Predicting Structures in NLP
Constrained Conditional Models
and
Integer Linear Programming

Dan Goldwasser, Vivek Srikumar, Dan Roth

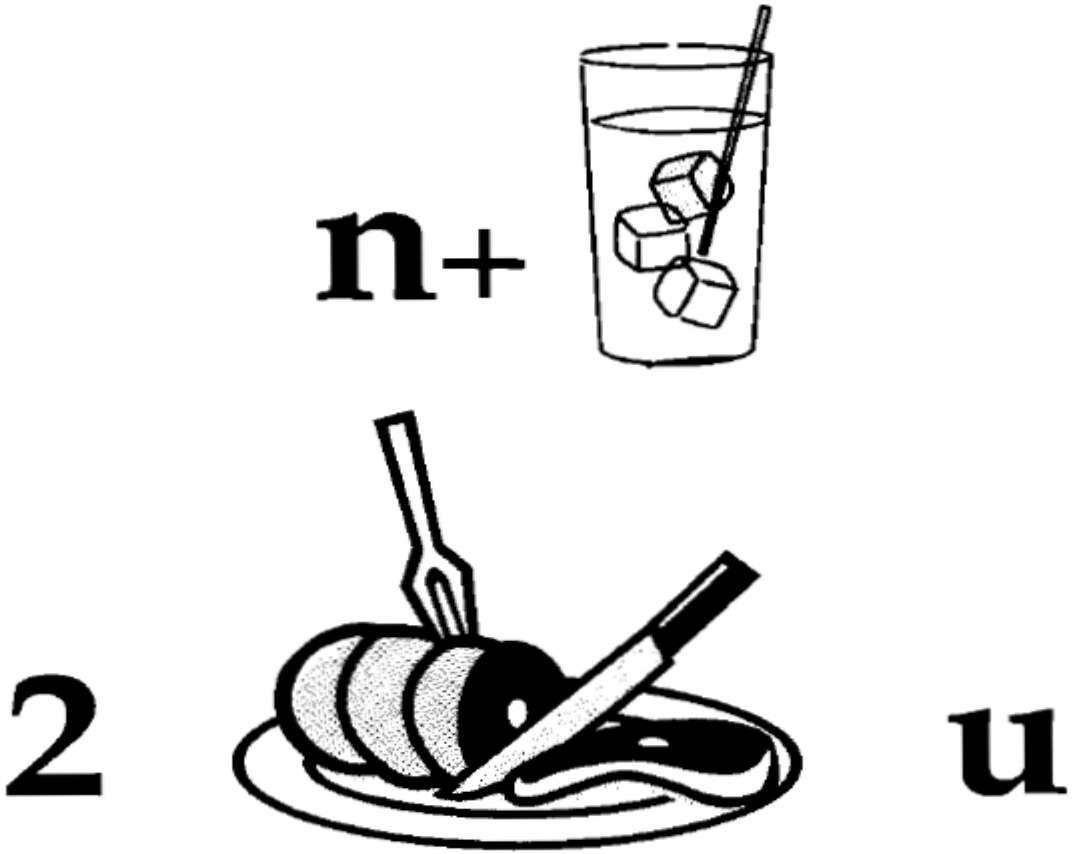
Department of Computer Science

University of Illinois at Urbana-Champaign

June 2012

NAACL

Nice to Meet You



Learning and Inference in NLP

- Natural Language Decisions are Structured
  - Global decisions in which several local decisions play a role but there are mutual dependencies on their outcome.
- It is essential to make coherent decisions in a way that takes the interdependencies into account. Joint, Global Inference.
- TODAY:
  - How to support making global, coherent decisions
  - How to learn models that are used, eventually, to make global decisions

  - A framework that allows one to exploit interdependencies among decision variables both in inference (decision making) and in learning.
  - **Inference:** A formulation for inference with expressive declarative knowledge.
  - **Learning:** Ability to learn simple models; amplify it power by exploiting interdependencies.

Constraints Driven Learning and Decision Making

- The focus of this tutorial is on
  - Augmenting statistical learning models with Declarative knowledge.
  - The knowledge will be expressed as constraints on the possible predictions our models can make.
- Why Constraints?
  - The Goal: Building a good NLP systems easily
  - We have prior knowledge at our hand
  - Within our framework we will see that we can use this knowledge to :
    - Improve decision making
    - Guide learning
    - Simplify the models we need to learn
    - Replace labeled data

Comprehension

(ENGLAND, June, 1989) - Christopher Robin is alive and well. He lives in England. He is the same person that you read about in the book, Winnie the Pooh. As a boy, Chris lived in a pretty home called Cotchfield Farm. When Chris was three years old, his father wrote a poem about him. The poem was printed in a magazine for others to read. Mr. Robin then wrote a book. He made up a fairy tale land where Chris lived. His friends were animals. There was a bear called Winnie the Pooh. There was also an owl and a young pig, called a piglet. All the animals were stuffed toys that Chris owned. Mr. Robin made them come to life with his words. The places in the story were all near Cotchfield Farm. Winnie the Pooh was written in 1925. Children still love to read about Christopher Robin and his animal friends. Most people don't know he is a real person who is grown now. He has written two books of his own. They tell what it is like to be famous.

1. Christopher Robin was born in England.
2. Winnie the Pooh is a title of a book.
3. Christopher Robin's dad was a magician.
4. Christopher Robin must be at least 65 now

This is an Inference Problem

Learning and Inference

- Global decisions in which several local decisions play a role but there are mutual dependencies on their outcome.
  - In current NLP we often think about simpler structured problems: Parsing, Information Extraction, SRL, etc.
  - As we move up the problem hierarchy (Textual Entailment, QA,….) not all component models can be learned simultaneously
  - We need to think about (learned) models for different sub-problems
  - Knowledge relating sub-problems (constraints) may appear only at evaluation time
- Goal: Incorporate models' information, along with prior knowledge (constraints) in making coherent decisions
  - decisions that respect the local models as well as domain & context specific knowledge/constraints.

Goal of the Tutorial

- By the end of the tutorial you should be able to:
  - Model structure prediction problems
    - **Injects declarative (domain, background) knowledge into your problem formulation**
  - Think about problem representation and the decomposition of the problem into natural components.
    - **Independently of algorithmic solutions**
  - Represent domain and other relevant knowledge as linear constraints
  - Think about possible way to support inference
  - Think about possible ways to learn your models
    - **Reason about several paradigms, their advantages and disadvantages.**

This Tutorial: Constrained Conditional Models (CCMs)

- Part 1: **Introduction to Constrained Conditional Models** (30min)
  - **Examples:**
    - **NE + Relations**
    - **Information extraction – correcting models with CCMS**
  - **First summary: What are CCMs**
  - **Problem Setting**
    - **Features and Constraints; some hints about training issues**

This Tutorial: Constrained Conditional Models

- Part 2: **Modeling NLP via CCMs** (45 minutes)
  - Introduction to ILP
  - Posing NLP Problems as ILP problems
    - **1. Sequence tagging** **(HMM/CRF + global constraints)**
    - **2. SRL** **(Independent classifiers + Global Constraints)**
    - **3. Sentence Compression** **(Language Model + Global Constraints)**
  - Less detailed examples
    - **1. Co-reference**
    - **2. A bunch more ...**
- Part 3: **Inference Algorithms** (15 minutes)
  - Exact Algorithms
  - Relaxation methods
  - Approximate Algorithms

BREAK

This Tutorial: Constrained Conditional Models (Part II)

- Part 4: **Training Paradigms for CCMs** (20 min)
  - Independently of constraints **(L+I);** Jointly with constraints **(IBT)**
  - Decomposed to simpler models
- Part 5: Constraints Driven Training (60 min)
  - Learning constraints' penalties
    - **Independently of learning the model**
    - **Jointly, along with learning the model**
  - Dealing with lack of supervision
    - **Constraints Driven Semi-Supervised learning (CODL)**
    - **Indirect Supervision**
  - Learning Constrained Latent Representations

This Tutorial: Constrained Conditional Models (Part II)

- Part 6: **Conclusion (& Discussion)** (10 min)
  - Summary
  - Building CCMs; Features and Constraints. Mixed models vs. Joint models;
  - Where is Knowledge coming from

THE END

PART 1: INTRODUCTION

This Tutorial: Constrained Conditional Models

Part 1: **Introduction to Constrained Conditional Models** (30min)

- **Examples:**
  - **NE + Relations**
  - **Information extraction – correcting models with CCMS**
- **First summary: What are CCMs**
- **Problem Setting**
  - **Features and Constraints; some hints about training issues**

Three Ideas Underlying Constrained Conditional Models

- **Idea 1:** Modeling

  Separate modeling and problem formulation from algorithms
  - Similar to the philosophy of probabilistic modeling

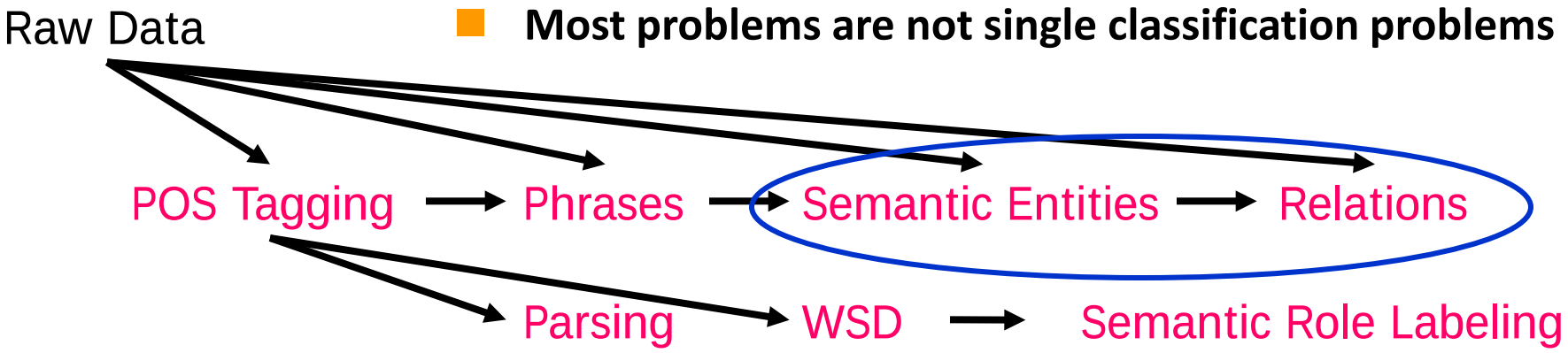
- **Idea 2:** Inference

  Keep model simple, make expressive decisions (via constraints)
  - Unlike probabilistic modeling, where models become more expressive

- **Idea 3:** Learning

  Expressive structured decisions can be supported by simply learned models
  - Global Inference can be used to amplify the simple models (and even minimal supervision).

Pipeline



- **Most problems are not single classification problems**
- Conceptually, Pipelining is a crude approximation
  - Interactions occur across levels and down stream decisions often interact with previous decisions.
  - Leads to propagation of errors
  - Occasionally, later stages are easier but cannot correct earlier errors.
- But, there are good reasons to use pipelines
  - Putting everything in one basket may not be right
  - How about choosing some stages and think about them jointly?

Inference with General Constraint Structure [Roth&Yih

Recognizing Entities and Relations

Improvement over no inference: 2-5%

other	0.05
irrelevant	0.05
spouse_of	**0.45**
born_in	0.50

other	0.10
irrelevant	0.10
spouse_of	0.05
born_in	**0.85**

other	0.05

Y = argm

= argm

Subject to Constraints

An Objective function that incorporates learned models with knowledge (constraints)

A constrained Conditional Model

Note:
Non Sequential Model

Models could be learned separately; constraints may come up only at decision time.

Task of Interests: Structured Output

- For each instance, assign values to a set of variables
- Output variables depend on each other
- Common NLP tasks
  - Parsing; Semantic Parsing; Summarization; Transliteration; Co-reference resolution, Textual Entailment…
- Common Information Extraction Tasks:
  - Entities, Relations,…
- Many pure machine learning approaches exist
  - Hidden Markov Models (HMMs); CRFs
  - Structured Perceptrons and SVMs…
- However, …

Information Extraction via Hidden Markov Models

Lars Ole Andersen . Program analysis and specialization for the C Programming language. PhD thesis. DIKU , University of Copenhagen, May 1994 .

Prediction result of a trained HMM

[AUTHOR]	Lars Ole Andersen . Program analysis and
[TITLE]	specialization for the
[EDITOR]	C
[BOOKTITLE]	Programming language
[TECH-REPORT]	. PhD thesis .
[INSTITUTION]	DIKU , University of Copenhagen , May
[DATE]	1994 .

Unsatisfactory results !

Many “natural constraints” are violated

Strategies for Improving the Results

- (Pure) Machine Learning Approaches
  - Higher Order HMM/CRF?
  - Increasing the window size?
  - Adding a lot of new features
    - Requires a lot of labeled examples
  - What if we only have a few labeled examples?

Increasing the model complexity

Increase difficulty of Learning

Can we keep the learned model simple and still make expressive decisions?

- Other options?
  - Constrain the output to make sense
  - Push the (simple) model in a direction that makes sense

Information extraction without Prior Knowledge

Lars Ole Andersen . Program analysis and specialization for the C Programming language. PhD thesis. DIKU , University of Copenhagen, May 1994 .

Prediction result of a trained HMM

[AUTHOR]	Lars Ole Andersen . Program analysis and
[TITLE]	specialization for the
[EDITOR]	C
[BOOKTITLE]	Programming language
[TECH-REPORT]	. PhD thesis .
[INSTITUTION]	DIKU , University of Copenhagen , May
[DATE]	1994 .

Violates lots of natural constraints!

Examples of Constraints

- Each field must be a consecutive list of words and can appear at most once in a citation.
- State transitions must occur on punctuation marks.
- The citation can only start with *AUTHOR* or *EDITOR*.
- The words *pp., pages* correspond to *PAGE*.
- Four digits starting with 20xx and 19xx are *DATE*.
- Quotations can appear only in *TITLE*
- …….

Easy to express pieces of "knowledge"

Non Propositional; May use Quantifiers

Information Extraction with Constraints

- Adding constraints, we get correct results!
  - **Without changing the model**

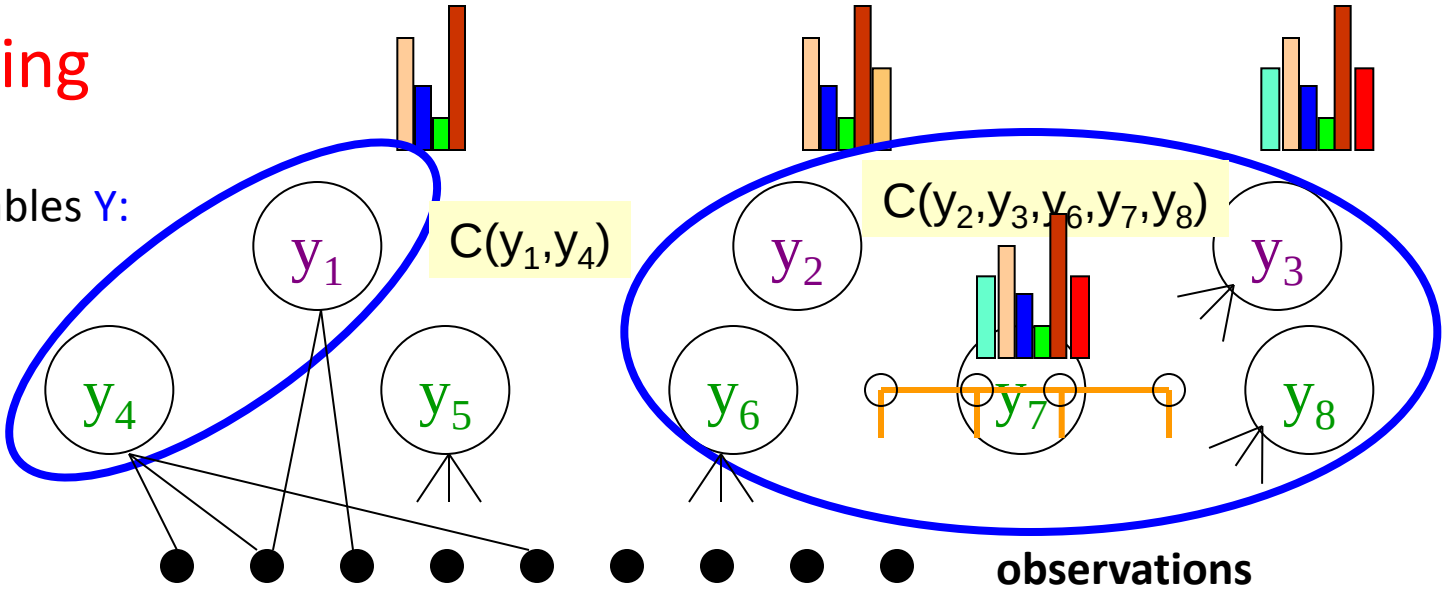
- [AUTHOR] Lars Ole Andersen .
  [TITLE] Program analysis and specialization for the C Programming language .
  [TECH-REPORT] PhD thesis .
  [INSTITUTION] DIKU , University of Copenhagen ,
  [DATE] May, 1994 .

Constrained Conditional Models Allow:

- **Learning a simple model**
- **Making decisions with a more complex model**
- **Accomplished by directly incorporating constraints to bias/re-ranks decisions made by the simpler model**

Problem Setting

- Random Variables Y:



- Conditional Distributions P (learned by models/classifiers)
- Constraints C– any Boolean function
  defined over partial assignments (possibly: + weights W)

- Goal: Find the "best" assignment
  - **The assignment that achieves the highest global performance.**
- This is an Integer Programming Problem

$Y^* = \text{argmax}_Y P \bullet Y \quad (+ W \bullet C)$ subject to constraints C

Constrained Conditional Models

$$\underset{y}{\operatorname{argmax}} \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

Penalty for violating the constraint.

(Soft) constraints component

Weight Vector for "local" models

Features, classifiers; log-linear models (HMM, CRF) or a combination

How far y is from a "legal" assignment

How to solve?

This is an Integer Linear Program

Solving using ILP packages gives an exact solution.

Cutting Planes, Dual Decomposition & other search techniques are possible

How to train?

Training is learning the objective function

Decouple? Decompose?

How to exploit the structure to minimize supervision?

What is a Constrained Conditional Model?

$$\underset{y}{\operatorname{argmax}}\, \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

Modeling NLP problem • Variables, Features and constraints	**Objective function** • Constrained Conditional Model
Constrained optimization language • How to represent inference?	Integer linear program
Inference • How to solve it?	Several inference algorithms: Exact ILP, search, relaxation; dynamic prog.
Learning • How to learn the objective function?	Learning λ and ρ. Several learning strategies: L+I, IBT, others.

Examples: CCM Formulations

$$\underset{y}{\operatorname{argmax}} \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

CCMs can be viewed as a general interface to easily combine declarative domain knowledge with data driven statistical models

Formulate NLP Problems as ILP problems (inference may be done otherwise)

- 1. Sequence tagging (HMM/CRF + Global constraints)
- 2. Sentence Compression (Language Model + Global Constraints)
- 3. SRL (Independent classifiers + Global Constraints)

Sentence Compression/Summarization:

Language Model based:

Argmax $\sum \lambda_{ijk}$ x_{ijk}

Linguistics Constraints

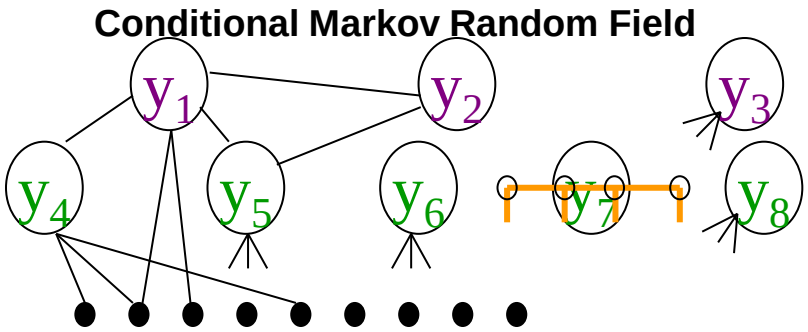
If a modifier chosen, include its head

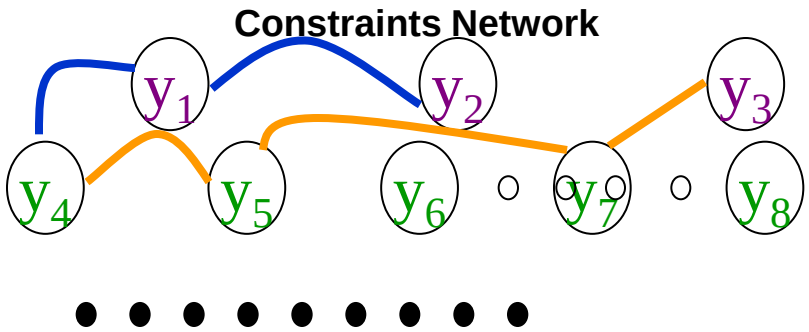
If verb is chosen, include its arguments

Context: There are Many Formalisms

- Our goal is to assign values to multiple interdependent discrete variables
- These problems can be formulated and solved with multiple approaches
  - Markov Random Fields (MRFs) provide a general framework for it. But:
- The decision problem for MRFs can be written as an ILP too
  - [Roth & Yih 04,07, Taskar 04]
- Key difference: In MRF approaches the model is learned globally.
  - Not easy to systematically incorporate problem understanding and knowledge
  - Our approach, on the other hand, is designed to address also cases in which some of the component models are learned in other contexts and at other times, or incorporated as background knowledge.
  - **That is, some components of the global decision need not, or cannot, be trained in the context of the decision problem.**
  - Markov Logic Networks (MLNs) attempt to compile knowledge into an MRF, thus provide one example of a global training approach.
- Caveat: Everything can be done with everything, but there are key conceptual differences that impact what is easy to do

Context: Constrained Conditional Models





$$y^* = \text{argmax}_y \sum w_i \phi(x; y)$$

- Linear objective functions
- Typically $\phi(x,y)$ will be local functions, or $\phi(x,y) = \phi(x)$

$$- \sum_i \rho_i d_C(x,y)$$

- Expressive constraints over output variables
- Soft, weighted constraints
- Specified declaratively as FOL formulae

- Clearly, there is a joint probability distribution that represents this mixed model.
- We would like to:
  - Learn a simple model or several simple models
  - Make decisions with respect to a complex model

Key difference from MLNs which provide a concise definition of a model, but the whole joint one.

$$\operatorname*{argmax}_{y} \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1} \rho_i d(y, 1_{C_i(x)})$$

Features Versus Constraints in CCMs

- $F_i : X \times Y \rightarrow \{0,1\}$ or R; $C_i : X \times Y \rightarrow \{0,1\}$;
  - In principle, constraints and features can encode the same properties
- In practice, they are very different

- Features
  - Local , short distance properties – to allow tractable inference
  - Propositional (grounded):
  - E.g. True if: "the" followed by a Noun occurs in the sentence"
- Constraints
  - Global properties
  - Quantified, first order logic expressions
  - E.g.True if: "all y_is in the sequence y are assigned different values."

Indeed, used differently

Role of Constraints: Encoding Prior Knowledge

- Consider encoding the knowledge that:
  - Entities of type A and B cannot occur simultaneously in a sentence
- The "Feature" Way
  - Many new (possible) features: propsitionalizing;
  - Only a "suggestion" to the learning algorithm; need to learn weights
  - Wastes parameters to learn indirectly knowledge we have.
  - Results in higher order models; may require tailored models
- The Constraints Way
  - Tell the model what it should attend to
  - Keep the model simple; add expressive constraints directly
  - A small set of constraints
  - Allows for decision time incorporation of constraints

A form of supervision

Details depend on whether (1) learned model use $\phi(x,y)$ or $\phi(x)$
(2) hard or soft constraints

Constrained Conditional Models—Before a Summary

- Constrained Conditional Models – ILP formulations – have been shown useful in the context of many NLP problems
- [Roth&Yih, 04,07: Entities and Relations; Punyakanok et. al: SRL …]
  - Summarization; Co-reference; Information & Relation Extraction; Event Identifications; Transliteration; Textual Entailment; Knowledge Acquisition; Sentiments; Temporal Reasoning, Dependency Parsing,…
- Some theoretical work on training paradigms [Punyakanok et. al., 05 more; Constraints Driven Learning, PR, Constrained EM…]

- We will provide some insights into theoretical issues and cover some of the applications.

- **Summary of work & a bibliography: http://L2R.cs.uiuc.edu/tutorials.html**

Constrained Conditional Models – 1st Summary

- Introduced CCMs as a formalisms that allows us to
  - Learn simpler models than we would otherwise
  - Make decisions with expressive models, augmented by declarative constraints
- Focused on modeling – posing NLP problems as CCMs
  - 1. Sequence tagging (HMM/CRF + global constraints)
  - 2. SRL (Independent classifiers + Global Constraints)
  - 3. Sentence Compression (Language Model + Global Constraints)
- Next: More Modeling & Inference
  - From declarative constraints to CCMs; solving ILP, exactly & approximately
- Second half – Learning
  - Supervised setting, and supervision-lean settings

PART 2: MODELING

This Tutorial: Constrained Conditional Models

- Part 2: **Modeling NLP via CCMs** (45 minutes)
  - Introduction to ILP
  - Posing NLP Problems as ILP problems
    - 1. Sequence tagging (HMM/CRF + global constraints)
    - 2. SRL (Independent classifiers + Global Constraints)
    - 3. Sentence Compression (Language Model + Global Constraints)
  - Less detailed examples
    - 1. Co-reference
    - 2. A bunch more ...

What is a Constrained Conditional Model?

$$\underset{y}{\operatorname{argmax}} \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

Modeling NLP problem • Variables, Features and constraints	**Objective function** • Constrained Conditional Model
Constrained optimization language • How to represent inference?	Integer linear program
Inference • How to solve it?	Several inference algorithms: Exact ILP, search, relaxation; dynamic prog.
Learning • How to learn the objective function?	Learning λ and ρ. Several learning strategies: L+I, IBT, others.

Modeling NLP via CCMs

- Inference is a discrete optimization problem
  - Goal: To assign values to a collection of variables of interest
- We choose to model inference step using the language of Integer Linear Programming
- CCMs provides:
  - A way to focus on problem definition rather than how to solve it
  - Simple (to write down) but expressive formulation
  - A way to use of declarative knowledge

Integer Linear Programming: Review

- Telfa Co. produces tables and chairs; wants to maximize profit
  - Each table makes $8 profit, each chair makes $5 profit.
  - A table requires 1 hour of labor and 9 sq. feet of wood
  - A chair requires 1 hour of labor and 5 sq. feet of wood
  - We have only 6 hours of work and 45sq. feet of wood

- Variables
- Objective function

y_1: Number of tables manufactured
y_2: Number of chairs manufactured

- Constraints
  - Labor
  - Wood
  - Variable

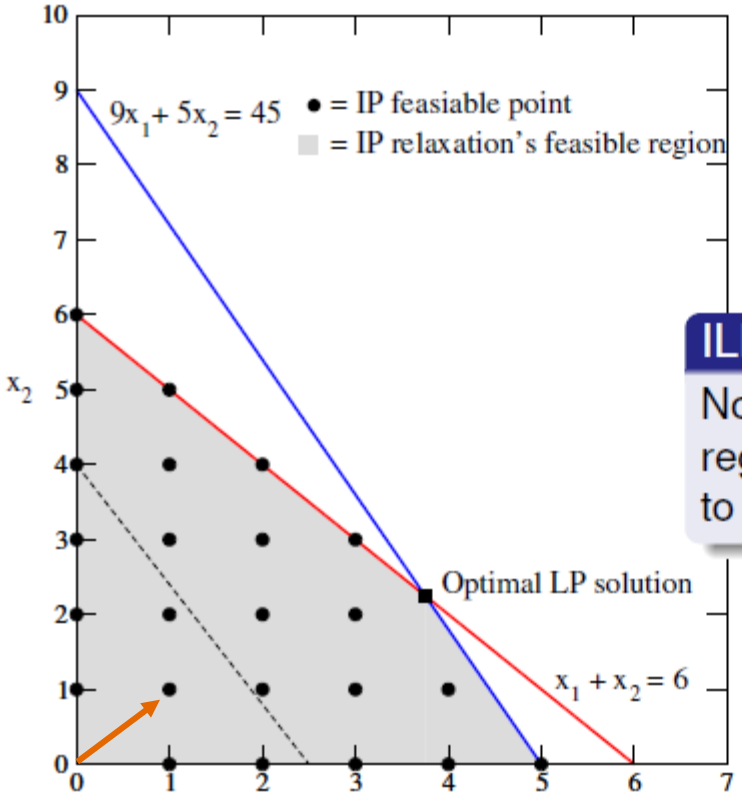
$$\max_{y_1,y_2} 8y_1 + 5y_2$$
$$y_1 + y_2 \leq 6$$
$$9y_1 + 5y_2 \leq 45$$
$$y_1, y_2 \geq 0$$
$$y_1, y_2 \text{ integers}$$

$$\max_{\vec{y}} z = \vec{c} \cdot \vec{y}$$
$$\text{subject to } \mathbf{A}\vec{y} \leq \vec{b}$$
$$y_i \text{ integer for all } i$$

We cannot build fractional tables or chairs!

Geometry of integer linear programs

$$\max_{y_1, y_2} 8y_1 + 5y_2$$
$$y_1 + y_2 \leq 6$$
$$9y_1 + 5y_2 \leq 45$$
$$y_1, y_2 \geq 0$$
$$y_1, y_2 \text{ integers}$$



ILP Solutions

Not all points within feasible region of an LP will be solutions to ILP problem.

Cost (profit) vector

Modeling Your Problem

$$\operatorname*{argmax}_{y} \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

Penalty for violating the constraint.

(Soft) constraints component

Weight Vector for "local" models

A collection of Classifiers; Log-linear models (HMM, CRF) or a combination

How far y is from a "legal" assignment

- How do we write our models in this form?
  - What goes in an objective function?
  - How to design constraints?

$$\max_{\vec{y}} \quad \vec{c} \cdot \vec{y}$$
$$\text{subj. to} \quad \mathbf{A}\vec{y} \leq \vec{b}$$
$$y_i \in \{0, 1\}$$

We will consider the case when y's are restricted to 0 or 1.

