Injecting output constraints into neural NLP models

Jay Yoon Lee

May 19th, 2020

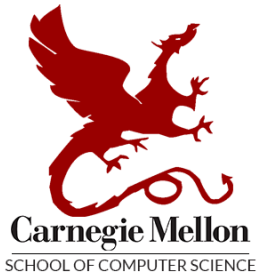
Jaime Carbonell, Co-Chair

William Cohen, Co-Chair

Graham Neubig

Yulia Tsvetkov

Dan Roth (UPenn)



Carnegie Mellon University

What do I mean by “output constraints”?

Injecting **output constraints** into neural NLP models

Constraints?

- **Example of a grammar**
  - I are presenting virtually.
  - I am presenting virtually.

  - The keys is on the table.
  - The keys are on the table.

  - The keys to the cabinet is on the table.
  - The keys to the cabinet are on the table.

Some examples are from
Linzen et al, Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies, 2016

Constraints?

- **Example of a grammar**
  - I are presenting virtually.
  - I am presenting virtually.

  - The keys is on the table.
  - The keys are on the table.

  - The keys to the cabinet is on the table.
  - The keys to the cabinet are on the table.

Constraint violating

If subject, verb pair does not agree. (singular/plural)

Some examples are from
Linzen et al, Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies, TACL 2016

Constraints?

- The constraints I focus here are simple rules on the output space.

- Defined on the discrete sequence output $Y=(y_1, y_2, \ldots, y_T)$
  - Output intrinsic constraint e.g. Subject-Verb agreement.
  - Input, output relation constraint. e.g. Syntactic parsing.
  - Exterior knowledge on the problem. e.g. Syntactic information on SRL.

Motivation

- Enforcing output constraint is important for real service.
  - A minimal condition for a correct output.

Machine Translation

Should preserve meaning.

Subject-verb agreement (singular/ plural).

Morphologically consistent.

Motivation

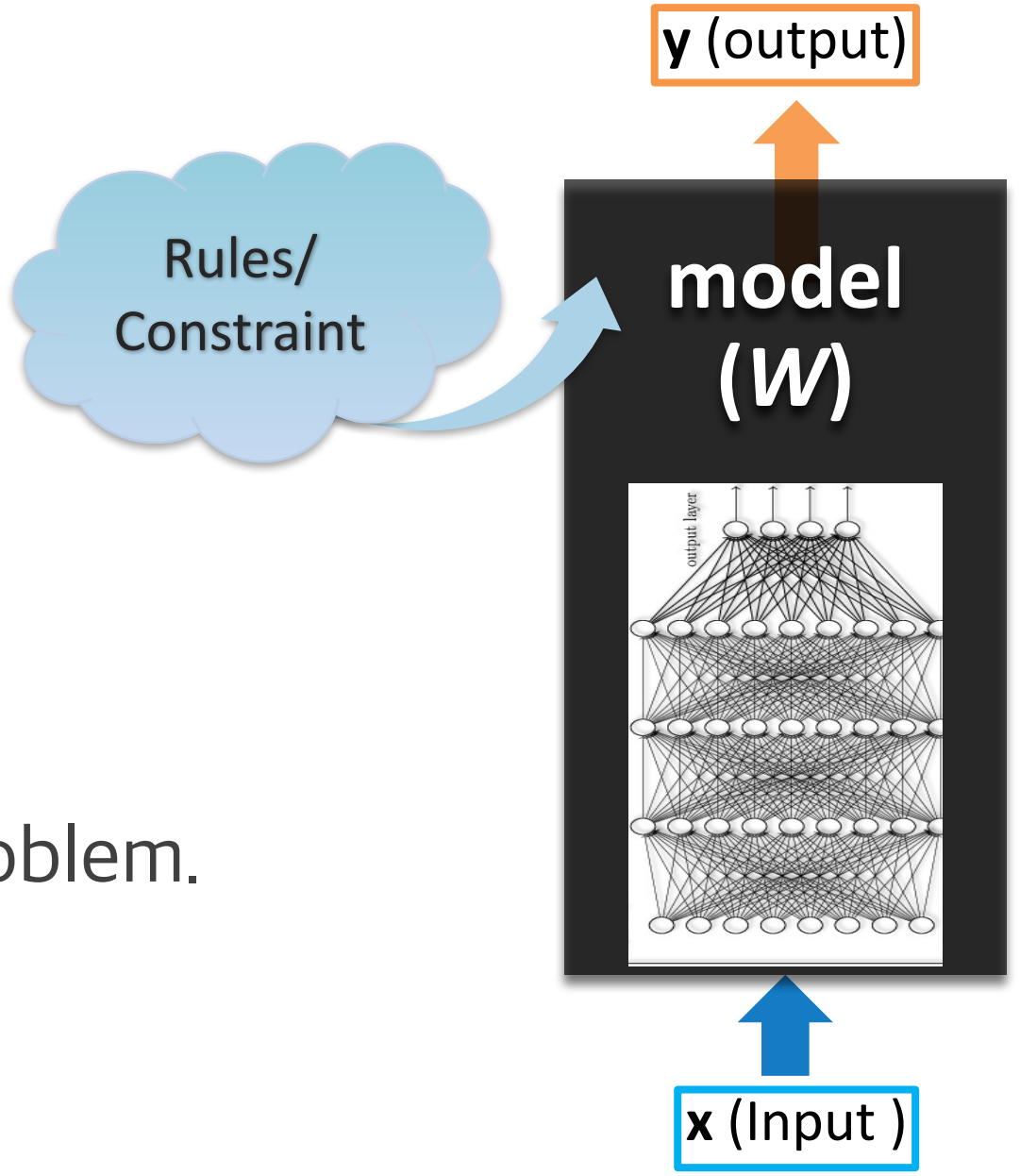
- Enforcing output constraint is important for real service.
  - A minimal condition for a correct output.

- Constraint can serve as an extra signal to learn better model.
  - Structured prediction
    - Rush et al., 2010, Chang et al., 2012, Riedel and McCallum, 2011
  - Learning with constraints
    - Ganchev., 2010, Chang et al., 2012, Samdani et al., 2011, …

Goal of this thesis

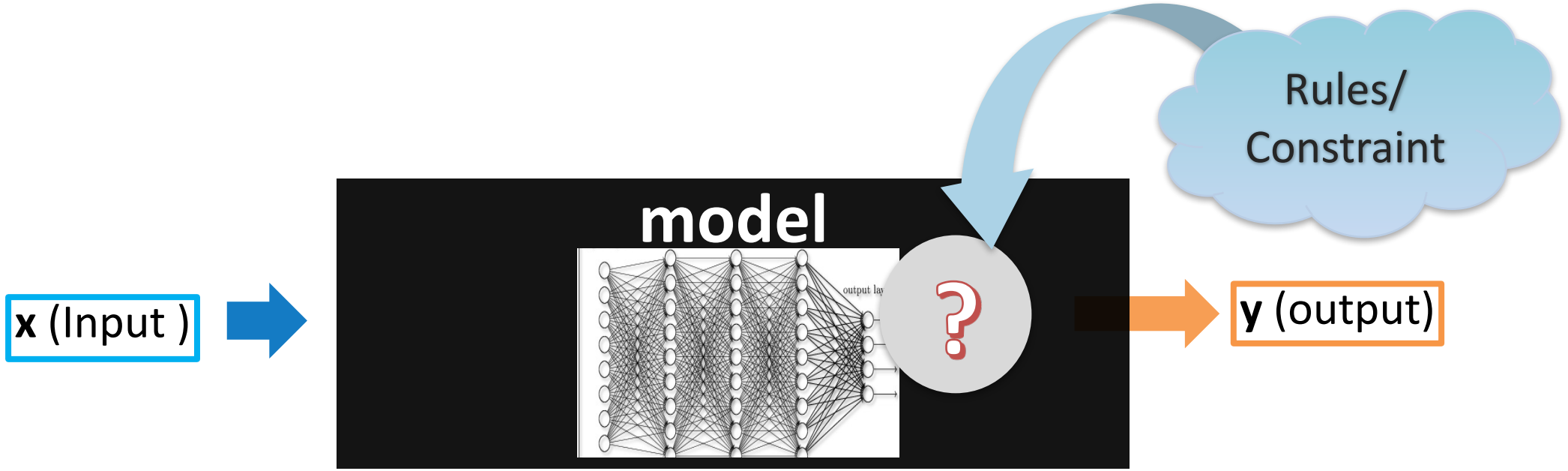
Enforcing output constraint to

1. **Provide** more logical output,
2. **Improve** the model performance,
3. **Overcome** the data **deficiency** problem.



Challenges

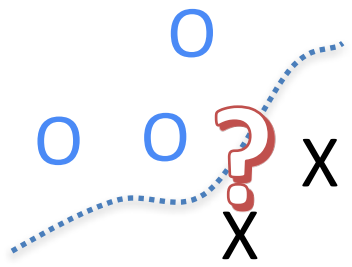
- It is hard to enforce output constraints on neural models.
  - Black box: no explicit relation between model parameters & output



Challenges

- It is hard to enforce output constraints on neural models.
  - Black box: no explicit relation between model parameters & output

- Constraint is not explicit in training data.
  - Consider general classification problems.
  - Training set only consists of *constraint-satisfying* instances.



Challenges

- It is hard to enforce output constraints on neural models.
  - Black box: no explicit relation between model parameters & output
- Constraint is not explicit in training data.
- Thus, constraint is a prior knowledge that we have to inject.

Overall approach

- Inject prior knowledge in the form of a scoring function.
  - *Constraint function* $g(\mathbf{y}, \mathcal{L}^{\mathbf{x}})$ measures the degree of violation.

- Enforcing constraints in a model-agnostic way.
  - For the usability in general ML applications.
  - For the approach to be useful as neural models improve.

Roadmap

- Introduction.
- Applications.
- Inference with constraints. (Ch.3) [Lee, Mehta, Wick, Tristan, Carbonell AAAI19]
- Learning with constraints. (Ch.4) [Mehta*, Lee*, Carbonell EMNLP18]
- Agreement as constraint. (Ch.5) [Lim*, Lee*, Carbonell, Poibeau AAAI20]

* co-first authorship with equal contribution.

