Structured Learning with Constrained Condition Models

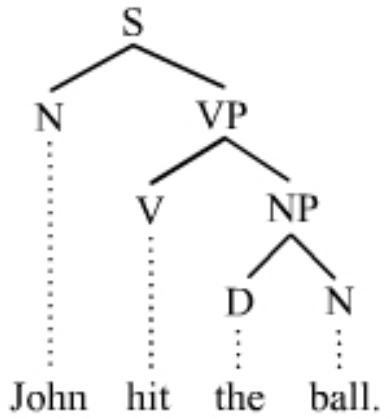
Ming-Wei Chang, Lev Ratinov and Dan Roth

Presenter: Sihao Chen
CIS-700 Spring 2020

Structured Prediction

- Classification problems with structured, (usually) interdependent outputs

- Structured Prediction is everywhere in NLP
  - Parsing
  - Sequence Labeling
    - E.g. Named Entity Recognition (NER)



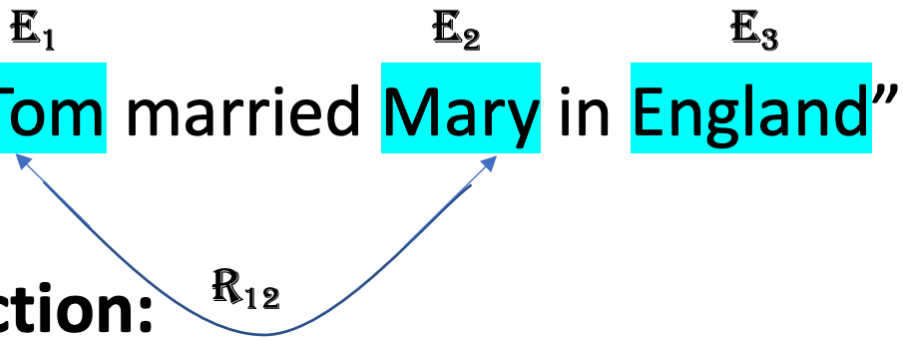
Constituency-based parse tree

- In fact, not just in NLP…

y	U-ORG	O	U-PER	O	O	U-LOC
x	U.N	Official	Ekeus	heads	for	Baghdad

Example 1 – Entity & Relation Extraction

- **Task:** Identify Entities and Relations
- **Sentence:** “Tom married Mary in England”



- **Entity Extraction:**
  - (Tom, *person*)
  - (Mary, *person*)
  - (England, *location*)
- **Relation Extraction:**
  - *Marry*(Tom, Mary)

The relation tag “marry" is **constrained** by the two entity labels for “Tom” and “Mary”

Borrowed From Aditya Kashyap’s presentation earlier this semester. “A Linear Programming Formulation for Global Inference in Natural Language Tasks”