Modeling Your Problem: Decision Variables

$$\underset{y}{\operatorname{argmax}} \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

Functions from (x,y) to 0 or 1

$$F(x, y) = \{fi(x, y)\}$$

- F(x,y) is a collection of features from x and y
  - Eg: Does the i^{th} word have the POS tag NN?
  - Several such features, not all active in a given (x,y) instance

- Define indicator variables
  - Is f_i is active in an input?

$$1_{f_i} = \begin{cases} 1 & \text{if } f_i(x, y) \text{ active} \\ 0 & \text{otherwise} \end{cases}$$

- F(x,y) can be rewritten using indicator variables as $\sum_i 1_{f_i} f_i(x, y)$

$$arg\max_{y} \sum_{i} 1_{f_i} (\lambda \cdot f_i(x, y)) - \sum_{i=1}^{K} \rho_i d\left(y, 1_{C_i(x)}\right)$$

Modeling Your Problem: Constraints

Score for this variable

Penalty for violating the constraint.

$$arg\max_y \sum_i 1_{f_i}(\lambda \cdot f_i(x,y)) - \sum_{i=1}^{K} \rho_i d\left(y, 1_{C_i(x)}\right)$$

Inference variables can be 0 or 1

How far y is from a "legal" assignment

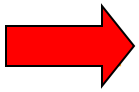
- Suppose we want to disallow all "illegal" assignments.
  - Make all the ρ_i infinity
  - ***Hard constraints***; can be written as linear inequalities in terms of the inference variables

$$\begin{array}{ll} arg\max & \sum_i 1_{f_i}(\lambda \cdot f_i(x,y)) \\ \text{subj. to} & \mathbf{A1_f} \geq b \end{array}$$

$\mathbf{1_f}$ is the vector of all inference variables

CCM Examples

- Many works in NLP make use of constrained conditional models, implicitly or explicitly.
- Next we describe three examples in detail.



- Example 1: Semantic Role Labeling
  - The use of inference with constraints to improve semantic parsing
- Example 2: Sequence Tagging
  - Adding long range constraints to a simple model
- Example 3: Sentence Compression
  - Simple language model with constraints outperforms complex models

Example 1: Semantic Role Labeling

Who did what to whom, when, where, why,…

Semantic Role Labeling Output

Input Text:

A car bomb that exploded outside the U.S. military base in Beniji killed 11 Iraqi citizens.

Result: Complete!

⊞ General Explanation of Argument Labels

A	bomb [A1]		killer [A0]
car			
bomb			
that	bomb (Reference) [R-A1]		
exploded	V: explode		
outside	location [AM-LOC]		
the			
U.S.			
military	temporal [AM-TMP]		
base			
in	location [AM-LOC]		
Beniji			
killed			V: kill
11			corpse [A1]
Iraqi			
citizens			
.			

Demo:http://L2R.cs.uiuc.edu/~cogcomp

Top ranked system in CoNLL'05 shared task

Key difference is the Inference

A simple sentence

I *left* my pearls to my daughter in my will .

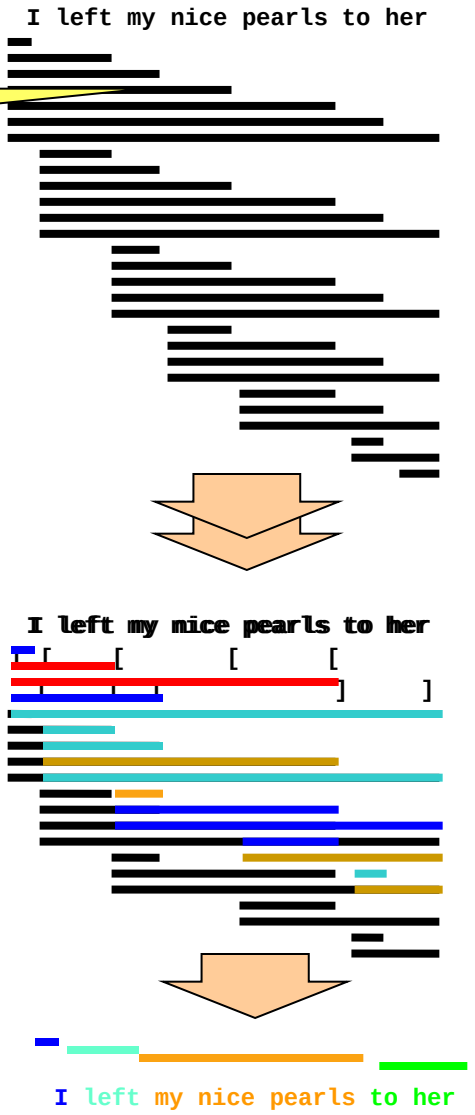
[I]$_{A0}$ *left* [my pearls]$_{A1}$ [to my daughter]$_{A2}$ [in my will]$_{AM\text{-}LOC}$.

- ***A0*** Leaver
- ***A1*** Things left
- ***A2*** Benefactor
- ***AM-LOC*** Location

I *left* my pearls to my daughter in my will .

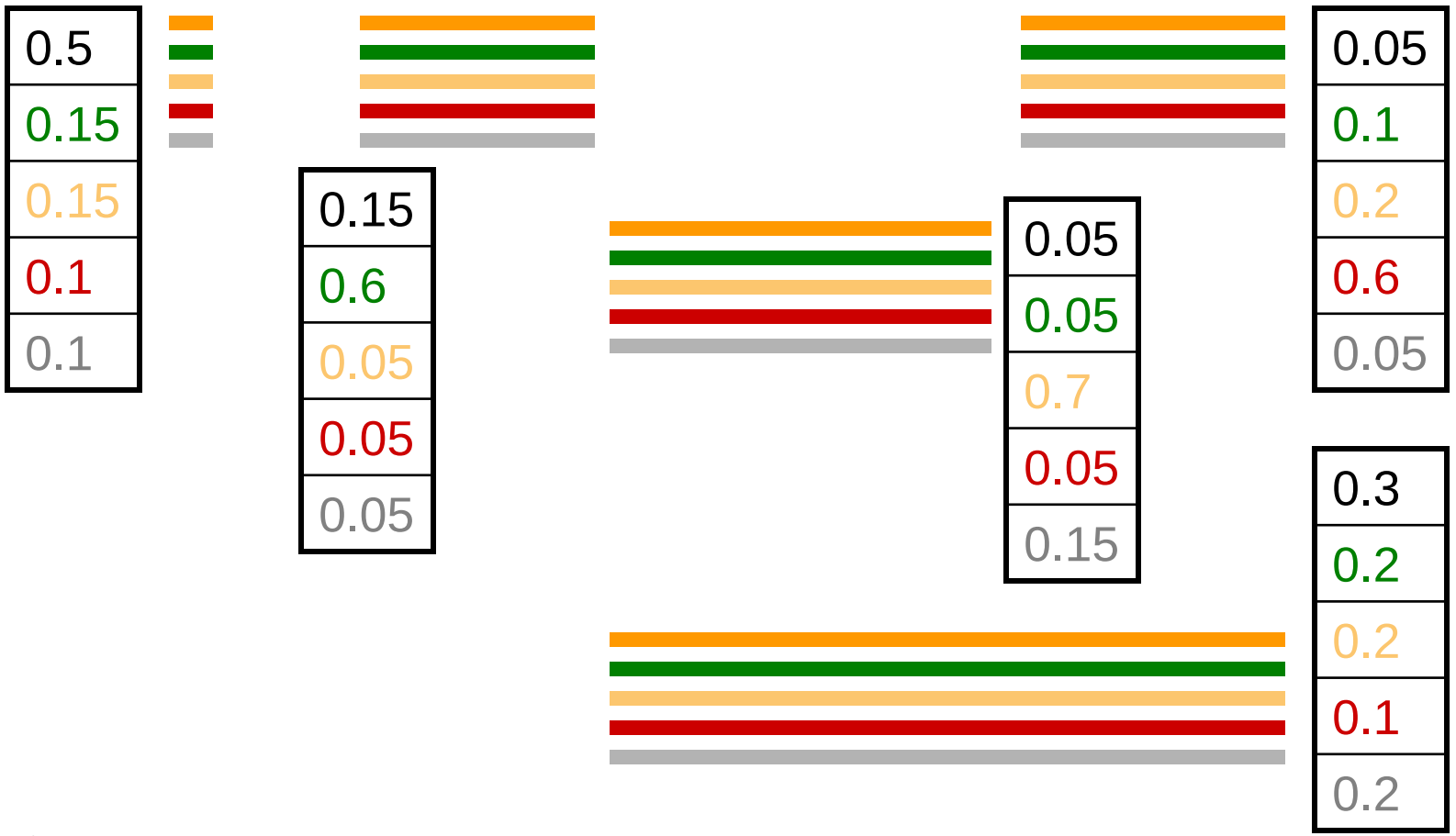
Algorithmic Approach

- Identify argument candidates
  - Pruning [Xue&Palmer, EMNLP'04]
  - Argument Identifier
    - Binary classification
- Classify argument candidates
  - Argument Classifier
    - Multi-class classification
- Inference
  - Use the estimated probability distribution given by the argument classifier
  - Use structural and linguistic constraints
  - Infer the optimal global output



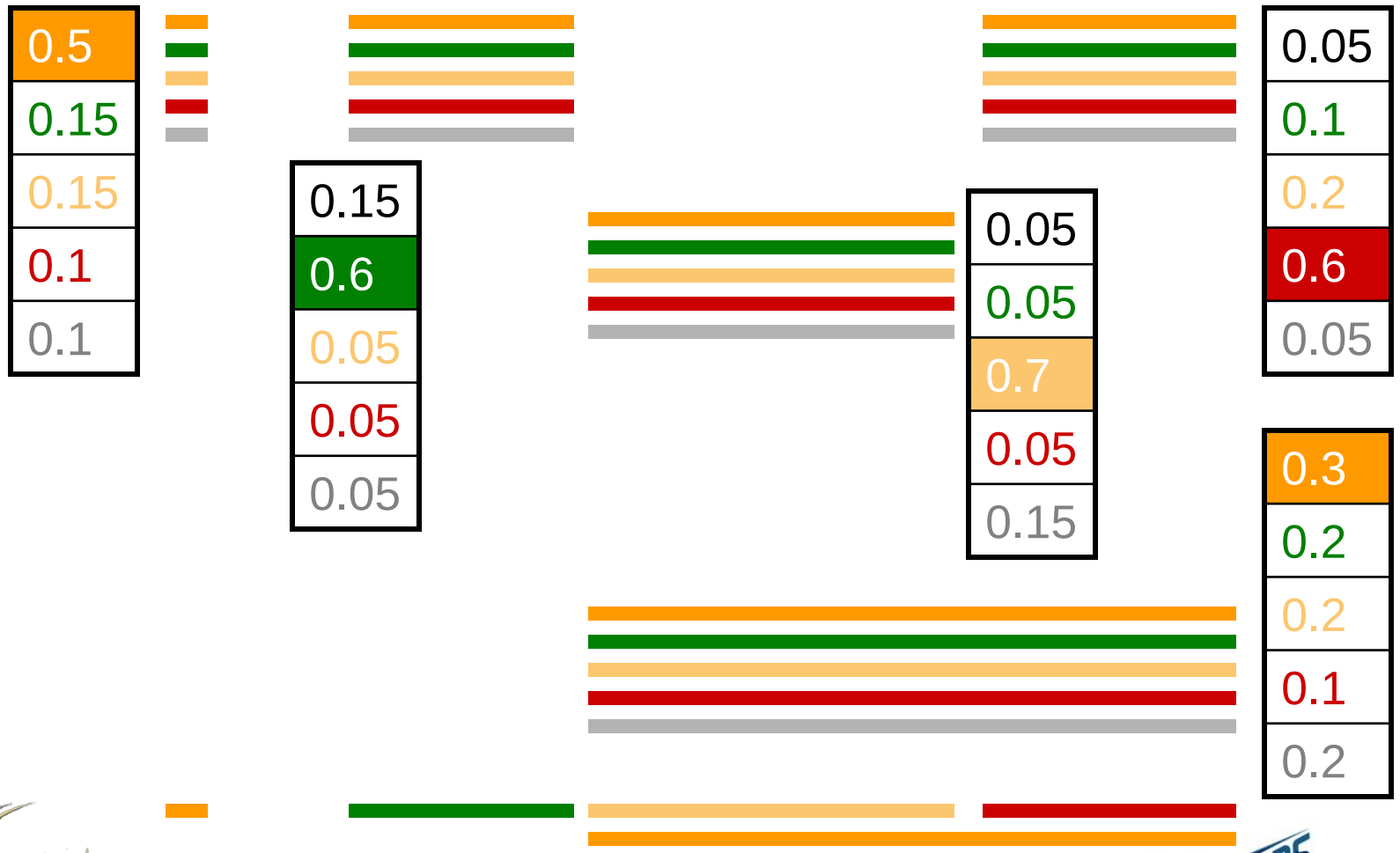
Semantic Role Labeling (SRL)

I *left* my pearls to my daughter in my will .



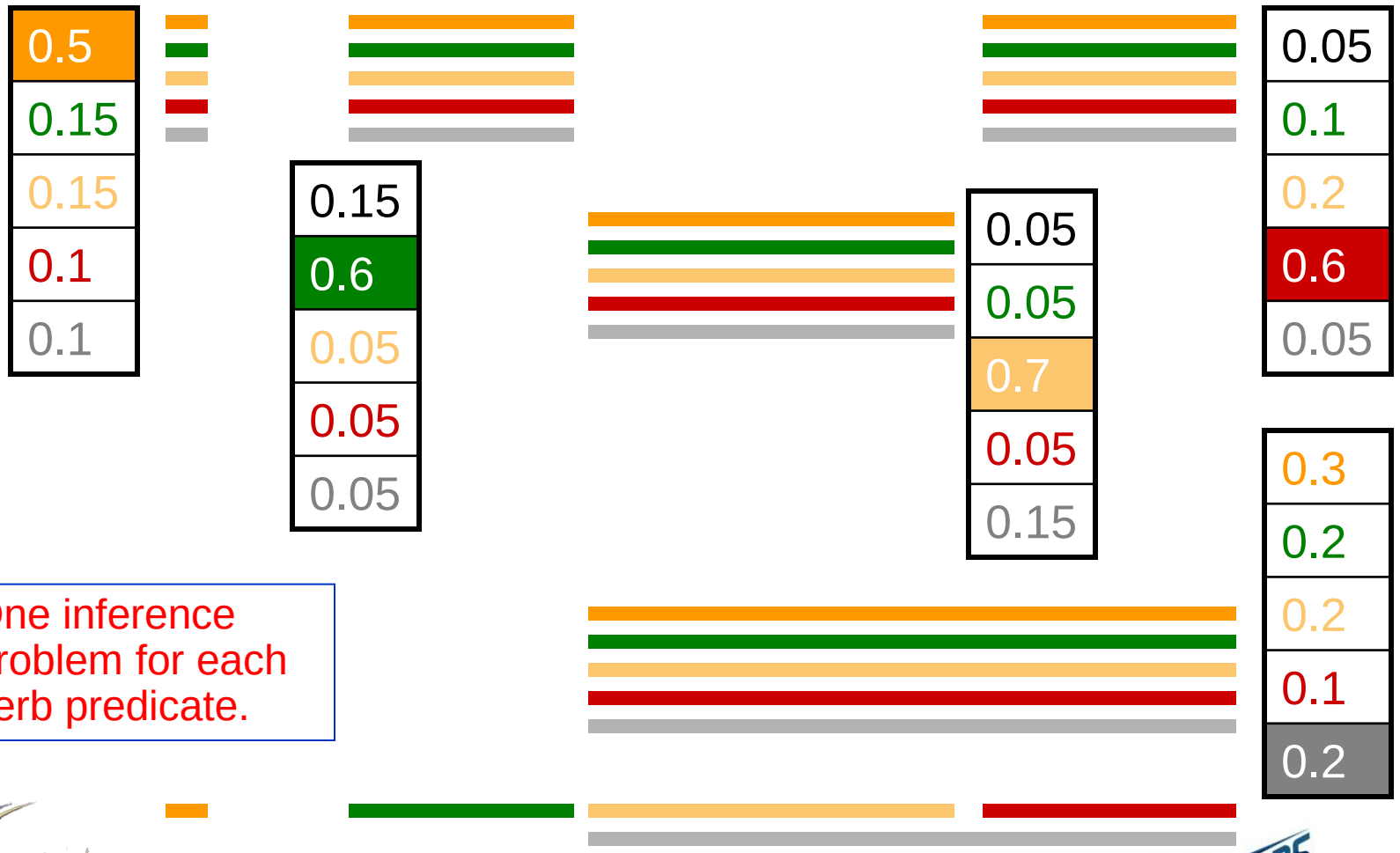
Semantic Role Labeling (SRL)

I *left* my pearls to my daughter in my will .



Semantic Role Labeling (SRL)

I *left* my pearls to my daughter in my will .



One inference problem for each verb predicate.

Constraints

- No duplicate argument classes

$$\forall y \in \mathcal{Y}, \sum_{i=0}^{n-1} 1_{\{y_i=y\}} \leq 1$$

Any Boolean rule can be encoded as a set of linear inequalities.

- Reference-Ax

$$\forall y \in \mathcal{Y}_R, \sum_{i=0}^{n-1} 1_{\{y_i=y=\text{“R-Ax”}\}} \leq \sum_{i=0}^{n-1} 1_{\{y_i=\text{“Ax”}\}}$$

If there is an Reference-Ax phrase, there is an Ax

- Continuation-Ax

$$\forall j, y \in \mathcal{Y}_C, \ 1_{\{y_j=y=\text{“C-Ax”}\}} \leq \sum_{i=0}^{j} 1_{\{y_i=\text{“Ax”}\}}$$

If there is an Continuation-x phrase, there is an Ax … e it

Universally quantified rules

- Many other possible constraints:
  - Unique labels
  - No overlapping or embedding
  - Relations between number of arguments; order constraints
  - If verb is of type A, no argument of type B

Learning Based Java: allows a developer to encode constraints in First Order Logic; these are compiled into linear inequalities automatically.

SRL: Posing the Problem

$$\text{maximize} \quad \sum_{i=0}^{n-1} \sum_{y \in \mathcal{Y}} \lambda_{\mathbf{x}_i, y} 1_{\{y_i = y\}}$$

$$\text{where} \quad \lambda_{\mathbf{x},y} = \lambda \cdot F(\mathbf{x}, y) = \lambda_y \cdot F(\mathbf{x})$$

$$\text{subject to} \quad \forall i, \sum_{y \in \mathcal{Y}} 1_{\{y_i = y\}} = 1$$

$$\forall y \in \mathcal{Y}, \sum_{i=0}^{n-1} 1_{\{y_i = y\}} \leq 1$$

$$\forall y \in \mathcal{Y}_R, \sum_{i=0}^{n-1} 1_{\{y_i = y = \text{“R-Ax”}\}} \leq \sum_{i=0}^{n-1} 1_{\{y_i = \text{“Ax”}\}}$$

$$\forall j, y \in \mathcal{Y}_C, \; 1_{\{y_j = y = \text{“C-Ax”}\}} \leq \sum_{i=0}^{j} 1_{\{y_i = \text{“Ax”}\}}$$

A	bomb [A1]		killer [A0]
car			
bomb			
that	bomb (Reference) [R-A1]		
exploded	V: explode		
outside	location [AM-LOC]		
the			
U.S.			
military	temporal [AM-TMP]		
base			
in	location [AM-LOC]		
Beniji			
killed			V: kill
11			corpse [A1]
Iraqi			
citizens			
.			

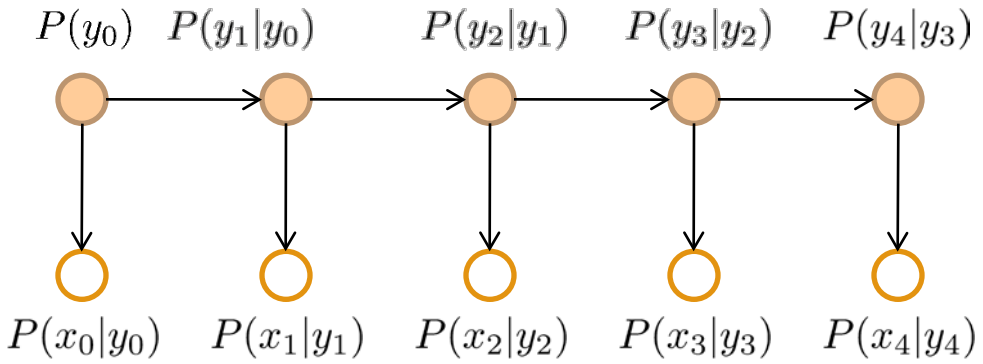
CCM Examples

- Many works in NLP make use of constrained conditional models, implicitly or explicitly.
- Next we describe three examples in detail.

- Example 1: Semantic Role Labeling
  - The use of inference with constraints to improve semantic parsing
- Example 2: Sequence Tagging
  - Adding long range constraints to a simple model
- Example 3: Sentence Compression
  - Simple language model with constraints outperforms complex models

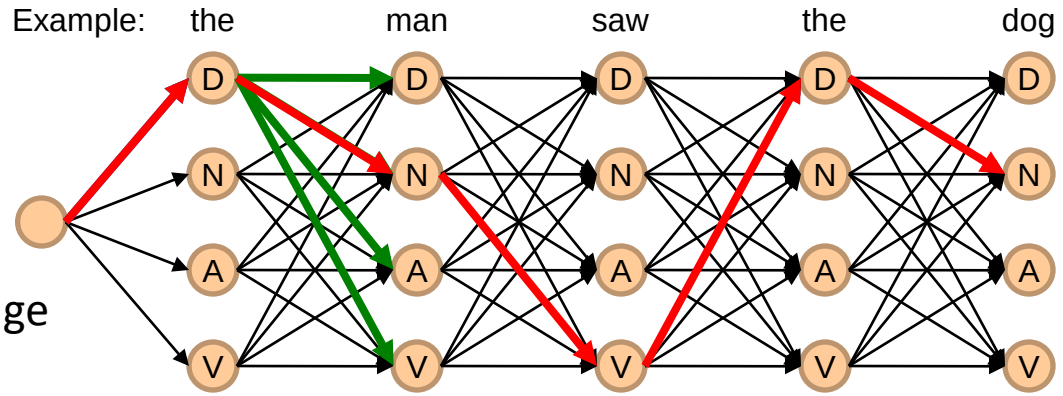
Example 2: Sequence Tagging

HMM : $$\mathbf{y}^* = \underset{\mathbf{y}\in\mathcal{Y}}{\operatorname{argmax}} P(y_0)P(x_0|y_0)\prod_{i=1}^{n-1} P(y_i|y_{i-1})P(x_i|y_i)$$



Here, *y*'s are labels; *x*'s are observations.

The ILP's objective function must include all entries of the Conditional Probability Table.



Every edge is a Boolean variable that selects a transition CPT entry.

They are related: if we choose

y_0 = D then we must choose an edge

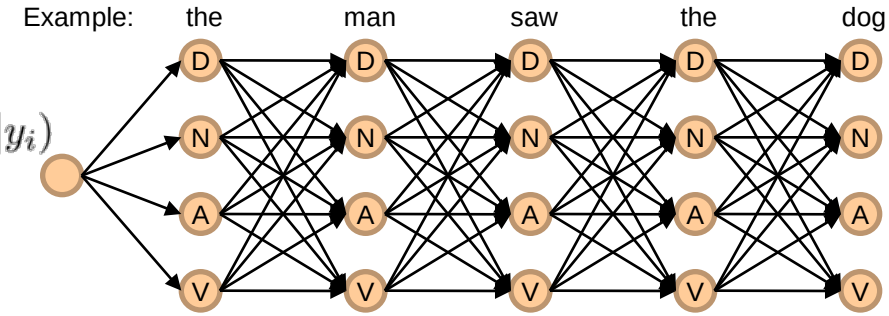
y_0 = D ∧ y_1 = ? .

Every assignment to the *y*'s is a path.

Example 2: Sequence Tagging

HMM:

$$\mathbf{y}^* = \underset{\mathbf{y}\in\mathcal{Y}}{\operatorname{argmax}}\, P(y_0)P(x_0|y_0)\prod_{i=1}^{n-1} P(y_i|y_{i-1})P(x_i|y_i)$$



As an ILP:

Inference Variables

$$\text{maximize} \sum_{y\in\mathcal{Y}} \lambda_{0,y} 1_{\{y_0=y\}} + \sum_{i=1}^{n-1}\sum_{y\in\mathcal{Y}}\sum_{y'\in\mathcal{Y}} \lambda_{i,y,y'} 1_{\{y_i=y \wedge y_{i-1}=y'\}}$$

$$\text{subject to}$$

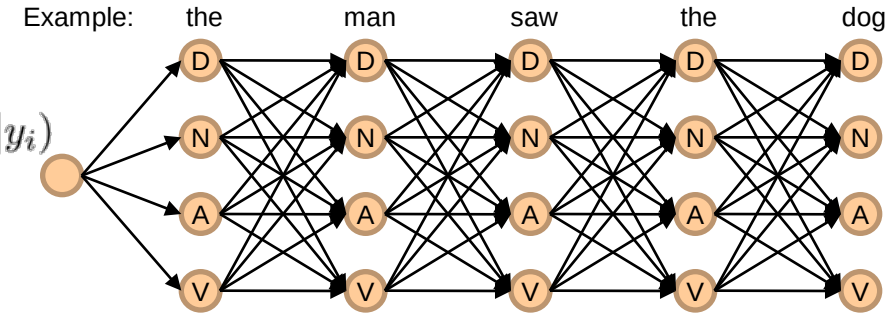
Learned Parameters

$$\lambda_{0,y} = \log(P(y)) + \log(P(x_0|y))$$
$$\lambda_{i,y,y'} = \log(P(y|y')) + \log(P(x_i|y))$$

Example 2: Sequence Tagging

HMM:

$$\mathbf{y}^* = \underset{\mathbf{y}\in\mathcal{Y}}{\operatorname{argmax}}\, P(y_0)P(x_0|y_0)\prod_{i=1}^{n-1} P(y_i|y_{i-1})P(x_i|y_i)$$



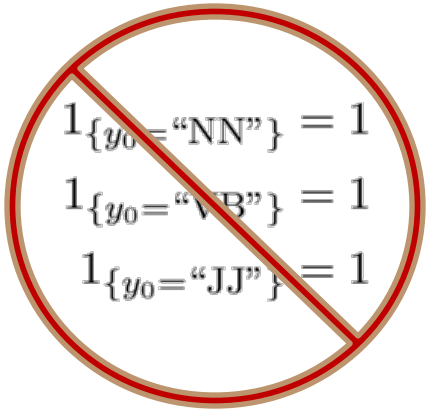
As an ILP:

$$\text{maximize} \quad \sum_{y\in\mathcal{Y}} \lambda_{0,y} 1_{\{y_0=y\}} + \sum_{i=1}^{n-1}\sum_{y\in\mathcal{Y}}\sum_{y'\in\mathcal{Y}} \lambda_{i,y,y'} 1_{\{y_i=y \,\wedge\, y_{i-1}=y'\}}$$

$$\lambda_{0,y} = \log(P(y)) + \log(P(x_0|y))$$
$$\lambda_{i,y,y'} = \log(P(y|y')) + \log(P(x_i|y))$$

$$\text{subject to} \quad \sum_{y\in\mathcal{Y}} 1_{\{y_0=y\}} = 1$$

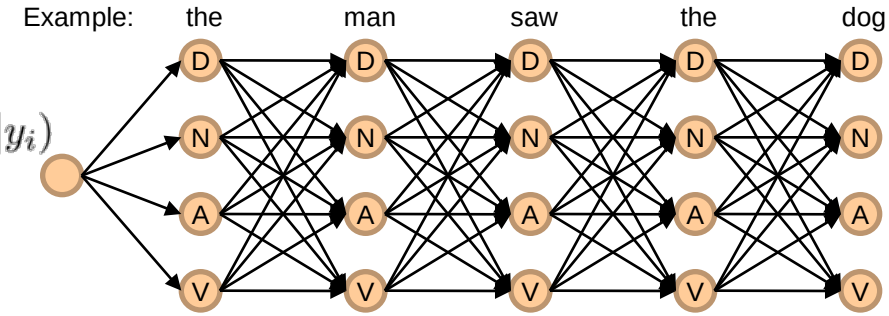
Unique label for each word



Example 2: Sequence Tagging

HMM :

$$\mathbf{y}^* = \underset{\mathbf{y}\in\mathcal{Y}}{\operatorname{argmax}} P(y_0)P(x_0|y_0)\prod_{i=1}^{n-1} P(y_i|y_{i-1})P(x_i|y_i)$$



As an ILP:

$$\text{maximize} \quad \sum_{y\in\mathcal{Y}} \lambda_{0,y} 1_{\{y_0=y\}} + \sum_{i=1}^{n-1}\sum_{y\in\mathcal{Y}}\sum_{y'\in\mathcal{Y}} \lambda_{i,y,y'} 1_{\{y_i=y \wedge y_{i-1}=y'\}}$$

$$\lambda_{0,y} = \log(P(y)) + \log(P(x_0|y))$$
$$\lambda_{i,y,y'} = \log(P(y|y')) + \log(P(x_i|y))$$

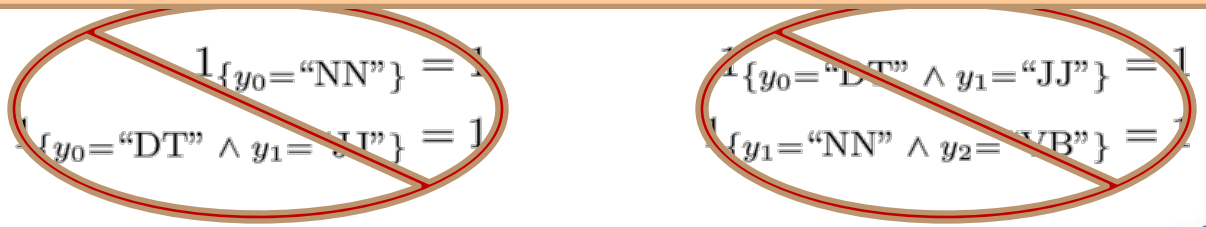
$$\text{subject to} \quad \sum_{y\in\mathcal{Y}} 1_{\{y_0=y\}} = 1$$

Unique label for each word

$$\forall y, \quad 1_{\{y_0=y\}} = \sum_{y'\in\mathcal{Y}} 1_{\{y_0=y \wedge y_1=y'\}}$$

$$\forall y, i>1 \quad \sum_{y'\in\mathcal{Y}} 1_{\{y_{i-1}=y' \wedge y_i=y\}} = \sum_{y''\in\mathcal{Y}} 1_{\{y_i=y \wedge y_{i+1}=y''\}}$$

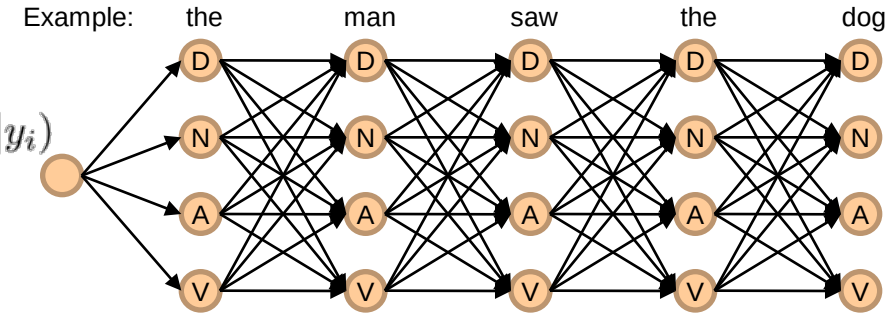
Edges that are chosen must form a path



Example 2: Sequence Tagging

HMM :

$$\mathbf{y}^* = \underset{\mathbf{y}\in\mathcal{Y}}{\operatorname{argmax}} P(y_0)P(x_0|y_0)\prod_{i=1}^{n-1} P(y_i|y_{i-1})P(x_i|y_i)$$



As an ILP:

$$\text{maximize} \quad \sum_{y\in\mathcal{Y}} \lambda_{0,y} 1_{\{y_0=y\}} + \sum_{i=1}^{n-1}\sum_{y\in\mathcal{Y}}\sum_{y'\in\mathcal{Y}} \lambda_{i,y,y'} 1_{\{y_i=y \wedge y_{i-1}=y'\}}$$

$$\lambda_{0,y} = \log(P(y)) + \log(P(x_0|y))$$
$$\lambda_{i,y,y'} = \log(P(y|y')) + \log(P(x_i|y))$$

$$\text{subject to} \quad \sum_{y\in\mathcal{Y}} 1_{\{y_0=y\}} = 1$$

Unique label for each word

$$\forall y, \quad 1_{\{y_0=y\}} = \sum_{y'\in\mathcal{Y}} 1_{\{y_0=y \wedge y_1=y'\}}$$

$$\forall y, i>1 \quad \sum_{y'\in\mathcal{Y}} 1_{\{y_{i-1}=y' \wedge y_i=y\}} = \sum_{y''\in\mathcal{Y}} 1_{\{y_i=y \wedge y_{i+1}=y''\}}$$

Edges that are chosen must form a path

$$1_{\{y_0=\text{“V”}\}} + \sum_{i=1}^{n-1}\sum_{y\in\mathcal{Y}} 1_{\{y_{i-1}=y \wedge y_i=\text{“V”}\}} \geq 1$$

There must be a verb!