Roadmap

- ☑ Introduction.
- Applications.
- Inference with constraints. (Ch.3) [Lee, Mehta, Wick, Tristan, Carbonell AAAI19]
- Learning with constraints. (Ch.4) [Mehta*, Lee*, Carbonell EMNLP18]
- Agreement as constraint. (Ch.5) [Lim*, Lee*, Carbonell, Poibeau AAAI20]

* co-first authorship with equal contribution.

Syntactic parsing (sequence-to-sequence)

- *Syntactic Parsing*
  - Finding structure of the provided (input) sentence.

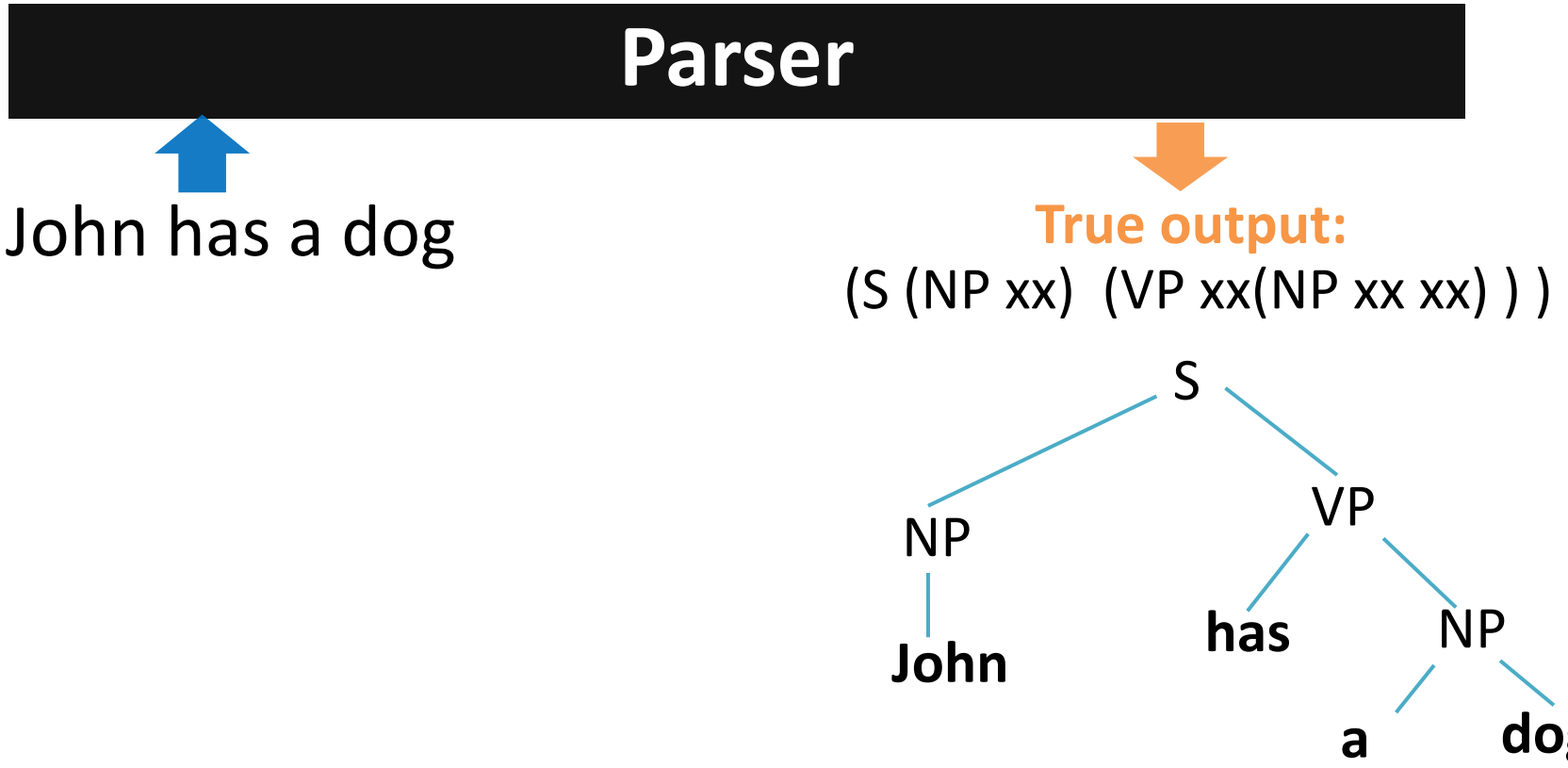
John has a dog

True output:

(S (NP xx) (VP xx(NP xx xx)))

Syntactic parsing (sequence-to-sequence)

- *Syntactic Parsing*
  - Finding structure of the provided (input) sentence.



Syntactic parsing (sequence-to-sequence)

- *Syntactic Parsing*
  - Finding structure of the provided (input) sentence.

John has a dog

True output:

(S (NP xx) (VP xx(NP xx xx)))

Others possible output:

(S (NP xx) (VP xx (NP xx xx)))))
((S (NP xx) (VP xx (NP xx xx))
(S (NP xx)))(VP xx (NP xx xx)))
(S (NP xx xx xx xx xx))
(S (NP xx) (VP xx xx xx))

Syntactic parsing (sequence-to-sequence)

- *Syntactic Parsing*
  - Finding structure of the provided (input) sentence.

John has a dog

True output:

(S (NP xx) (VP xx(NP xx xx)))

Constraint:

"# open - # closing bracket = 0"

Others possible output:

(S (NP xx) (VP xx (NP xx xx)))))

((S (NP xx) (VP xx (NP xx xx))

(S (NP xx)))(VP xx (NP xx xx)))

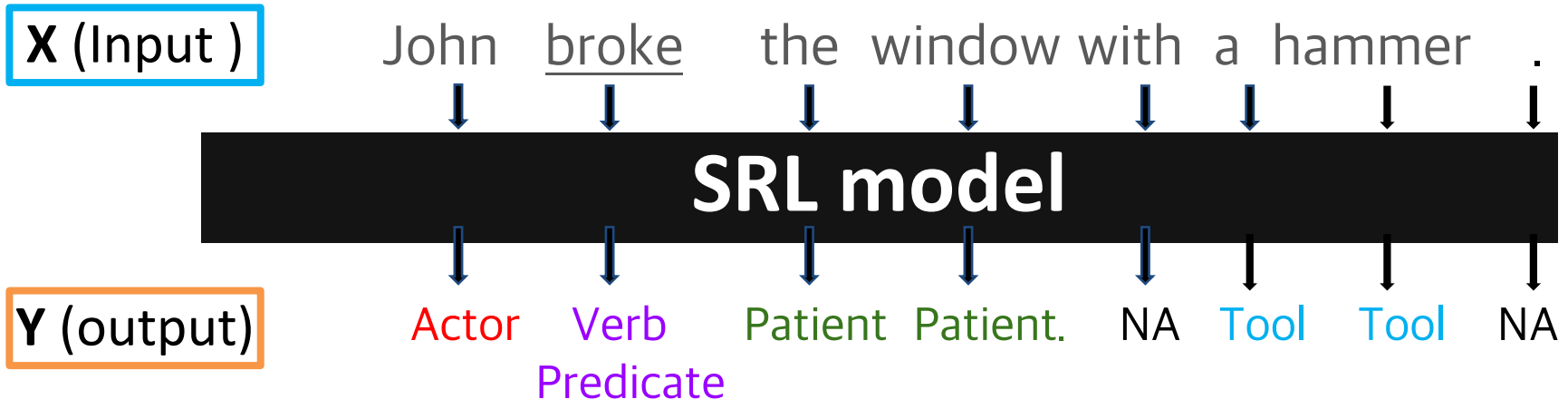
"# of input tokens has to match # of children on the tree"

(S (NP xx xx xx xx xx))

(S (NP xx) (VP xx xx xx))

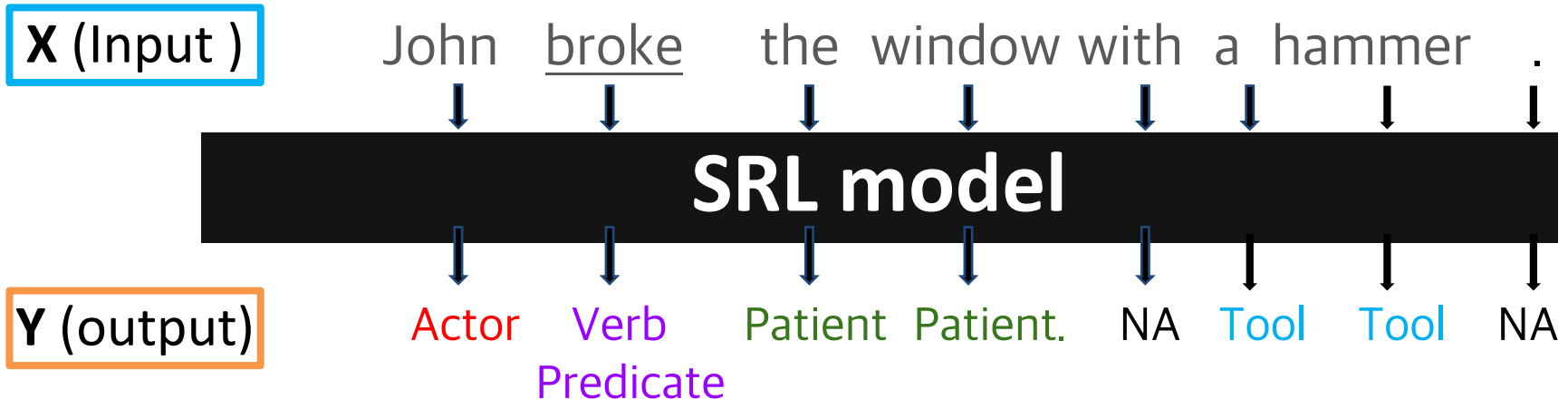
Semantic Role Labeling

- *Semantic Role Labeling (SRL)*
  - Given verb predicate and a sentence, answers *who, what, where, when, why, how* (5W1H)