Example 2 – Syntactic Parsing

- **Input**: A sentence
- **Output**: A sequence representing its syntactic tree

Input: John kissed Mary

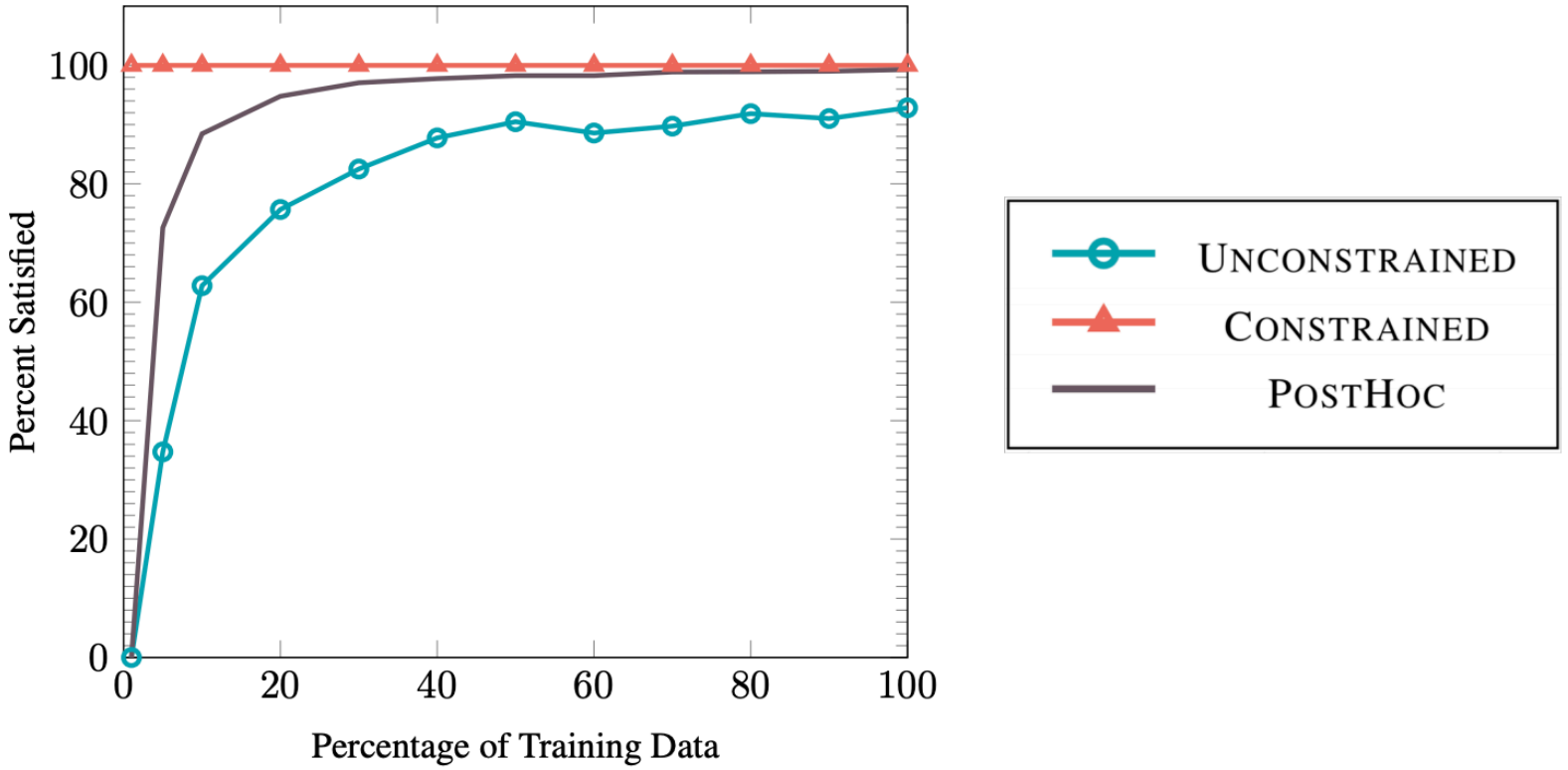
Gold Parse: (S (NP XX) (VP XX (NP XX)))

Invalid Parse: (S (NP) (VP XX XX (NP XX))) ✖

Empty phrase

Example 2 – Syntactic Parsing

- Example model: A seq2seq parser (outputs the tree sequence from Left to Right)
- Only >90% of the outputs are valid (but not necessarily correct) parse trees
  - (When we <100% training data, it's even worse!)



"A General-Purpose Algorithm for Constrained Sequential Inference" Deutsch, Upadhyay, Roth. 2019

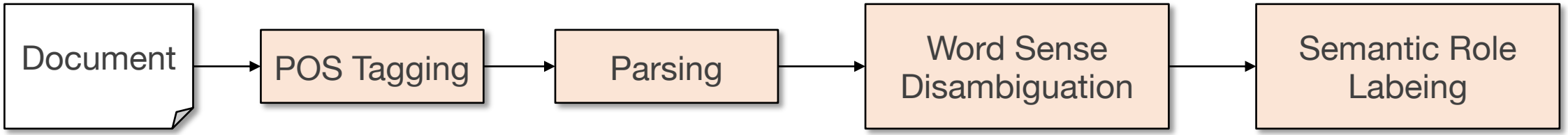
Constrained Conditional Model – Three Motivations