Constraints

- We have seen three different constraints in this example
  1. Unique label for each word
  2. Chosen edges must form a path
  3. There must be a verb
- All three can be expressed as linear inequalities

A conventional model

- In terms of modeling, there is a difference
  - The first two define the output structure (in this case, a sequence)
  - The third one adds knowledge to the problem

In CCMs, knowledge is an integral part of the modeling

CCM Examples

- Many works in NLP make use of constrained conditional models, implicitly or explicitly.
- Next we describe three examples in detail.

- Example 1: Semantic Role Labeling
  - The use of inference with constraints to improve semantic parsing
- Example 2: Sequence Tagging
  - Adding long range constraints to a simple model
- Example 3: Sentence Compression
  - Simple language model with constraints outperforms complex models

Example 3: Sentence Compression (Clarke & Lapata)

He became a power player in Greek Politics in 1974, when he founded the socialist Pasok Party.

He became a player in politics.

We took these troubled youth who don't have fathers, and brought them into the room to Dads who don't have their children.

We took these youth and brought them into the room to Dads.

Example

Trigram Objective Function

$$\max \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} \gamma_{ijk} \cdot P(x_k | x_i, x_j)$$

Example:

0	1	2	3	4	5	6	7	8
Big	fish	eat	small	fish	in	a	small	pond
Big	fish				in	a		pond

$$\delta_0 = \delta_1 = \delta_5 = \delta_6 = \delta_8 = 1$$

$$\gamma_{015} = \gamma_{156} = \gamma_{568} = 1$$

Language model-based compression

Trigram Objective Function

$$\max \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} \gamma_{ijk} \cdot P(x_k | x_i, x_j)$$

Decision Variables

$$\delta_i = \begin{cases} 1 & \text{if } x_i \text{ is in the compression} \\ 0 & \text{otherwise} \end{cases} \quad (1 \leq i \leq n)$$

Auxiliary Variables

$$\gamma_{ijk} = \begin{cases} 1 & \text{if word sequence } x_i, x_j, x_k \text{ is in the compression} \\ 0 & \text{otherwise} \end{cases}$$

Example: Summarization

Trigram Objective Function

$$\max \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} \gamma_{ijk} \cdot P(x_k | x_i, x_j)$$

This formulation requires some additional constraints

Big fish eat small fish in a small pond

No selection of decision variables can make these trigrams appear consecutively in output.

We skip these constraints here.

Trigram model in action

He became a power player in Greek Politics in 1974, when he founded the socialist Pasok Party.

He became a player in the Pasok.

We took these troubled youth who don't have fathers, and brought them into the room to Dads who don't have their children.

We don't have, and don't have children.

Modifier Constraints

Modifier Constraints

- Ensure the relationships between head words and their modifiers remain grammatical.
- If a modifier is in the compression, its head word must be included:

$$\delta_{head} - \delta_{modifer} \geq 0$$

- Do not drop *not* if the head word is in the compression (same for words like *his*, *our* and genitives).

Example

He became a power player in Greek Politics in 1974, when he founded the socialist Pasok Party.

He became a player in the Pasok.

We took these troubled youth who don't have fathers, and brought them into the room to Dads who don't have their children.

We don't have, and don't have children.

Example

He became a power player in Greek Politics in 1974, when he founded the socialist Pasok Party.

He became a player in the Pasok Party.

We took these troubled youth who don't have fathers, and brought them into the room to Dads who don't have their children.

We don't have them don't have their children.

Sentential Constraints

Sentential Constraints

- Take the overall sentence structure into account.
- If a verb is in the compression then so are its arguments, and vice-versa:

$$\delta_{subject/object} - \delta_{verb} = 0$$

- The compression must contain at least one verb.

Example

He became a power player in Greek Politics in 1974, when he founded the socialist Pasok Party.

He became a player in the Pasok Party.

We took these troubled youth who don't have fathers, and brought them into the room to Dads who don't have their children.

We don't have them don't have their children.

Example

He became a power player in Greek Politics in 1974, when he founded the socialist Pasok Party.

He became a player in politics.

We took these troubled youth who don't have fathers, and brought them into the room to Dads who don't have their children.

We took these youth and brought them into the room to Dads.

More constraints

Discourse Constraints

- Preserve the discourse flow of the original document.
- Focus on local discourse.
- Retain personal pronouns.
  $\delta_{pronoun} = 1$
- Centering constraint over adjacent sentences.
  $\delta_{center} = 1$
- Lexical chains constraint on nouns in prevalent chains.
  $\delta_{topical} = 1$

Sentence Compression: Posing the Problem

Learned Parameters

Inference Variables

$$\text{maximize} \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} \lambda_{k,i,j}\ \gamma_{i,j,k}$$

If the three corresponding auxiliary variables are on, the inference variable must be on.

$$\text{subject to}$$

$$\forall i, j, k,\ 0 \leq i < j < k \leq n,$$

$$3\gamma_{i,j,k} \leq \delta_i + \delta_j + \delta_k$$

$$2 + \gamma_{i,j,k} \geq \delta_i + \delta_j + \delta_k$$

$$(k - i - 2)\gamma_{i,j,k} + \sum_{s=i+1}^{j-1} \delta_s + \sum_{s=j+1}^{k-1} \delta_s \leq k - i - 2$$

If the inference variable is on, no intermediate auxiliary variables may be on.

The tutorial web page has good notes on how to convert Boolean constraints to linear inequalities. .

Other examples: Coreference Resolution

- K. Chang et. al 2011. Inference Protocols for Coreference Resolution
  - Also Denis and Baldridge, 2009

Example

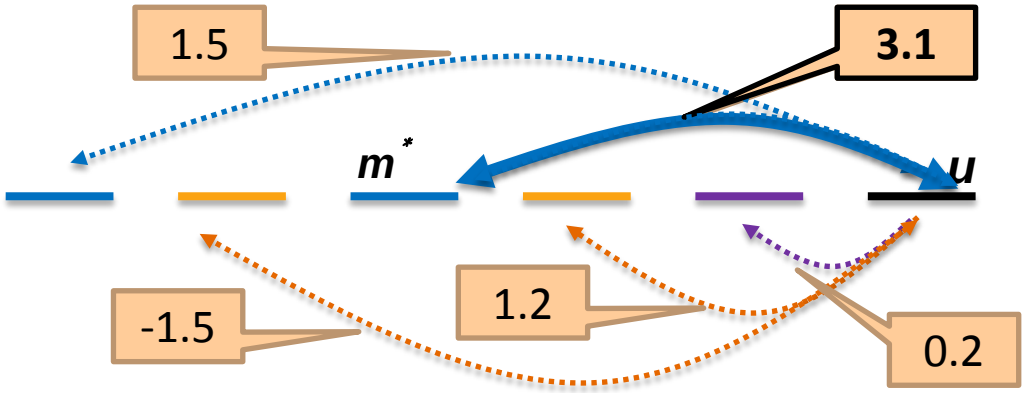
Clinton told National Public Radio that his answers to questions about Lewinsky were constrained by Starr's investigation. NPR reporter Mara Liasson asked Clinton "whether you had any conversations with her about her testimony, had any conversations at all."

Input: a set of pairwise mention scores over a document

Output: globally consistent cliques representing entities

Best-Link Inference

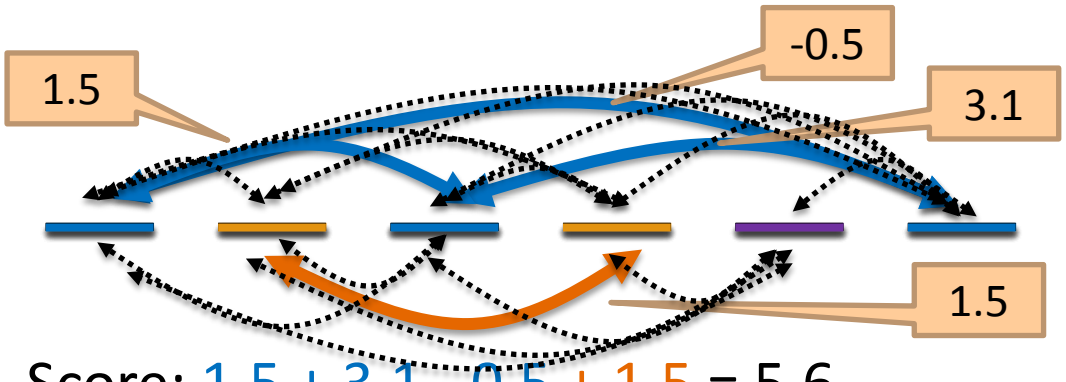
- For each mention *u*, Best-Link considers the best mention on its left to connect to
- Then, it creates a link between them if the score is above some threshold (typically 0)



- Best-Link inference is simple and effective (Bengtson and Roth, 2008)

All-Link Inference

- It scores a clustering of mentions by including **all possible pairwise links** in the score:



Score: 1.5 + 3.1 - 0.5 + 1.5 = 5.6

- McCallum and Wellner, 2003; Finley and Joachims, 2005

Integer Linear Programming (ILP) Formulation for Inference

- Best-Link

$$\arg\max_y \quad \sum_{u,v} w_{uv} y_{uv}$$

$$\text{s.t} \quad \sum_{u<v} y_{uv} \leq 1 \; \forall v,$$

$$y_{uw} \in \{0, 1\}.$$

Pairwise mention score

Binary variable

- All-Link

$$\arg\max_y \quad \sum_{u,v} w_{uv} y_{uv}$$

$$\text{s.t} \quad y_{uw} \geq y_{uv} + y_{vw} - 1 \; \forall u, w, v,$$

$$y_{uw} \in \{0, 1\}.$$

Enforce the transitivity closure of the clustering

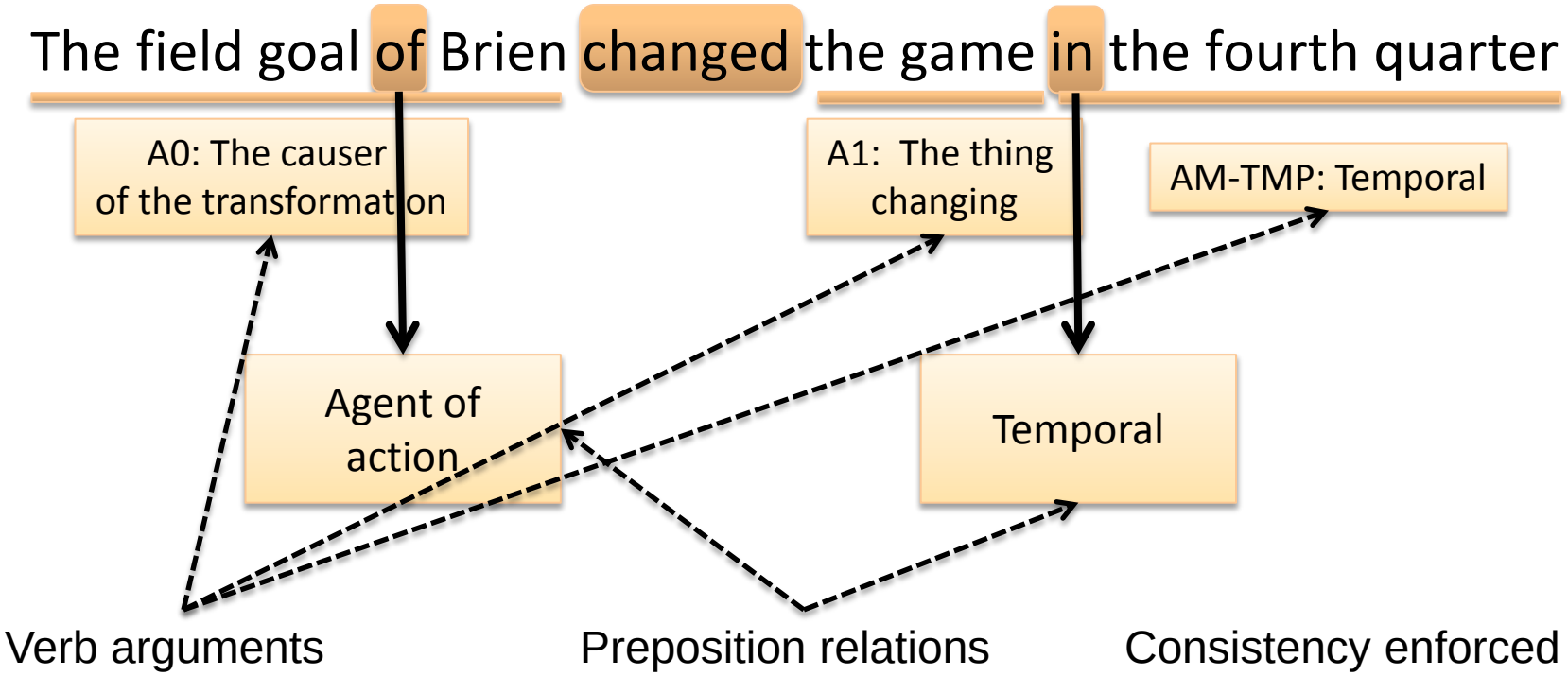
Opinion Recognition

- Y. Choi, E. Breck, and C. Cardie. Joint Extraction of Entities and Relations for Opinion Recognition EMNLP-2006

> [*Bush*]$^{(1)}$ intends$^{(1)}$ to curb the increase in harmful gas emissions and is counting on$^{(1)}$ the good will$^{(2)}$ of [*US industrialists*]$^{(2)}$.

- Semantic parsing variation:
  - Agent=entity
  - Relation=opinion
- Constraints:
  - An agent can have at most two opinions.
  - An opinion should be linked to only one agent.
  - The usual non-overlap constraints.

Extending Semantic role labeling



V. Srikumar and D. Roth. A Joint Model for Extended Semantic Role Labeling. EMNLP 2011.

Temporal Ordering

- N. Chambers and D. Jurafsky. Jointly Combining Implicit Constraints Improves Temporal Ordering. EMNLP-2008.

*Trustcorp Inc. will become(**e1**) Society Bank & Trust when its merger(**e3**) is completed(**e4**) with Society Corp. of Cleveland, the bank said(**e5**). Society Corp., which is also a bank, agreed(**e6**) in June(**t15**) to buy(**e8**) Trustcorp for 12.4 million shares of stock with a market value of about $450 million. The transaction(**e9**) is expected(**e10**) to close(**e2**) around year end(**t17**).*

Temporal Ordering

- N. Chambers and D. Jurafsky. Jointly Combining Implicit Constraints Improves Temporal Ordering. EMNLP-2008.

*Trustcorp Inc. will become(**e1**) Society Bank & Trust when its merger(**e3**) is completed(**e4**) with Society Corp. of Cleveland, the bank said(**e5**). Society Corp., which is also a bank, agreed(**e6**) in June(**t15**) to buy(**e8**) Trustcorp for 12.4 million shares of stock with a market value of about $450 million. The transaction(**e9**) is expected(**e10**) to close(**e2**) around year end(**t17**).*

Three types of edges:

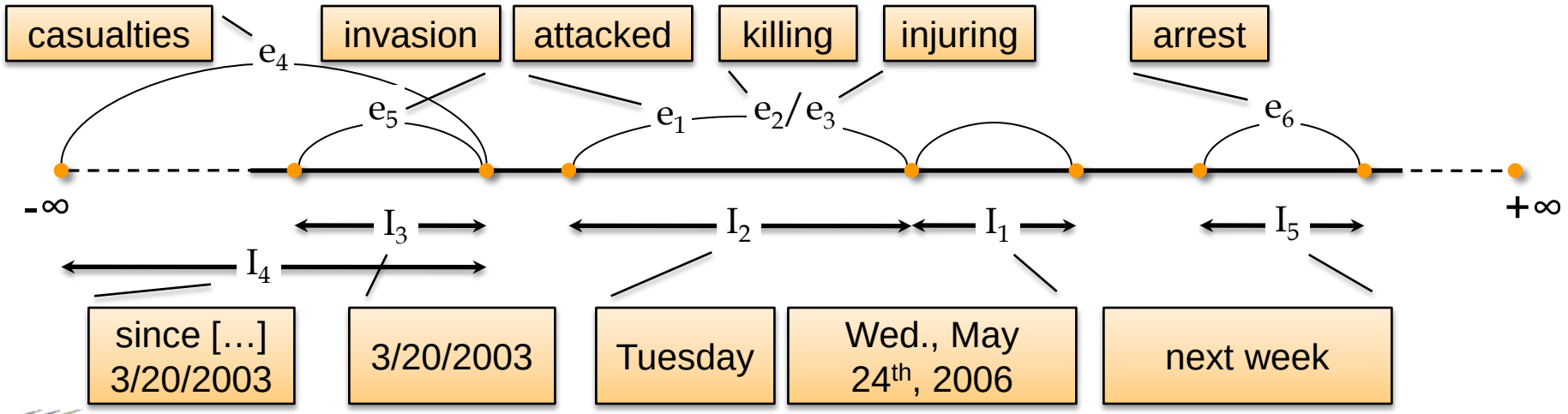
1) Annotation relations before/after
2) Transitive closure constraints
3) Time normalization constraints

Event Timeline construction

- Q. Do, L. Wei and D. Roth. Joint inference for Event Timeline Construction. EMNLP 2012

[…] The Iraqi insurgents *attacked* a police station in Tal Afar on Tuesday *killing* 6 policemen and *injuring* 8 other people. This action brings the *casualties* to over 3000 since the *invasion* of the coalition armies on 3/20/2003. Police wants to *arrest* the insurgents in a campaign next week. […]

Publishing date: Wed., May 24th, 2006



Language generation.

- R. Barzilay and M. Lapata. Aggregation via Set Partitioning for Natural Language Generation.HLT-NAACL-2006.

Passing

PLAYER	CP/AT	YDS	AVG	TD	INT
Cundiff	22/37	237	6.4	1	1
Carter	23/47	237	5.0	1	4
...	...	...	...	...	...

Rushing

PLAYER	REC	YDS	AVG	LG	TD
Hambrick	13	33	2.5	10	1
...	...	...	...	...	...

1 (*Passing* (Cundiff 22/37 237 6.4 1 1))
(*Passing* (Carter 23/47 237 5.0 1 4))
2 (*Interception* (Lindell 1 52 1))
(*Kicking* (Lindell 3/3 100 38 1/1 10))
3 (*Passing* (Bledsoe 17/34 104 3.1 0 0))
4 (*Passing* (Carter 15/32 116 3.6 1 0))
5 (*Rushing* (Hambrick 13 33 2.5 10 1))
6 (*Fumbles* (Bledsoe 2 2 0 0 0))

- Constraints:
  - Transitivity: if (e_i,e_j)were aggregated, and (e_i,e_{jk}) were too, then (e_i,e_k) get aggregated.
  - Max number of facts aggregated, max sentence length.

MT & Alignment

- Ulrich Germann, Mike Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. Fast decoding and optimal decoding for machine translation. ACL 2001.
- John DeNero and Dan Klein. The Complexity of Phrase Alignment Problems. ACL-HLT-2008.

Learning Based Java: Translating to ILP

```
constraint References(SRLSentence sentence)
{
  for (int i = 0; i < sentence.verbCount(); ++i)
  {
    ParseTreeWord verb = sentence.getVerb(i);
    LinkedList forVerb = sentence.getCandidates(verb);

    (exists (Argument a in forVerb) ArgumentTypeLearner(a) :: "R-A0")
      => (exists (Argument a in forVerb) ArgumentTypeLearner(a) :: "A0");
    (exists (Argument a in forVerb) ArgumentTypeLearner(a) :: "R-A1")
      => (exists (Argument a in forVerb) ArgumentTypeLearner(a) :: "A1");
```

- Constraint syntax based on First Order Logic
  - Declarative; interspersed within pure Java
  - Grounded in the program's Java objects
- Automatic run-time translation to linear inequalities
  - Creates auxiliary variables
  - Resulting ILP size is linear in size of propositionalization

Summary of Examples

- We have shown several different NLP solutions that use CCMs.

- In all cases, knowledge about the problem can be stated as constraints in a high level language, and then transformed into linear inequalities.

- Learning based Java (LBJ) [Rizzolo&Roth '07, '10] describes an automatic way to compile high level description of constraint into linear inequalities.

  http://cogcomp.cs.illinois.edu/page/software

PART 3: INFERENCE

This Tutorial: Constrained Conditional Models

- Part 3: **Inference Algorithms** (15 minutes)
  - Exact Algorithms
  - Relaxation methods
  - Approximate Algorithms

What is a Constrained Conditional Model?

$$\underset{y}{\operatorname{argmax}} \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

Modeling NLP problem • Variables, Features and constraints	**Objective function** • Constrained Conditional Model
Constrained optimization language • How to represent inference?	Integer linear program
Inference • How to solve it?	Several inference algorithms: Exact ILP, search, relaxation; dynamic prog.
Learning • How to learn the objective function?	Learning λ and ρ. Several learning strategies: L+I, IBT, others.

Inference strategies

1. Solving an ILP directly
2. Problem specific approaches
   - Dynamic programming
3. Relaxing constraints
   - Cutting place strategy
   - Dual decomposition and Lagrangian relaxation
   - LP relaxation
4. Approximation methods
   - Beam Search
5. Amortized inference
   - Data set optimization rather than instance based optimization

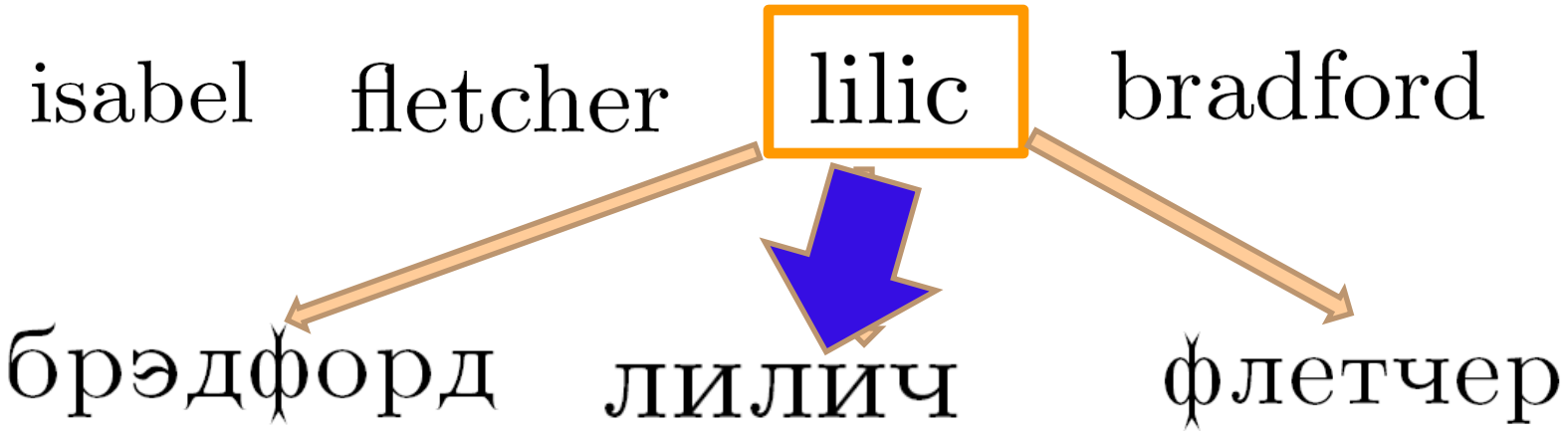
1. Inference by solving an ILP directly

- Several powerful off-the-shelf solvers available
  - Gurobi
  - Xpress-MP
  - GLPK
  - LPsolve
  - R
  - Mosek
  - CPLEX

- Most solvers have good APIs
- LBJ provides hooks for Gurobi, Xpress-MP, GLPK

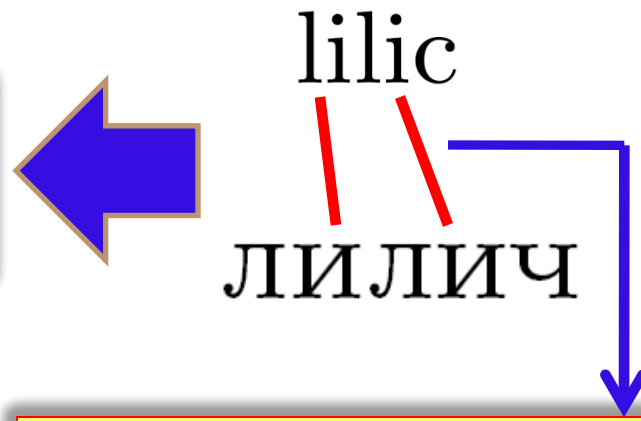
- Inference problems in NLP
  - Sometimes actually easy for ILP solvers
    - Semantic role labeling
      - 5217 instances, with an average of 146 variables and 51 constraints each takes ~13 seconds to solve
    - Entities-Relations [Roth and Yih, 2004]
      - Problem is known to be NOT **totally unimodular**
      - ILP solver still efficient!

2. Problem specific approaches



- How to get a score for the pair?
- The CCM approach:
  - Introduce an internal structure (characters)
  - Constrain character mappings to “make sense”.

Transliteration Discovery with CCM



Score = sum of the mappings' weight
s. t. mapping satisfies constraints

A weight is assigned to each edge.

Include it or not? A binary decision.

- **Natural constraints**
  - Pronunciation constraints
  - One-to-One
  - Non-crossing
  - …

- The problem now: inference
  - How to find the best mapping that satisfies the constraints?

Finding The Best Character Mappings

- **An Integer Linear Programming Problem**

Maximize the mapping score

$$\max \sum_{i \in S, j \in T} c_{ij} x_{ij}$$

$$0 \le x_{ij} \le 1, x_{ij} \in Z$$

Pronunciation constraint

$$\forall (i,j) \in B, x_{ij} = 0,$$

One-to-one constraint

$$\forall i, \sum_{j} x_{ij} = 1,$$

Non-crossing constraint

$$\forall i, j, k, m, i > k, m > j,$$

$$x_{ij} + x_{km} \le 1$$

...

- **What is the best inference algorithm?**

Finding The Best Character Mappings

A Dynamic Programming Algorithm

lilic

лилич

Weighted edit distance!

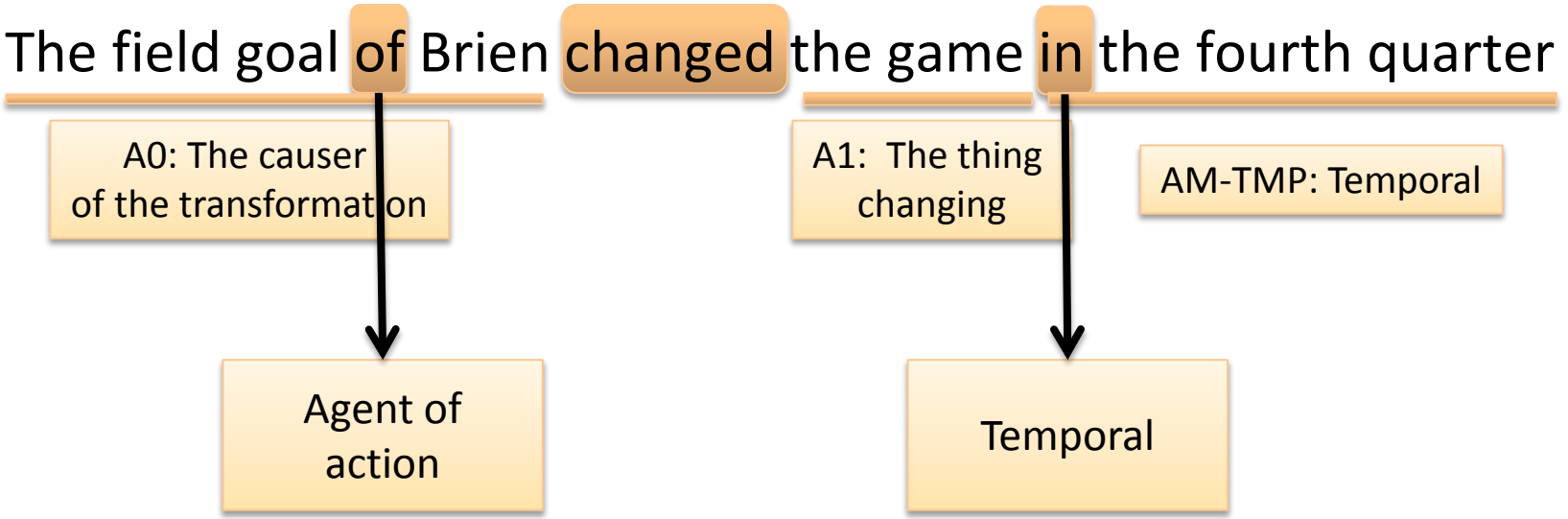
Take Home Message:
Although ILP can solve most problems, the fastest inference algorithm depends on the constraints and can be simpler

- **Exact** and fast!