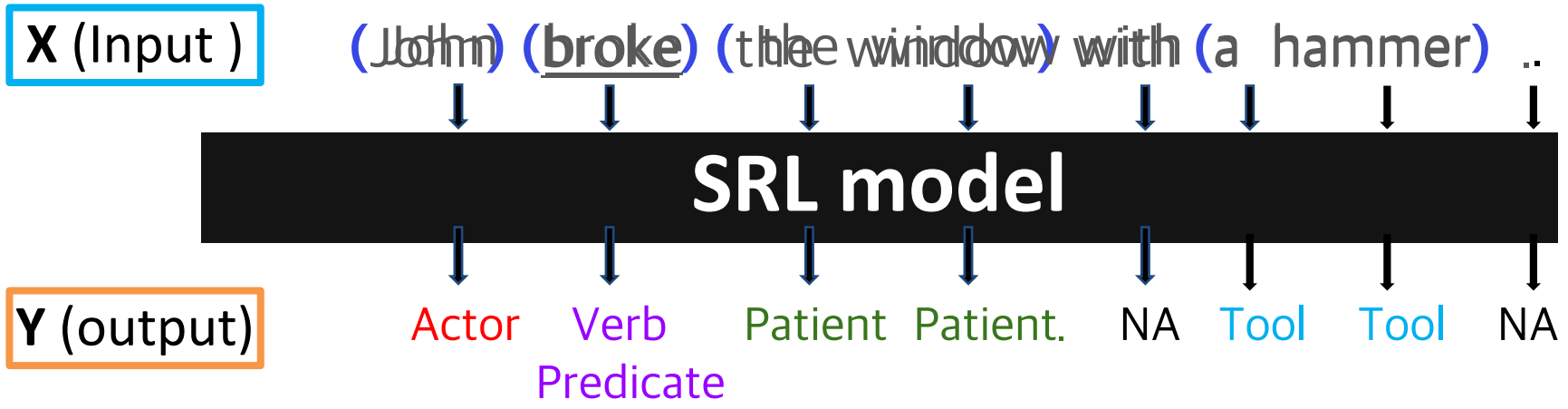
Semantic Role Labeling

- SRL can be viewed as a “span labeling” problem.



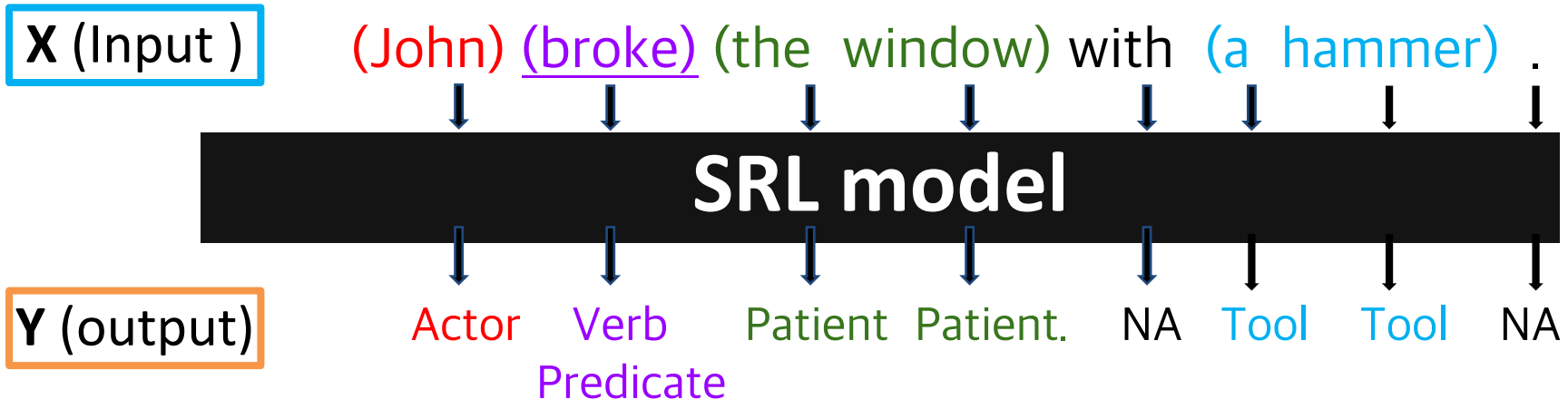
Semantic Role Labeling

- SRL can be viewed as a “span labeling” problem.
  1. “Span” identification (Span= sequence of words.)



Semantic Role Labeling

- SRL can be viewed as a “span labeling” problem.
  1. “Span” identification (Span= sequence of words.)
  2. “Labeling” of provided span.
- Many neural models solve the two jointly.



Syntactic constraint on SRL

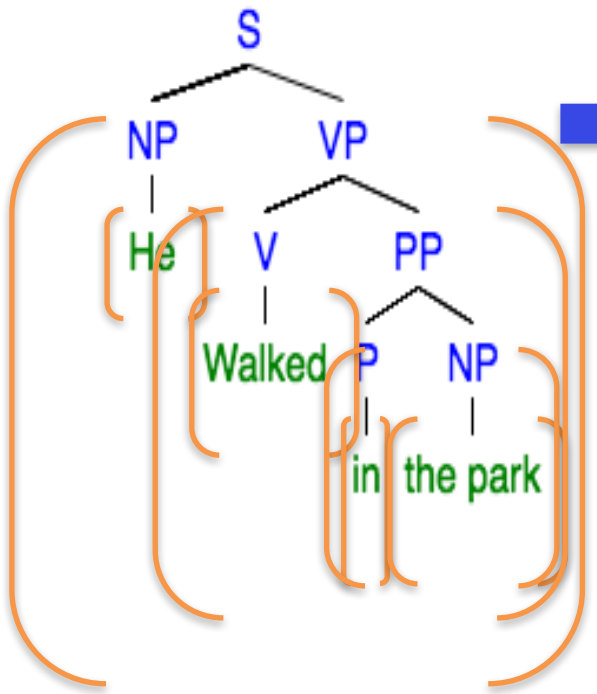
- Traditionally, SRL span candidates were ***limited*** by syntactic tree.

Syntactic constraint on SRL

- Traditionally, SRL span candidates were ***limited*** by syntactic tree.

X (Input)

"He walked in the park"



Possible span candidate

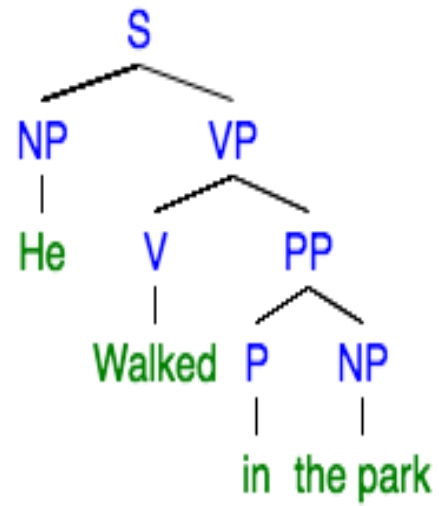
(He), (walked),(in),(the park), (in the park), (walked in the park), (He walked in the park)

Syntactic constraint on SRL

- Traditionally, SRL span candidates were ***limited*** by syntactic tree.

X (Input)

"He walked in the park"



Possible span candidate

(He), (walked),(in),(the park), (in the park), (walked in the park), (He walked in the park)

$\mathcal{L}^x$

Constraint space

Constraint violating

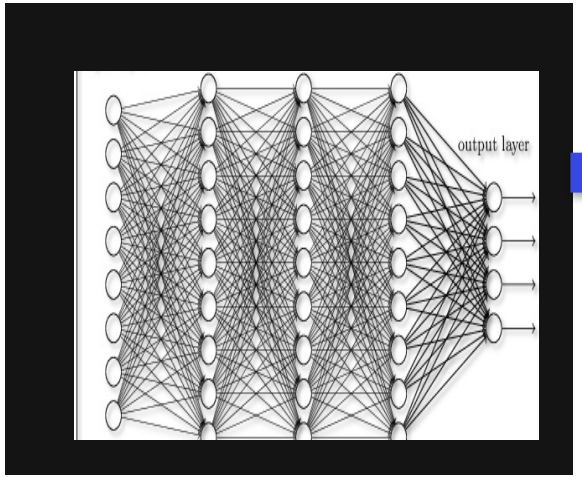
If output has a span that is not on the constraint space

Syntactic constraint on SRL

- How about neural SRL models?

X (Input)

"He walked in the park"



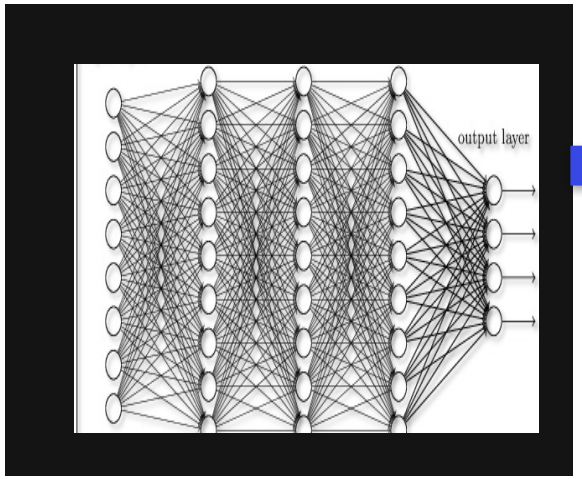
In contrast, neural model without syntactic information can generate **all possible** "spans" given sentence.

Syntactic constraint on SRL

- How about neural SRL models?