1. Separate modeling and problem formulation from algorithms — Modeling
2. Keep model simple, make expressive decisions via constraints — Inference
3. Expressive structured decisions can be supported by simply learned models — Learning
   - ☐ Do Global Inference on top of simple models

(As we will get into more detailed explanations in later slides)

Constrained Conditional Model – Three Motivations

- Let's revisit the pipeline example –



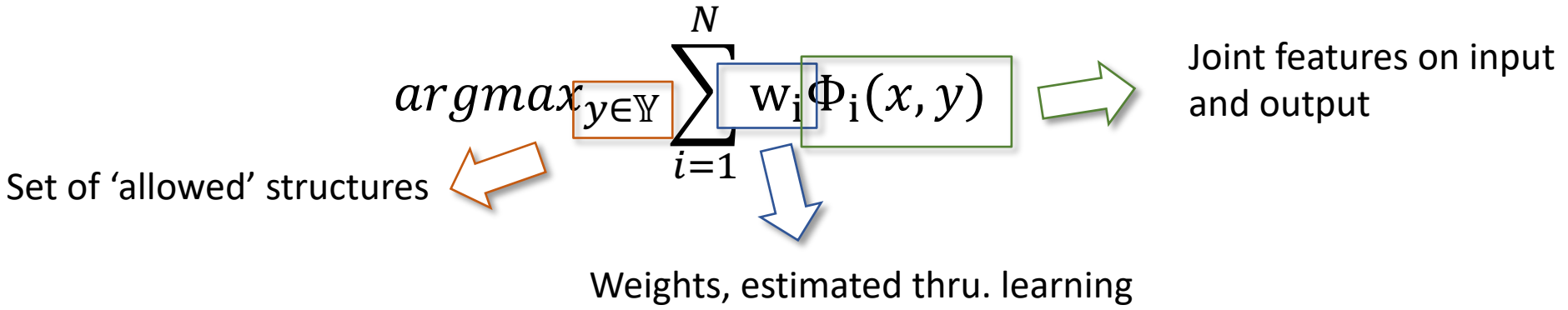
- Later stages can't go back and correct the previous results
- Easier to learn model for each individual task. Difficult to learn jointly.
- What if we impose constraints at decision time?

Ingredients of Structured Prediction

Ingredients	Example: Part-of-Speech tagging
■ Input x	x is the input sentence
■ Output Structure y	y is the output PoS tag sequence
■ A list of Feature Extractors $\Phi_i(x, y)$	
■ Inference □ e.g. $argmax_y \sum_{i=1}^{N} w_i \Phi_i(x, y)$	Finding the ”best” sequence of PoS tags

Structured Prediction - Inference

- Inference is expressed as a maximization of a scoring function



- Labels y_i are interdependent. You can't greedily predict from left to right.

- When given an example $x \in \mathbb{X}$, Inference requires looking at all $y \in \mathbb{Y}$ at decision time (for exact inference)
  - Exact inference is NP-hard in general
  - Sometimes done through approximation. E.g. LP relaxation, Beam search, A*

Constrained Conditional Model

$$argmax_y \sum_{i=1}^{N} \mathrm{w_i} \Phi_\mathrm{i}(x, y)$$

Given K "constraints" C_i for i $\in [1, K]$

Suppose we define a "violation function" $d\left(y, 1_{C_i(x,y)}\right)$, which measures the degree to which the current output structure y violates the constraint C_i.