3. Relaxing constraints

- Given an ILP, find a set of constraints that make the problem "hard" to solve
  - Eg. In sequence labeling, without long-range constraints, the inference is tractable. The long-range constraints make the problem difficult.
- Solve the easier problem without these constraints
  - Maybe incrementally introduce the "difficult" constraints into the problem
- Examples
  - Cutting plane approach [Riedel and Clarke, EMNLP 2006]
  - Dual decomposition/ Lagrangian relaxation [Rush and Collins. 2010, 2011, Chang and Collins 2011]
  - LP relaxation [Roth and Yih, ICML 2005]
    - Dropping the integrality constraints
    - Exact solution if the constraints are totally unimodular

Extending Semantic role labeling



Verb arguments + Preposition relations + Consistency enforced

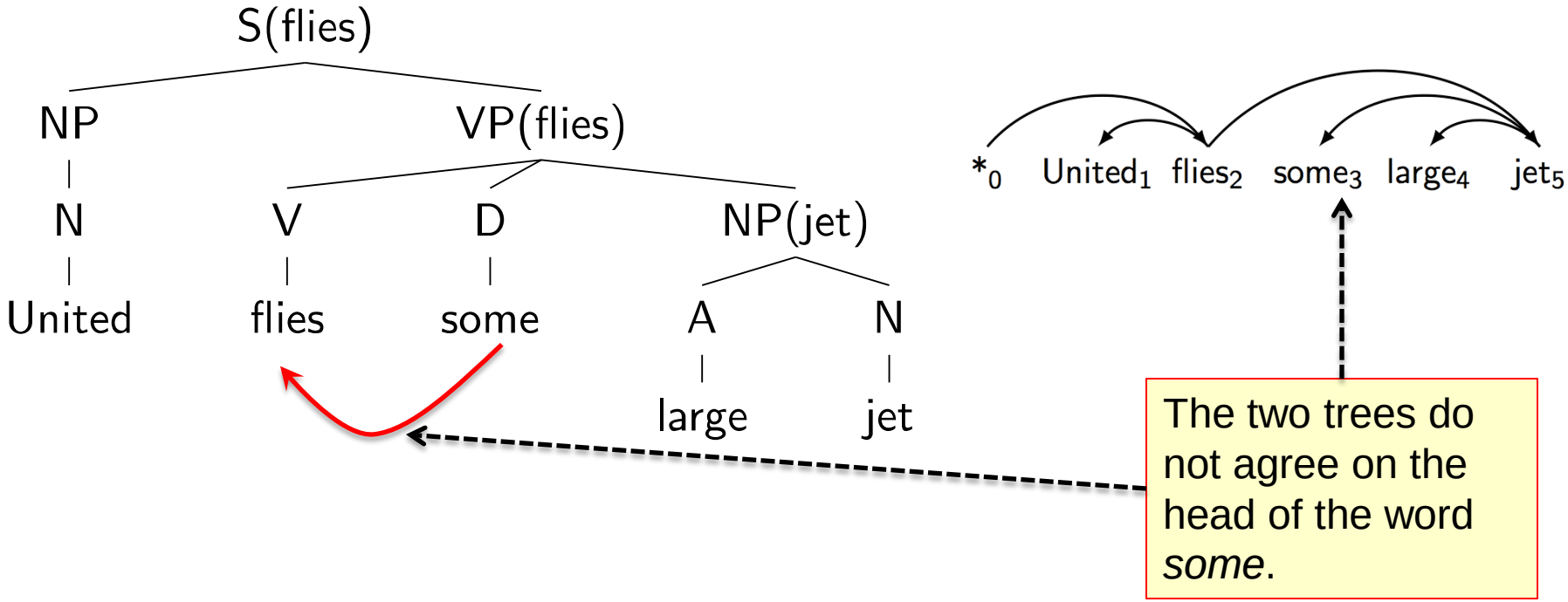
Uses cutting plane approach: Introduce ***agreement constraints*** ONLY if they are violated

V. Srikumar and D. Roth. A Joint Model for Extended Semantic Role Labeling. EMNLP 2011.

Dual Decomposition: Combining different parsers

[Rush et al, 2010]

- Combining dependency parser and constituent parser
- The parsers should agree on their dependencies



Inference iteratively removes disagreements to reach consensus

4. Approximate inference

- When ILP solver isn't fast enough, and one can resort to approximate solutions.

- Beam search
  - We will see an example next

Example: Search based Inference for SRL

- The objective function

$$\max \sum_{i,j} c_{ij} \cdot x_{ij}$$

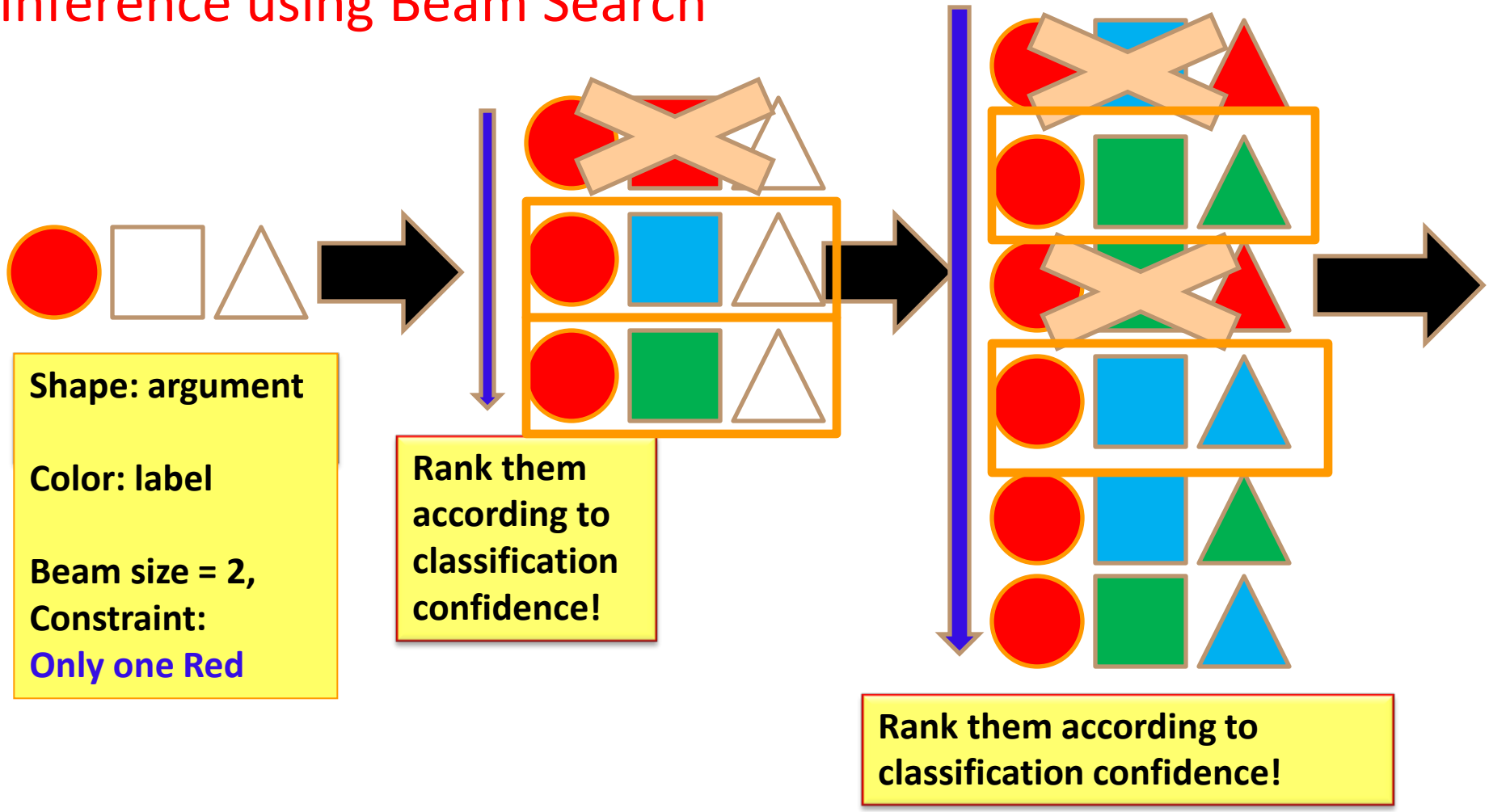
Maximize total score subject to linguistic constraints

Classification confidence

Indicator variable
assigns the j-th class for the i-th token

- **Constraints**
  - Unique labels
  - No overlapping or embedding
  - If verb is of type A, no argument of type B
  - …
- **Intuition**: check constraints' violations on **partial assignments**

Inference using Beam Search



- For each step, discard partial assignments that violate constraints!

Heuristic Inference

- Problems of heuristic inference
  - Problem 1: Possibly, sub-optimal solution
  - Problem 2: May not find a feasible solution
    - Drop some constraints, solve it again

- Using search on SRL gives comparable results to using ILP, but is much faster.

Other Inference Options

- **Amortized inference**
  - [Srikumar et. al, EMNLP 2012]
  - A way of speeding up any inference algorithm
  - Key idea:
    - Consider inference over an entire data set
    - Identify examples for which previous solutions can be re-used
    - **Speedup obtained by *not* running inference**
- **Other search algorithms**
  - ***A-star***, Hill Climbing…
  - Gibbs Sampling Inference [Finkel et. al, ACL 2005]
    - Named Entity Recognition: enforce long distance constraints
    - Can be considered as : Learning + Inference
    - One type of constraints only

Inference Methods – Summary

- Why ILP? A powerful way to **formalize** the problems
  - **However, not the only algorithmic solution**
- Heuristic inference algorithms are useful sometimes!
  - Beam search
  - Other approaches: annealing …
- Sometimes, a specific inference algorithm can be designed
  - According to your constraints

Constrained Conditional Models – 1st Part

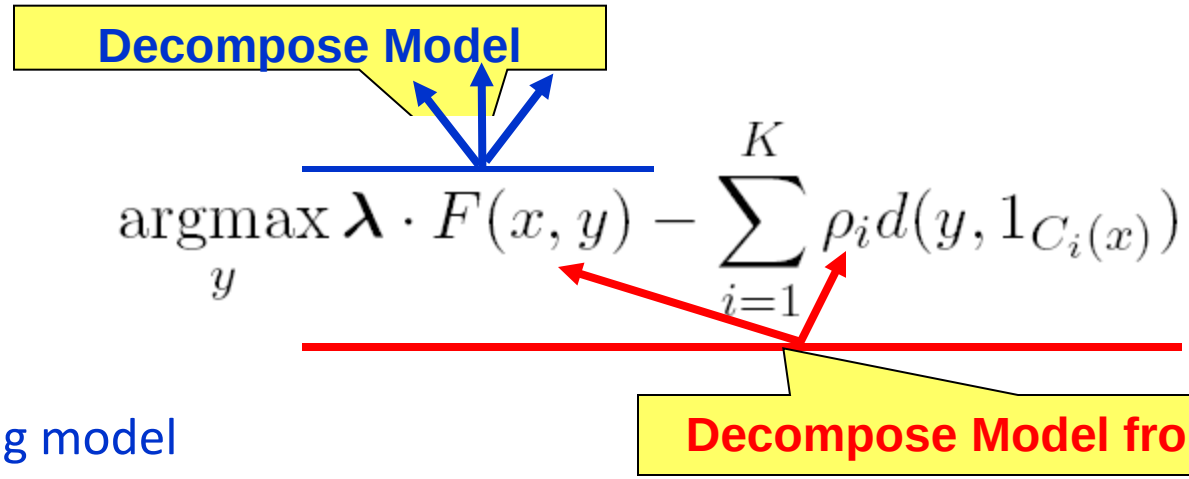
- Introduced CCMs as a formalisms that allows us to
  - Learn simpler models than we would otherwise
  - Make decisions with expressive models, augmented by declarative constraints
- Focused on modeling – posing NLP problems as ILP problems
  - 1. Sequence tagging (HMM/CRF + global constraints)
  - 2. SRL (Independent classifiers + Global Constraints)
  - 3. Sentence Compression (Language Model + Global Constraints)
- Described Inference
  - Solving inference problems, exactly & approximately
- Second half – Learning
  - Supervised setting, and supervision-lean settings

PART 4: LEARNING PARADIGMS

This Tutorial: Constrained Conditional Models (Part II)

- Part 4: Training Paradigms (20 min)
  - Learning models
    - Independently of constraints (L+I); Jointly with constraints (IBT)
    - Decomposed to simpler models

Training Constrained Conditional Models



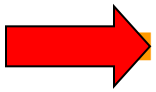
Learning model

- Independently of the constraints (L+I)
- Jointly, in the presence of the constraints (IBT)
- Decomposed to simpler models

Where are we?

Modeling & Algorithms for Incorporating Constraints

- Showed that CCMs allow for formalizing many problems
- Showed several ways to incorporate global constraints in the decision.



Training: Coupling vs. Decoupling Training and Inference.

- Incorporating global constraints is important but
- Should it be done only at evaluation time or also at training time?
- How to decompose the objective function and train in parts?
- Issues related to:
  - **Modularity, efficiency and performance, availability of training data**
  - **Problem specific considerations**

Training Constrained Conditional Models

$$\underset{y}{\operatorname{argmax}}\, \boldsymbol{\lambda} \cdot F(x,y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

Learning model

- Independently of the constraints (L+I)
- Jointly, in the presence of the constraints (IBT)

Decompose Model from constraints

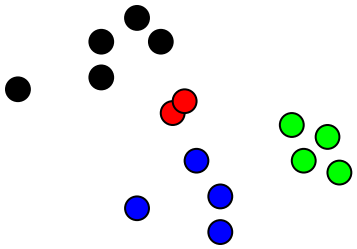
- Note that structured prediction algorithms (S-SVM, S-Perceptron, CRFs) can be used both
  - **When we decide to learn jointly (IBT)**
  - **When the left part is structured but we still want to add additional constraints at inference time**
- It is possible to train a model (left side), but make decisions with additional information (right side) that was not incorporated during learning model. (Not available, not needed, or just too expensive)

Comparing Training Methods

- Option 1: Learning + Inference (with Constraints)
  - Ignore (some) constraints during training
- Option 2: Inference (with Constraints) Based Training
  - Consider constraints during training
- In both cases: Global Decision Making with Constraints
- Question: Isn't Option 2 always better?
- Not so simple…
  - Next, the "Local model story"

Intuition: Solving Multi-Class with Binary Classifiers

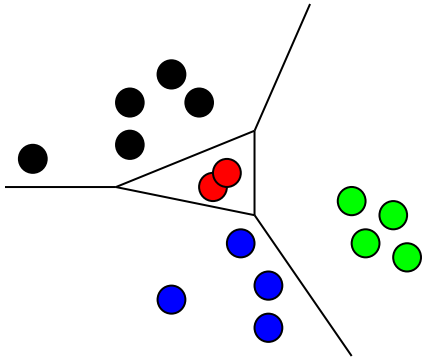
- MultiClass classifier
  - Function $f : \mathbf{R}^d \rightarrow \{1,2,3,\ldots,k\}$



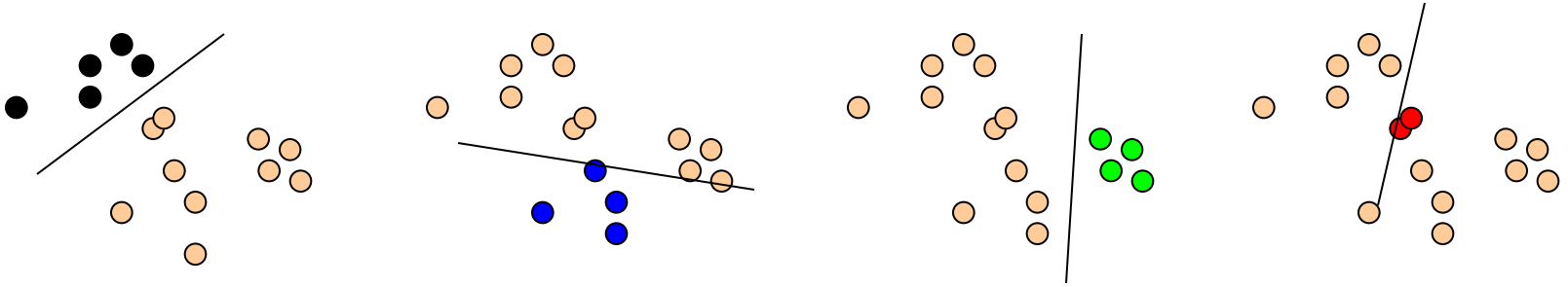
- Not always easy
- Constrained Classification:

[Har-Peled et. el 2002; Crammer et. al 2002]

Real Problem:



- But the way we typically address it is via 1-vs- all:
  - **Decompose into binary problems**
  - **It works quite well even though 1-vs-all is not expressive enough**

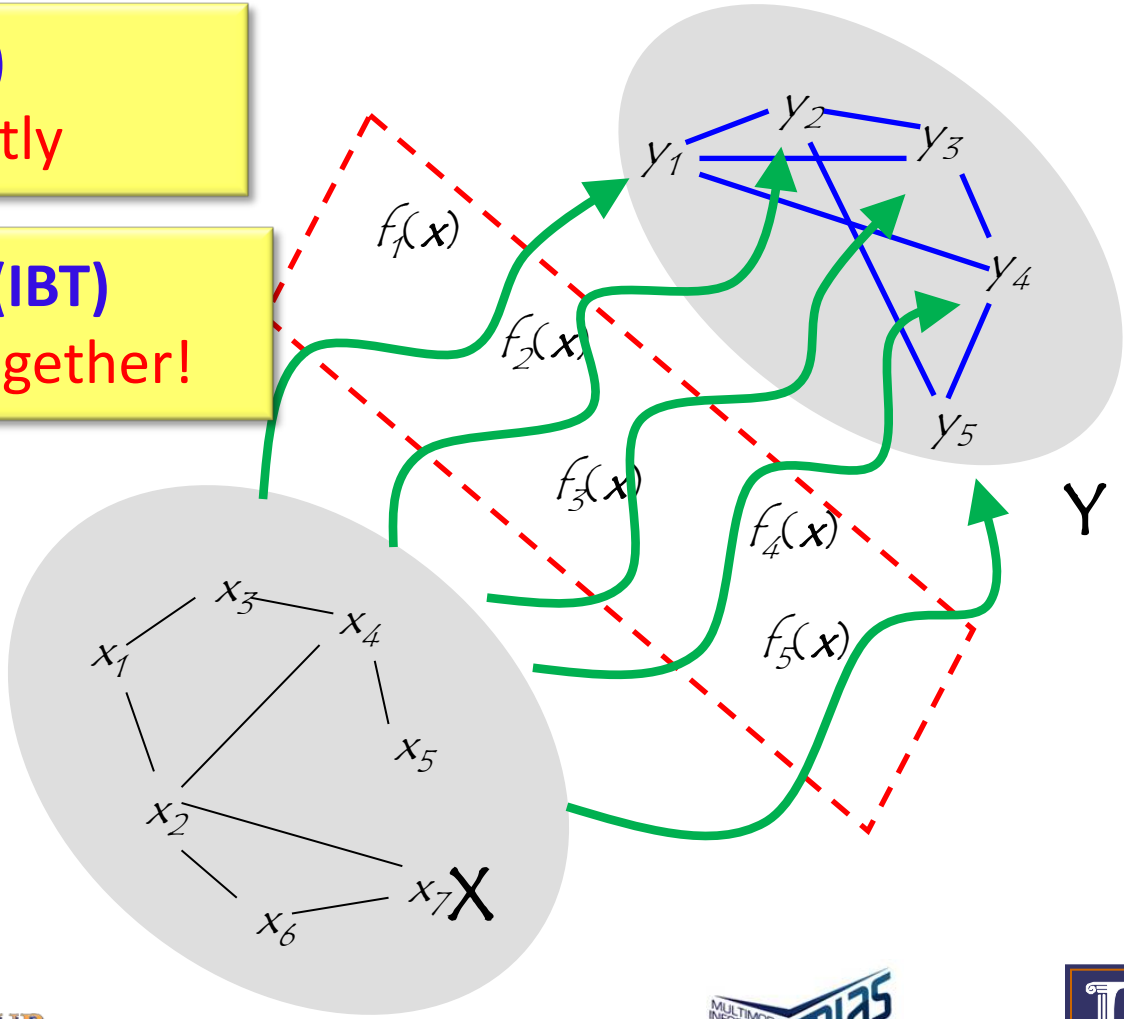


Training Methods

Learning + Inference (L+I)
Learn models independently

Inference Based Training (IBT)
Learn one model, all y's together!

Intuition: Learning with constraints may make learning more difficult

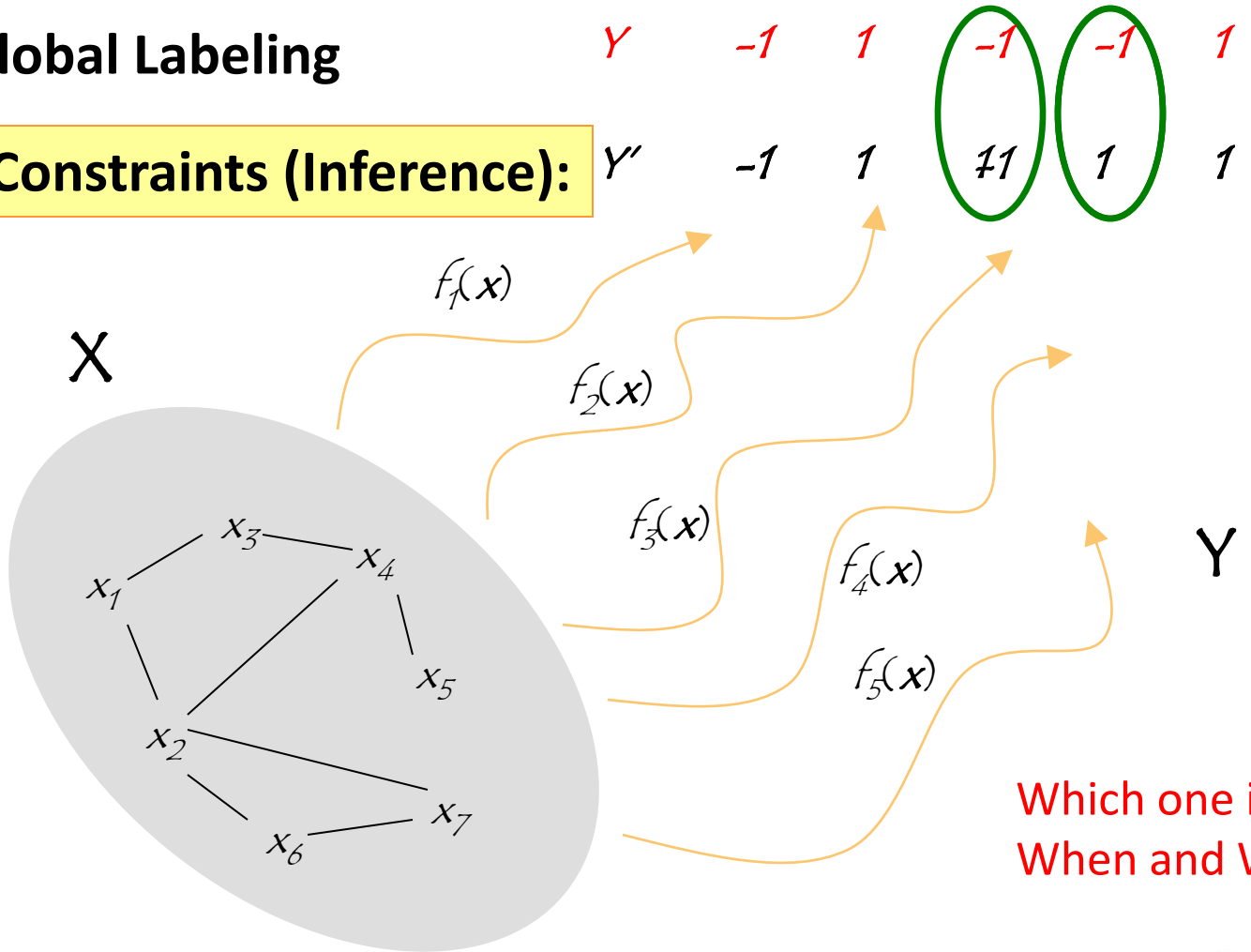


Training with Constraints

Example: Perceptron-based Global Learning: Structured Perceptron

True Global Labeling

Apply Constraints (Inference):



Which one is better?
When and Why?

L+I & IBT: General View – Structured Perceptron

- Graphics for the case: F(x,y) = F(x)

For each iteration

For each (X, $\mathbf{Y}_{GOLD}$) in the training data

$$Y_{PRED} = \operatorname*{argmax}_{y} \boldsymbol{\lambda} \cdot F(x,y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

If $\mathbf{Y}_{PRED}$!= $\mathbf{Y}_{GOLD}$

$\lambda = \lambda + F(X, \mathbf{Y}_{GOLD}) - F(X, \mathbf{Y}_{PRED})$

endif

endfor

The difference between L+I and IBT

Claims [Punyakanok et. al , IJCAI 2005]

- Theory applies to the case of local models
  - F(x,y) = F(x); applies broadly, e.g., SRL
- When the local modes are "easy" to learn, L+I outperforms IBT.
  - In many applications, the components are *identifiable* and easy to learn (e.g., argument, open-close, PER).
- Only when the local problems become difficult to solve in isolation, IBT outperforms L+I, but needs a larger number of training examples.

> L+I: cheaper computationally; modular
> IBT is better in the limit, and when there is strong interaction among y's

- Other training paradigms are possible
  - Pipeline-like Sequential Models: [Roth, Small, Titov: AI&Stat'09]
    - Identify a preferred ordering among components
    - Learn k-th model jointly with previously learned models
  - Constrained Driven Learning [Chang et. al'07,12; later]

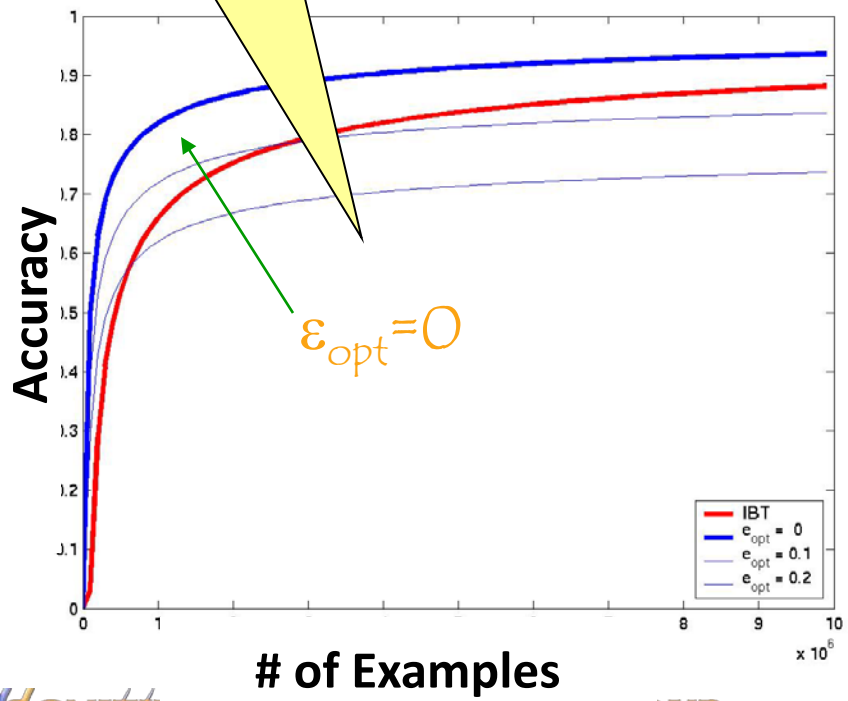
L+I vs IBT

L+I vs. IBT: the more identifiable individual problems are, the better overall performance is with L+I

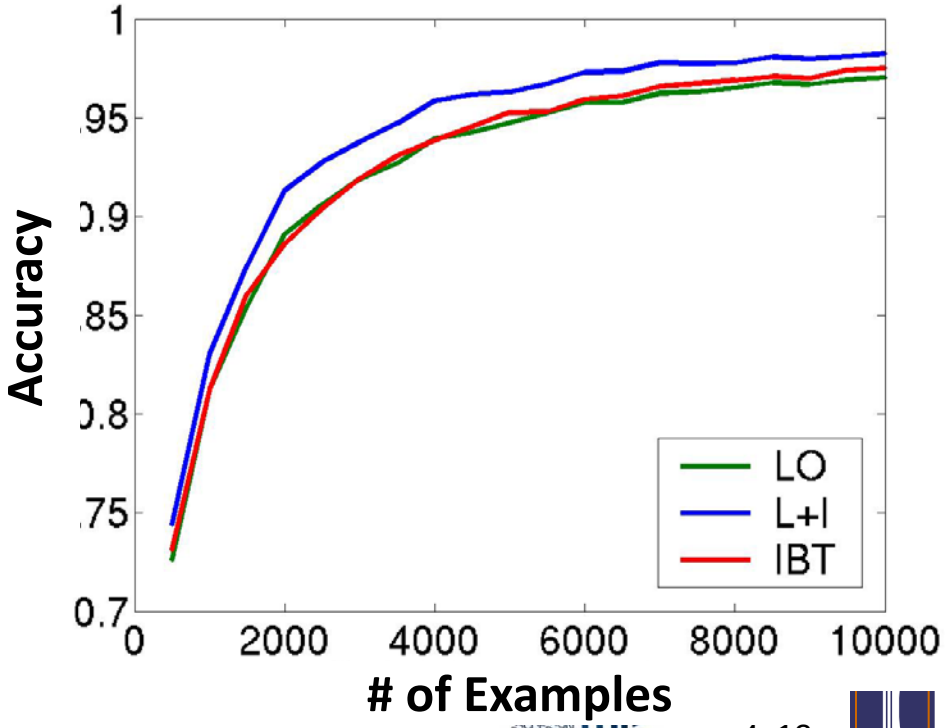
- Local $\varepsilon \le \varepsilon_{opt} + \left(\left(d \log m + \log 1/\delta \right) / m \right)^{1/2}$
- Global $\varepsilon \le 0 + \left(\left(cd \log m + c^2 d + \log 1/\delta \right) / m \right)^{1/2}$

Indication for hardness of problem

Bounds



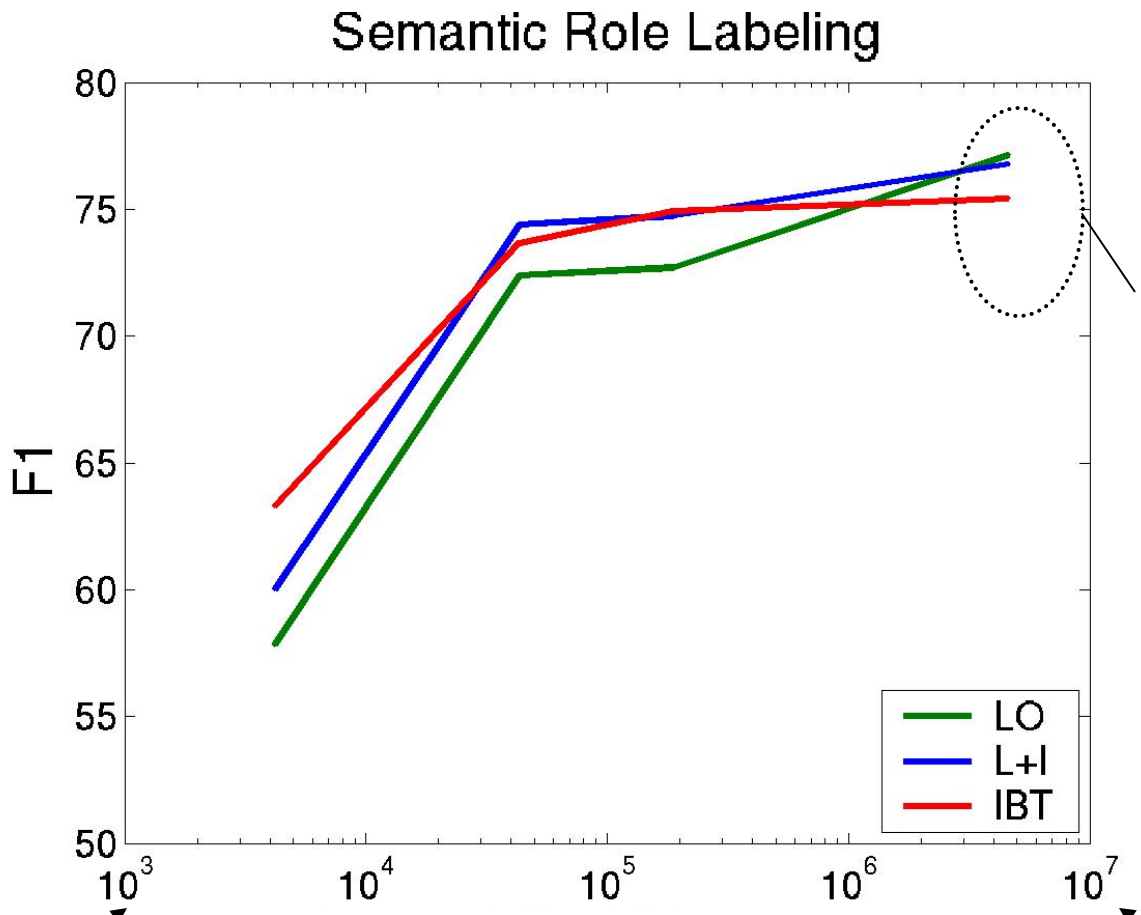
Simulated Data



Relative Merits: SRL

In some cases problems are hard due to lack of training data.