X (Input)

"He walked in the park"



Subset of all the possible output space

(He) (walked) (in the park)
(He) (walked in)(the park)
(He walked) (in the park)
(He walked in the)(park)
(He)(walked in the park)
(He walked in the park)

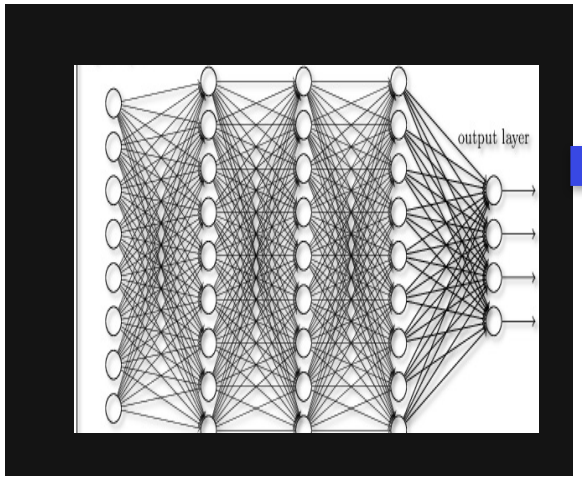
In contrast, neural model without syntactic information can generate **all possible** "spans" given sentence.

Syntactic constraint on SRL

- How about neural SRL models?

X (Input)

"He walked in the park"



Subset of all the possible output space

(He) (walked) (in the park)
(He) (walked in)(the park)
(He walked) (in the park)
(He walked in the)(park)
(He)(walked in the park)
(He walked in the park)

Comparing to the constraint space obtained from the parse tree.

Possible span candidate

(He), (walked),(in),(the park), (in the park), (walked in the park), (He walked in the park)

$\mathcal{L}^{\mathbf{x}}$

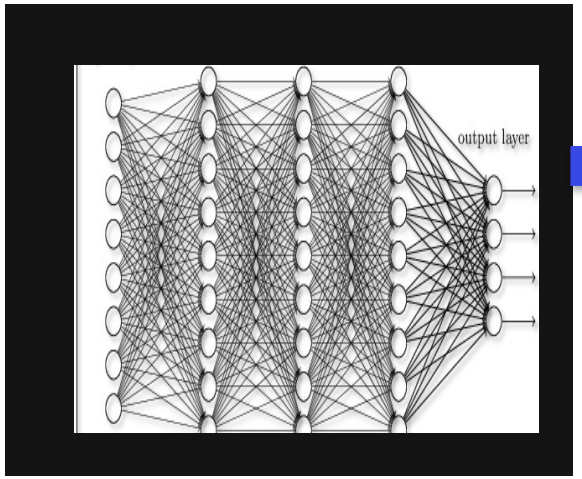
Constraint space

Syntactic constraint on SRL

- How about neural SRL models?

X (Input)

"He walked in the park"



Subset of all the possible output space

(He) (walked) (in the park)

~~(He)~~ ~~(walked in)~~~~(the park)~~

(He walked) (in the park)

(He walked in the)(park)

(He)(walked in the park)

(He walked in the park)

Comparing to the constraint space obtained from the parse tree.

Possible span candidate

(He), (walked),(in),(the park), (in the park), (walked in the park), (He walked in the park)

$\mathcal{L}^{\mathbf{x}}$

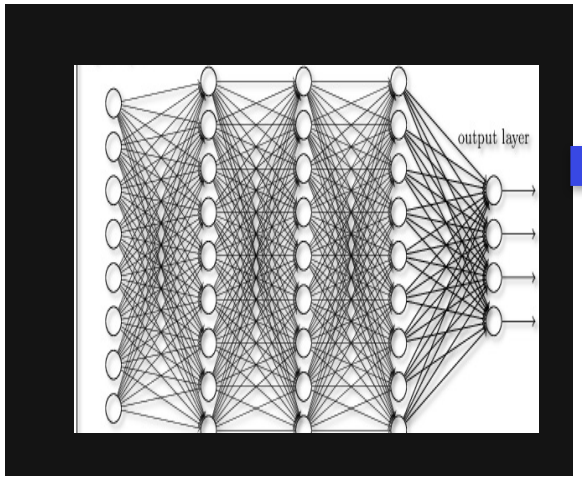
Constraint space

Syntactic constraint on SRL

- How about neural SRL models?

X (Input)

"He walked in the park"



Subset of all the possible output space

(He) (walked) (in the park)

~~(He)~~ ~~(walked in)~~~~(the park)~~

~~(He walked)~~ ~~(in the park)~~

(He walked in the)(park)

(He)(walked in the park)

(He walked in the park)

Comparing to the constraint space obtained from the parse tree.

Possible span candidate

(He), (walked),(in),(the park), (in the park), (walked in the park), (He walked in the park)

$\mathcal{L}^{\mathbf{x}}$

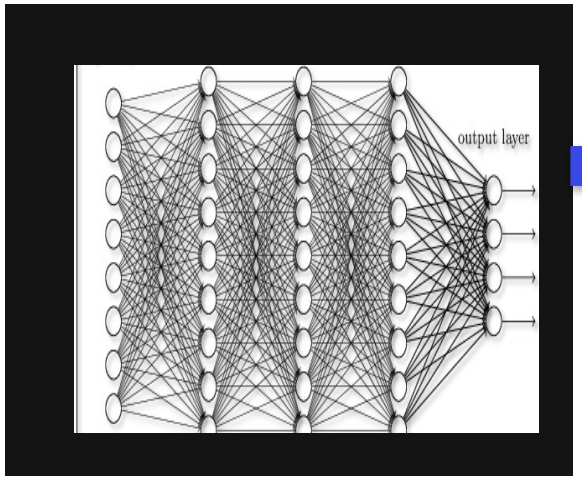
Constraint space

Syntactic constraint on SRL

- How about neural SRL models?

X (Input)

"He walked in the park"



Subset of all the possible output space

(He) (walked) (in the park)

~~(He)~~ ~~(walked in)~~~~(the park)~~

~~(He walked)~~ ~~(in the park)~~

~~(He walked in the)(park)~~

(He)(walked in the park)

(He walked in the park)

Comparing to the constraint space obtained from the parse tree.

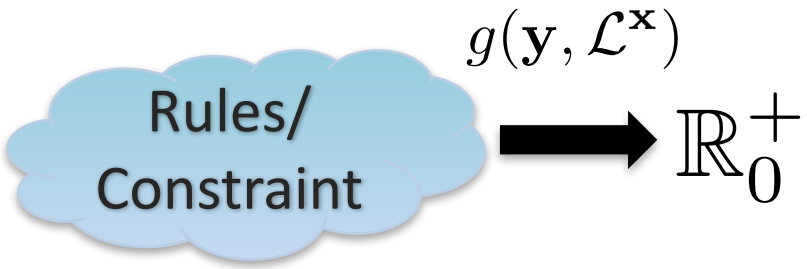
Possible span candidate

(He), (walked),(in),(the park), (in the park), (walked in the park), (He walked in the park)

$\mathcal{L}^{\mathbf{x}}$

Constraint space

Constraint function on SRL.



- Measuring degree of constraint violation.
- g(y, L^x) as *normalized error counts.*

$$g(y, L^x) = \frac{\text{\# of spans not in parse tree}}{\text{\# of total spans emitted}}$$

g(y, L^x)	Subset of all the possible output space
0.0	(He) (walked) (in the park)
0.33	~~(He)~~ ~~(walked in)~~~~(the park)~~
0.5	~~(He walked)~~ ~~(in the park)~~
1.0	~~(He walked in the)(park)~~
0.0	(He)(walked in the park)
0.0	(He walked in the park)

$$\mathbf{y} \in \mathcal{L}^{\mathbf{x}} \longleftrightarrow g(\mathbf{y}, \mathcal{L}^{\mathbf{x}}) = 0$$

Summary of constraint types

Task	model	Source of constraint	Checking constraint
Syntactic Parsing	seq2seq	input-output relation + output-only	Count comparison
SRL	Sequence tagger	Exterior knowledge	Existence in constraint space

How can we enforce constraints to neural models ?

Roadmap

- ☑ Introduction.
- ☑ Applications.
- Inference with constraints. (Ch.3) [Lee, Mehta, Wick, Tristan, Carbonell AAAI19]
  [Lee, Wick, Tristan, Carbonell AKBC17]
- Learning with constraints. (Ch.4) [Mehta*, Lee*, Carbonell EMNLP18]
- Agreement as constraint. (Ch.5) [Lim*, Lee*, Carbonell, Poibeau AAAI20]

* co-first authorship with equal contribution.

Roadmap

- ☑ Introduction.
- ☑ Applications.
- Inference with constraints. (Ch.3) [Lee, Mehta, Wick, Tristan, Carbonell AAAI19]
  [Lee, Wick, Tristan, Carbonell AKBC17]
- Learning with constraints. (Ch.4) [Mehta*, Lee*, Carbonell EMNLP18]
- Agreement as constraint. (Ch.5) [Lim*, Lee*, Carbonell, Poibeau AAAI20]

* co-first authorship with equal contribution.

Overview for “Inference with constraints”

Can we enforce global constraints in the inference process without searching exponential space ?

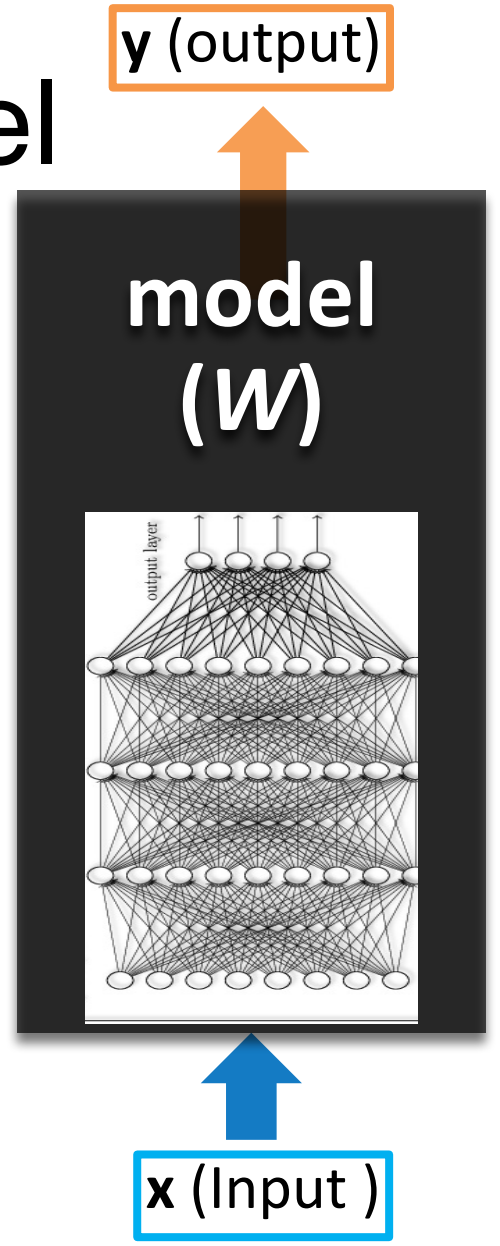
- **Proposed method**
  - Gradient-Based Inference (GBI).
  - Updating model parameter to search constraint-satisfying output.