Constrained Conditional Model

$$argmax_y \sum_{i=1}^{N} w_i \Phi_i(x,y) - \boxed{\sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x,y)})}$$

Before we get into explanation, there are still problems with the objective…

- Only expresses "soft" constraints, what about "hard" constraints?
  - (Technically you can do it by setting ρ to $+\infty$, but that way "hard" constraints overshadow the "soft" ones)

Constrained Conditional Model

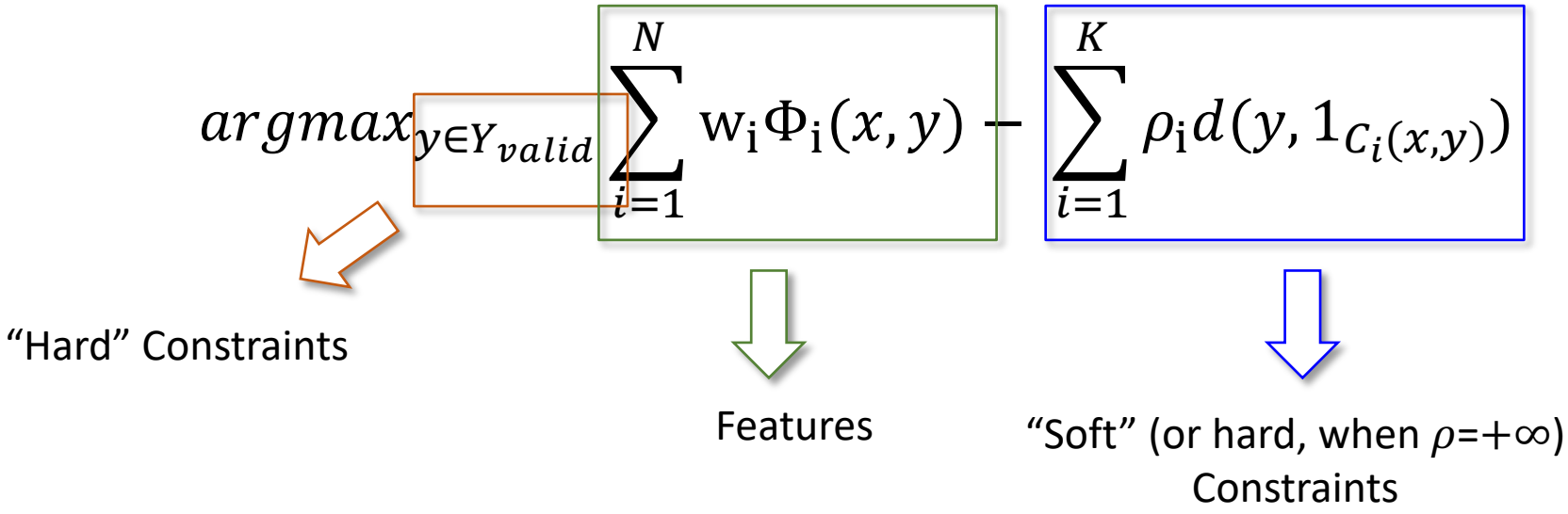
$$argmax_{y \in Y_{valid}} \sum_{i=1}^{N} \mathrm{w_i} \Phi_\mathrm{i}(x, y) - \sum_{i=1}^{K} \rho_\mathrm{i} d(y, 1_{C_i(x,y)})$$

Before we get into explanation, there are still problems with the objective…

- Only expresses "soft" constraints, what about "hard" constraints?
  - "Hard" constraints can be imposed directly on the output space $Y \rightarrow Y_{valid}$
  - E.g. can be formulated as an ILP problem

$$argmax_y \sum_{i=1}^{N} 1_{\{\Phi(x,y)\}} W_i$$
$$subject\ to\ C_j\ for\ j \in [1, k]$$

Constrained Conditional Model



Observation: The "soft"constraints term seems similar to the features term.

Key Question

What's the benefit of separating features from constraints?

