Univ. of Pittsburgh

Conditional Random Fields

Zhipeng Luo

zpluo@cs.pitt.edu

Oct. 2014

Computer Science Department

Univ. of Pittsburgh

Subjects of CRF

Part 0

Outline

Univ. of Pittsburgh

Introduction

Part 1

Introduction

Univ. of Pittsburgh

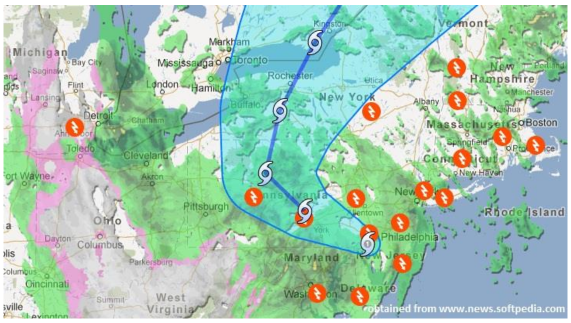
Problem Description

Part 1

Introduction

- Given ***X*** (observations), find ***Y*** (predictions)
- For example,

$$\begin{cases} X = \{temperature, moisture, pressure, \ldots\} \\ Y = \{Sunny, Rainy, Stormy, \ldots\} \end{cases}$$



Univ. of Pittsburgh

Part 2
Modeling

CRF Modeling

- 1. Introduction
- *2. CRF Modeling*
  - ***Related Models***
  - ***Discriminative vs. Generative***
  - ***Chain CRF***
  - ***General CRF***
- 3. Inference using CRF
- 4. Training CRF
- 5. Applications of CRF

Univ. of Pittsburgh

Part 2
Modeling

Related Models

- Markov Random Fields
- Bayesian Network
- Factor Graph
- Sequencing Model

Univ. of Pittsburgh

Part 2 Modeling

Undirected Graph Model: MRF

- On an undirected graph, the joint distribution of variables $\mathbf{y}$

$$p(\mathbf{y}) = \frac{1}{Z}\prod_C \psi_C(\mathbf{y}_C),\ Z = \sum_{\mathbf{y}}\prod_C \psi_C(\mathbf{y}_C)$$

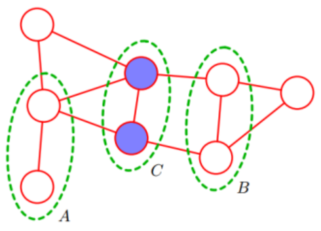
  - Potential Functions: $\psi_C(\mathbf{y}_C) \geq 0$
  - Partition Functions: Z
  - Energy Functions: $\psi_C(\mathbf{y}_C) = \exp\{-E(\mathbf{y}_C)\}$
- In MRF, $\psi_C(\mathbf{y}_C) \geq 0$ defined on cliques
- Markov property (next slide)
- A generative model

Univ. of Pittsburgh

Part 2 Modeling

Independence in MRF

- Markov property: for any two variables (or sets of Variables) A, B in MRF, the variable A is independent of B conditioned on A's neighbors.
- $\boldsymbol{A \perp B \mid C}$



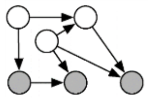
Univ. of Pittsburgh

Part 2 Modeling

Directed Graph: Bayesian Network

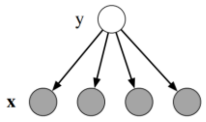
- Local Conditional Distributions
  - $\pi(s)$ Indices the parent of y_s

$$p(\mathbf{y}) = \prod_{s=1}^{S} p(y_s|\mathbf{y}_{\pi(s)})$$



- Naïve Bayes: once the class label is known, all the features are independent

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



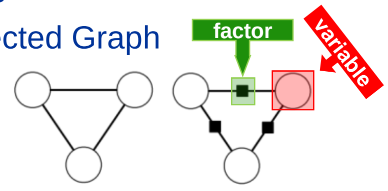
- Again, a generative model

Univ. of Pittsburgh

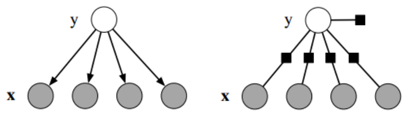
Part 2 Modeling

Factor Graph

- An explicit way to represent the factors in graphs
- Undirected Graph



- Directed Graph



Univ. of Pittsburgh

Part 2 Modeling

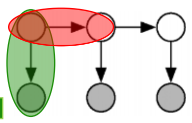
Sequence Prediction

- NER, POS problems
  - Set of observations: $X = \{x_t\}_{t=1}^{T}$
  - Set of underlying sequence of states $Y = \{y_t\}_{t=1}^{T}$
- HMM is generative:

Transition probability

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

Observation probability



- Basic Independent Assumptions
  - Observation is only dependent of its corresponding state;
  - Current state is only dependent of its previous state.
  - Strong assumptions!

Univ. of Pittsburgh

Part 2 Modeling

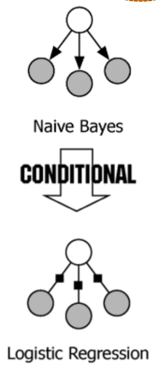
Discriminative Vs. Generative

- Generative: describes how a label vector **y** can probabilistically "generate" a feature vector **x**. $p(\mathbf{y},\mathbf{x})$
- Discriminative: describes how to take a feature vector **x** and assign it a label **y**. $p(\mathbf{y}\,|\,\mathbf{x})$
- Naïve Bayes:

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

- MaxEnt classifier:

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

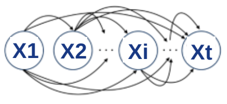


Univ. of Pittsburgh

Part 2 Modeling

Discriminative Vs. Generative

- Limitations of generative models
  - Modeling the joint distribution can lead to difficulties
  - features may have complex dependencies
    - Models often make strong independence assumptions



Fully dependent

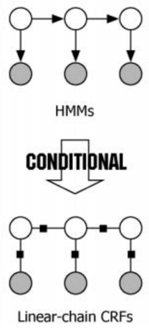
- Discriminative models:
  - No independence assumption is made for **X**
  - We care about conditional independences among **Y**
  - And how the **Y** can depend on **X**.
  - Best suits rich, **overlapping** features.

Univ. of Pittsburgh

Part 2 Modeling

Chain CRFs

- CRF is simply a conditional distribution with an associated graphical structure $p(\mathbf{y}|\mathbf{x})$
- *X* are constant with respect to *Y*
- Convert HMM to Chain CRF:



Univ. of Pittsburgh

Part 2 Modeling

Convert HMM to Chain CRF:

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

- As: $$p(\mathbf{y},\mathbf{x}) = \frac{1}{Z}\prod_{t=1}^{T}\exp\left\{\sum_{i,j\in S}\theta_{ij}\mathbf{1}_{\{y_t=i\}}\mathbf{1}_{\{y_{t-1}=j\}} + \sum_{i\in S}\sum_{o\in O}\mu_{oi}\mathbf{1}_{\{y_t=i\}}\mathbf{1}_{\{x_t=o\}}\right\},$$

- Where: $$\theta_{ij} = \log p(y'=i|y=j)$$ $$\mu_{oi} = \log p(x=o|y=i)$$ $$Z=1$$

Univ. of Pittsburgh

Part 2 Modeling

Convert HMM to Chain CRF:

- Step 2: introduce feature functions

$$p(\mathbf{y},\mathbf{x}) = \frac{1}{Z}\prod_{t=1}^{T}\exp\left\{\sum_{i,j\in S}\theta_{ij}\underbrace{\boxed{\mathbf{1}_{\{y_t=i\}}\mathbf{1}_{\{y_{t-1}=j\}}}}_{f_{ij}(y,y',x)} + \sum_{i\in S}\sum_{o\in O}\mu_{oi}\underbrace{\boxed{\mathbf{1}_{\{y_t=i\}}\mathbf{1}_{\{x_t=o\}}}}_{f_{io}(y,y',x)}\right\},$$

$f_{io}(y,y',x)$ $f_{ij}(y,y',x)$

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

Univ. of Pittsburgh

Part 2

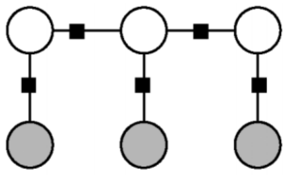
Modeling

Convert HMM to Chain CRF:

- Step 3: More compactly as

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

- Where we refer to a feature function generally as f_k , which ranges over both all the f_{ij} and all the f_{io}



Univ. of Pittsburgh

Part 2

Modeling

Convert HMM to Chain CRF:

- Step 4: Conditional Distribution

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

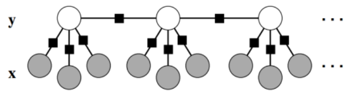
$$p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{y},\mathbf{x})}{\sum_{\mathbf{y}'} p(\mathbf{y}',\mathbf{x})} = \frac{\prod_{t=1}^{T}\exp\left\{\sum_{k=1}^{K}\theta_k f_k(y_t, y_{t-1}, x_t)\right\}}{\sum_{\mathbf{y}'}\prod_{t=1}^{T}\exp\left\{\sum_{k=1}^{K}\theta_k f_k(y'_t, y'_{t-1}, x_t)\right\}}$$

Univ. of Pittsburgh

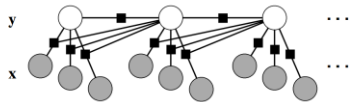
Part 2 Modeling

Chain CRFs

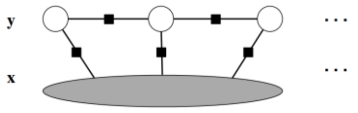
- We can change it so that each state depends on more observations:



- Or inputs at previous steps:



- Or all inputs:



Univ. of Pittsburgh

Part 2 Modeling

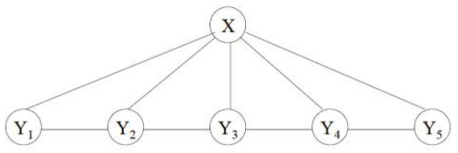
General CRF

- If $G = (V, E)$, and $\mathbf{Y} = (\mathbf{Y}_v)_{v \in V}$;
- $w \sim v \Leftrightarrow$ w and v are neighbors;
- $(\mathbf{X}, \mathbf{Y})$ is a CRF, if

$$p(\mathbf{Y}_v \,|\, \mathbf{X}, \mathbf{Y}_w, w \neq v) = p(\mathbf{Y}_v \,|\, \mathbf{X}, \mathbf{Y}_w, w \sim v).$$

- Example:

Suppose $P(Y_v \mid X$, all other $Y) = P(Y_v \mid X$, neighbors$(Y_v))$
then X with Y is a **conditional** random field



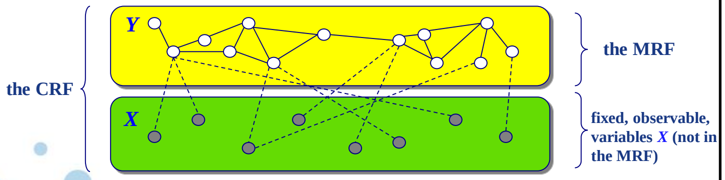
- $P(Y_3 \mid X$, all other $Y) = P(Y_3 \mid X, Y_2, Y_4)$
- Think of X as observations and Y as labels

Univ. of Pittsburgh

Part 2 Modeling

General CRFs: Visualization

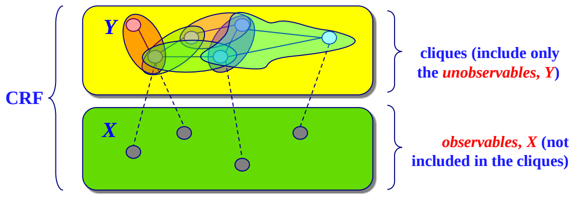
- According to the definition of CRF, the random variables $\mathbf{Y} = (\mathbf{Y}_v)_{v \in V}$ still obey the Markov Property with respect to the graph.