- **Take away**
  - Constraint violation signal can guide the model weight updates.
  - Leading to better & faster results to competing method.

Training & Inference of a neural model

- **Training**
  - Maximizing log-likelihood of (input, true output) pair using stochastic gradient descent.
    - x_L, y_L : labeled input, output.
    - Ψ: score (log probability, energy)

$$\max_{W} \quad \Psi(\mathbf{x_L}, \mathbf{y_L}, W)$$



Training & Inference of a neural model

- **Training**
  - Maximizing log-likelihood of (input, true output) pair using stochastic gradient descent.
    - x_L, y_L : labeled input, output.
    - Ψ: score (log probability, energy)

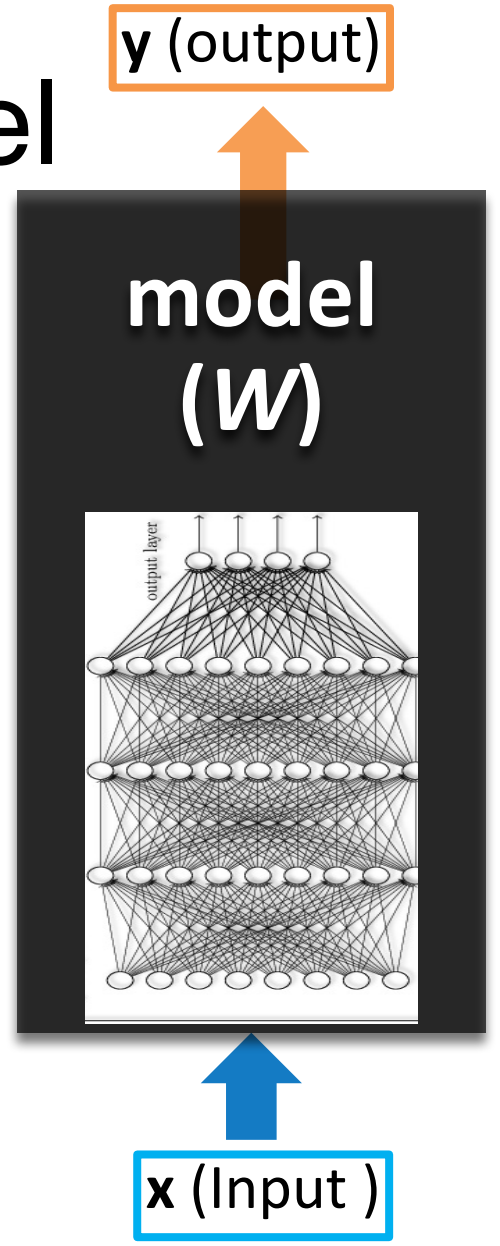
$$\max_{W} \quad \Psi(\mathbf{x_L}, \mathbf{y_L}, W)$$

- **Inference**
  - Finding the output **y** with the highest model score (probability).
    - x: a test input
    - y: length T, vocab size V

$$\max_{\mathbf{y}} \quad \Psi(\mathbf{x}, \mathbf{y}, W)$$

  - Greedy: ***O(TV)***
  - Global constraint
    - ***$O(V^T)$ (A* algorithm is often used)***

Learned model



Gradient-Based Inference (GBI)

- **Training**
  - Maximizing log-likelihood of (input, true output) pair using stochastic gradient descent.
    - x_L, y_L : labeled input, output.
    - Ψ: score (log probability, energy)

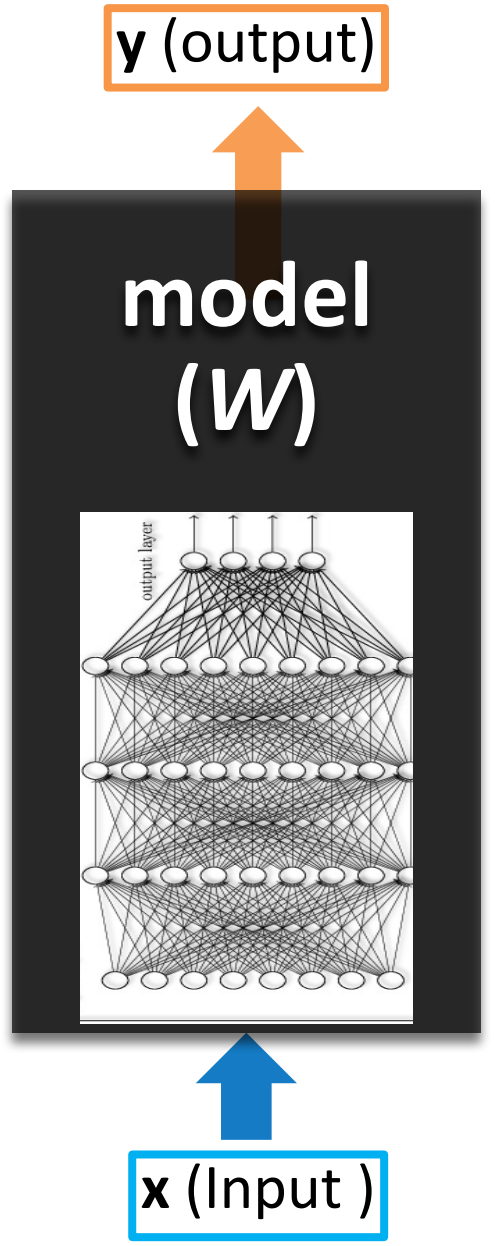
$$\max_{W} \quad \Psi(\mathbf{x_L}, \mathbf{y_L}, W)$$

- **Inference**
  - Finding the output **y** with the highest model score (probability).
    - x: a test input
    - y: length T, vocab size V

$$\max_{\mathbf{y}} \quad \Psi(\mathbf{x}, \mathbf{y}, W)$$

  - Greedy: ***O(TV)***
  - Global constraint
    - ***$O(V^T)$ (A* algorithm is often used)***

Learned model



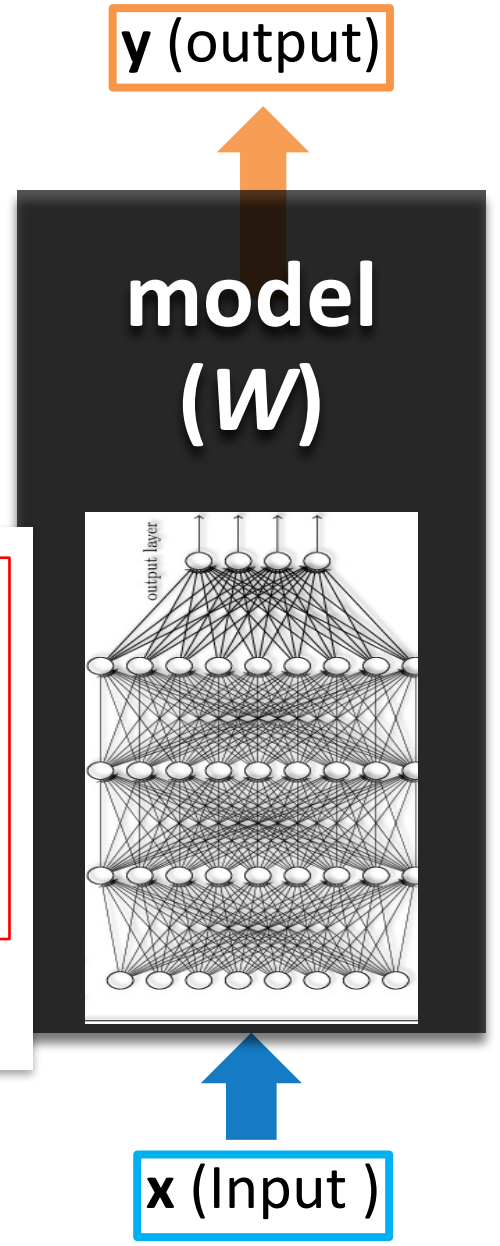
Gradient-Based Inference (GBI)

- **Training**
  - Maximizing log-likelihood of (input, true output) pair using stochastic gradient descent.
    - x_L, y_L : labeled input, output.
    - Ψ: score (log probability, energy)

- **Inference**
  - Finding the output **y**
    - x: a test input
    - y: length T, vocab si
  - Greedy: ***O(TV)***
  - Global constraint
    - ***$O(V^T)$ (A* algorithm is often used)***

Post-processing is often used instead:

Get multiple candidates and filter out constraint-violating candidates.