Semi-supervised learning



L+I is better.

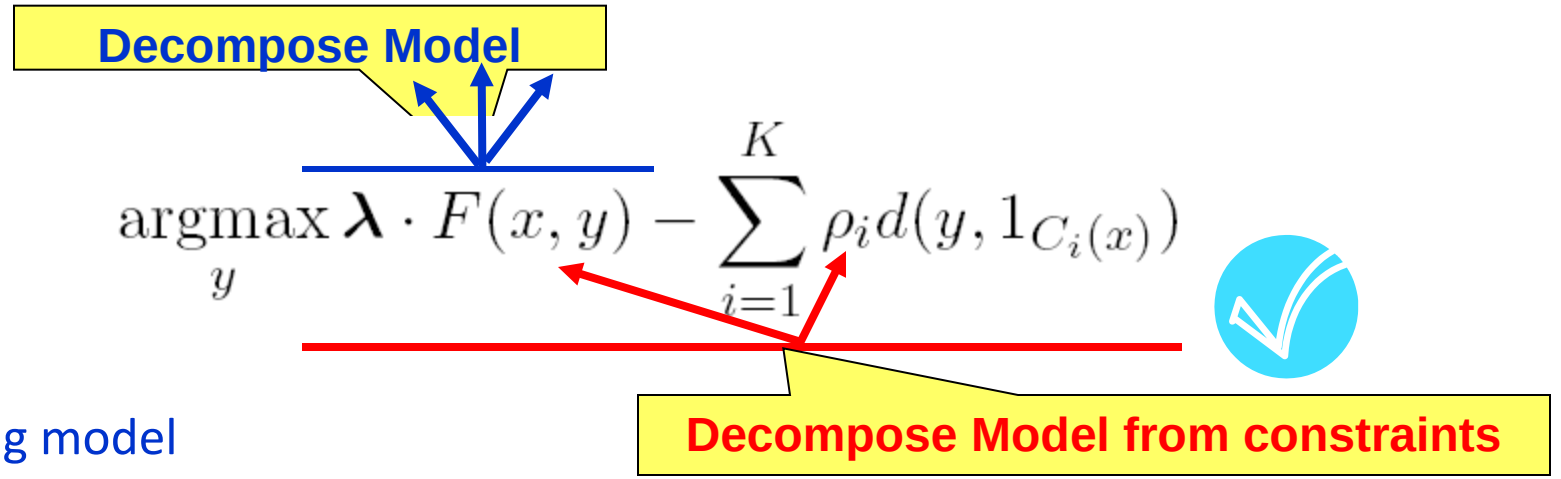
When the problem is artificially made harder, the tradeoff is clearer.

hard

Difficulty of the learning problem (# features)

easy

Training Constrained Conditional Models (II)

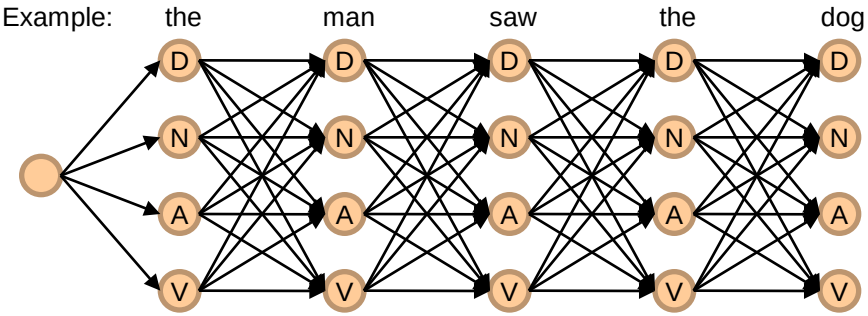


- Learning model
  - Independently of the constraints (L+I)
  - Jointly, in the presence of the constraints (IBT)
  - Decomposed to simpler models
- Local Models (trained independently) vs. Structured Models
  - In many cases, structured models might be better due to expressivity
- But, what if we use constraints?
- Local Models + Constraints vs. Structured Models + Constraints
  - Hard to tell: Constraints are expressive
  - For tractability reasons, structured models have less expressivity than what's possible with constraints; L+I could be better then, and easier to learn.

Recall: Example 1: Sequence Tagging (HMM/CRF)

HMM / CRF:

$$\mathbf{y}^* = \underset{\mathbf{y}\in\mathcal{Y}}{\operatorname{argmax}} P(y_0)P(x_0|y_0)\prod_{i=1}^{n-1} P(y_i|y_{i-1})P(x_i|y_i)$$



As an ILP:

$$\text{maximize} \quad \sum_{y\in\mathcal{Y}} \lambda_{0,y} 1_{\{y_0=y\}} + \sum_{i=1}^{n-1}\sum_{y\in\mathcal{Y}}\sum_{y'\in\mathcal{Y}} \lambda_{i,y,y'} 1_{\{y_i=y \wedge y_{i-1}=y'\}}$$

$$\lambda_{0,y} = \log(P(y)) + \log(P(x_0|y))$$
$$\lambda_{i,y,y'} = \log(P(y|y')) + \log(P(x_i|y))$$

$$\text{subject to} \quad \sum_{y\in\mathcal{Y}} 1_{\{y_0=y\}} = 1$$

Discrete predictions

$$\forall y, \quad 1_{\{y_0=y\}} = \sum_{y'\in\mathcal{Y}} 1_{\{y_0=y \wedge y_1=y'\}}$$

$$\forall y, i>1 \quad \sum_{y'\in\mathcal{Y}} 1_{\{y_{i-1}=y' \wedge y_i=y\}} = \sum_{y''\in\mathcal{Y}} 1_{\{y_i=y \wedge y_{i+1}=y''\}}$$

Feature consistency

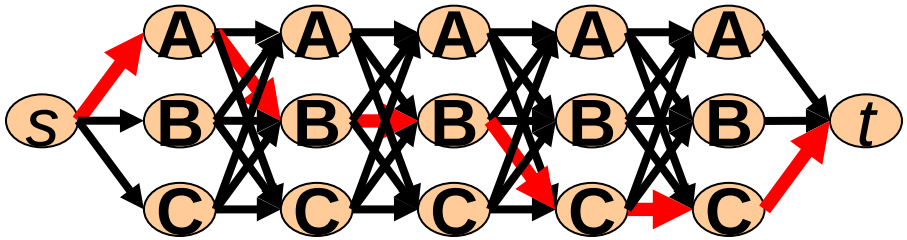
$$1_{\{y_0=\text{“V”}\}} + \sum_{i=1}^{n-1}\sum_{y\in\mathcal{Y}} 1_{\{y_{i-1}=y \wedge y_i=\text{“V”}\}} \geq 1$$

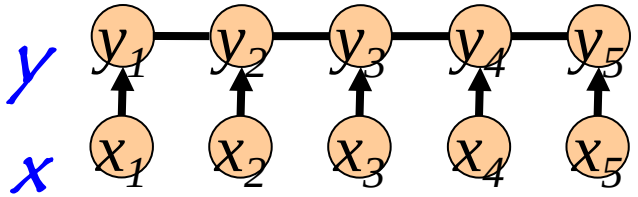
There must be a verb!

Example: CRFs are CCMs

But, you can do better

- Consider a common model for sequential inference: HMM/CRF
  - Inference in this model is done via the Viterbi Algorithm.

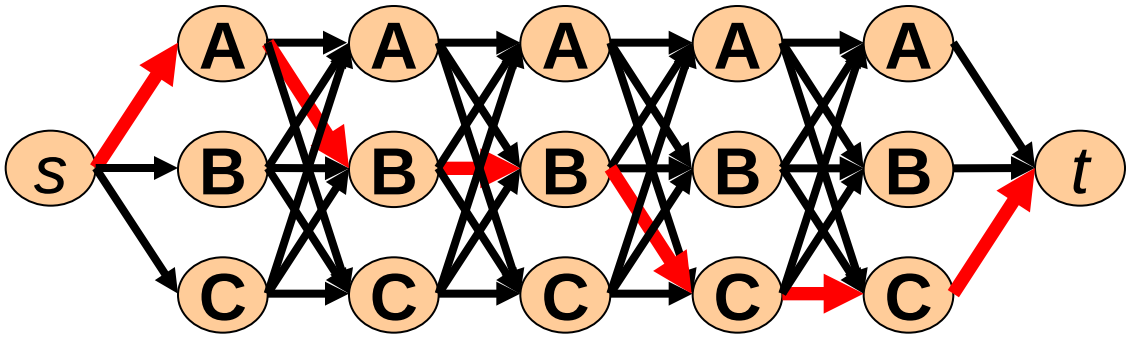




- Viterbi is a special case of the Linear Programming based Inference.
  - It is a shortest path problem, which is a LP, with a canonical matrix that is totally unimodular. Therefore, you get integrality constraints for free.
  - One can now incorporate non-sequential/expressive/declarative constraints by modifying this canonical matrix
    - No value can appear twice; a specific value must appear at least once; A→B
  - And, run the inference as an ILP inference.

Learn a rather simple model; make decisions with a more expressive model

Experiment: Semantic Role Labeling Revisited



- Sequential Models
  - Conditional Random Field
  - Global perceptron
- Training: Sentence based
- Testing: Find best global assignment (shortest path)
  - **+ with constraints**

- Local Models
  - Logistic Regression
  - Avg. Perceptron
- Training: Token based
- Testing: Find best assignment locally
  - **+ with constraints (Global)**

Which Model is Better? Semantic Role Labeling

- Experiments on SRL: [Roth and Yih, ICML 2005]
  - Story: Inject expressive Constraints into conditional random field

	Sequential Models			Local
	L+I		IBT	L+I
Model	**CRF-ML**	**CRF-D**	**CRF-IBT**	**Avg. P**

Local Models are now better than Sequential Models!
(With constraints)

Summary: Training Methods – Supervised Case

- Many choices for training a CCM
  - Learning + Inference (Training w/o constraints; add constraints later)
  - Inference based Learning (Training with constraints)
- Based on this, what kind of models should you use?
  - Decomposing models can be better that structured models
- Advantages of L+I
  - Require fewer training examples
  - More efficient; most of the time, better performance
  - Modularity; easier to incorporate already learned models.
- Bottom line: L+I is better when y-level interactions are not very strong
- Next: Soft Constraints; Supervision-lean models

PART 5: CONSTRAINTS DRIVEN LEARNING

Training Constrained Conditional Models

$$\underset{y}{\operatorname{argmax}}\, \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

- Learning constraints' weights
  - Independently of learning the model
  - Jointly, along with learning the model
- Dealing with lack of supervision
  - Constraints Driven Semi-Supervised learning (CODL)
  - Learning Constrained Latent Representations
  - Indirect Supervision

Soft Constraints

$$-\sum_{i=1}^{K} \rho_k d(y, 1_{C_i(x)})$$

- Hard Versus Soft Constraints
  - Hard constraints: Fixed Penalty $\rho_i = \infty$
  - Soft constraints: Need to set the penalty
- Why soft constraints?
  - Some constraint violations are more serious than others
  - An example can violate multiple constraints, multiple times!
  - Sometime we cannot make a prediction that violates no constraint
  - Degree of violation is only meaningful when constraints are soft!

Information extraction without Prior Knowledge

Lars Ole Andersen . Program analysis and specialization for the C Programming language. PhD thesis. DIKU , University of Copenhagen, May 1994 .

Prediction result of a trained HMM

[AUTHOR]	Lars Ole Andersen . Program analysis and
[TITLE]	specialization for the
[EDITOR]	C
[BOOKTITLE]	Programming language
[TECH-REPORT]	. PhD thesis .
[INSTITUTION]	DIKU , University of Copenhagen , May
[DATE]	1994 .

Violates lots of natural constraints!

Examples of Constraints

- Each field must be a consecutive list of words and can appear at most once in a citation.

- State transitions must occur on punctuation marks.

- The citation can only start with *AUTHOR* or *EDITOR*.

- The words *pp., pages* correspond to *PAGE*.
- Four digits starting with 20xx and 19xx are *DATE*.
- Quotations can appear only in *TITLE*
- …….

Easy to express pieces of "knowledge"

Non Propositional; May use Quantifiers

Degree of Violations

One possibility: Count how many times the assignment y violates a constraint

$$d(y, 1_{C(x)}) = \sum_{j=1}^{T} \phi_C(y_j)$$

$$\phi_C(y_j) = \begin{cases} \text{1 - if assigning } y_i \text{ to } x_i \text{ violates the constraint C with respect to assignment } (x_1,..,x_{i-1};y_1,...,y_{i-1}) \\ \text{0 - otherwise} \end{cases}$$

State transition must occur on punctuations.

$\forall i, y_{i-1} \neq y_i \Rightarrow x_{i-1}$ is a punctuation

Lars	Ole	Andersen	.
AUTH	BOOK	EDITOR	EDITOR
$\Phi_c(y_1)=0$	$\Phi_c(y_2)=1$	$\Phi_c(y_3)=1$	$\Phi_c(y_4)=0$

$\sum \Phi_c(y_j) = 2$

Reason for using degree of violation

- An assignment might violate a constraint multiple times
- Allows us to chose a solution with fewer constraint violations

Lars	Ole	Andersen	.
AUTH	AUTH	EDITOR	EDITOR
$\Phi_c(y_1)=0$	$\Phi_c(y_2)=0$	$\Phi_c(y_3)=1$	$\Phi_c(y_4)=0$

Lars	Ole	Andersen	.
AUTH	BOOK	EDITOR	EDITOR
$\Phi_c(y_1)=0$	$\Phi_c(y_2)=1$	$\Phi_c(y_3)=1$	$\Phi_c(y_4)=0$

The first one is better because of $d(y,1_{c(X)})$!

Learning the constraints' weights

$$\lambda \cdot F(x, y) - \sum_{i=1}^{K} \rho_k d(y, 1_{C_i(x)})$$

- **Strategy 1: Independently of learning the model**
  - Handle the learning parameters λ and the penalty ρ separately
  - Learn a feature model and a constraint model
  - Similar to L+I, but also learn the penalty weights
  - Keep the model simple

- **Strategy 2: Jointly, along with learning the model**
  - Handle the learning parameters λ and the penalty ρ together
  - Treat soft constraints as high order features
  - Similar to IBT, but also learn the penalty weights

Strategy 1: Independently of learning the model

- Model: (First order) Hidden Markov Model P_λ (x,y)
- Constraints: long distance constraints
  - The i-th the constraint:
  - The probability that the i-th constraint C_i is violated $P(C_i = 1)$
- Assumption: Product of Experts

- The learning problem
  - Given labeled data, estimate λ and $P(C_i = 1)$
  - For one labeled example,

$$\text{SCORE}(x, y) = \text{HMM Probability} \times \text{Constraint Violation Score}$$

  - Training: Maximize the score of all labeled examples!

Strategy 1: Independently of learning the model (cont.)

$$\text{SCORE}(x, y) = \text{HMM Probability} \times \text{Constraint Violation Score}$$

- The new scoring function is a CCM!
  - Setting $\rho_i = -\log \frac{P(C_i=1)}{P(C_i=0)}$
  - New score:

$$\log \text{SCORE}(x, y) = \lambda \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)}) + c$$

- Maximize this new scoring function on labeled data
  - Learn a HMM separately
  - Estimate $P(C_i=1)$ separately by counting how many times the constraint is violated by the training data!
- The product of experts assumption justifies optimizing the model and the constraints' weights separately

Strategy 2: Jointly, along with learning the model

- The problem is now a standard structured learning problem
  - Structured perceptron, Structured SVM
  - Need to supply the inference algorithm: $\max_y w^T \phi(x, y)$
  - For example, Structured SVM

$$\min_w \frac{\|w\|^2}{2} + C \sum_{i=1}^{l} L_S(x_i, y_i, w),$$

  - The function $L_S(x,s,w)$ measures the distance between gold label and the inference result for this example!
- Simple solution for Joint parameter learning
  - Add constraints directly into the inference problem
  - w=[λ ρ], ϕ(x,y) contains both features and constraint violations
- It's also possible to learn the joint weight vector with CRF:
  - Sum problem in inference during training [cannot use ILP]

Summary: Learning constraints' weights

- The need for soft constraints
  - Constraints can be violated by gold data; some are more important
  - Want to have degree of violation
  - Experimental Evidence: Domain Specific – soft constraints help for the Citation & Advertisement Domain
- Learning constraints' weights
  - Independent approach: fix the model
    - **Learn constraints weights by counting violations**
  - Joint approach
    - **Treat constraints as long distance/abstract features**
    - **Structured learning problem: can use Sum or Max (easier)**
  - Experimental evidence: Joint learning of constraints improves only with enough training data.
- See details in: [Chang, Ratinov, Roth, Machine Learning Journal 2012]

Training Constrained Conditional Models

$$\underset{y}{\operatorname{argmax}}\, \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

- Learning constraints' weights
  - Independently of learning the model
  - Jointly, along with learning the model
- Dealing with lack of supervision
  - Constraints Driven Semi-Supervised learning (CODL)
  - Learning Constrained Latent Representations
  - Indirect Supervision

Dealing with lack of supervision

- Goal of this tutorial: learning structured models
- Learning structured models requires annotating structures.
  - Very expensive process

- Constraints can easy the burden in two ways:

  - Constraints can serve as a supervision source
    - **Will be discussed in the context of semi-supervised learning**
    - **Constrained EM**
  - The presence of constraints can help amplify simple forms of supervision
    - **Use binary supervision to learn structure**
    - **Indirect supervision**

Role of Constraints: Encoding Prior Knowledge

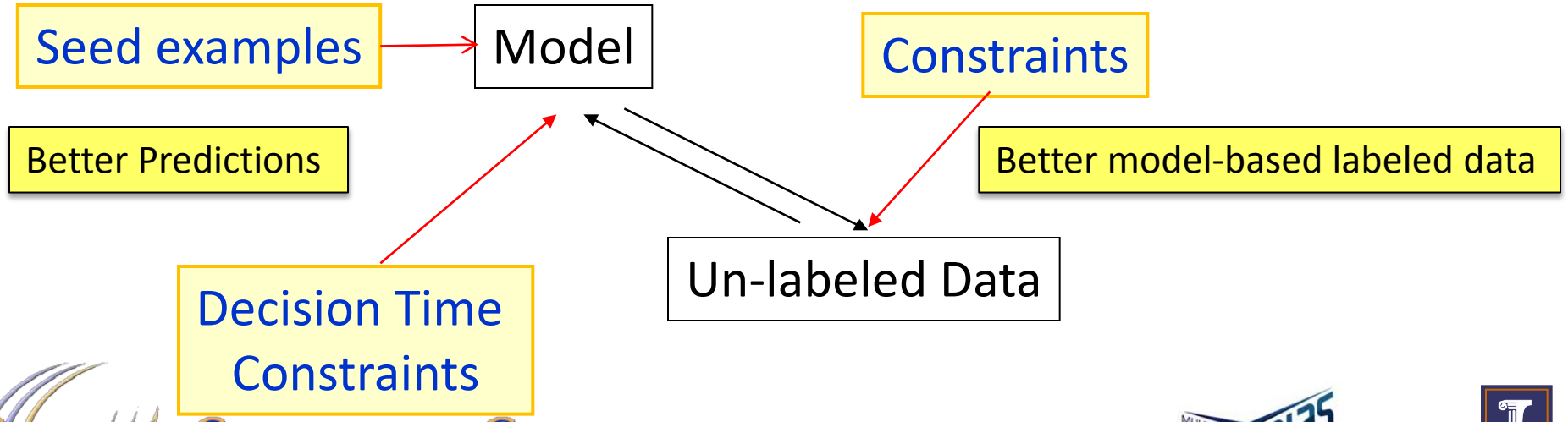
- Consider encoding the knowledge that:
  - Entities of type A and B cannot occur simultaneously in a sentence
- The "Feature" Way
  - Requires larger models
  - Needs more training data
- The Constraints Way
  - Tells the model what it should attend to
  - Keep the model simple; add expressive constraints directly
  - A small set of constraints
  - Allows for decision time incorporation of constraints

A form of supervision

We can use constraints to replace training data

Guiding (Semi-Supervised) Learning with Constraints

- In traditional Semi-Supervised learning the model can drift away from the correct one.
- Constraints can be used to generate better training data
  - At training to improve labeling of un-labeled data (and thus improve the model)
  - At decision time, to bias the objective function towards favoring constraint satisfaction.



Constraints Driven Learning (CoDL)

[Chang, Ratinov, Roth, ACL'07;ICML'08,MLJ'12]

(w_0,ρ_0)=learn(L)

Supervised learning algorithm parameterized by (w,ρ). Learning can be justified as an optimization procedure for an objective function

For N iterations do

T=ϕ

For each x in unlabeled dataset

$h \leftarrow \text{argmax}_y\ w^T\ \phi(x,y) - \sum \rho_k\ d_C(x,y)$

$T=T \cup \{(x, h)\}$

Inference with constraints: augment the training set

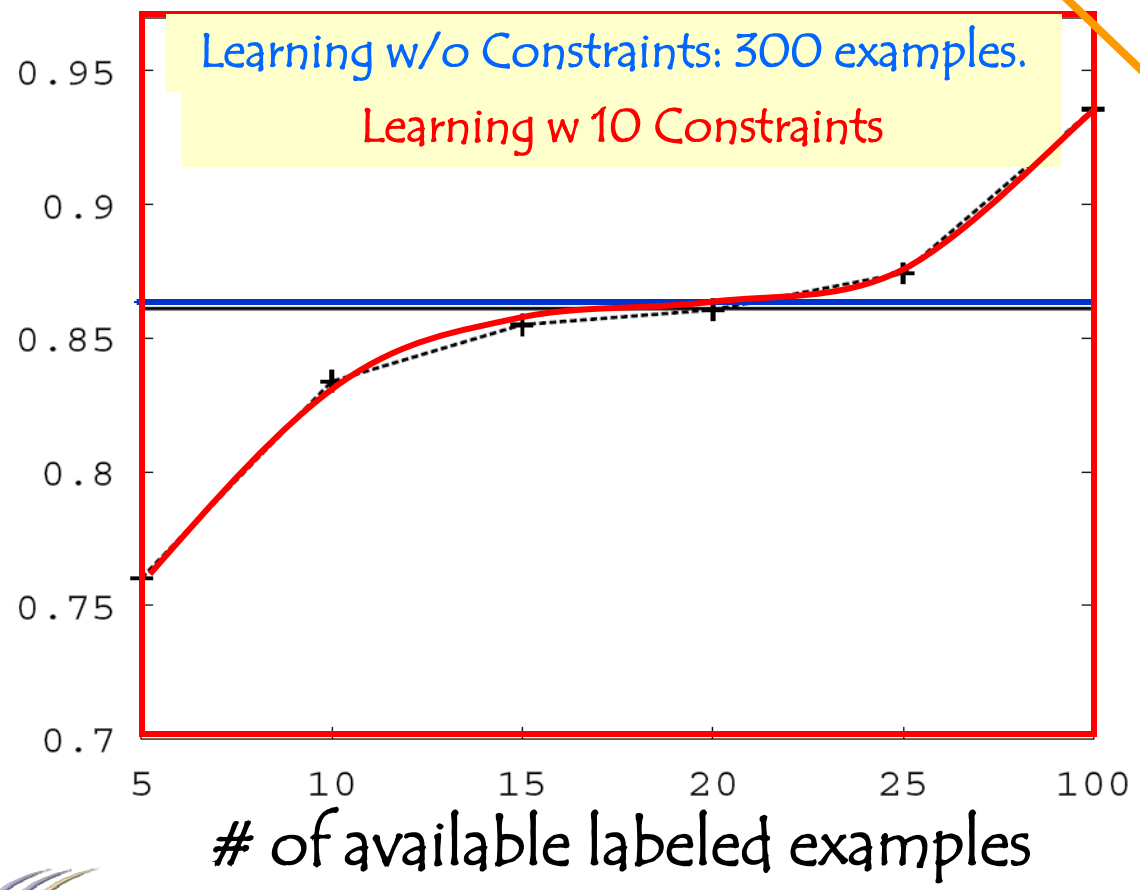
$(w,\rho) = \gamma\ (w_0,\rho_0) + (1- \gamma)$ learn(T)

Learn from new training data Weigh supervised & unsupervised models.

Excellent Experimental Results showing the advantages of using constraints, especially with small amounts on labeled data [Chang et. al, Others]

Value of Constraints in Semi-Supervised Learning

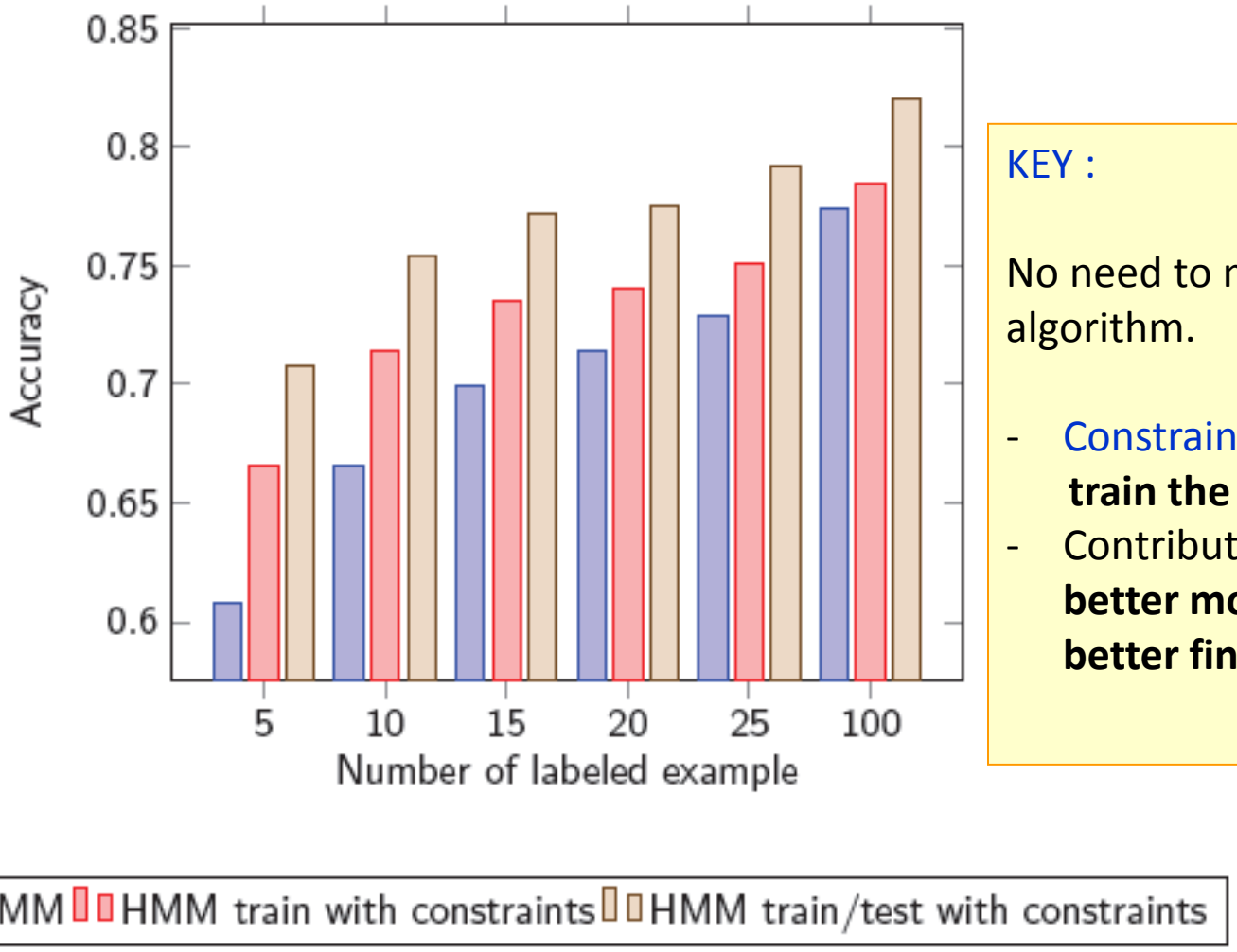
Objective function: $f_{\Phi,C}(\mathbf{x},\mathbf{y}) = \sum w_i \phi_i(\mathbf{x},\mathbf{y}) - \sum \rho_i d_{C_i}(\mathbf{x},\mathbf{y}).$



Constraints are used to Bootstrap a semi-supervised learner

Poor model + constraints used to annotate unlabeled data, which in turn is used to keep training the model.

Train and Test With Constraints



KEY :

No need to modify the HMM algorithm.

- Constraints are used to **train the model**
- Contribute both to a **better model** and to **better final predictions**.