Univ. of Pittsburgh

Part 2 Modeling

General CRFs: Visualization


Y
X
CRF
cliques (include only the unobservables, Y)
observables, X (not included in the cliques)

- Divide MRF into cliques. The parameters inside each template are tied $\Phi_c(\mathbf{y}_c, \mathbf{x})$ --*potential functions*; functions for the template

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} e^{Q(\mathbf{y},\mathbf{x})} = \frac{1}{\sum_{\mathbf{y}'} e^{Q(\mathbf{y}',\mathbf{x})}} e^{Q(\mathbf{y},\mathbf{x})}, \quad Q(\mathbf{y},\mathbf{x}) = \sum_{c \in C} \Phi_c(\mathbf{y}_c, \mathbf{x})$$

- The cliques contain only unobservables (**y**); **x** is an argument to Φ_c
- The probability P_M(**y**|**x**) is a *joint distribution* over the unobservables *Y*

Univ. of Pittsburgh

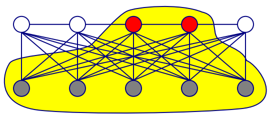
Part 2 Modeling

General CRFs: Visualization

- Φ_c is typically decomposed into a weighted sum of feature sensors f_i, producing:

$$\left.\begin{aligned} p(\mathbf{y}\mid\mathbf{x}) &= \frac{1}{Z}e^{Q(\mathbf{y},\mathbf{x})} \\ Q(\mathbf{y},\mathbf{x}) &= \sum_{c\in C}\Phi_c(\mathbf{y}_c,\mathbf{x}) \\ \Phi_c(\mathbf{y}_c,\mathbf{x}) &= \sum_{i\in F}\lambda_i f_i(y_c,\mathbf{x}) \end{aligned}\right\} \quad P(\mathbf{y}\mid\mathbf{x}) = \frac{1}{Z}e^{\sum_{c\in C}\sum_{i\in F}\lambda_i f_i(y_c,\mathbf{x})}$$

- Back to the chain-CRF!
- *Cliques* can be identified as *pairs* of adjacent Ys:



Univ. of Pittsburgh

Part 3 Inference

Inference using CRF

- 1. Introduction
- 2. CRF Modeling
- ***3. Inference using CRF***
  - ***General CRF***
  - ***Chain CRF***
- 4. Training CRF
- 5. Applications of CRF

Univ. of Pittsburgh

Part 3

Inference

General CRF

- Given the observations, and parameters, we target to find the best state sequence:
  $$\mathbf{y}^* = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})$$
- For the general CRF:
- $$\mathbf{y}^* = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}) =$$
  $$\arg\max_{\mathbf{y}} \frac{1}{Z} e^{\sum_{c\in C} \Phi_c(\mathbf{y}_c, \mathbf{x})} = \arg\max_{\mathbf{y}} \sum_{c\in C} \Phi_c(\mathbf{y}_c, \mathbf{x})$$
- But, exact inference in CRFs is intractable…
- Approximate methods!
  - MCMC, Belief Propagation

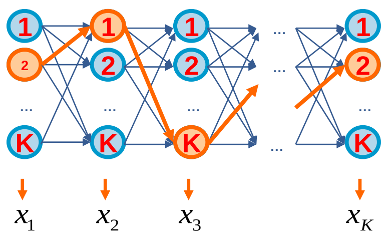
Univ. of Pittsburgh

Part 3

Inference

Inference in HMM

- Dynamic Programming:
  - Forward
  - Backward
  - Viterbi



Univ. of Pittsburgh

Chain CRFs

Part 3 Inference

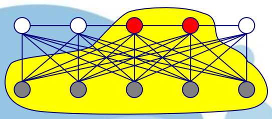
- Chain CRF could be done using dynamic programming

$$p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\lambda}) = \frac{1}{Z(\boldsymbol{x})}\exp\left(\sum_j \lambda_j F_j(\boldsymbol{y},\boldsymbol{x})\right)$$

$$F_j(\boldsymbol{y},\boldsymbol{x}) = \sum_{i=1}^{n} f_j(y_{i-1}, y_i, \boldsymbol{x}, i),$$

- Define a matrix $\{M_i(\boldsymbol{x})|i = 1,\ldots,n+1\}$ with size $|\mathcal{Y}\times\mathcal{Y}|$
  - $\mathcal{Y}$ is a finite label alphabet

$$M_i(y', y|\boldsymbol{x}) = \exp\left(\sum_j \lambda_j f_j(y', y, \boldsymbol{x}, i)\right)$$



$$p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\lambda}) = \frac{1}{Z(\boldsymbol{x})}\prod_{i=1}^{n+1} M_i(y_{i-1}, y_i|\boldsymbol{x})$$

$$Z(\boldsymbol{x}) = \left[\prod_{i=1}^{n+1} M_i(\boldsymbol{x})\right]_{\texttt{start},\texttt{end}}$$