Gradient-Based Inference (GBI)

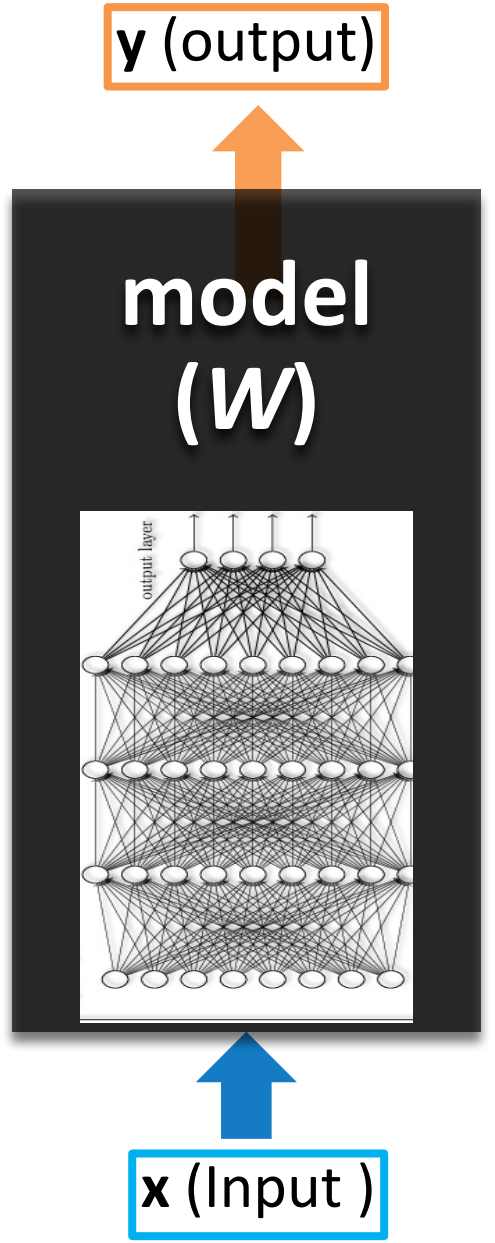
- **Training**
  - Maximizing log-likelihood of (input, true output) pair using stochastic gradient descent.
    - x_L, y_L : labeled input, output.
    - Ψ: score (log probability, energy)

$$\max_{W} \quad \Psi(\mathbf{x_L}, \mathbf{y_L}, W)$$

- **Inference**
  - Finding the output **y** with the highest model score (probability).
    - x: a test input
    - y: length T, vocab size V

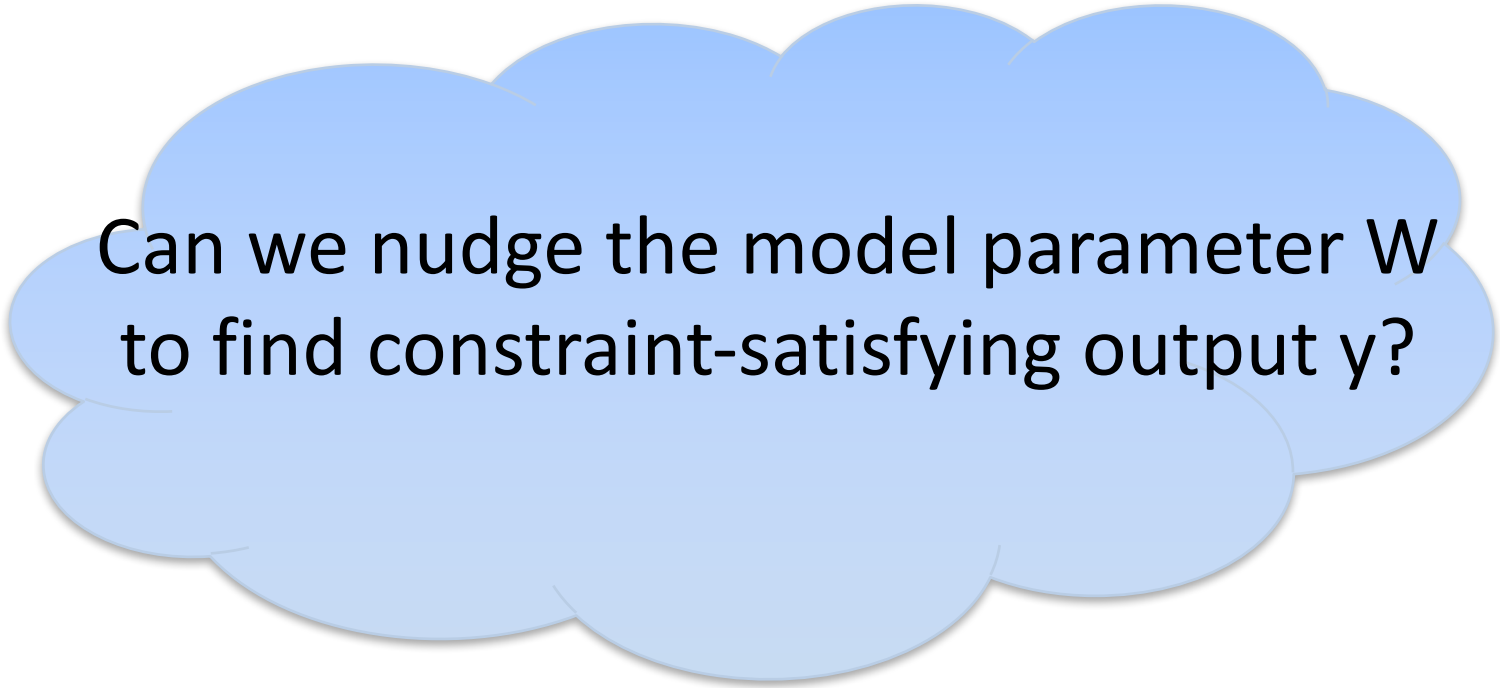
$$\max_{\mathbf{y}} \quad \Psi(\mathbf{x}, \mathbf{y}, W)$$

Can we do "continuous space" search to find constraint-satisfying output y? (like Training)



Key Idea for the method

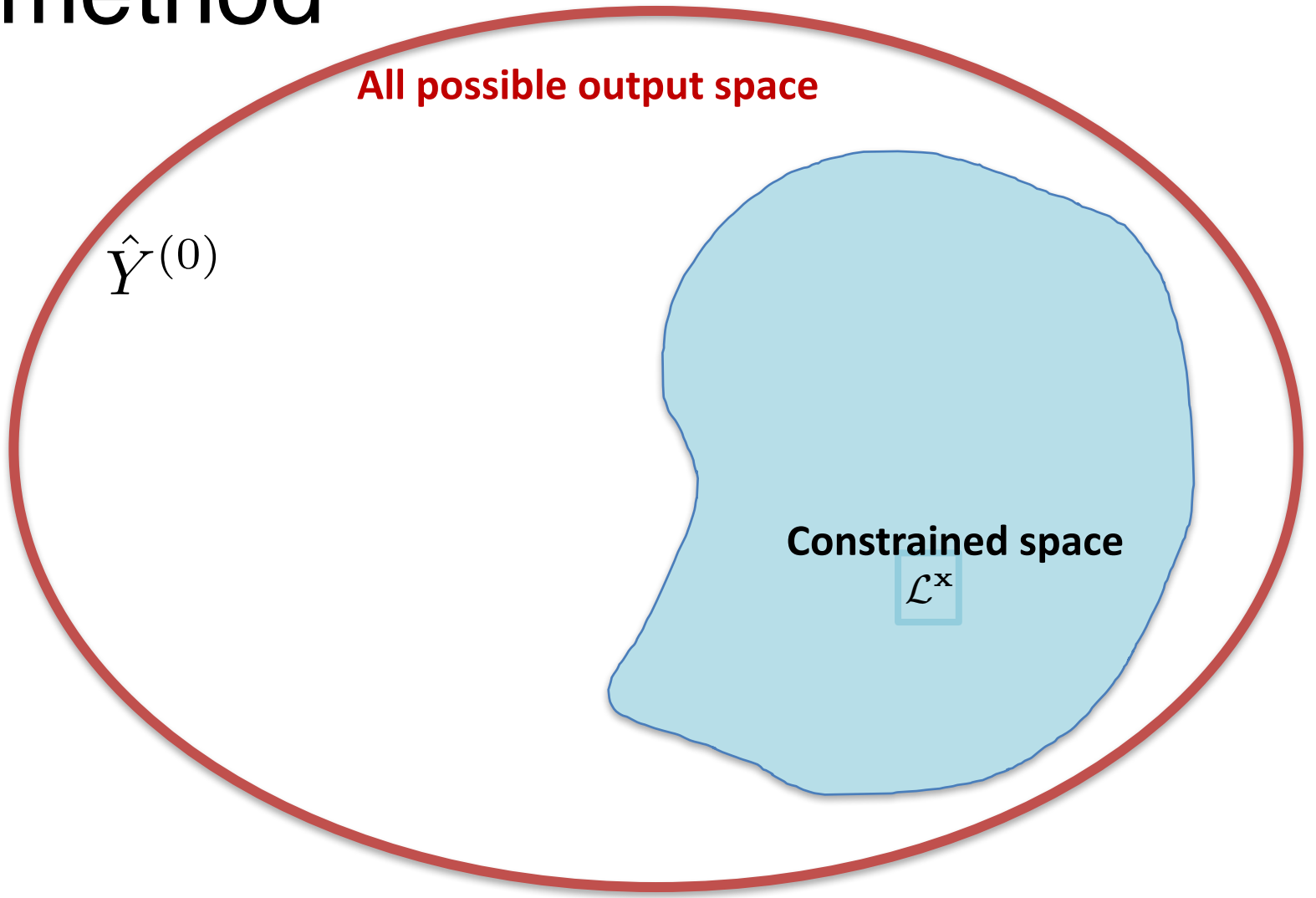
1. Deterministic y given fixed W, x.
2. Model parameter W is in continuous space.
3. Computational architectures for easy gradient updates.