Benefit #1: Using Constraints as Supervision

$$argmax_{y \in Y_{valid}} \sum_{i=1}^{N} \mathrm{w_i} \Phi_\mathrm{i}(x, y) - \sum_{i=1}^{K} \rho_\mathrm{i} d(y, 1_{C_i(x,y)})$$

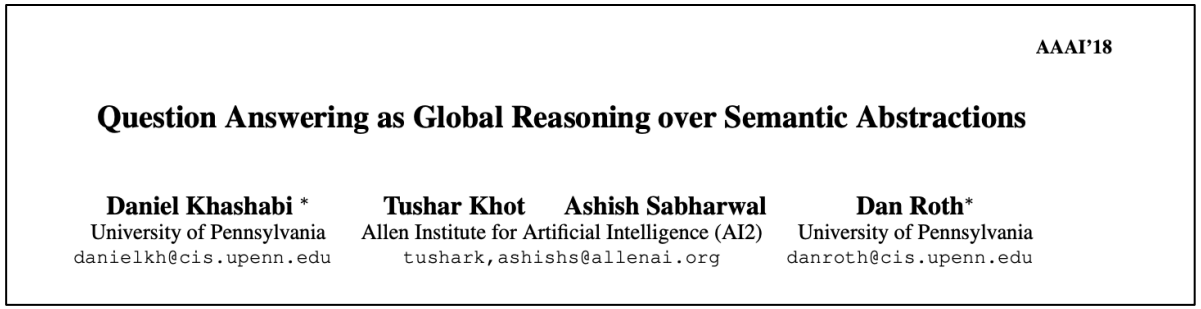
- In scarce labeled data scenario, this becomes very important
- Constraints are more reliable. And more importantly we as human have a sense of "how reliable it is"
  - The penalty parameter ρ can be set manually, doesn't have to be learned

Benefit #2: Separating "Modeling" from the "Problem"

$$argmax_{y \in Y_{valid}} \sum_{i=1}^{N} w_i \Phi_i(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x,y)})$$

- Constraints can be more expressive
  - E.g. $C_i(x, y)$ can be written as first-order logic expression

- Constraints are more suitable for "defining the problem formulation", while features doesn't have to be "problem specific".
  - We've seen the same ideas from many previous presentations before



AAAI'18

Question Answering as Global Reasoning over Semantic Abstractions

Daniel Khashabi *
University of Pennsylvania
danielkh@cis.upenn.edu

Tushar Khot **Ashish Sabharwal**
Allen Institute for Artificial Intelligence (AI2)
tushark,ashishs@allenai.org

Dan Roth*
University of Pennsylvania
danroth@cis.upenn.edu

Benefit #2: Separating "Modeling" from the "Problem"

$$argmax_{y \in Y_{valid}} \sum_{i=1}^{N} w_i \Phi_i(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x,y)})$$

- People usually model problems in the way that the **feature functions Φ really only depend on x.**
- The dependency on y is expressed via $C_i(x, y)$ as structural constraints

Benefit #3: Efficiency

$$argmax_{y \in Y_{valid}} \sum_{i=1}^{N} w_i \Phi_i(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

- For most problems, the number of features N is fewer than the number of constraints K.
  - In such cases, using constraints as supervision is more efficient.

- Also because ρ should always be positive, we could potentially speedup exact inference.

  (E.g. by using A* search)

CCM - Brief Summary

$$argmax_y \, \mathrm{F}(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

Replaced the term with a general notion of "collection of classifiers"

- Introducing CCMs as a formalism that allows us to –
  - Learn simpler models than we would otherwise
  - Make decisions with expressive models, augmented by declarative constraints/knowledge
- Cast NLP problems as CCMs
  - Sequence Tagging (Hidden Markov Model + Global Constraints)
  - Semantic Role Labeling (Independent/Local classifiers + Global Constraints)

Inference is challenging

$$argmax_y \mathrm{F}(x, y) - \sum_{i=1}^{K} \rho_i d(y, 1_{C_i(x)})$$

- Requires discrete optimization
- Naturally this is an Integer Linear Programming (ILP) problem
  - Every structured prediction inference is an ILP.
  - But it doesn't have to be solved as an ILP problem (e.g. if you don't need exact inference)

Quick Revisit - ILP

- 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

$$\max_{y_1,y_2} 8y_1 + 5y_2$$

- Constraints
  - Labor
  - Wood
  - Variable

$$\begin{aligned} y_1 + y_2 &\leq 6 \\ 9y_1 + 5y_2 &\leq 45 \\ y_1, y_2 &\geq 0 \\ y_1, y_2 &\text{ integers} \end{aligned}$$

We cannot build fractional tables or chairs!