Univ. of Pittsburgh

Chain CRFs

Part 3 Inference

- By defining the following forward and backward parameters,

$$\alpha_i(\boldsymbol{x})^T = \alpha_{i-1}(\boldsymbol{x})^T M_i(\boldsymbol{x})$$

$$\alpha_0(y|\boldsymbol{x}) = \begin{cases} 1 & \text{if } y = \texttt{start} \\ 0 & \text{otherwise} \end{cases}$$

$$Z(\mathbf{x}) = \textstyle\sum_{i\in S}\alpha_{\mathrm{T}}(i).$$

$$\beta_i(\boldsymbol{x}) = M_{i+1}(\boldsymbol{x})\beta_{i+1}(\boldsymbol{x})$$

$$\beta_{n+1}(y|\boldsymbol{x}) = \begin{cases} 1 & \text{if } y = \texttt{stop} \\ 0 & \text{otherwise} \end{cases}$$

$$Z(\mathbf{x}) = \beta_0(y_0)$$

Univ. of Pittsburgh

Part 3

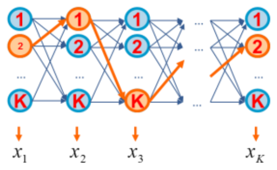
Inference

Chain CRFs

- The inference of linear-chain CRF is very similar to that of HMM
- We can write the marginal distribution:

$$p(Y_{i-1} = y', Y_i = y | \boldsymbol{x}^{(k)}, \boldsymbol{\lambda}) = \frac{\alpha_{i-1}(y'|\boldsymbol{x}) M_i(y', y|\boldsymbol{x}) \beta_i(y|\boldsymbol{x})}{Z(\boldsymbol{x})}$$

- Solve Chain-CRF using Dynamic Programming (Similar to Viterbi)!
  - 1. First computing α for all t (forward), then compute β for all t (backward).
  - 2. Return the marginal distributions computed.
  - 3. Run viterbi to find the optimal sequence



Univ. of Pittsburgh

Part 4

Learning

Training CRF

- 1. Introduction
- 2. CRF Modeling
- 3. Inference using CRF
- ***4. Training CRF***
  - ***General CRF***
  - ***Intro to approximate algorithms***
- 5. Applications of CRF

Univ. of Pittsburgh

Parameter learning

Part 4
Learning

- Given the training data, $\{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{N}$ we wish to learn parameters of the model.
- For chain or tree structured CRFs, they can be trained by maximum likelihood
- The objective function for chain-CRF is convex(see Lafferty et al(2001)).
- General CRFs are intractable hence approximation solutions are necessary

Univ. of Pittsburgh

Parameter learning

Part 4
Learning

- Conditional log-likelihood for a general CRF:

$$\mathcal{L}(\boldsymbol{\lambda}) = \sum_k \left[\log \frac{1}{Z(\boldsymbol{x}^{(k)})} + \sum_j \lambda_j F_j(\boldsymbol{y}^{(k)}, \boldsymbol{x}^{(k)}) \right]$$

$$\frac{\partial \mathcal{L}(\boldsymbol{\lambda})}{\partial \lambda_j} = \underbrace{E_{\tilde{p}(\boldsymbol{Y},\boldsymbol{X})}}_{\text{Empirical Distribution}} \left[F_j(\boldsymbol{Y}, \boldsymbol{X}) \right] - \sum_k E_{p(\boldsymbol{Y}|\boldsymbol{x}^{(k)},\boldsymbol{\lambda})} \left[F_j(\boldsymbol{Y}, \boldsymbol{x}^{(k)}) \right]$$

- It is not possible to analytically determine the parameter values that maximize the log-likelihood – setting the gradient to zero and solving for λ does not always yield a closed form solution. (Almost always)