Key Idea for the method

Can we find constraint-satisfying output by updating the model parameter ?

$$\hat{Y}^{(k)} = \arg\max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W^{(k)})$$

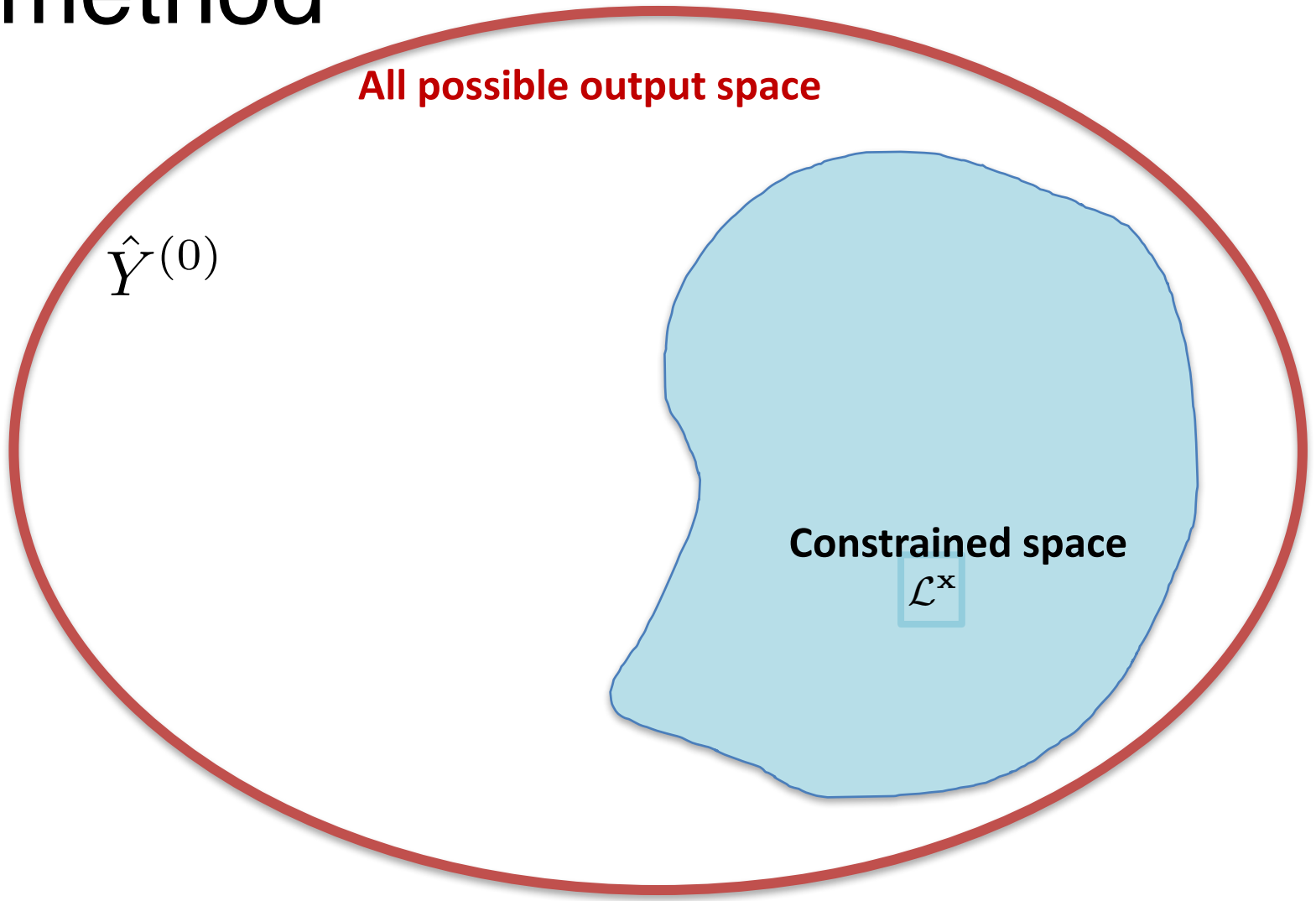


Key Idea for the method

Can we find constraint-satisfying output by updating the model parameter ?

$$\hat{Y}^{(k)} = \arg\max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W^{(k)})$$

input	model	unconstrained $\arg\max_y$
X	$W^{(0)}$	$\hat{Y}^{(0)}$

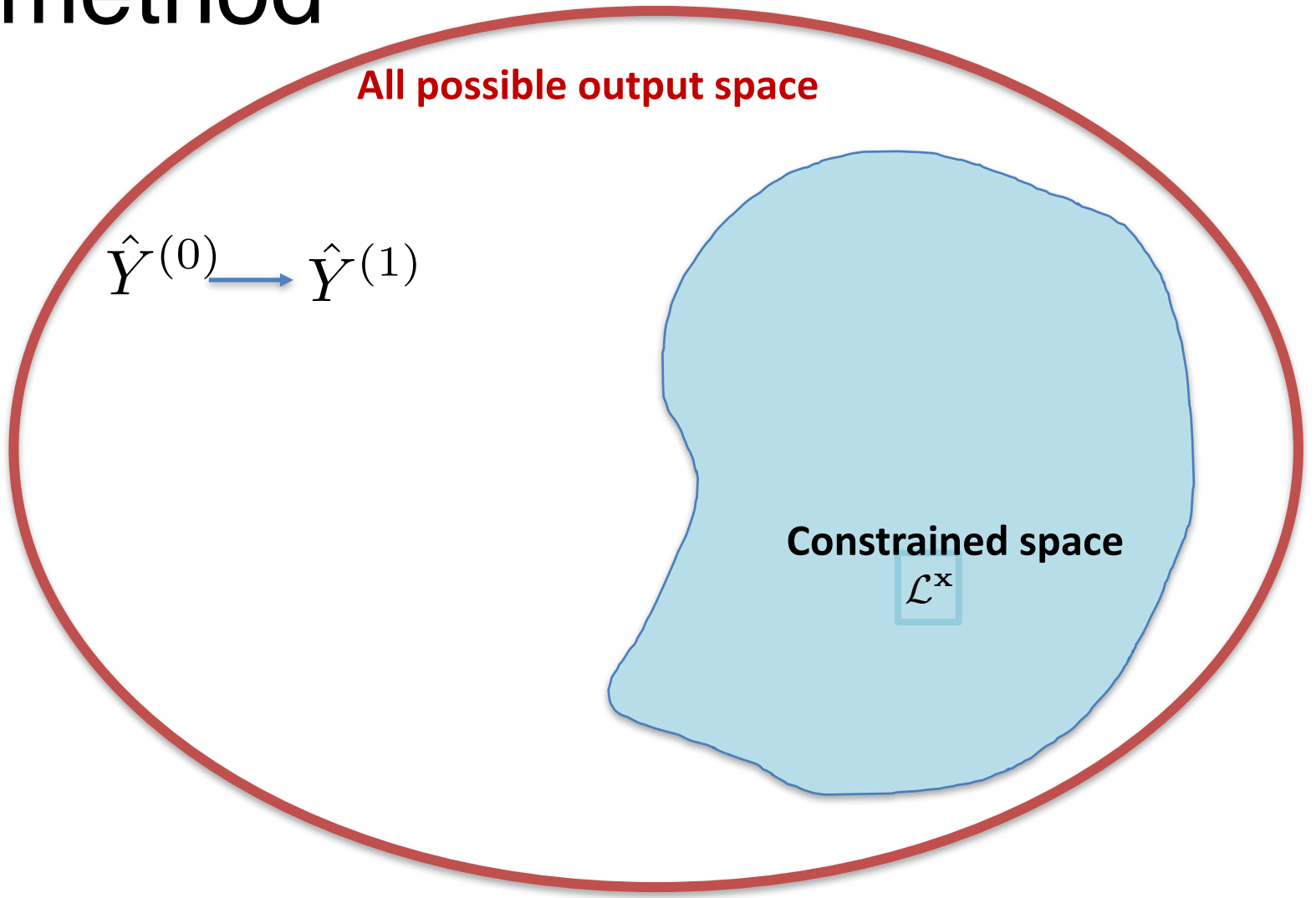


Key Idea for the method

Can we find constraint-satisfying output by updating the model parameter ?

$$\hat{Y}^{(k)} = \arg\max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W^{(k)})$$

input	model	unconstrained $\arg\max_y$
X	$W^{(0)}$	$\hat{Y}^{(0)}$
X	$W^{(1)}$	$\hat{Y}^{(1)}$

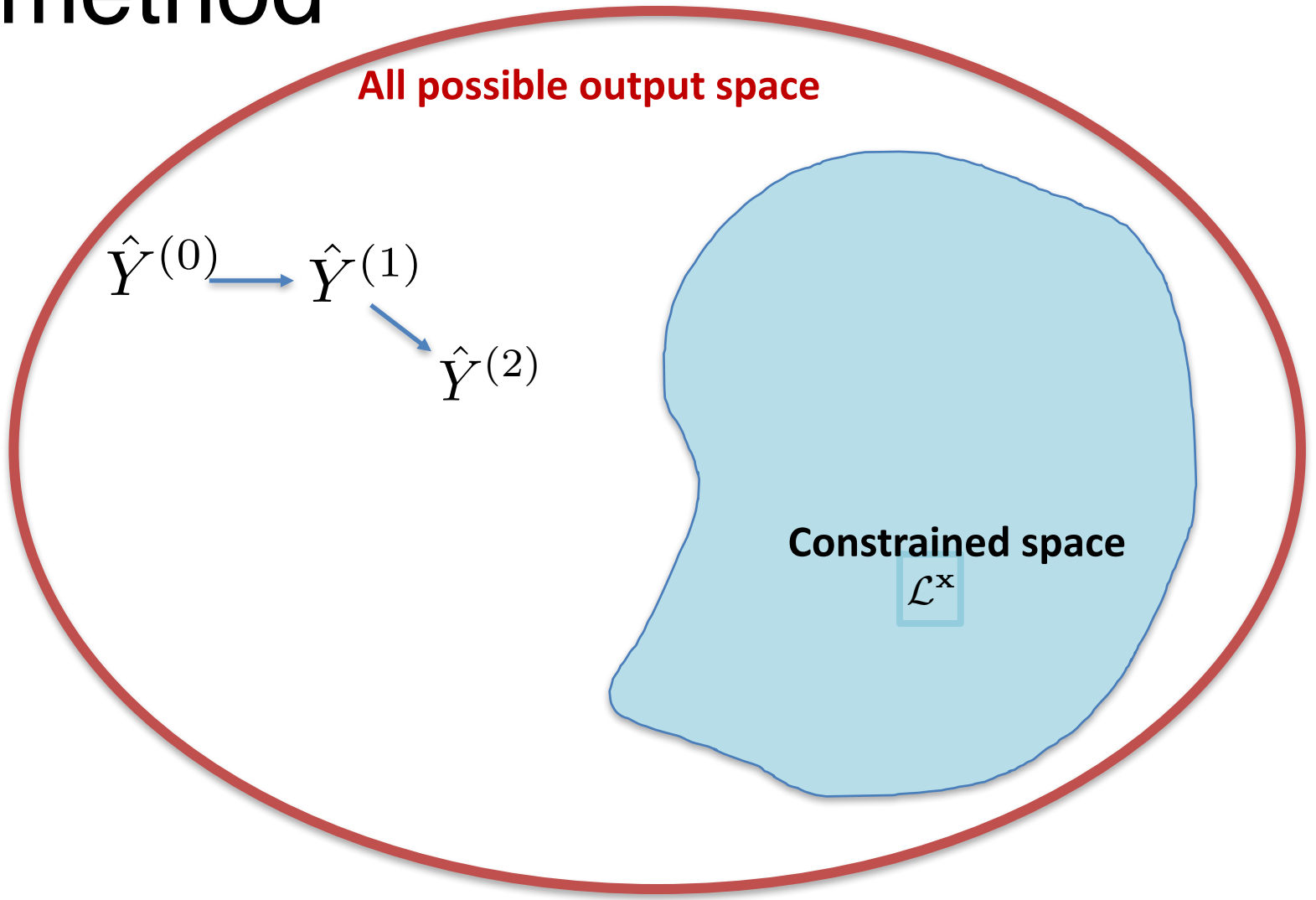


Key Idea for the method

Can we find constraint-satisfying output by updating the model parameter ?