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

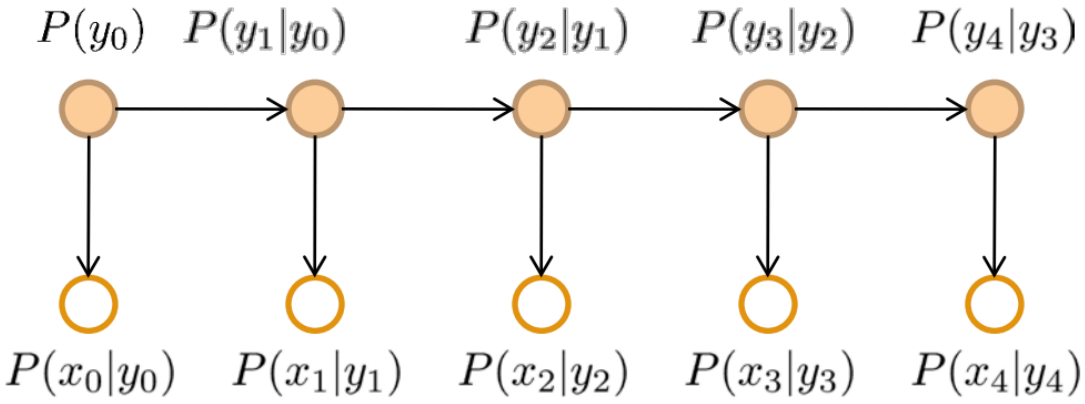
CCM example – Sequence Labeling

- Hidden Markov Model
  - (Here y would a a single label instead of structure)

$$argmax_y \, P(y_0)P(\mathrm{x}_0|y_0)\prod_{i=1}^{T} P(y_i|y_{i-1})P(x_i|y_i)$$

- Models the joint probability $P(x, y)$ under independence assumptions (next prediction only looks at previous prediction)

- Only captures local relationships



CCM example – Sequence Labeling

- Hidden Markov Model
  - (Here y would a a single label instead of structure)

$$argmax_y \, P(y_0)P(\mathrm{x}_0|y_0)\prod_{i=1}^{T} P(y_i|y_{i-1})P(x_i|y_i)$$

- Step #1 – Represent the objective in ILP formulation –
  - You don't have to read this too carefully…

$$maximize \sum_{y\in Y} \lambda_{o,y} 1_{\{y_o=y\}} + \sum_{i=1}^{n-1}\sum_{y\in Y}\sum_{y'\in Y} \lambda_{i,y,y'} 1_{\{y_i=y \wedge y_{i-1}=y'\}}$$

(Learned parameters)

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

CCM example – Sequence Labeling

$$Maximize \sum_{y \in Y} \lambda_{o,y} 1_{\{y_o = y\}} + \sum_{i=1}^{n-1} \sum_{y \in Y} \sum_{y' \in Y} \lambda_{i,y,y'} 1_{\{y_i = y \wedge y_{i-1} = y'\}}$$

- **Step #2**: Come up with constraints according to the problem
- Let's take Part-of-Speech tagging as example again…
  - Constraint #1: Unique label for each word

$$\sum_{y \in Y} 1_{\{y_0 = y\}} = 1$$

CCM example – Sequence Labeling

$$Maximize \sum_{y \in Y} \lambda_{o,y} 1_{\{y_o = y\}} + \sum_{i=1}^{n-1} \sum_{y \in Y} \sum_{y' \in Y} \lambda_{i,y,y'} 1_{\{y_i = y \wedge y_{i-1} = y'\}}$$

- **Step #2**: Come up with constraints according to the problem
- Let's take Part-of-Speech tagging as example again…
  - Constraint #2: There must be at least one verb in the sentence

$$1_{\{y_0 = 'verb'\}} + \sum_{i=1}^{n-1} \sum_{y \in Y} 1_{\{y_{i-1} = y \wedge y_i = 'verb'\}} \geq 1$$