Univ. of Pittsburgh

Part 4

Learning

Parameter learning

- This could be done using gradient descent

$$\lambda \propto \max_\lambda \mathrm{L}(\lambda; y \mid x) \propto \max_\lambda \log \sum_{i=1}^{N} p(\mathbf{y} \mid \mathbf{x}; \lambda)$$

$$\lambda_{i+1} \leftarrow \lambda_i + \alpha . \nabla_\lambda \mathrm{L}(\lambda; y \mid x)$$

- Until we reach convergence

$$| \mathrm{L}(\lambda_{i+1}; y \mid x) - \mathrm{L}(\lambda_i; y \mid x) | < \text{ò}$$

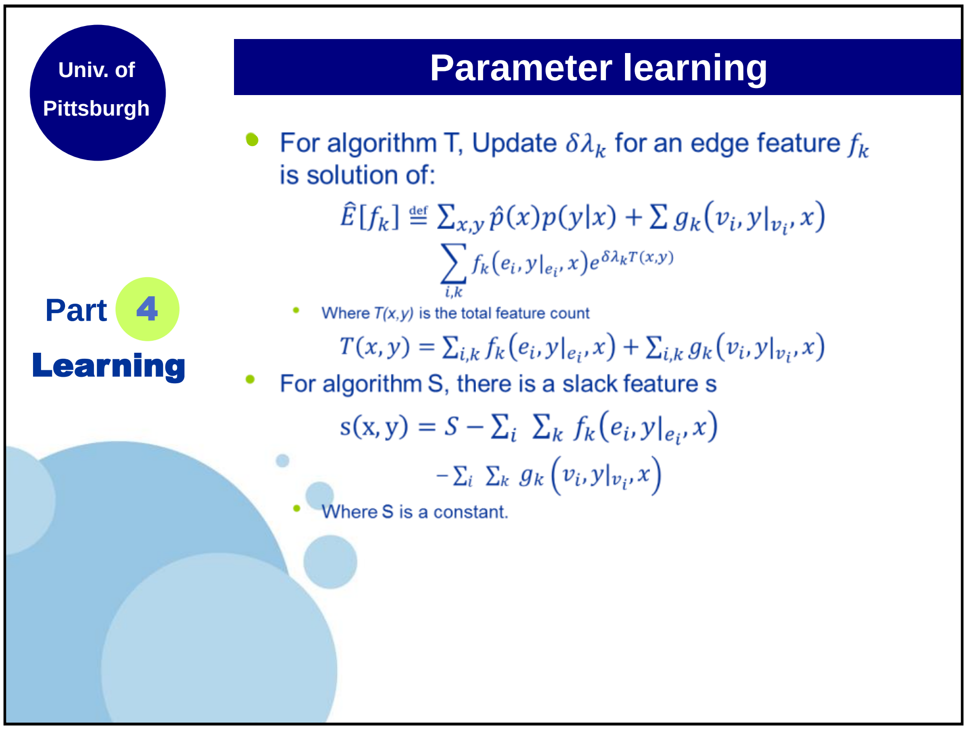
Univ. of Pittsburgh

Part 4

Learning

Parameter learning

- 2 algorithms based on improved iterative scaling are used:
  - Algorithm S
  - Algorithm T
- Improved iterative scaling algorithm updates weights as:
  - $\lambda_k \leftarrow \lambda_k + \delta\lambda_k$ for appropriately chosen δ



Univ. of Pittsburgh

Part 4 Learning

Parameter learning

- For algorithm T, Update $\delta\lambda_k$ for an edge feature f_k is solution of:

$$\hat{E}[f_k] \overset{\text{def}}{=} \sum_{x,y} \hat{p}(x)p(y|x) + \sum g_k(v_i, y|_{v_i}, x)$$
$$\sum_{i,k} f_k(e_i, y|_{e_i}, x) e^{\delta\lambda_k T(x,y)}$$