CoDL as Constrained Hard EM

- Hard EM is a popular variant of EM
- While EM estimates a distribution over all y variables in the E-step,
- … Hard EM predicts the best output in the E-step

$$y^* = \mathbf{argmax}_{\mathrm{y}}\ P(\mathrm{y} \mid \mathrm{x}, \mathrm{w})$$

- Alternatively, hard EM predicts a peaked distribution

$$q(y) = \delta_{y=y^*}$$

- Constrained-Driven Learning (CODL) – can be viewed as a constrained version of hard EM:

Constraining the feasible set

$$y^* = \mathbf{argmax}_{\mathrm{y:Uy} \leq \mathrm{b}}\ P_w(y \mid x)$$

Constrained EM: Two Versions

- While Constrained-Driven Learning [CODL; Chang et al, 07] is a constrained version of hard EM:

Constraining the feasible set

$$y^* = \text{argmax}_{\mathbf{y}:\mathbf{U}\mathbf{y}\leq \mathbf{b}} P_w(y|x)$$

- … It is possible to derive a constrained version of EM:
- To do that, constraints are relaxed into expectation constraints on the posterior probability q:

$$\mathrm{E}_q[Uy] \leq b$$

- The E-step now becomes:

$$q' = \underset{q:q(\mathbf{y})\geq 0, E_q[\mathbf{U}\mathbf{y}]\leq \mathbf{b}, \sum_{\mathbf{y}} q(\mathbf{y})=1}{\arg\min} KL(q(\mathbf{y})||P(\mathbf{y}|\mathbf{x},\mathbf{w}))$$

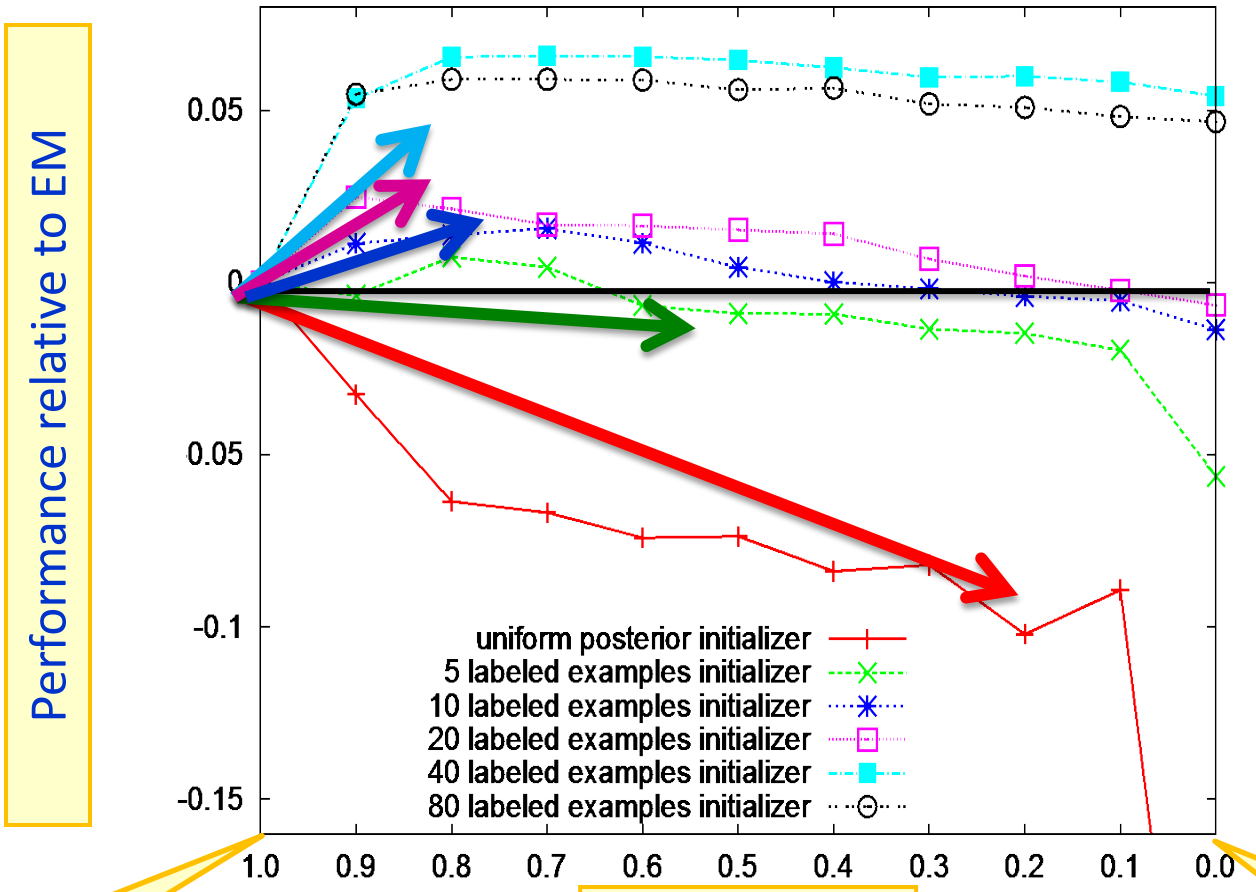
- This is the Posterior Regularization model [PR; Ganchev et al, 10]

Which (Constrained) EM to use?

- There is a lot of literature on EM vs hard EM
  - Experimentally, the bottom line is that with a good enough (???) initialization points , hard EM is probably better (and more efficient).
    - **E.g., EM vs hard EM (Spitkovsky et al, 10)**
- Similar issues exist in the constrained case: CoDL vs. PR
- New – Unified EM (UEM)
  - [Samdani et. al., Talk this Wednesday, ML-II session]
  - UEM is a family of EM algorithms,
    - Parameterized by a single parameter γ that
    - Provides a continuum of algorithms – from EM to hard EM, and infinitely many new EM algorithms in between.
    - Implementation wise, not more complicated than EM

Unsupervised POS tagging: Different EM instantiations

- Measure percentage accuracy relative to EM



Initialization with 40-80 examples

Initialization with 20 examples

Initialization with 10 examples

Initialization with 5 examples

Uniform Initialization

Summary: Constraints as Supervision

- Introducing domain knowledge-based constraints can help guiding semi-supervised learning
  - E.g. “the sentence must have at least one verb”, “a field y appears once in a citation”
- Constrained Driven Learning (CoDL) : Constrained hard EM
- PR: Constrained soft EM
- UEM : Beyond “hard” and “soft”

- Related literature:
  - **Unified EM (Samdani et al 2012: Talk on Wednesday)**
  - Constraint-driven Learning (Chang et al, 07),
  - Posterior Regularization (Ganchev et al, 10),
  - Generalized Expectation Criterion (Mann & McCallum, 08),
  - Learning from Measurements (Liang et al, 09)

Training Constrained Conditional Models

$$\underset{y}{\operatorname{argmax}}\, \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

- Learning constraints' penalties
  - Independently of learning the model
  - Jointly, along with learning the model
- Dealing with lack of supervision
  - Constraints Driven Semi-Supervised learning (CODL)
  - Learning Constrained Latent Representations
  - Indirect Supervision

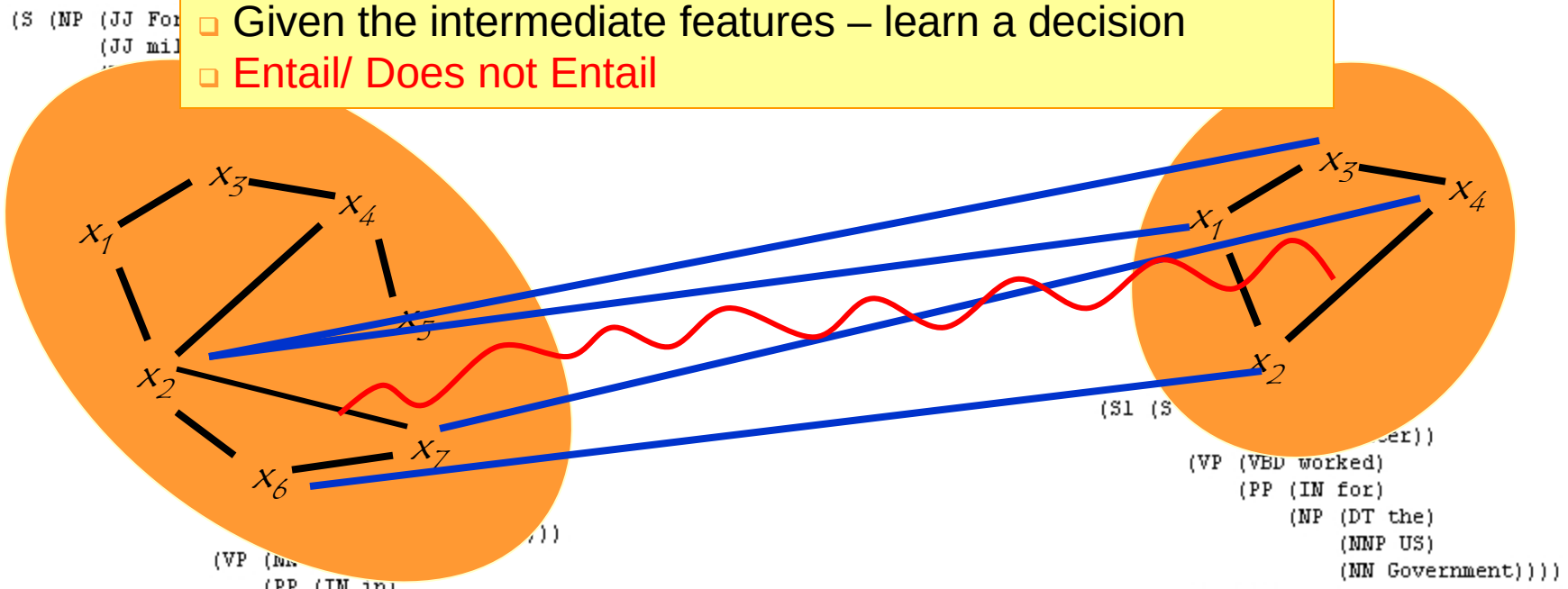
Different types of structured learning tasks

- Type 1: Structured output prediction
  - **Dependencies** between different output decisions
  - We can add constraints on the output variables
  - Examples: information extraction, parsing, pos tagging, ….
- Type 2: Binary output tasks with latent structures
  - Output: binary, but requires an intermediate representation (structure)
  - The intermediate representation is hidden
  - Examples: paraphrase identification, TE, …

Textual Entailment

Former military specialist Carpenter took the helm at FictitiousCom Inc. after t … in the Unite…

- Entailment Requires an Intermediate Representation
- Alignment based Features
- Given the intermediate features – learn a decision
- Entail/ Does not Entail



But only positive entailments are expected to have a meaningful intermediate representation

Jim Carpenter worked for the US Government.

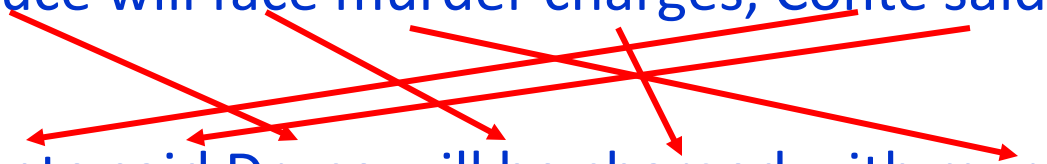
Paraphrase Identification

Given an input $x \in X$
Learn a model $f : X \rightarrow \{-1, 1\}$

- Consider the following sentences:

- S1: Druce will face murder charges, Conte said.

- S2: Conte said Druce will be charged with murder .



We need latent variables that explain:
why this is a positive example.

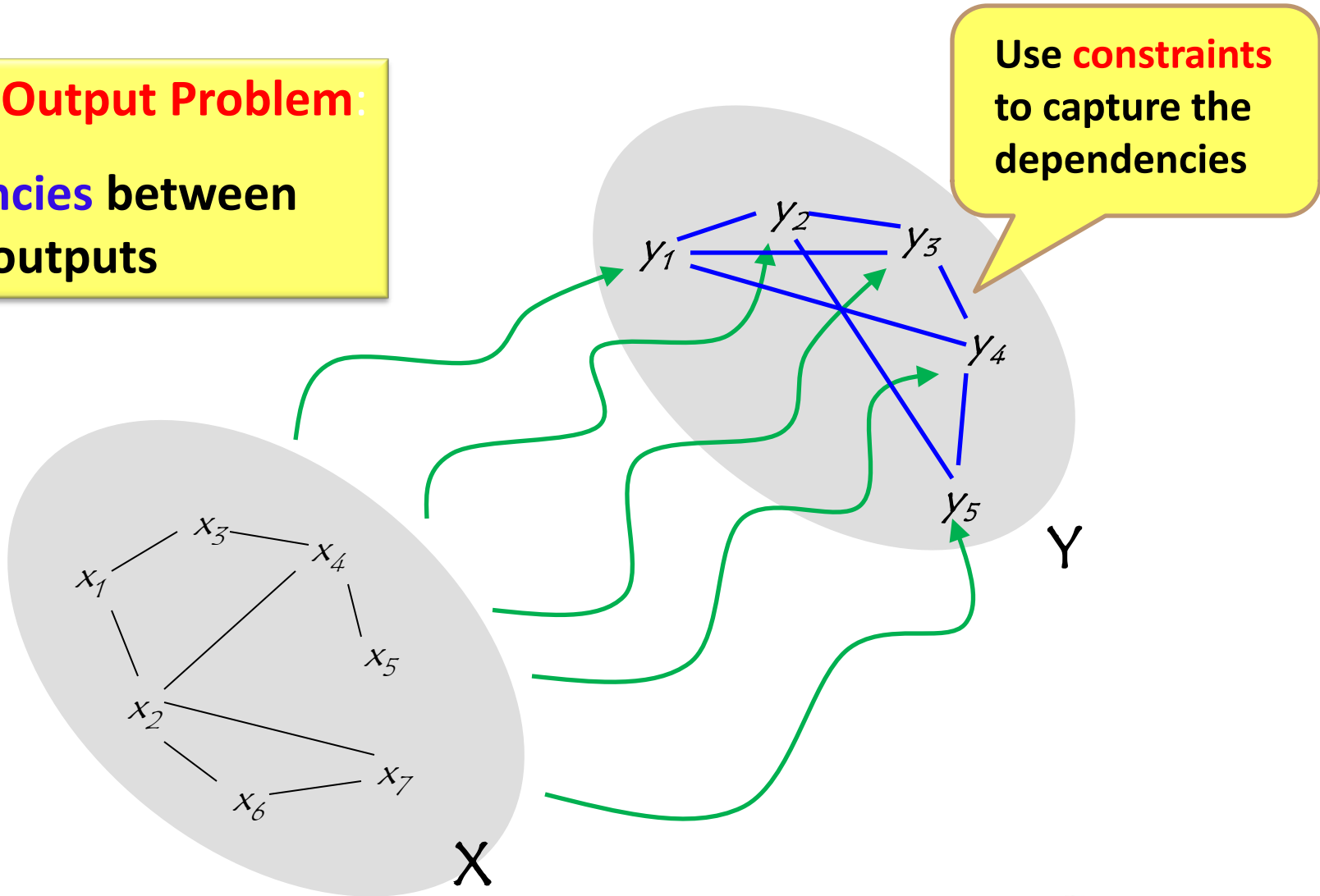
- Are S1 and S2 a paraphrase of each other?
- There is a need for an intermediate representation to justify this decision

Given an input $x \in X$
Learn a model $f : X \rightarrow H \rightarrow \{-1, 1\}$

Structured output learning

Structure Output Problem:

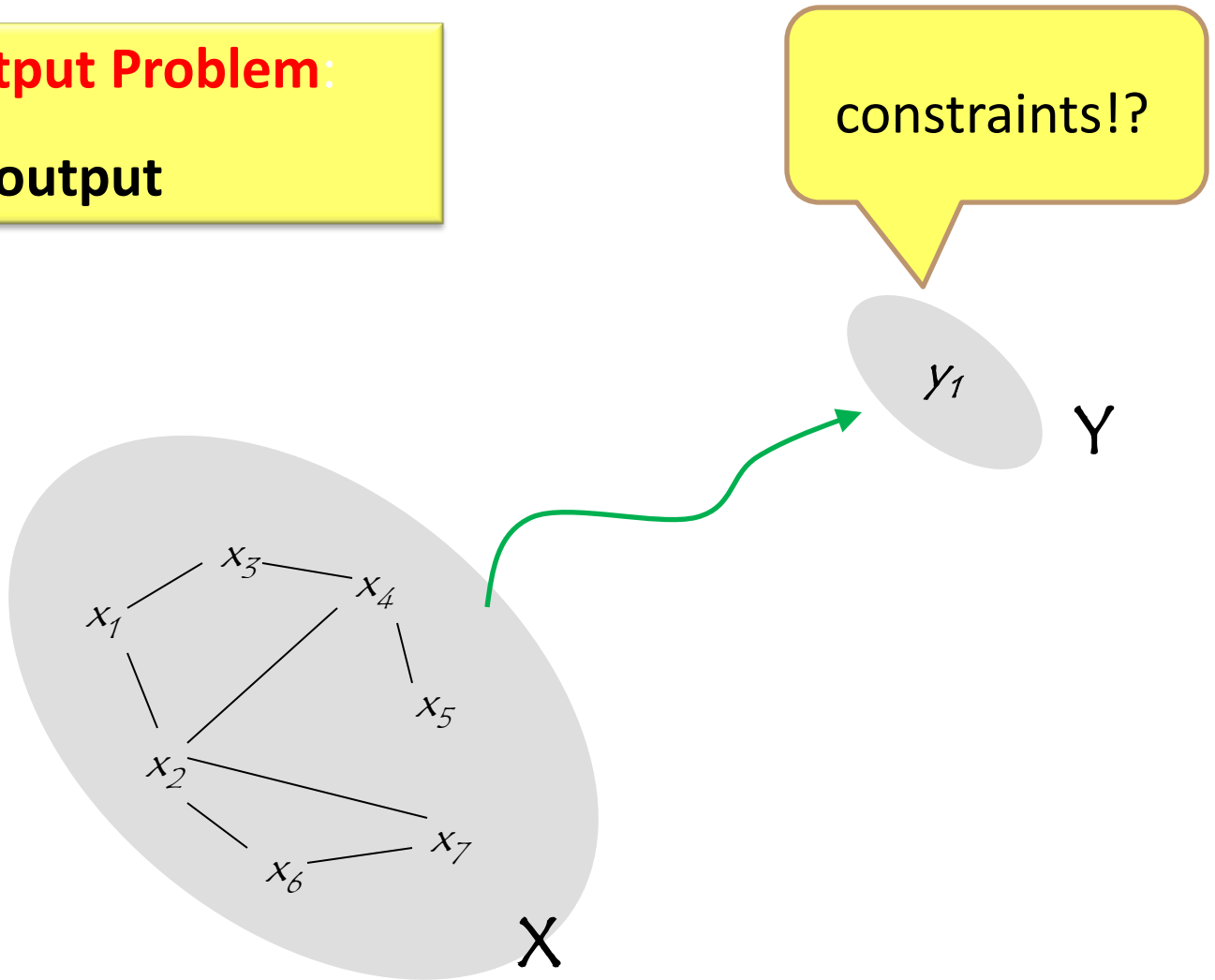
Dependencies between different outputs



Standard Binary Classification problem

Single Output Problem:

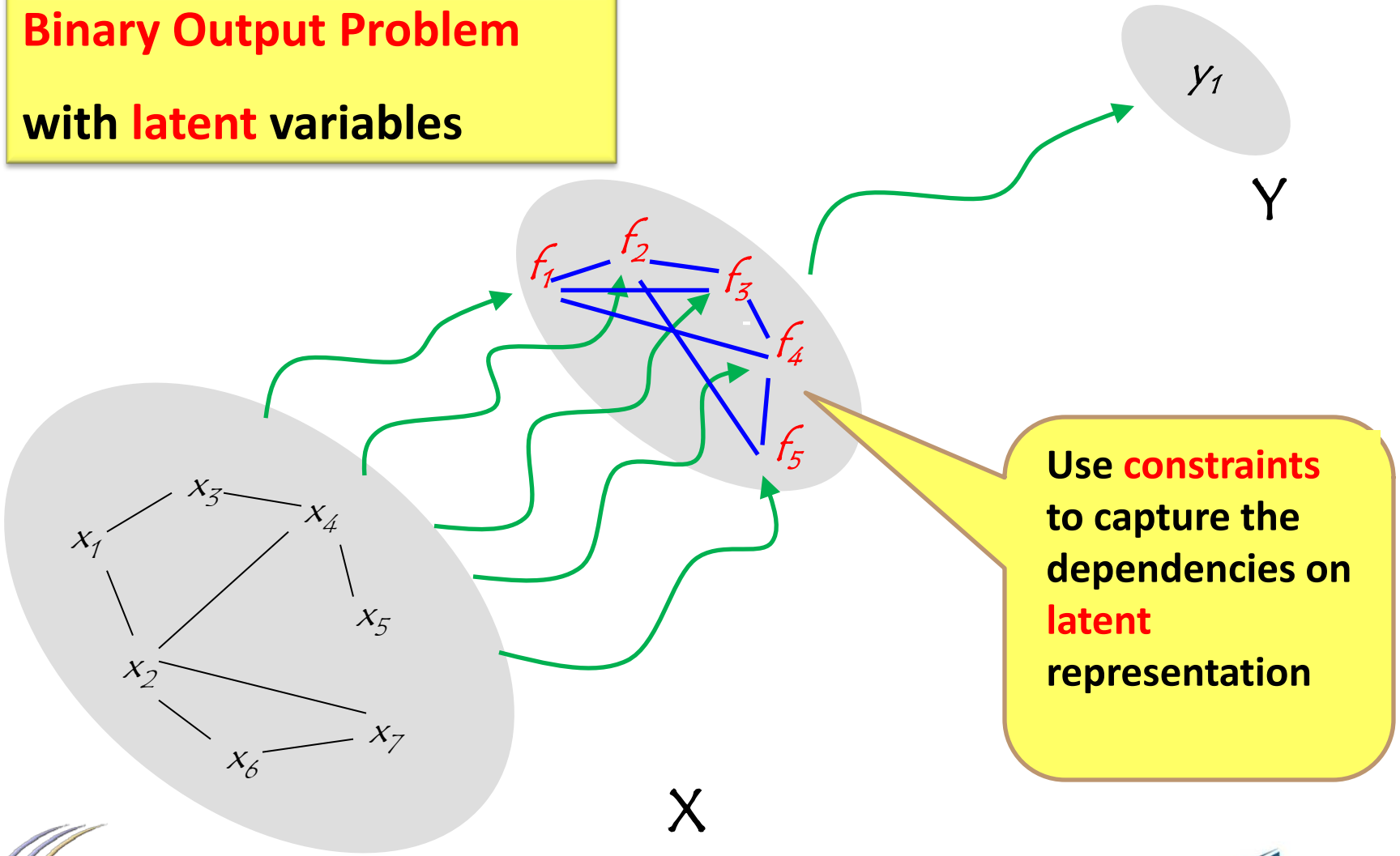
Only one output



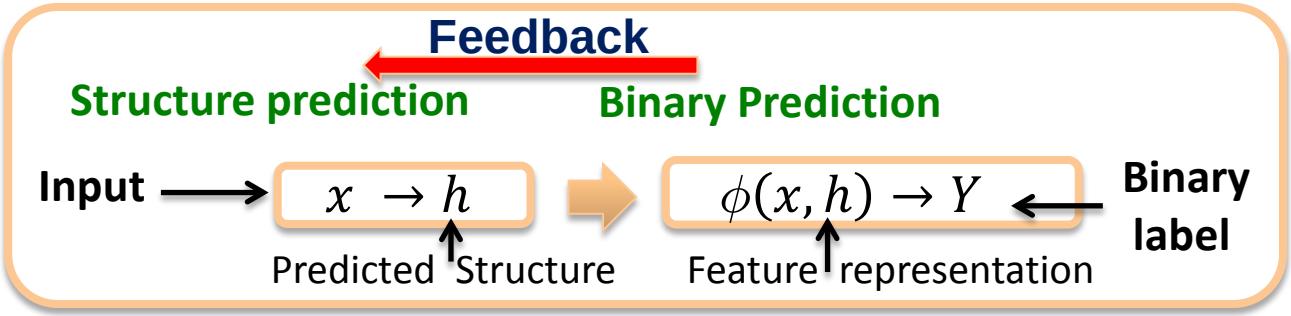
Binary classification problem with latent representation

Binary Output Problem

with latent variables



Algorithms: Two Conceptual Approaches



- Two stage approach (typically used for TE and paraphrase identification)
  - Learn hidden variables; fix it
    - **Need supervision for the hidden layer (or heuristics)**
  - For each example, extract features over x and (the fixed) h.
  - Learn a binary classier
- Proposed Approach: Joint Learning
  - Drive the learning of h from the binary labels
  - Find the best h(x) [Use constraints here to search only for "legitimate" h's]
  - **An intermediate structure representation is good to the extent is supports better final prediction.**
  - Algorithm?

Learning with Constrained Latent Representation (LCLR): Intuition

- If x is positive
  - There must exist a good explanation (intermediate representation)
  - $\exists$ h, $w^T \phi(x,h) \geq 0$
  - or, $\max_h w^T \phi(x,h) \geq 0$

This is an inference step that will gain from the CCM formulation CCM on the latent structure

- If x is negative
  - No explanation is good enough to support the answer
  - $\forall$ h, $w^T \phi(x,h) \leq 0$
  - or, $\max_h w^T \phi(x,h) \leq 0$

New feature vector for the final decision. Chosen h selects a representation.

- Altogether, this can be combined into an objective function:

$$\text{Min}_w \; 1/2||w||^2 \; + \; C\sum_i L(1-z_i \max_{h \in \mathcal{C}} w^T \sum_{\{s\}} h_s \phi_s(x_i))$$

Inference: best h subject to constraints C

Optimization

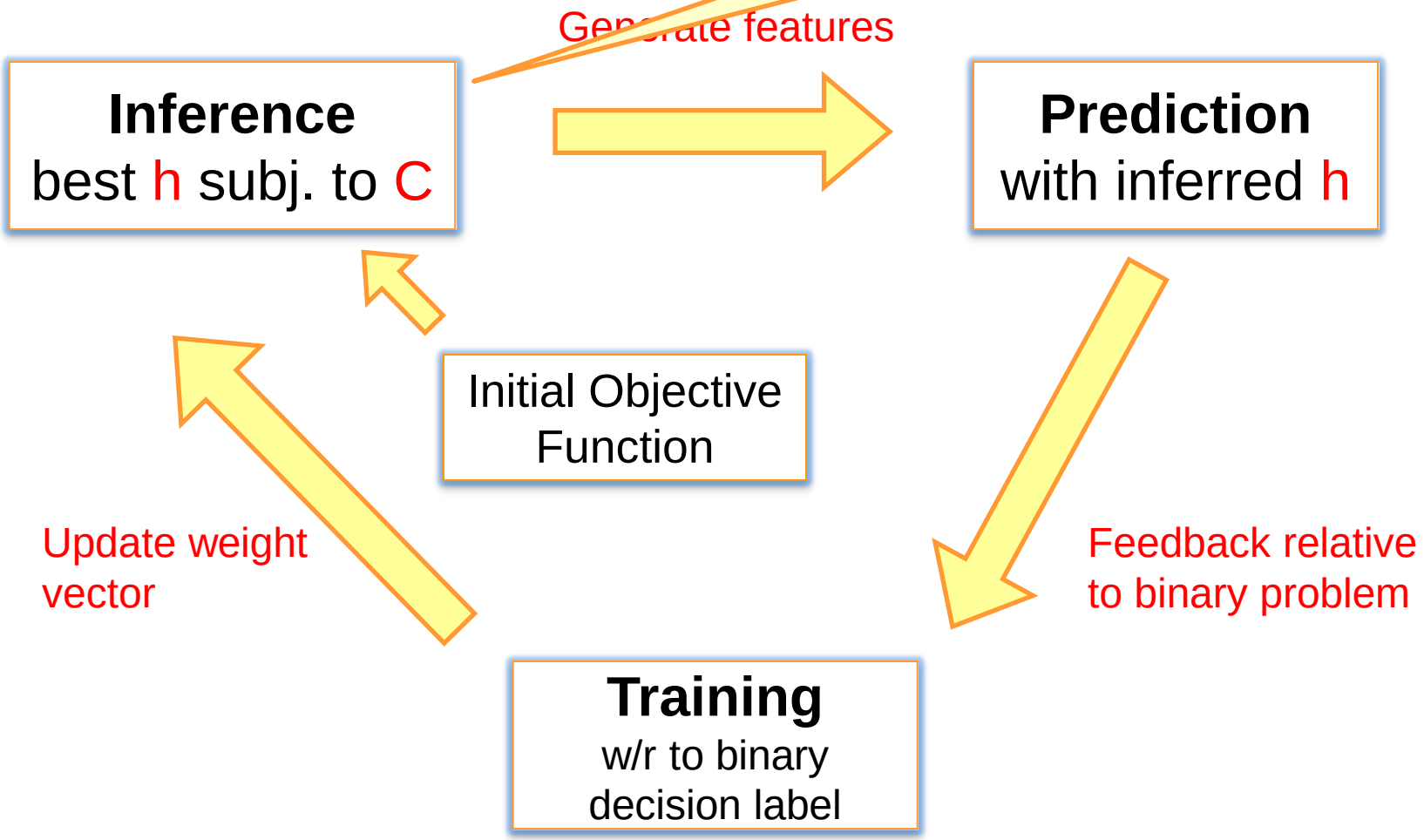
- Non Convex, due to the maximization term inside the global minimization problem
- In each iteration:
  - Find the best feature representation h* for all positive examples (off-the shelf ILP solver)
  - Having fixed the representation for the positive examples, update w solving the convex optimization problem:

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i:z_i=1} \ell(1 - \mathbf{w}^T \sum_s h^*_{i,s} \Phi_s(\mathbf{x}_i)) + C \sum_{i:z_i=-1} \ell(1 + \max_{\mathbf{h}\in\mathcal{H}} \mathbf{w}^T \sum_s h_s \Phi_s(\mathbf{x}_i))$$

  - Not the standard SVM/LR: need inference
- Asymmetry: Only positive examples require a good intermediate representation that justifies the positive label.
  - Consequently, the objective function decreases monotonically

Iterative Objective Function Learning

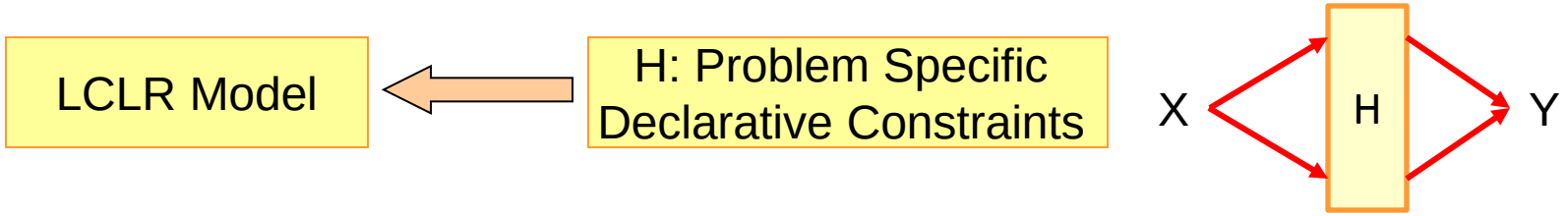
A CCM goes here: restrict possible hidden structures considered.