CCM example – Sequence Labeling

$$Maximize \sum_{y \in Y} \lambda_{o,y} 1_{\{y_o = y\}} + \sum_{i=1}^{n-1} \sum_{y \in Y} \sum_{y' \in Y} \lambda_{i,y,y'} 1_{\{y_i = y \wedge y_{i-1} = y'\}}$$

Subject to:

(1 PoS tag per word)

$$\sum_{y \in Y} 1_{\{y_0 = y\}} = 1$$

You can write many declarative constraints like this, as 'prior knowledge' for the problem

(>= 1 verb)

$$1_{\{y_0 = 'verb'\}} + \sum_{i=1}^{n-1} \sum_{y \in Y} 1_{\{y_{i-1} = y \wedge y_i = 'verb'\}} \geq 1$$

.....

CCM – Empirical Evaluation

- Citation Field Parsing

(a) [*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 .]

(b) [*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 .]

CCM – Empirical Evaluation

- Citation Field Parsing – Constraints used

Table 1 The list of constraints used in the citations domain. Some constraints are relatively difficult to represents in traditional models

Citations	
Start	The citation can only start with author or editor.
AppearsOnce	Each field must be a consecutive list of words, and can appear at most once in a citation.
Punctuation	State transitions must occur on punctuation marks.
BookJournal	The words *proc, journal, proceedings, ACM* are *JOURNAL* or *BOOKTITLE*.
Date	Four digits starting with 20xx and 19xx are *DATE*.
Editors	The words *ed, editors* correspond to *EDITOR*.
Journal	The word *journal* are *JOURNAL*.
Note	The words *note, submitted, appear* are *NOTE*.
Pages	The words *pp., pages* correspond to *PAGE*.
TechReport	The words *tech, technical* are *TECH_ REPORT*.
Title	Quotations can appear only in titles.
Location	The words *CA, Australia, NY* are *LOCATION*.

CCM – Empirical Evaluation

- Results
  - With small amount of training examples

    Supervision from constraints > Supervision from examples

# labeled samples	Supervised		Semi-supervised	
	HMM	HMM^{CCM}	HMM	HMM^{CCM}
Citations				
5	58.48	71.64 (31.69 %)	64.55	77.65 (36.96 %)
10	63.37	75.44 (32.94 %)	69.86	81.51 (38.67 %)
20	70.78	81.15 (35.49 %)	75.35	85.11 (39.61 %)
300	86.69	93.92 (54.29 %)	87.89	94.32 (53.07 %)

CCM – Hard vs. Soft Constraints

- Soft-constraints work better when there's sufficient training data

Table 6 Comparison of using hard and soft constraints in semi-supervised learning

Training samples	5	10	20	300
(a)-Citations				
semi-HMM^{CCM}	77.65	81.51	85.11	94.32
semi-HMM$^{CCM}_{\infty}$	78.18	81.11	85.16	92.80
(b)-Advertisement				
semi-HMM^{CCM}	70.79	75.40	77.56	82.00
semi-HMM$^{CCM}_{\infty}$	69.91	73.46	75.25	79.59

All Constraints are hard ($\rho = +\infty$)

Constraints are not perfect! Some training examples will violate the constraints.

CCM in 2020? Challenges? Opportunities?

- We try to solve everything in Neural Networks these days.
- From a pure inference perspective, declarative constraints are almost always good to have
- What are the challenges in incorporating constraints in NN training?
  - If we want to do inference based training, how do we write a differentiable $d(y, 1_{C_i(x)})$

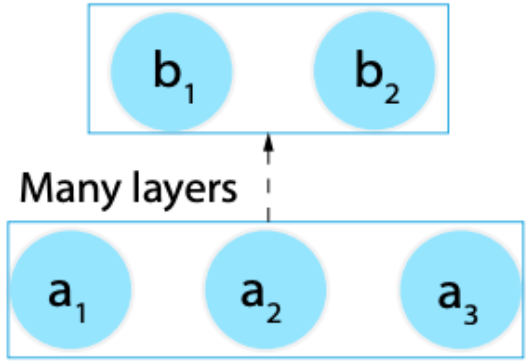
Augmenting Neural Networks with First-order Logic