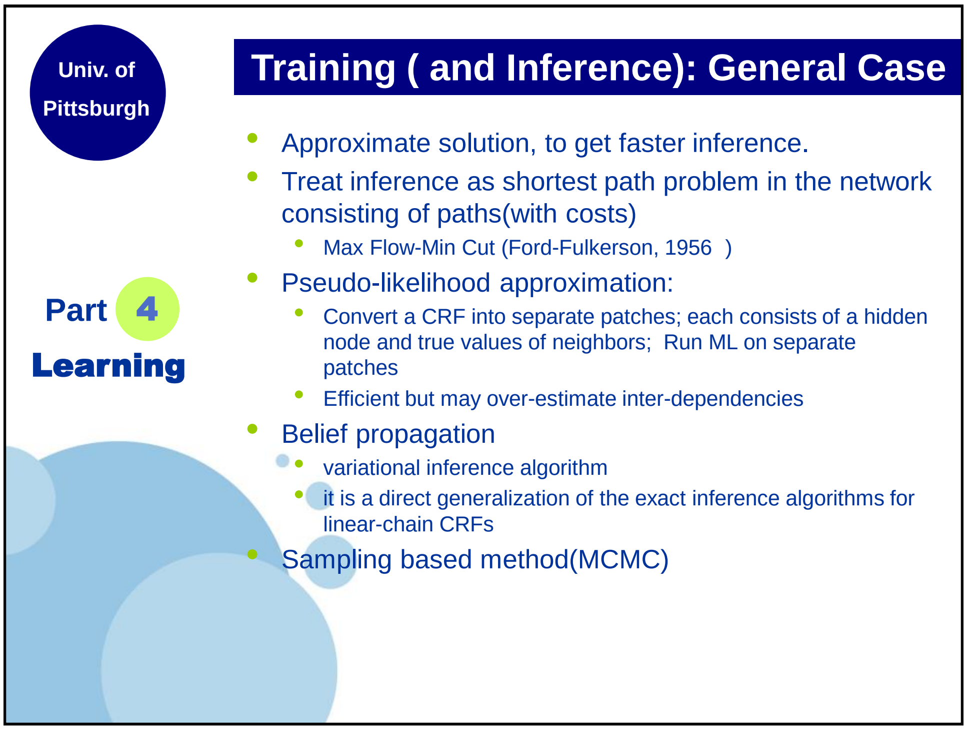
  - Where *T(x,y)* is the total feature count

$$T(x,y) = \sum_{i,k} f_k(e_i, y|_{e_i}, x) + \sum_{i,k} g_k(v_i, y|_{v_i}, x)$$

- For algorithm S, there is a slack feature s

$$\mathrm{s}(\mathrm{x},\mathrm{y}) = S - \sum_i \ \sum_k \ f_k(e_i, y|_{e_i}, x)$$
$$-\sum_i \ \sum_k \ g_k\left(v_i, y|_{v_i}, x\right)$$

  - Where S is a constant.



Univ. of Pittsburgh

Part 4 Learning

Training (and Inference): General Case

- Approximate solution, to get faster inference.
- Treat inference as shortest path problem in the network consisting of paths(with costs)
  - Max Flow-Min Cut (Ford-Fulkerson, 1956)
- Pseudo-likelihood approximation:
  - Convert a CRF into separate patches; each consists of a hidden node and true values of neighbors; Run ML on separate patches
  - Efficient but may over-estimate inter-dependencies
- Belief propagation
  - variational inference algorithm
  - it is a direct generalization of the exact inference algorithms for linear-chain CRFs
- Sampling based method(MCMC)

Univ. of Pittsburgh

Applications of CRF

Part 5 **Application**

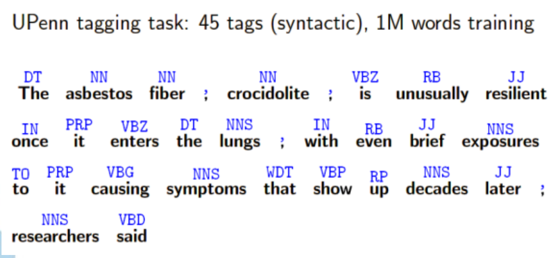
- 1. Introduction
- 2. CRF Modeling
- 3. Inference using CRF
- 4. Training CRF
- ***5. Applications of CRF***

Univ. of Pittsburgh

Part-of-Speech-Tagging

Part 5 **Application**

- POS(part of speech) tagging; the identification of words as nouns, verbs, adjectives, adverbs, etc.