- Formalized as Structured SVM + Constrained Hidden Structure
- **LCRL: Learning Constrained Latent Representation**

Learning with Constrained Latent Representation (LCLR): Framework

- LCLR provides a general inference formulation that allows the use of expressive constraints to determine the hidden level
  - Flexibly adapted for many tasks that require latent representations.



- Paraphrasing: Model input as graphs, $V(G_{1,2})$, $E(G_{1,2})$
  - Four (types of) Hidden variables:
    - **$h_{v1,v2}$ – possible vertex mappings; $h_{e1,e2}$ – possible edge mappings**

$$\forall v_1 \in V(G_1), \sum_{v_2 \in V(G_2)} h_{v_1,v_2} + h_{v_1,*} = 1, \quad \forall v_2 \in V(G_2), \sum_{v_1 \in V(G_1)} h_{v_1,v_2} + h_{*,v_2} = 1$$

$$\forall e_1 \in E(G_1), \sum_{e_2 \in E(G_2)} h_{e_1,e_2} + h_{e_1,*} = 1, \quad \forall e_2 \in E(G_2), \sum_{e_1 \in E(G_1)} h_{e_1,e_2} + h_{*,e_2} = 1$$

$$h_{v_1,v_2} + h_{v'_1,v'_2} - h_{e_1,e_2} \leq 1, \quad h_{v_1,v_2} \geq h_{e_1,e_2}, \quad h_{v'_1,v'_2} \geq h_{e_1,e_2}$$

Experimental Results

Transliteration:

Transliteration System	Acc	MRR
(Goldwasser and Roth 2008)	N/A	89.4
Alignment + Learning	80.0	85.7
LCLR	**92.3**	**95.4**

Recognizing Textual Entailment:

Entailment System	Acc
Median of TAC 2009 systems	61.5
Alignment + Learning	65.0
LCLR	**66.8**

Paraphrase Identification:*

Alignment + Learning	72.00
LCLR	**72.75**

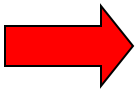
Summary

- Many important NLP problems require latent structures
- LCLR:
  - An algorithm that applies CCM to a latent structure
  - Can be used for many different NLP tasks
  - Easy to inject linguistic constraints on latent structures
  - A general learning framework that is good for many loss functions
- Take home message:
  - It is possible to apply constraints on many important problems with latent variables!

Training Constrained Conditional Models

$$\underset{y}{\operatorname{argmax}}\, \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

- Learning constraints' penalties
  - Independently of learning the model
  - Jointly, along with learning the model
- Dealing with lack of supervision
  - Constraints Driven Semi-Supervised learning (CODL)
  - Learning Constrained Latent Representations
  - Indirect Supervision



Indirect supervision for Structured Prediction

- Our goal is to exploit constraints to aid learning structures.
- Intuition:
  - If the y variables we are after are tightly coupled (via constraints)
  - …perhaps supervising some of them could be propagated to others and amplify the weak supervision
- Before, the structure was in the intermediate level
  - We cared about the structured representation only to the extent it helped the final binary decision
  - The binary decision variable was given as supervision
- What if we care about the structure?
  - Information & Relation Extraction; POS tagging, Semantic Parsing
- Invent a companion binary decision problem!

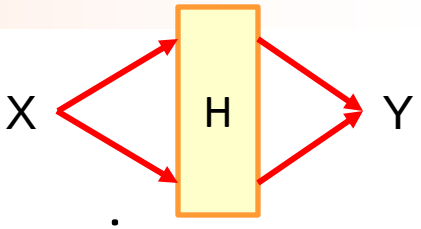
Information extraction

Lars Ole Andersen . Program analysis and specialization for the C Programming language. PhD thesis. DIKU , University of Copenhagen, May 1994 .

Prediction result of a trained HMM

[AUTHOR]	Lars Ole Andersen . Program analysis and
[TITLE]	specialization for the
[EDITOR]	C
[BOOKTITLE]	Programming language
[TECH-REPORT]	. PhD thesis .
[INSTITUTION]	DIKU , University of Copenhagen , May
[DATE]	1994 .

Structured Prediction



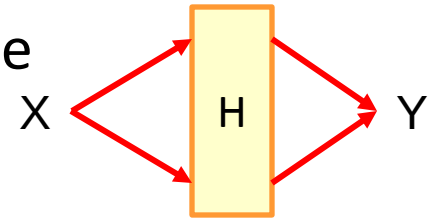
- The problem we have now is a "real" structure learning problem.
  - We will reduce the previous problem to this one by introducing a "fictitious" companion binary variable that is easy to supervise.

- Invent a companion binary decision problem!
- **Parse Citations: Lars Ole Andersen . Program analysis and specialization for the C Programming language. PhD thesis. DIKU , University of Copenhagen, May 1994 .**
  - **Companion**: Given a citation; does it have a legitimate citation parse?
- **POS Tagging**
  - **Companion:** Given a word sequence, does it have a legitimate POS tagging sequence?
- Binary Supervision is almost free

Companion Task Binary Label as Indirect Supervision

- The two tasks are related just like the binary and structured tasks discussed earlier

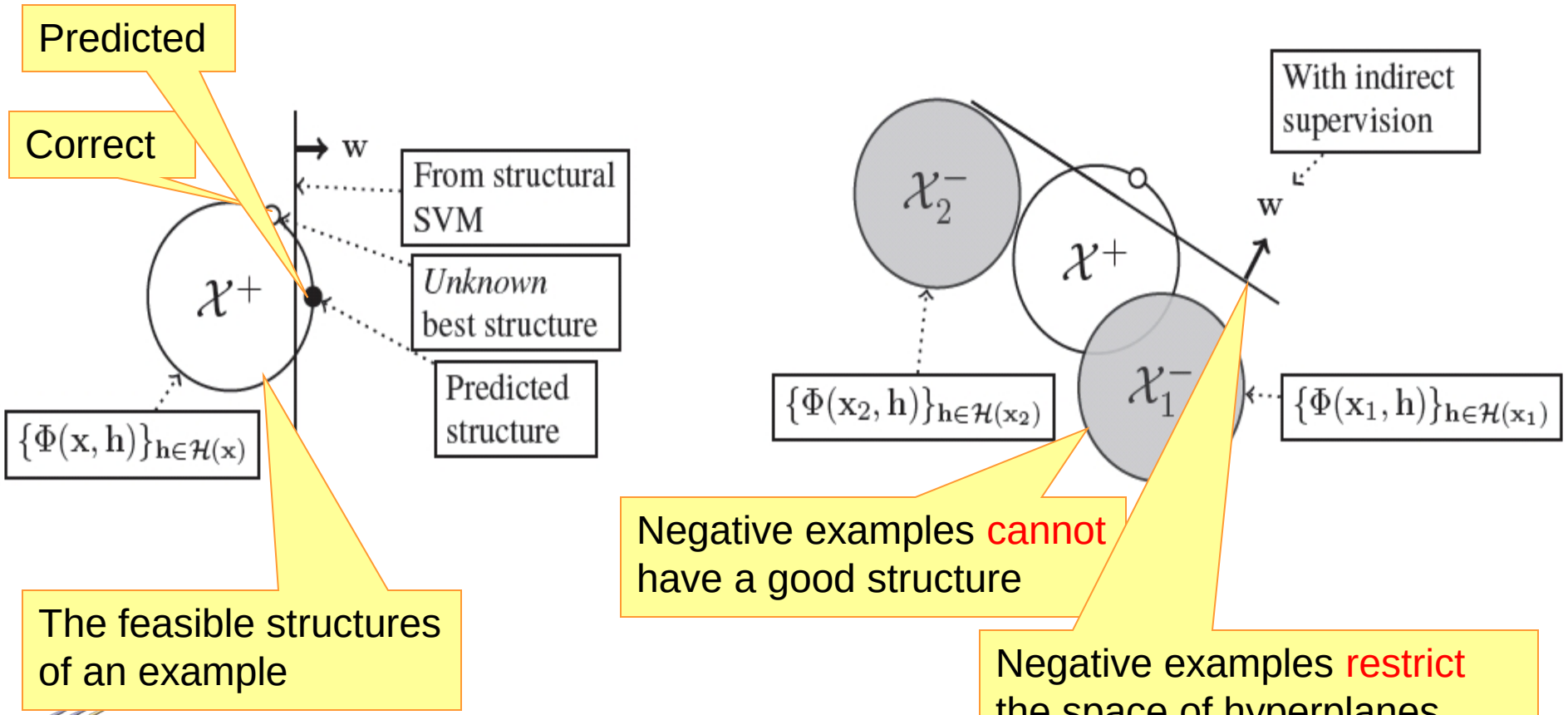
- All positive examples must have a good structure
- Negative examples cannot have a good structure
- We are in the same setting as before
  - Binary labeled examples are easier to obtain
  - We can take advantage of this to help learning a structured model



- Algorithm: combine binary learning and structured learning

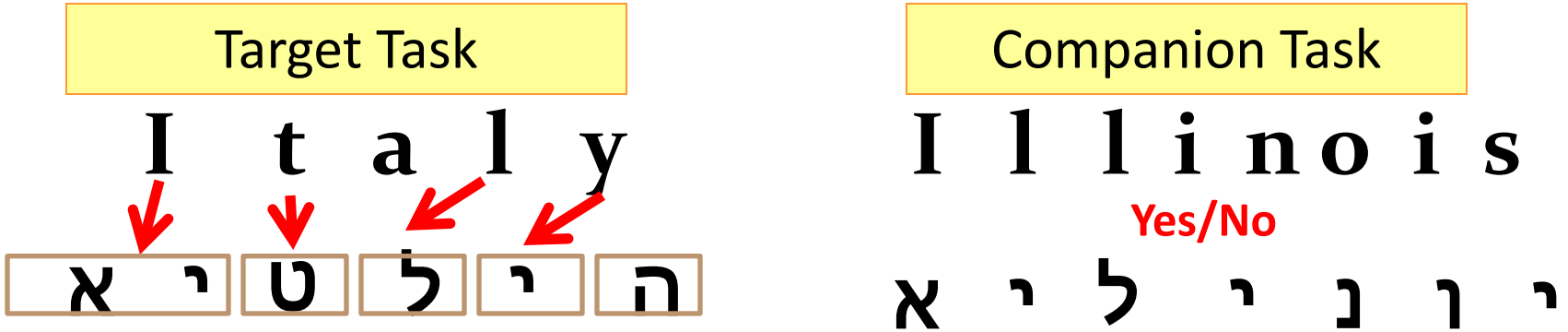
Learning Structure with Indirect Supervision

- In this case we care about the predicted structure
- Use both Structural learning and Binary learning



Joint Learning Framework

- Joint learning : If available, make use of both supervision types



Loss function – same as described earlier.

Key: the same parameter **w** for both components

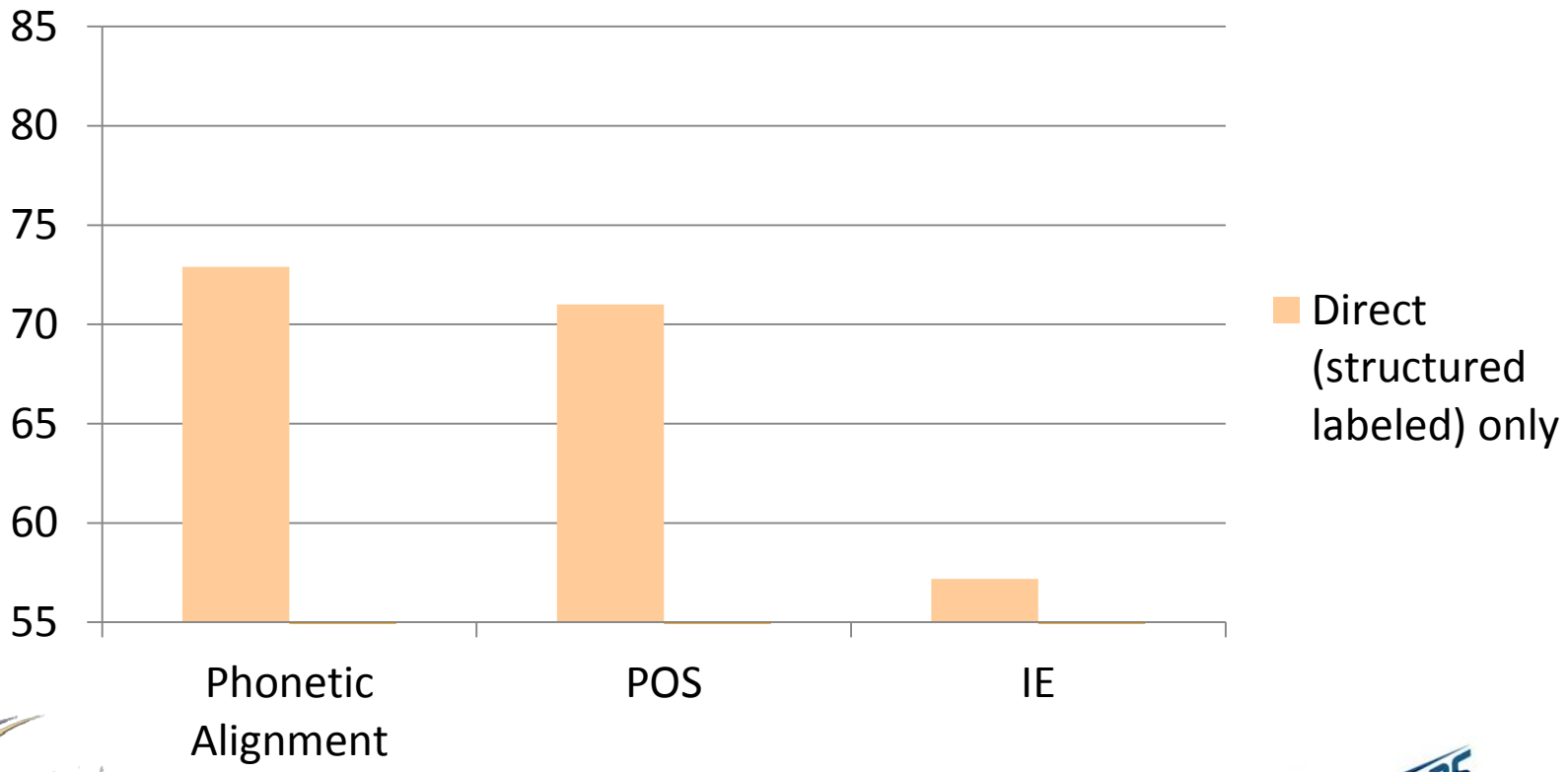
$$\min_{w} \frac{1}{2} w^T w + C_1 \sum_{i \in S} L_S(x_i, y_i; w)$$

Loss on Target Task

Loss on Companion Task

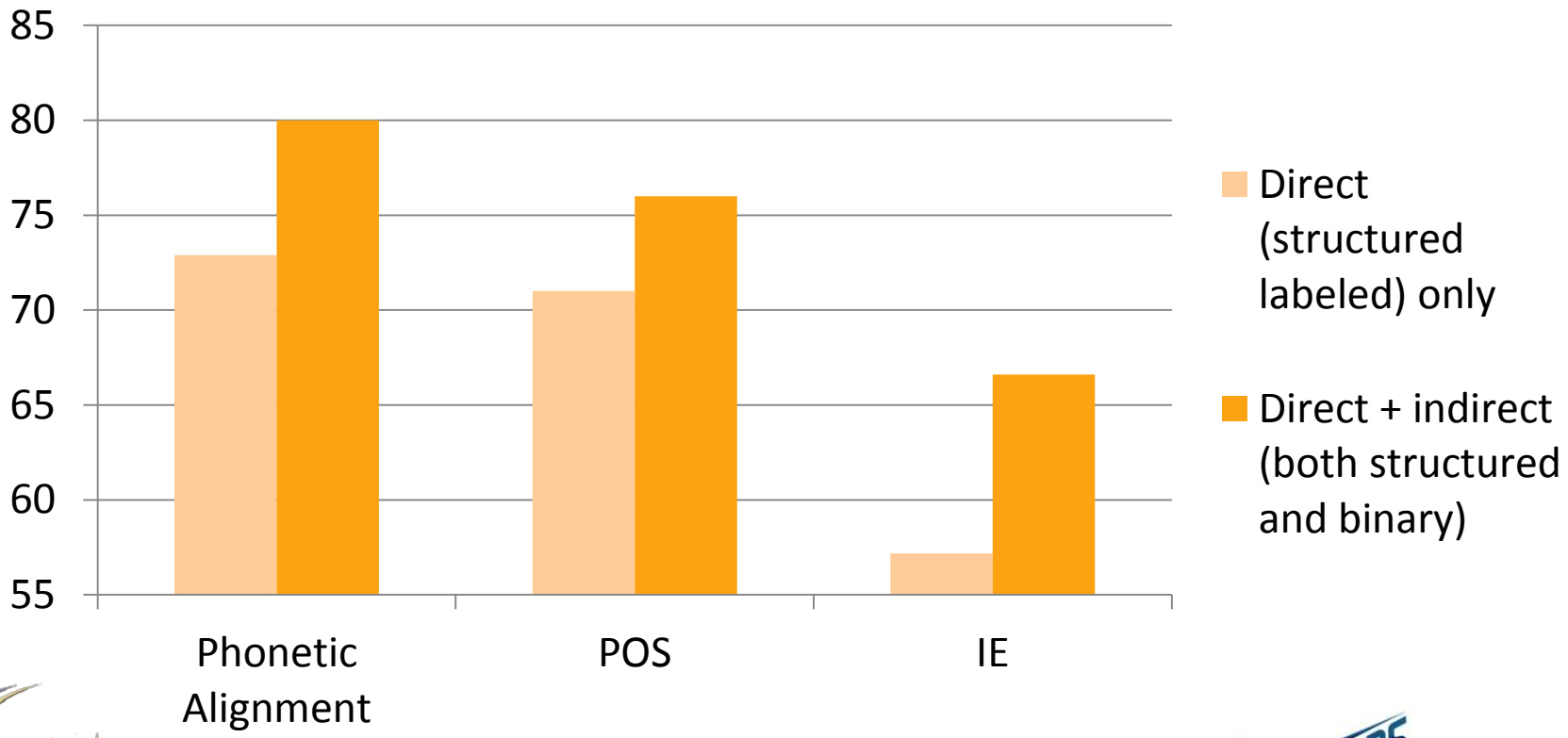
Experimental Result

- Very little direct (structured) supervision.



Experimental Result

- Very little direct (structured) supervision.
- (Almost free) Large amount binary indirect supervision



More on Dealing with minimal of supervision

- Constraint Driven Learning
  - Use constraints to guide semi-supervised learning!

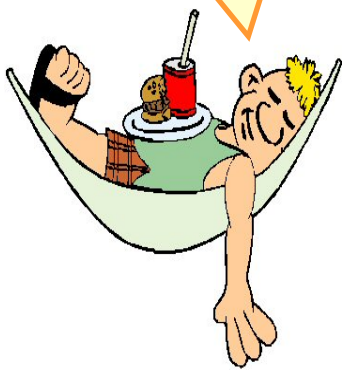
- Use Binary Labeled data to help structured output prediction
  - Training Structure Predictors by Inventing (easy to supervise) binary labels

  - Driving supervision signal from World's Response
    - **Efficient Semantic Parsing = CCM + world's response**

Connecting Language to the World

Can we rely on this interaction to provide supervision (and, eventually, recover meaning) ?

Can I get a coffee with sugar and no milk



Semantic Parser

MAKE(COFFEE,SUGAR=YES,MILK=NO)

Great!

Arggg

- How to recover meaning from text?
- Annotate with meaning representation; use (standard) “example based” ML
  - Teacher needs deep understanding of the learning agent
  - Annotation burden; not scalable.
- Instructable computing
  - Natural communication between teacher/agent

Real World Feedback

Supervision = Expected Response

Traditional approach: learn from logical forms and gold alignments

EXPENSIVE!

Our approach: use only the responses

x NL Query

y Logical Query

r Query Response:

"What is the largest state that borders NY?"

largest(state(next_to(const(NY))))

Pennsylvania

Pennsylvania

Interactive Computer System

Binary Supervision

Check if Predicted response == Expected response

Semantic parsing is a structured prediction problem: identify mappings from text to a meaning representation

Expected : Pennsylvania
Predicted : Pennsylvania

Positive Response

Expected : Pennsylvania
Predicted : NYC

Negative Response

We will use a CCM formulation with a lot of "legitimacy" constraints

Train a structured predictor with this binary supervision !

Response Based Learning

Input: Inputs $\{\mathbf{x}^l\}_{l=1}^N$,
Feedback : $\mathcal{X} \times \mathcal{Y} \rightarrow \{+1, 1\}$,
initial weight vector $\mathbf{w}$

1: **repeat**
2: **for** $l = 1, \ldots, N$ **do**
3: $\hat{\mathbf{h}}, \hat{\mathbf{y}} = \arg\max_{\mathbf{h},\mathbf{y}} \mathbf{w}^T \Phi(\mathbf{x}^l, \mathbf{h}, \mathbf{y})$
4: $f = Feedback(\mathbf{x}^l, \hat{\mathbf{y}})$
5: add $(\Phi(\mathbf{x}^l, \hat{\mathbf{h}}, \hat{\mathbf{y}}), f)$ to B
6: **end for**
7: $\mathbf{w} \leftarrow TRAIN(B)$
8: **until** Convergence
9: **return** w

Difficulty:
- **Need to generate training examples**
- **Negative examples give no information**

Basic Algorithm:
- Try to generate good structures
- Update parameters based on current examples
- Coarse use of incorrect structures

Repeat
for all input sentences **do**
Find best logical representation y given current w
Query *feedback* function
end for
Learn new W using feedback
Until Convergence

TRAIN: Try to get more positive examples (representations with positive feedback)
Direct (Binary) protocol: a binary classifier on **Positive/Negative ex's**
(**Problem: many good structures are being demoted**)
Structured Protocol: Use only correct structures.
(Problem: ignores negative feedback)

Empirical Evaluation [CoNLL'10,ACL'11, IJCAI'11]

- Key Question: **Can we learn from this type of supervision?**

Algorithm	# training structures	Test set accuracy
No Learning: Initial Objective Fn	0	22.2%
Binary signal: **Binary Protocol**	0	69.2 %
Binary signal: **Structured Protocol**	0	**73.2 %**
Improved Protocol:	0	**79.6%**
WM*2007 (fully supervised – uses gold structures)	**310**	**75 %**

*[WM] Y.-W. Wong and R. Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. ACL.

Summary

- Constrained Conditional Models: Computational Framework for global inference and a vehicle for incorporating knowledge
- Direct supervision for structured NLP tasks is **expensive**
  - Indirect supervision is cheap and easy to obtain
- We suggested learning protocols for Indirect Supervision
  - Make use of simple, easy to get, binary supervision
  - Showed how to use it to learn structure and latent structures
  - **CCM Inference is key in propagating the simple supervision**
- Learning Structures from Real World Feedback
  - Obtain binary supervision from "real world" interaction
  - Indirect supervision replaces direct supervision

This Tutorial: Constrained Conditional Models (Part II)

- Part 6: **Conclusion (& Discussion)** (10 min)
  - Building CCMs; Features and Constraints. Mixed models vs. Joint models;
  - where is Knowledge coming from

THE END

Conclusion

- Constrained Conditional Models combine
  - Learning conditional models with using declarative expressive constraints
  - Within a constrained optimization framework

- Our goal was to describe:
  - A clean way of incorporating constraints to bias and improve decisions of learned models
  - A clean way to use (declarative) prior knowledge to guide semi-supervised learning
  - Ways to make use of (declarative) prior knowledge when choosing intermediate (latent) representations.

- Provide examples for the diverse usage CCMs have already found in NLP
  - Significant success on several NLP and IE tasks (often, with ILP)

What is a Constrained Conditional Model?

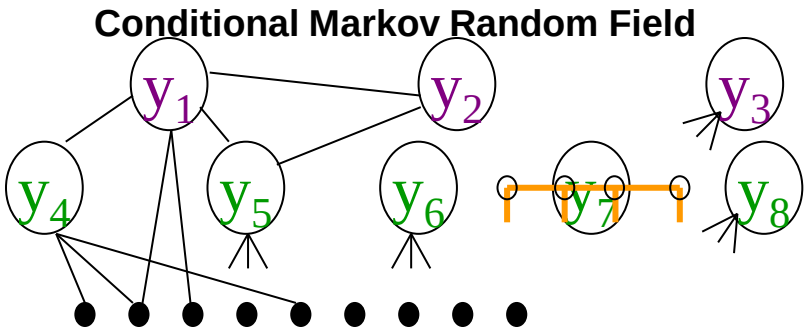
$$\underset{y}{\operatorname{argmax}}\, \boldsymbol{\lambda} \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

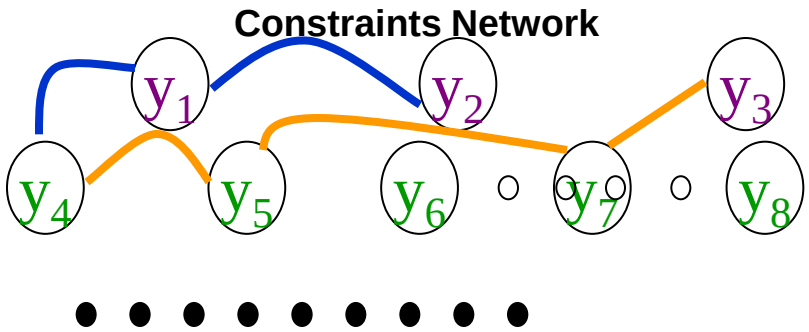
Modeling NLP problem • Variables, Features and constraints	**Objective function** • Constrained Conditional Model
Constrained optimization language • How to represent inference?	Integer linear program
Inference • How to solve it?	Several inference algorithms: Exact ILP, search, relaxation; dynamic prog.
Learning • How to learn the objective function?	Learning λ and ρ. Several learning strategies: L+I, IBT, others.

Technical Conclusions

- Presented and discussed modeling issues
  - How to improve existing models using declarative information
  - Incorporating expressive global constraints into simpler learned models
- Discussed Inference issues
  - Often, the formulation is via an Integer Linear Programming formulation, but algorithmic solutions can employ a variety of algorithms.
- Training issues – Training protocols matters
  - Training with/without constraints; soft/hard constraints;
  - Performance, modularity and ability to use previously learned models.
  - Supervision-lean models
- We did not attend to the question of "how to find constraints"
  - Emphasis on: background knowledge is important, exists, use it.
  - But, it's clearly possible to learn constraints.

Summary: Constrained Conditional Models





$$y^* = \text{argmax}_y \sum w_i \phi(x; y)$$

- Linear objective functions
- Typically $\phi(x,y)$ will be local functions, or $\phi(x,y) = \phi(x)$

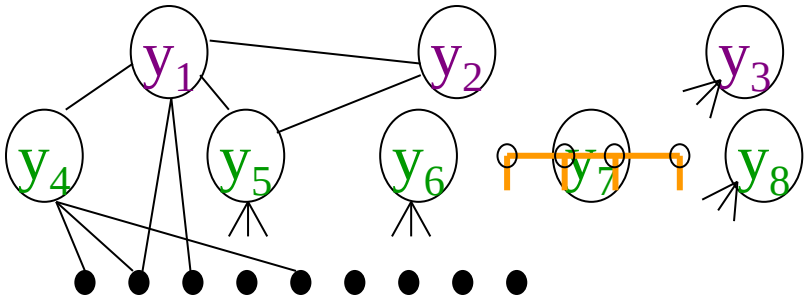
$$- \sum_i \rho_i d_C(x,y)$$

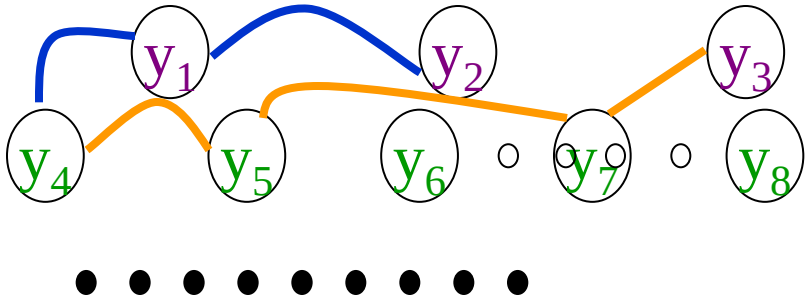
- Expressive constraints over output variables
- Soft, weighted constraints
- Specified declaratively as FOL formulae

- Clearly, there is a joint probability distribution that represents this mixed model.
- We would like to:
  - Learn a simple model or several simple models
  - Make decisions with respect to a complex model

Designing CCMs

Thanks!





$$y^* = \text{argmax}_y \sum w_i \phi(x; y)$$

- Linear objective functions
- Typically $\phi(x,y)$ will be local functions, or $\phi(x,y) = \phi(x)$

$$- \sum_i \rho_i d_C(x,y)$$

- Expressive constraints over output variables
- Soft, weighted constraints
- Specified declaratively as FOL formulae

LBJ (Learning Based Java): http://L2R.cs.uiuc.edu/~cogcomp
A modeling language for Constrained Conditional Models. Supports programming along with building learned models, high level specification of constraints and inference with constraints

Questions?

- Thank you!

Global Inference Using Integer Linear Programming

Wen-tau Yih

August 15, 2004

1 Introduction

This report is a supplemental document of some of our papers [5, 3, 4]. It gives a simple but complete step-by-step case study, which demonstrates how we apply integer linear programming to solve a global inference problem in natural language processing. This framework first transforms an optimization problem into an integer linear program. The program can then be solved using existing numerical packages.

The goal here is to provide readers an easy-to-follow example to model their own problems in this framework. There are two main parts in this report. Sec. 2 describe a problem of labeling entities and relations simultaneously as our inference task. It then discusses the constraints among the labels and shows how the objective function and constraints are transformed to an integer linear program. Although transforming the constraints to their linear forms is not difficult in this entity and relation example, sometimes it can be tricky, especially when more variables are involved. Therefore, we discuss how to handle different types of constraints in Sec. 3.

2 Labeling Entities & Relations

Given a sentence, the task is to assign labels to the entities in this sentence, and identify the relation of each pair of these entities. Each entity is a phrase and we assume the boundaries of these entities are given.

Figure 1 gives an example of the sentence "Dole's wife, Elizabeth, is a native of N.C." In this sentence, there are three entities, *Dole*, *Elizabeth*, and *N.C.* We use E_1, E_2, and E_3 to represent their entity labels. In this example, possible entity labels include ***other***, ***person***, and ***location***. In addition, we would like to know the relation between each pair of the entities. For a pair of two entities E_i and E_j, the relation is represented by R_{ij}. In this example, there will be 6 relation variables – $R_{12}, R_{21}, R_{13}, R_{31}, R_{23}, R_{32}$. Since most entities have no special relation, the value of most relation variables should be ***irrelevant***. Besides this special label, the relations of interest in this example are ***spouse_of*** and ***born_in***.

