 | 85.17 |
| | CRF | 97.45 | 93.83 | 86.13 |
| | LSTM-CRF | 97.54 | 94.27 | 88.36 |
| | BI-LSTM-CRF | **97.55** | **94.46** | **88.83 (90.10)** |

**Bidirectional LSTM-CRF Models for Sequence Tagging**

# Future Directions

- New model combinations of CRFs
- Better training approximations (i.e. besides MCMC sampling methods)
- Additional ‘hidden’ architectures

# Bibliography


Trevor A. Cohn. *Scaling Conditional Random Fields for Natural Language Processing*. 2007.

Roman Klinger and Katrin Tomanek. *Classical Probabilistic Models and Conditional Random Fields*. 2007.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data". In: *Proceedings of the Eighteenth International Conference on Machine Learning*. ICML '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 282–289. ISBN: 1-55860-778-1. URL: `http://dl.acm.org/citation.cfm?id=645530.655813`.

Laurens Maaten, Max Welling, and Lawrence K. Saul. *Hidden-Unit Conditional Random Fields*.

Andrew McCallum. "Efficiently Inducing Features of Conditional Random Fields". In: *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*. UAI'03. Acapulco, Mexico: Morgan Kaufmann Publishers Inc., 2003, pp. 403–410. ISBN: 0-127-05664-5. URL: `http://dl.acm.org/citation.cfm?id=2100584.2100633`.

Charles A. Sutton. "Efficient Training Methods for Conditional Random Fields". AAI3315485. PhD thesis. 2008. ISBN: 978-0-549-66363-8.

# Bibliography


A. Pass, J. Zhang, and D. Stewart. "Hidden Conditional Random Fields for Visual Speech Recognition". In: *Machine Vision and Image Processing Conference, 2009. IMVIP '09. 13th International.* Sept. 2009, pp. 117–122. DOI: 10.1109/IMVIP.2009.28.

Fei Sha and Fernando Pereira. "Shallow Parsing with Conditional Random Fields". In: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*. NAACL '03. Edmonton, Canada: Association for Computational Linguistics, 2003, pp. 134–141. DOI: 10.3115/1073445.1073473. URL: http://dx.doi.org/10.3115/1073445.1073473.

Jeremy J. Morris. "A Study on the Use of Conditional Random Fields for Automatic Speech Recognition". AAI3417603. PhD thesis. Columbus, OH, USA, 2010. ISBN: 978-1-124-15313-1.

Charles Sutton and Andrew McCallum. "An Introduction to Conditional Random Fields". In: *Found. Trends Mach. Learn.* 4.4 (Apr. 2012), pp. 267–373. ISSN: 1935-8237. DOI: 10.1561/2200000013. URL: http://dx.doi.org/10.1561/2200000013.

Xiaojin Zhu. *CS838-1 Advanced NLP: Conditional Random Fields*. 2007.

Hanna M. Wallach. *Conditional random fields: An introduction*. Tech. rep. 2004.

# Bibliography

## Other Resources


- Tom Mitchell's *Machine Learning.* Online draft version of 'Generative and Discriminative Classifiers: Naive Bayes and Logistic Regression.' http://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf. Accessed January 25, 2016.

- Edwin Chen Blog. *Introduction to Conditional Random Fields*. http://blog.echen.me/2012/01/03/introduction-to-conditional-random-fields/. Accessed February 2, 2016.

- LingPipe Tutorial: http://alias-i.com/lingpipe/demos/tutorial/crf/read-me.html. Accesed February 2, 2016.

# Thank You

Questions?