UPenn tagging task: 45 tags (syntactic), 1M words training

DT	NN	NN		NN		VBZ	RB	JJ		
The	asbestos	fiber	,	crocidolite	,	is	unusually	resilient		
IN	PRP	VBZ	DT	NNS		IN	RB	JJ	NNS	
once	it	enters	the	lungs	,	with	even	brief	exposures	
TO	PRP	VBG	NNS	WDT	VBP	RP	NNS	JJ		
to	it	causing	symptoms	that	show	up	decades	later	,	
NNS	VBD									
researchers	said									

Univ. of Pittsburgh

Part 5 Application

Part-of-Speech-Tagging

- Each word to be labeled with one of 45 syntactic tags.
- 50%-50% train-test split
- Compared HMMs, MEMMs, and CRFs on Penn treebank POS tagging
- oov = out-of-vocabulary (not observed in the training set)

model	*error*	*oov error*
HMM	5.69%	45.99%
MEMM	6.37%	54.61%
CRF	5.55%	48.05%

Univ. of Pittsburgh

Part 5 Application

Part-of-Speech-Tagging

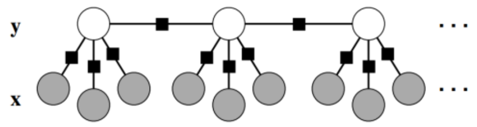
- But…
- Add a small set of orthographic features: whether a spelling begins with a number or upper case letter, whether it contains a hyphen, and if it contains one of the following suffixes: -ing, -ogy, -ed, -s, -ly, -ion, -tion, -ity, -ies

Feature Type	Description
Transition	$\forall k,k'$ $y_i = k$ and $y_{i+1}=k'$
Word	$\forall k,w$ $y_i = k$ and $x_i=w$ $\forall k,w$ $y_i = k$ and $x_{i-1}=w$ $\forall k,w$ $y_i = k$ and $x_{i+1}=w$ $\forall k,w,w'$ $y_i = k$ and $x_i=w$ and $x_{i-1}=w'$ $\forall k,w,w'$ $y_i = k$ and $x_i=w$ and $x_{i+1}=w'$
Orthography: Suffix	$\forall s$ in {"ing","ed","ogy","s","ly","ion","tion", "ity", …} and $\forall k$ $y_i=k$ and x_i ends with s
Orthography: Punctuation	$\forall k$ $y_i = k$ and x_i is capitalized $\forall k$ $y_i = k$ and x_i is hyphenated …

Univ. of Pittsburgh

Part 5 Application

Part-of-Speech-Tagging



model	*error*	*oov error*
HMM	5.69%	45.99%
MEMM	6.37%	54.61%
CRF	5.55%	48.05%
MEMM$^+$	4.81%	26.99%
CRF$^+$	4.27%	23.76%

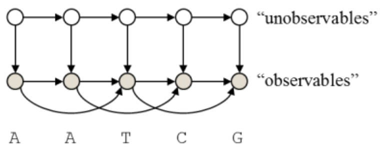
$^+$Using spelling features

Univ. of Pittsburgh

Part 5 Application

Is HMM(Gen.) better or CRF(Disc.)

- If your application gives you good structural information such that could be easily modeled by dependent distributions, and could be learnt tractably, go the generative way!
- Ex. Higher-order emissions from individual states



Univ. of Pittsburgh

Other Applications

Part 5
Application

- Application in computational biology
  - DNA and protein sequence alignment
  - Sequence homolog searching in databases
  - Protein secondary structure prediction
  - RNA secondary structure analysis
- Application in computational linguistics & computer science
  - Text and speech processing, including topic segmentation, part-of-speech (POS) tagging
  - Information extraction
  - Syntactic disambiguation

Univ. of Pittsburgh

References

- J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proc. ICML01, 2001.
- Charles Sutton and Andrew McCallum, “An Introduction to Conditional Random Fields for Relational Learning,” MIT Press, 2006
- Daniel Khashabi, “Conditional Random Fields and beyond ”, UIUC CS 546, 2013
- Zitao Liu, “Probabilistic Models of Time Series and Sequences”, 2014
- Bishop, Christopher M., “Pattern recognition and machine learning”. Vol. 1. New York: springer, 2006
- Peter, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data”, 2012

Univ. of Pittsburgh

Conditional Random Fields

Thank you!

Computer Science Department