Assume that magically some local classifiers have already provided some *confidence* scores on possible labels, as shown in Table 1.

If we want to choose the labels that maximize the sum of those confidence scores, it's the same as choosing the label that has the highest score for each variable. The global labeling then becomes:

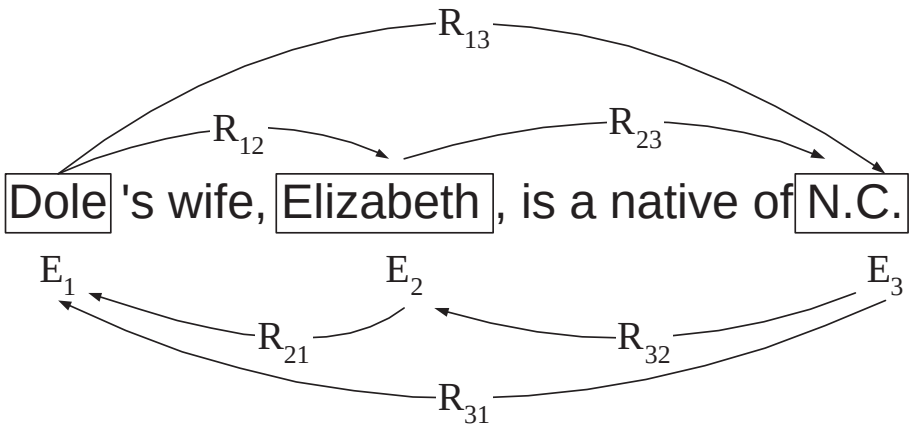


Figure 1: A sentence that has 3 entities

variable	other	person	location
E_1	0.05	0.85	0.10
E_2	0.10	0.60	0.30
E_3	0.05	0.50	0.45

variable	irrelevant	spouse_of	born_in
R_{12}	0.05	0.45	0.50
R_{21}	0.75	0.10	0.15
R_{13}	0.85	0.05	0.10
R_{31}	0.80	0.05	0.15
R_{23}	0.10	0.05	0.85
R_{32}	0.65	0.20	0.15

Table 1: The confidence scores on the labels of each variable.

variable	label	score
E_1	person	0.85
E_2	person	0.60
E_3	person	0.50
R_{12}	born_in	0.50
R_{21}	irrelevant	0.75
R_{13}	irrelevant	0.85
R_{31}	irrelevant	0.80
R_{23}	born_in	0.85
R_{32}	irrelevant	0.65
	sum	6.35

At this point, the problem seems to have been solved by the magic local classifiers. However, after a second look at this labeling, we can easily find the inconsistency between entity and relation labels. For example, R_{12} cannot be *born_in* if both entities E_1 and E_2 are *persons*. Indeed, there exists some natural constraints between the labeling of entity and relation variables that the local classifiers may not know or respect. In our example, we know the global labeling also stratifies the following two constraints.

- if R_{ij} = spouse_of, then E_i = person AND E_j = person
- if R_{ij} = born_in, then E_i = person AND E_j = location

In summary, the problem we want to solve here really is to find the best legitimate global labeling, which is subject to the constraints and maximizes the sum of the confidence scores.

Note that although exhaustive search seems plausible in this toy problem, it soon becomes intractable when the number of variables or the number of possible labels grows. In the rest of this section, we are going to show that how we transfer this problem to an integer linear program, and let the numerical packages help us to find the answer.

2.1 Indicator Variables

In order to apply (integer) linear programming, both the objective function and constraints have to be linear. Since the confidence score could be any real number, the original function is not linear. In addition, the logical constraints we have are not linear as well.

To overcome this difficulty, the first step of the transformation is to introduce several *indicator* (binary) variables, which represent the assignment of the original variables. For each entity or relation variable a and each legitimate label k, we introduce a binary variable $x_{a,k}$. When the original variable a is assigned label k, $x_{a,k}$ is set to 1. Otherwise, $x_{a,k}$ is 0. In our toy example, we then have 27 such indicator variables:

$$
\begin{array}{ccc}
x_{E_1,\text{other}}, & x_{E_1,\text{person}} & x_{E_1,\text{location}}, \\
x_{E_2,\text{other}}, & x_{E_2,\text{person}}, & x_{E_2,\text{location}}, \\
x_{E_3,\text{other}}, & x_{E_3,\text{person}}, & x_{E_3,\text{location}}, \\
x_{R_{12},\text{irrelevant}}, & x_{R_{12},\text{spouse\_of}}, & x_{R_{12},\text{born\_in}}, \\
x_{R_{21},\text{irrelevant}}, & x_{R_{21},\text{spouse\_of}}, & x_{R_{21},\text{born\_in}}, \\
x_{R_{13},\text{irrelevant}}, & x_{R_{13},\text{spouse\_of}}, & x_{R_{13},\text{born\_in}}, \\
x_{R_{31},\text{irrelevant}}, & x_{R_{31},\text{spouse\_of}}, & x_{R_{31},\text{born\_in}}, \\
x_{R_{23},\text{irrelevant}}, & x_{R_{23},\text{spouse\_of}}, & x_{R_{23},\text{born\_in}}, \\
x_{R_{32},\text{irrelevant}}, & x_{R_{32},\text{spouse\_of}}, & x_{R_{32},\text{born\_in}}.
\end{array}
$$

To simplify the notation, let $L_E = \{\text{other}, \text{person}, \text{location}\}$ and $L_R = \{\text{irrelevant}, \text{spouse\_of}, \text{born\_in}\}$ represent the sets of entity and relation labels, respectively. Assume $n = 3$ means the number of entities we have in the sentence. The indicator variables we introduce are:

$$
\begin{array}{ll}
x_{E_i,l_e}, & \text{where } 1 \le i \le n \text{ and } l_e \in L_E \\
x_{R_{ij},l_r}, & \text{where } 1 \le i,j \le n, i \ne j, \text{ and } l_r \in L_R
\end{array}
$$

2.2 Objective Function

Suppose c_{E_i,l_e} represents the confidence score of E_i being l_e, where $1 \le i \le n$ and $l_e \in L_E$, and c_{R_{ij},l_r} represents the confidence score of R_{ij} being l_r, where $1 \le i,j \le n, i \ne j$ and $l_r \in L_R$. The objective function f (i.e., the sum of confidence scores) can be represented by

$$
f = \sum_{1 \le i \le n, l_e \in L_E} c_{E_i,l_e} x_{E_i,l_e} + \sum_{1 \le i,j \le n, i \ne j, l_r \in L_R} c_{R_{ij},l_r} x_{R_{ij},l_r}
$$

If we plug in the numbers in Table 1, the function f is:

$$
f = 0.05 \cdot x_{E_1,\text{other}} + 0.85 \cdot x_{E_1,\text{person}} + \cdots + 0.65 \cdot x_{R_{32},\text{irrelevant}} + 0.20 \cdot x_{R_{32},\text{spouse\_of}} + 0.15 \cdot x_{R_{32},\text{born\_in}}
$$

Inevitably, this transformation also brings new constraints, which come from the fact that one entity/relation variable can only have one label, and must have one label. For example, only exact one of the labels *other*, *person*, *location* can be assigned to E_1. As a result, only one of the indicator variables $x_{E_1,other}, x_{E_1,person}, x_{E_1,location}$ can and must be 1. This restriction can be easily written as the following linear equations.

$$\begin{aligned} \textstyle\sum_{l_e \in L_E} x_{E_i,l_e} &= 1 \quad \forall 1 \le i \le n \\ \textstyle\sum_{l_r \in L_R} x_{R_{ij},l_r} &= 1 \quad \forall 1 \le i,j \le n, i \ne j \end{aligned}$$

2.3 Logical Constraints

The other reason of introducing indicator variables is to handle the real constraints we have – the logical constraints between entity and relation labels. Let me remind you what they are in our example:

- if R_{ij} = spouse_of, then E_i = person AND E_j = person, where $1 \le i,j \le n$ and $i \ne j$
- if R_{ij} = born_in, then E_i = person AND E_j = location, where $1 \le i,j \le n$ and $i \ne j$

If we treat the indicator variables as boolean variables, where 1 means *true* and 0 means f*alse*, the constraints can be rephrased as:

$$\begin{aligned} x_{R_{ij},\text{spouse\_of}} &\rightarrow x_{E_i,person} \wedge x_{E_j,person} \quad 1 \le i,j \le n, \text{ and } i \ne j \\ x_{R_{ij},born\_in} &\rightarrow x_{E_i,person} \wedge x_{E_j,location} \quad 1 \le i,j \le n, \text{ and } i \ne j \end{aligned}$$

In fact, these two boolean constraints can be modeled by the following two linear inequalities.

$$\begin{aligned} 2 \cdot x_{R_{ij},\text{spouse\_of}} &\le x_{E_i,\text{person}} + x_{E_j,\text{person}} \quad 1 \le i,j \le n, \text{ and } i \ne j \\ 2 \cdot x_{R_{ij},\text{born\_in}} &\le x_{E_i,\text{person}} + x_{E_j,\text{location}} \quad 1 \le i,j \le n, \text{ and } i \ne j \end{aligned}$$

Let's do a simple check to see if they are correct. When $x_{R_{ij},\text{spouse\_of}}$ is 0 (false), $x_{E_i,\text{person}}$ and $x_{E_j,\text{person}}$ can be either 0 or 1, and the inequality still holds. However, when $x_{R_{ij},\text{spouse\_of}}$ is 1 (true), both $x_{E_i,\text{person}}$ and $x_{E_j,\text{person}}$ have to be 1 (true).

Transforming the logical constraints into linear forms is the key of this framework. It is not hard, but may be tricky sometimes (which makes it an interesting brain exercise). We will talk more about transforming other types of logical constraints in Sec. 3 later.

2.4 Solving the Integer Linear Program Using Xpress-MP

Figure 2 shows the complete integer linear program. Now, all we need to do now is to apply some numeric packages, such as Xpress-MP [7], CPlex [1], or the LP solver in R [6], to solve it. Transferring the solution back to the global labeling we want is straightforward – just find those indicator variables that have the value 1. In this section, I will demonstrate how to apply Xpress-MP to do the job.

The syntax in Xpress-MP is fairly easy and straightforward. Here I simply list the source code with some comments, which are the lines beginning with the "!" symbol.

$$\max \quad \sum_{1\leq i\leq n, l_e\in L_E} c_{E_i,l_e} x_{E_i,l_e} + \sum_{1\leq i,j\leq n, i\neq j, l_r\in L_R} c_{R_{ij},l_r} x_{R_{ij},l_r}$$

subject to:

$$x_{E_i,l_e} \in \{0,1\} \qquad \forall 1 \leq i \leq n \tag{1}$$

$$x_{R_{ij},l_r} \in \{0,1\} \qquad \forall 1 \leq i,j \leq n, i \neq j \tag{2}$$

$$\sum_{l_e\in L_E} x_{E_i,l_e} = 1 \qquad \forall 1 \leq i \leq n \tag{3}$$

$$\sum_{l_r\in L_R} x_{R_{ij},l_r} = 1 \qquad \forall 1 \leq i,j \leq n, i \neq j \tag{4}$$

$$2 \cdot x_{R_{ij},\text{spouse\_of}} \leq x_{E_i,\text{person}} + x_{E_j,\text{person}} \qquad 1 \leq i,j \leq n, \text{ and } i \neq j \tag{5}$$

$$2 \cdot x_{R_{ij},\text{born\_in}} \leq x_{E_i,\text{person}} + x_{E_j,\text{location}} \qquad 1 \leq i,j \leq n, \text{ and } i \neq j \tag{6}$$

Figure 2: The complete integer linear program

```
model "Entity Relation Inference"
 uses "mmxprs"

parameters
  DATAFILE = "er.dat"
  Num_Entities = 3;
end-parameters

declarations
  ENTITIES = 1..Num_Entities
  ENT_CLASSES = {"Other", "Person", "Location"}
  REL_CLASSES = {"Irrelevant", "SpouseOf", "BornIn"}

  scoreEnt: array(ENTITIES, ENT_CLASSES) of real
  scoreRel: array(ENTITIES, ENTITIES, REL_CLASSES) of real
end-declarations

! DATAFILE stores the confidence scores from the local classifiers.
initializations from DATAFILE
  scoreEnt   scoreRel
end-initializations

! These are the indicator variables. declarations
  ent : array(ENTITIES, ENT_CLASSES) of mpvar
  rel : array(ENTITIES, ENTITIES, REL_CLASSES) of mpvar
end-declarations

! The objective function: sum of confidence scores
Obj := sum(u in ENTITIES, e in ENT_CLASSES) scoreEnt(u,e)*ent(u,e)
     + sum(u,v in ENTITIES, r in REL_CLASSES | u <> v) scoreRel(u,v,r)*rel(u,v,r)
```

```
! Constraints (1) and (2): the indicator variables take only binary values
forall(u in ENTITIES, e in ENT_CLASSES)
  ent(u,e) is_binary
forall(e1,e2 in ENTITIES, r in REL_CLASSES | e1 <> e2)
  rel(e1,e2,r) is_binary

! Constraints (3) and (4): sum = 1
forall(u in ENTITIES) sum(e in ENT_CLASSES)
  ent(u,e) = 1
forall(u,v in ENTITIES | u <> v) sum(r in REL_CLASSES)
  rel(u,v,r) = 1

! Constraints (5) and (6): logical constraints on entity and relation labels
forall(e1,e2 in ENTITIES | e1 <> e2)
  2*rel(e1,e2,"SpouseOf") <= ent(e1,"Person") + ent(e2,"Person")
forall(e1,e2 in ENTITIES | e1 <> e2)
  2*rel(e1,e2,"BornIn") <= ent(e1,"Person") + ent(e2,"Location")

! Solve the problem
maximize(Obj)

! Output the indicator variables that are 1
forall(u in ENTITIES, e in ENT_CLASSES | getsol(ent(u,e)) >= 1)
  writeln(u, " ", e)
forall(e1,e2 in ENTITIES, r in REL_CLASSES | e1 <> e2 and getsol(rel(e1,e2,r)) >= 1)
  writeln(e1, " ", e2, " ", r)

end-model
```

3 Transforming Logical Constraints into Linear Forms

This section summarizes and revises some rules of transforming logical constraints to linear (in)equalities described in [2]. To simplify the illustration, symbols a, b, c and $x_1, x_2, \cdots, x_n$ are used to represent indicator variables, which are treated as boolean variables and binary variables at the same. As usual, the values $0, 1$ represents the truth values *false* and *true*, respectively.

3.1 Choice Among Several Possibilities

In our entity and relation example, we have already processed the constraint "exactly k variables among $x_1, x_2, \cdots, x_n$ are true", where $k = 1$. The general form of this linear equation is:

$$x_1 + x_2 + \cdots + x_n = k$$

Another constraint, "at most k variables among $x_1, x_2, \cdots, x_n$ can be true", can be represented in a similar inequality.

$$x_1 + x_2 + \cdots + x_n \leq k$$

Uninterestingly, "k or more variables among $x_1, x_2, \cdots, x_n$ must be true" will be

$$x_1 + x_2 + \cdots + x_n \geq k$$

3.2 Implications

Implications are usually the logical constraints we encounter. While handling two or three variables may be trivial, extending it to more variables may be tricky. Here we illustrate how to develop the ideas from the simplest case to complicated constraints.

Two variables Suppose there are only two indicator variables a, b in the implication. The constraint, $a \rightarrow b$, can be represented as $a \leq b$. This can be easily verified by the following truth table.

$a \leq b$	$b = 0$	$b = 1$
$a = 0$	true	true
$a = 1$	false	true

What if we need to deal with something like $a \rightarrow \bar{b}$? The value of the compliment of b is exactly $1 - b$. Therefore, the corresponding linear constraint is $a \leq 1 - b$, or $a + b \leq 1$.

The relation "if and only if" is straightforward too. $a \leftrightarrow b$ is identical to $a \rightarrow b$ and $b \rightarrow a$. The corresponding linear constraints are $a \leq b$ and $b \leq a$, which is in fact $a = b$.

Three variables Now, let's try to generalize the implication a little bit to cover three variables. Since $a \rightarrow b \wedge c$ can be separated as $a \rightarrow b$ and $a \rightarrow c$, the straightforward transformation is to put two linear inequalities $a \leq b$ and $a \leq c$. Alternatively, the transformation in our entity and relation example "$2a \leq b + c$" also suffice, which is easy to check using a truth table.

Another implication, $a \rightarrow b \vee c$, can be modeled by $a \leq b + c$. This is because when $a = 1$, at least one of b and c has to be 1 to make the inequality correct.

What about the inverse of the above two implications? They can be derived using the compliment and DeMorgan's Theorem. $b \wedge c \rightarrow a$ is equivalent to $\bar{a} \rightarrow \overline{b \wedge c}$, which is $\bar{a} \rightarrow \bar{b} \vee \bar{c}$. Use the above rule and the the compliment, it can be modeled by $(1-a) \leq (1-b)+(1-c)$, or $a \geq b+c-1$. $b \vee c \rightarrow a$ is equivalent to $b \rightarrow a$ and $c \rightarrow a$, so it can be modeled by two inequalities $b \leq a$ and $c \leq a$. Alternatively, this can be transformed to $\bar{a} \rightarrow \overline{b \vee c}$, which is $\bar{a} \rightarrow \bar{b} \wedge \bar{c}$. Therefore, it can be modeled by $2(1 - a) \leq (1 - b) + (1 - c)$, or $\frac{b+c}{2} \leq a$.

More variables A logical constraint that has more variables can be complicated. Therefore, we only discuss some common cases here. Suppose we want to model "if a, then k or more variables among $x_1, x_2, \cdots, x_n$ are true." We can extend the transformation of $a \rightarrow b \vee c$, and use the following linear inequality.

$$a \leq \frac{x_1 + x_2 + \cdots + x_n}{k}$$

This transforation is certainly valid for $k = 1$. It is also easy to verify for other cases. If $a = 0$, then the right-hand-side is always larger or equal to 0, and the inequality is satisfied. However, when $a = 1$, it forces at least k x's are true, which is exactly what we want.

The next case we would like to try is the inverse, which is "if k or more variables among $x_1, x_2, \cdots, x_n$ are true, then a is true." This might be somewhat tricker than others. Our first guess might be:

$$(x_1 + x_2 + \cdots + x_n) - (k - 1) \leq a$$

Original constraint	Linear form
Exactly k of $x_1, x_2, \cdots, x_n$	$x_1 + x_2 + \cdots + x_n = k$
At most k of $x_1, x_2, \cdots, x_n$	$x_1 + x_2 + \cdots + x_n \leq k$
At least k of $x_1, x_2, \cdots, x_n$	$x_1 + x_2 + \cdots + x_n \geq k$
$a \rightarrow b$	$a \leq b$
$a = \bar{b}$	$a = 1 - b$
$a \rightarrow \bar{b}$	$a + b \leq 1$
$\bar{a} \rightarrow b$	$a + b \geq 1$
$a \leftrightarrow b$	$a = b$
$a \rightarrow b \wedge c$	$a \leq b$ and $a \leq c$ or, $a \leq \frac{b+c}{2}$
$a \rightarrow b \vee c$	$a \leq b + c$
$b \wedge c \rightarrow a$	$a \geq b + c - 1$
$b \vee c \rightarrow a$	$a \geq \frac{b+c}{2}$
if a then at least k of $x_1, x_2, \cdots, x_n$	$a \leq \frac{x_1+x_2+\cdots+x_n}{k}$
if at least k of $x_1, x_2, \cdots, x_n$ then a	$a \geq \frac{x_1+x_2+\cdots+w_n-(k-1)}{n-(k-1)}$
$a = x_1 \cdot x_2 \cdots x_n$	$a \leq \frac{x_1+x_2+\cdots+x_n}{n}$ and $a \geq x_1 + x_2 + \cdots + x_n - (n-1)$

Table 2: Rules of mapping constraints to linear (in)equalities

Although this may seem correct at the first glance, we observe that the left-hand-side (LHS) will be larger than 1 when more than k of the x variables are 1. Because a can be either 0 or 1, this constraint will be infeasible. In fact, what we really need is to *squash* the LHS to less than 1. Currently, the largest possible value of the left-hand-side is $n - (k-1)$. Therefore, dividing the LHS by $n - (k-1)$ should suffice.

$$\frac{(x_1 + x_2 + \cdots + x_n) - (k-1)}{n - (k-1)} \leq a$$

Let's examine two special cases of this transformation to see if they are correct. Remember $b \vee c \rightarrow a$ is indeed one of these cases, given that $n = 2$ and $k = 1$. The linear inequality $\frac{b+c}{2} \leq a$ is exactly the same as what we derived previously. The other special case is "$x_1 \wedge x_2 \wedge \cdots \wedge x_n \rightarrow a$", which is equivalent to say $k = n$ here. Obviously, $a \geq x_1 + x_2 + \cdots + w_n - (n-1)$ is correct. One interesting observation is that the conjunction of a set of boolean variables is the same as the product of the corresponding binary variables. Therefore, the nonlinear constraint $a = x_1 \cdot x_2 \cdots x_n$ is the same as $a = x_1 \wedge x_2 \wedge \cdots \wedge x_n$. Its linear transformation is therefore $a \geq x_1 + x_2 + \cdots + x_n - (n-1)$ and $a \leq \frac{x_1+x_2+\cdots+x_n}{n}$.

Table 2 summarizes all the transformations we have discussed in this section.

4 Conclusions

Thanks to the theoretical developments of integer linear programming in the last two decades, and the tremendous improvement on hardware and software technology, numerical packages these days are able to solve many integer linear programming problems within very short time, even though ILP is in general NP-hard.

In this report, we have provided an entity and relation problem as example, and discussed several cases for transforming boolean constraints. We hope these illustrations are helpful to remodeling your inference problem, and allow you to take advantage of the numerical LP solvers as well.

References

[1] CPLEX. ILOG, Inc. CPLEX. http://www.ilog.com/products/cplex/, 2003.

[2] C. Guéret, C. Prins, and M. Sevaux. *Applications of optimization with Xpress-MP*. Dash Optimization, 2002. Translated and revised by Susanne Heipcke.

[3] V. Punyakanok, D. Roth, W. Yih, and D. Zimak. Semantic role labeling via integer linear programming inference. In *Proceedings of COLING 2004*, 2004.

[4] V. Punyakanok, D. Roth, W. Yih, D. Zimak, and Y. Tu. Semantic role labeling via generalized inference over classifiers. In *Proceedings of CoNLL 2004*, 2004.

[5] D. Roth and W. Yih. A linear programming formulation for global inference in natural language tasks. In *Proceedings of CoNLL-2004*, pages 1–8, 2004.

[6] The R Project for Statistical Computing. http://www.r-project.org/, 2004.

[7] Xpress-MP. Dash Optimization. Xpress-MP. http://www.dashoptimization.com/products.html, 2003.

Readings on Constrained Conditional Models

[1] E. Althaus, N. Karamanis, and A. Koller. Computing locally coherent discourses. In *ACL*, pages 399–406, Barcelona, Spain, July 2004.

[2] R. Barzilay and M. Lapata. Aggregation via set partitioning for natural language generation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 359–366, New York City, USA, June 2006. Association for Computational Linguistics.

[3] P. Bramsen, P. Deshpande, Y. K. Lee, and R. Barzilay. Inducing temporal graphs. In *EMNLP*, pages 189–198, Sydney, Australia, July 2006. Association for Computational Linguistics.

[4] N. Chambers and D. Jurafsky. Jointly combining implicit constraints improves temporal ordering. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2008.

[5] K. Chang, R. Samdani, A. Rozovskaya, N. Rizzolo, M. Sammons, and D. Roth. Inference protocols for coreference resolution. In *CoNLL Shared Task*, pages 40–44, Portland, Oregon, USA, 2011. Association for Computational Linguistics.

[6] M. Chang, D. Goldwasser, D. Roth, and V. Srikumar. Discriminative learning over constrained latent representations. In *NAACL*, 6 2010.

[7] M. Chang, L. Ratinov, N. Rizzolo, and D. Roth. Learning and inference with constraints. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, July 2008.

[8] M. Chang, L. Ratinov, and D. Roth. Guiding semi-supervision with constraint-driven learning. In *Proc. of the Annual Meeting of the ACL*, pages 280–287, Prague, Czech Republic, Jun 2007. Association for Computational Linguistics.

[9] M. Chang, L. Ratinov, and D. Roth. Constraints as prior knowledge. In *ICML Workshop on Prior Knowledge for Text and Language Processing*, pages 32–39, July 2008.

[10] M. Chang, L. Ratinov, and D. Roth. Structured learning with constrained conditional models. *Machine Learning Journal*, 2012.

[11] M. Chang, V. Srikumar, D. Goldwasser, and D. Roth. Structured output learning with indirect supervision. In *ICML*, 2010.

[12] Y. Chang and M. Collins. Exact decoding of phrase-based translation models through lagrangian relaxation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 26–37, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[13] W. Che, Z. Li, Y. Hu, Y. Li, B. Qin, T. Liu, and S. Li. A cascaded syntactic and semantic dependency parsing system. In *CoNLL*, pages 238–242, Manchester, England, August 2008. Coling 2008 Organizing Committee.

[14] Y. Choi, E. Breck, and C. Cardie. Joint extraction of entities and relations for opinion recognition. In *Proc. of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2006.

[15] J. Clarke, D. Goldwasser, M. Chang, and D. Roth. Driving semantic parsing from the world's response. In *CoNLL*, 7 2010.

[16] J. Clarke and M. Lapata. Constraint-based sentence compression: An integer programming approach. In *Proc. of the COLING/ACL 2006 Main Conference Poster Sessions (ACL)*, 2006.

[17] J. Clarke and M. Lapata. Modelling compression with discourse constraints. In *Proc. of the Conference on Empirical Methods in Natural Language Processing and on Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

[18] J. Clarke and M. Lapata. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research (JAIR)*, 31:399–429, 2008.

[19] Hal Daumé III. Cross-task knowledge-constrained self training. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 680–688, Honolulu, Hawaii, October 2008. Association for Computational Linguistics.

[20] J. DeNero and D. Klein. The complexity of phrase alignment problems. In *Proceedings of ACL-08: HLT, Short Papers*, pages 25–28, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[21] P. Denis and J. Baldridge. Joint determination of anaphoricity and coreference resolution using integer programming. In *Proc. of the Annual Meeting of the North American Chapter of the Association for Computational Linguistics - Human Language Technology Conference (NAACL-HLT)*, 2007.

[22] P. Denis and P. Muller. Predicting globally-coherent temporal structures from texts via endpoint inference and graph decomposition. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

[23] P. Deshpande, R. Barzilay, and D. Karger. Randomized decoding for selection-and-ordering problems. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 444–451, Rochester, New York, April 2007. Association for Computational Linguistics.

[24] Q. Do, Y. Chan, and D. Roth. Minimally supervised event causality identification. In *EMNLP*, Edinburgh, Scotland, 7 2011.

[25] Q. Do, W. Lu, and D. Roth. Joint inference for event timeline construction. In *EMNLP*, 2012.

[26] K. Filippova and M. Strube. Dependency tree based sentence compression. In *INLG*, 2008.

[27] K. Filippova and M. Strube. Sentence fusion via dependency graph compression. In *EMNLP*, pages 177–185, Honolulu, Hawaii, October 2008. Association for Computational Linguistics.

[28] J. R. Finkel and C. D. Manning. The importance of syntactic parsing and inference in semantic rolelabeling. In *Proc. of the Annual Meeting of the Association for Computational Linguistics - Human Language Technology Conference, Short Papers (ACL-HLT)*, 2008.

[29] K. Ganchev, J. Graça, J. Gillenwater, and B. Taskar. Posterior Regularization for Structured Latent Variable Models. *Journal of Machine Learning Research*, 2010.

[30] U. Germann, M. Jahr, K. Knight, D. Marcu, and K. Yamada. Fast decoding and optimal decoding for machine translation. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, pages 228–235, Toulouse, France, July 2001. Association for Computational Linguistics.

[31] J. V. Graca, K. Ganchev, and B. Taskar. Expectation maximization and posterior constraints. In *NIPS*, volume 20, 2007.

[32] M. Klenner. Grammatical role labeling with integer linear programming. In *EACL*, 2006.

[33] M. Klenner. Enforcing consistency on coreference sets. In *RANLP*, 2007.

[34] M. Klenner. Shallow dependency labeling. In *ACL*, pages 201–204, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[35] P. Koomen, V. Punyakanok, D. Roth, and W. Yih. Generalized inference with multiple semantic role labeling systems (shared task paper). In Ido Dagan and Dan Gildea, editors, *Proc. of the Annual Conference on Computational Natural Language Learning (CoNLL)*, pages 181–184, 2005.

[36] R. McDonald. A study of global inference algorithms in multi-document summarization. In *ECIR*, 2007.

[37] V. Punyakanok, D. Roth, W. Yih, and D. Zimak. Semantic role labeling via integer linear programming inference. In *Proc. the International Conference on Computational Linguistics (COLING)*, pages 1346–1352, Geneva, Switzerland, August 2004.

[38] V. Punyakanok, D. Roth, W. Yih, and D. Zimak. Learning and inference over constrained output. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2005.

[39] V. Punyakanok, D. Roth, W. Yih, D. Zimak, and Y. Tu. Semantic role labeling via generalized inference over classifiers (shared task paper). In Hwee Tou Ng and Ellen Riloff, editors, *Proc. of the Annual Conference on Computational Natural Language Learning (CoNLL)*, pages 130–133, 2004.

[40] S. Riedel and J. Clarke. Incremental integer linear programming for non-projective dependency parsing. In *EMNLP*, pages 129–137, Sydney, Australia, July 2006. Association for Computational Linguistics.

[41] N. Rizzolo and D. Roth. Modeling Discriminative Global Inference. In *Proc. of the First International Conference on Semantic Computing (ICSC)*, pages 597–604, Irvine, California, September 2007. IEEE.

[42] N. Rizzolo and D. Roth. Learning based java for rapid development of nlp systems. In *LREC*, Valletta, Malta, 5 2010.

[43] D. Roth. Learning based programming. 2005.

[44] D. Roth and W. Yih. Integer linear programming inference for conditional random fields. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 737–744, 2005.

[45] D. Roth and W. Yih. Global inference for entity and relation identification via a linear programming formulation. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.

[46] A.M. Rush, D. Sontag, M. Collins, and T. Jaakkola. On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1–11. Association for Computational Linguistics, 2010.

[47] K. Sagae, Y. Miyao, and J. Tsujii. Hpsg parsing with shallow dependency constraints. In *ACL*, pages 624–631, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[48] R. Samdhani, M. Chang, and D. Roth. Unified expectation maximization. In *NAACL*, 6 2012.

[49] V. Srikumar, G. Kundu, and D. Roth. On amortizing inference cost for structured prediction. In *EMNLP*, 2012.

[50] V. Srikumar and D. Roth. A joint model for extended semantic role labeling. In *EMNLP*, Edinburgh, Scotland, 2011.

[51] T.H. Tsai, C.W. Wu, Y.C. Lin, and W.L. Hsu. Exploiting full parsing information to label semantic roles using an ensemble of ME and SVM via integer linear programming. In *CoNLL*, pages 233–236, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.