Tao Li
University of Utah
tli@cs.utah.edu

Vivek Srikumar
University of Utah
svivek@cs.utah.edu

Incorporating declarative constraints in NN

- For a NN, suppose have a set of **named neurons**, for which you know their **semantic implications** (with respect to the problem).
- You can write declarative constraints in the form of first-order logic expressions over the named neurons.
- This is analogous to C_i and $d(y, 1_{C_i(x)})$ we saw in CCM.
- The challenge here is to make $d(y, 1_{C_i(x)})$ differentiable.



Incorporating declarative constraints in NN

- **Idea:** Use T-norm fuzzy logic to represent $d(y, 1_{C_i(x)})$

  - differentiable distance functions to measure "how much a constraint is violated"

Constrained Neural Layers Our goal is to augment the computation of y so that whenever Z is true, the pre-activated value of y increases if the literal Y is not negated (and decreases if it is). To do so, we define a *constrained neural layer* as

$$y = g\left(\mathbf{W}\mathbf{x} + \rho d\left(\mathbf{z}\right)\right). \quad (1)$$

(Looks exactly like CCM, doesn't it?)

Antecedent	Distance $d(\mathbf{z})$
$\bigwedge_i Z_i$	$\max(0, \sum_i z_i - \lvert Z\rvert + 1)$
$\bigvee_i Z_i$	$\min(1, \sum_i z_i)$
$\neg \bigvee_i Z_i$	$\max(0, 1 - \sum_i z_i)$
$\neg \bigwedge_i Z_i$	$\min(1, N - \sum_i z_i)$

Table 1: Distance functions designed using the Łukasiewicz T-norm. Here, $|Z|$ is the number of antecedent literals. z_i's are upstream neurons associated with literals Z_i's.

Incorporating declarative constraints in NN

- Example: Machine Reading Comprehension (SQuAD)
  - Consider the attention layer over words in paragraph and question

$K_{i,j}$ word p_i is related to word q_j in ConceptNet via edges {*Synonym*, *DistinctFrom*, *IsA*, *Related*}.

$\overleftarrow{A}_{i,j}$ unconstrained model decision that word q_j best matches to word p_i.

$\overleftarrow{A}'_{i,j}$ constrained model decision for the above alignment.

$$R_1: \quad \forall i, j \in C,\ K_{i,j} \rightarrow \overleftarrow{A}'_{i,j}.$$

$$R_2: \quad \forall i, j \in C,\ K_{i,j} \wedge \overleftarrow{A}_{i,j} \rightarrow \overleftarrow{A}'_{i,j}.$$

Rule 1: If a word in question and a word in paragraph is related in ConceptNet, their attention should be aligned

Rule 2: (Softer version of Rule 1 which also takes model prediction into account)

Results

%Train	L-DAtt	+N_1	+N_2	+N_3	+$N_{2,3}$
1%	61.2	**64.9**	63.9	62.5	64.3
2%	66.5	**70.5**	69.8	67.9	70.2
5%	73.4	76.2	**76.6**	74.0	76.4
10%	78.9	80.1	**80.4**	79.3	80.3
100%	**87.1**	86.9	**87.1**	87.0	86.9

Table 3: Impact of constraints on L-DAtt network. Each score represents the average accuracy on SNLI test set among 3 random runs. For both N_1 and N_2, we set $\rho = (8, 8, 8, 8, 4)$ for the five different percentages. For the noisy constraint N_3, $\rho = (2, 2, 1, 1, 1)$.

Conclusion

- CCM is a framework that augments simpler, (linear) models with expressive declarative constraints.
- Declarative constraints can not only be used at decision time, it can also be viewed as a form of supervision for learning.
- (Li and Srikumar, 2019) is a good attempt in using declarative constraints to drive learning in Neural Networks
- Separating models from problem formulation seems to be a key intuition that we keep re-visiting in this class.