$$\hat{Y}^{(k)} = \arg\max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W^{(k)})$$

input	model	unconstrained $\arg\max_y$
X	$W^{(0)}$	$\hat{Y}^{(0)}$
X	$W^{(1)}$	$\hat{Y}^{(1)}$
X	$W^{(2)}$	$\hat{Y}^{(2)}$

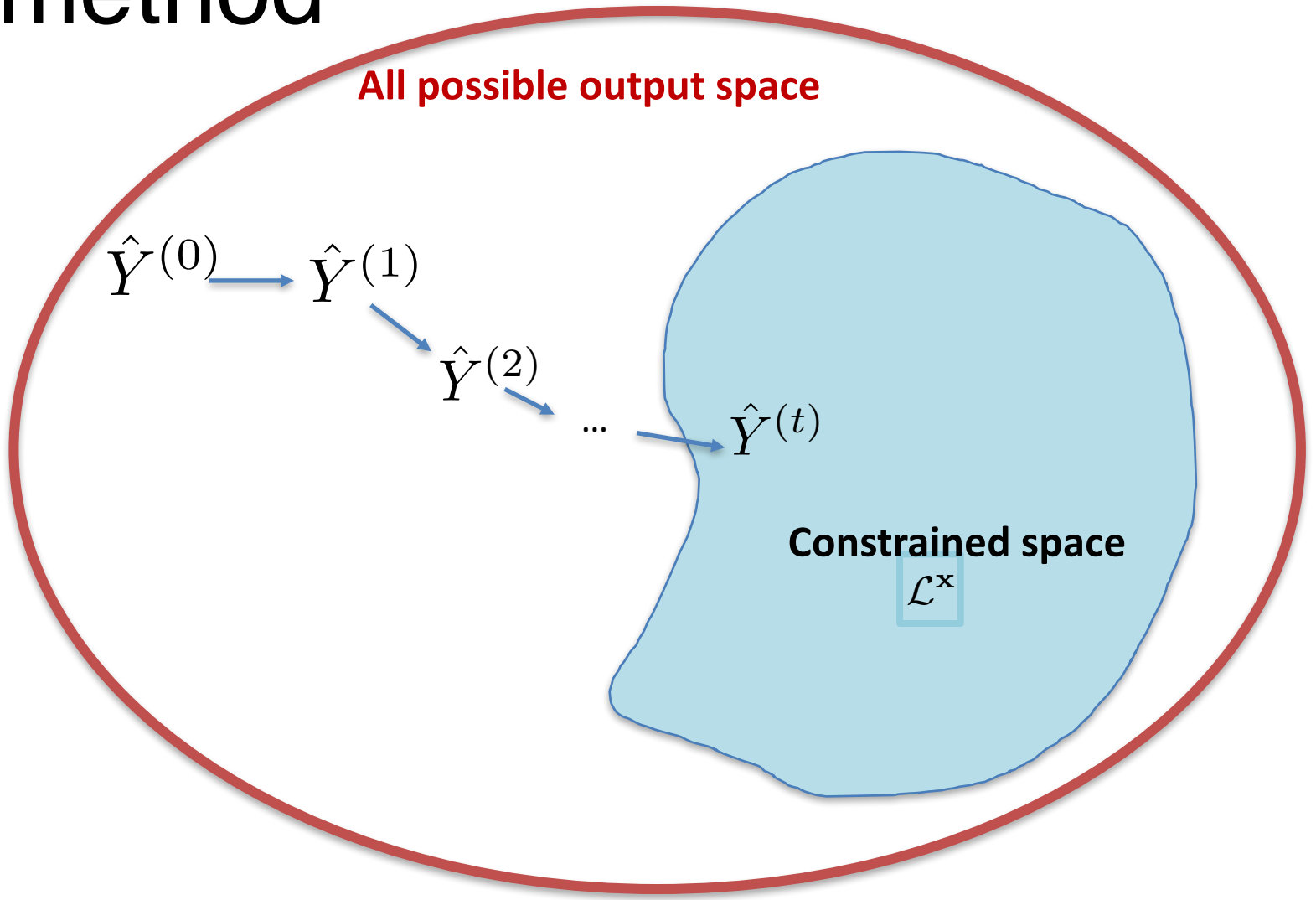


Key Idea for the method

Can we find constraint-satisfying output by updating the model parameter ?

$$\hat{Y}^{(k)} = \arg\max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W^{(k)})$$

input	model	unconstrained $\arg\max_y$
X	$W^{(0)}$	$\hat{Y}^{(0)}$
X	$W^{(1)}$	$\hat{Y}^{(1)}$
X	$W^{(2)}$	$\hat{Y}^{(2)}$
...	...	...
X	$W^{(t)}$	$\hat{Y}^{(t)}$



Key Idea for the method

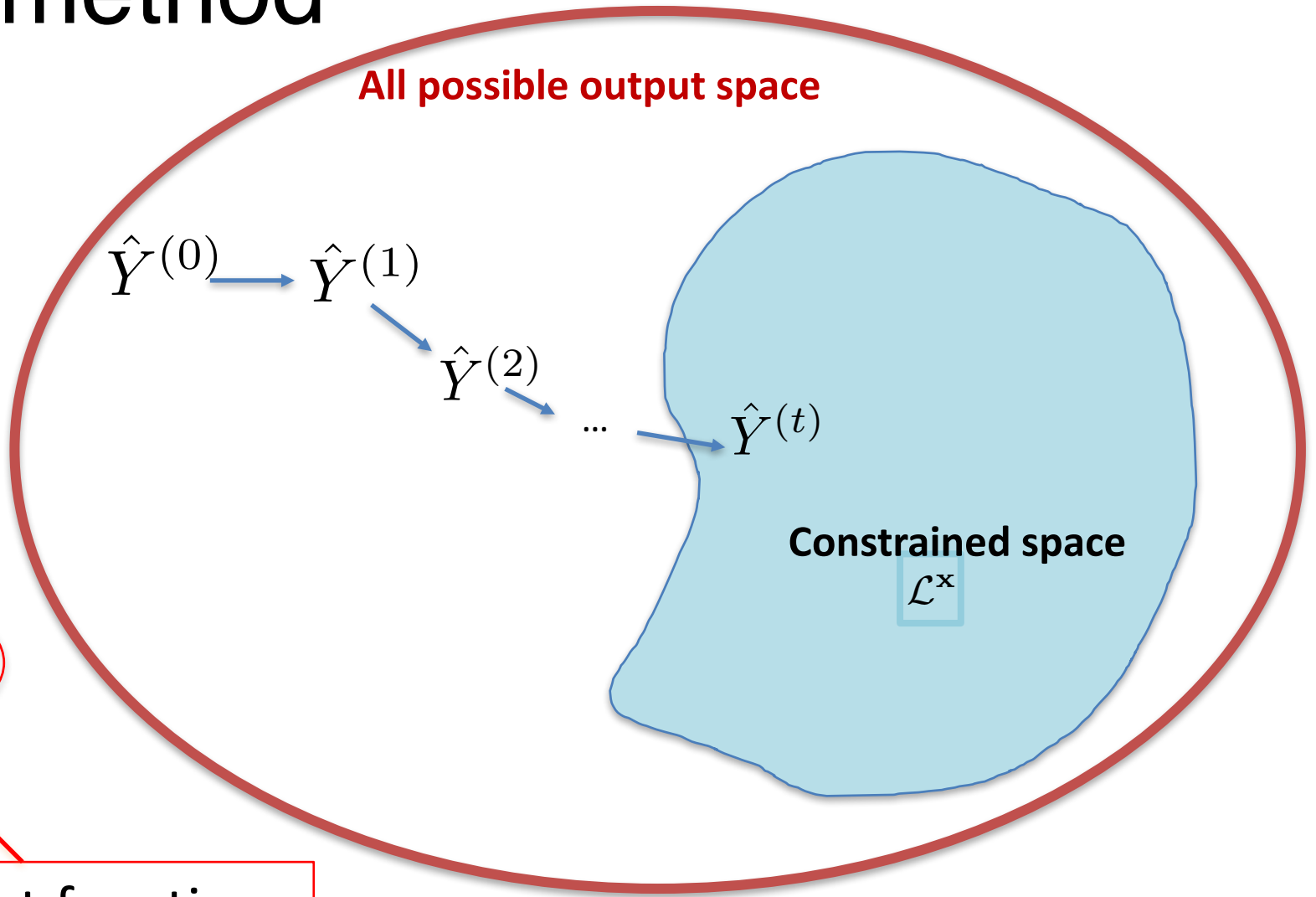
Can we find constraint-satisfying output by updating the model parameter ?

$$\hat{Y}^{(k)} = \arg\max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W^{(k)})$$

$$-\nabla_W \Psi(\mathbf{x}, \mathbf{y}, W) g(\mathbf{y}, \mathcal{L}^{\mathbf{x}})$$

Likelihood

Constraint function



e.g.) $g(y, L^x) = \frac{\text{\# of spans not in parse tree}}{\text{\# of total spans emitted}}$

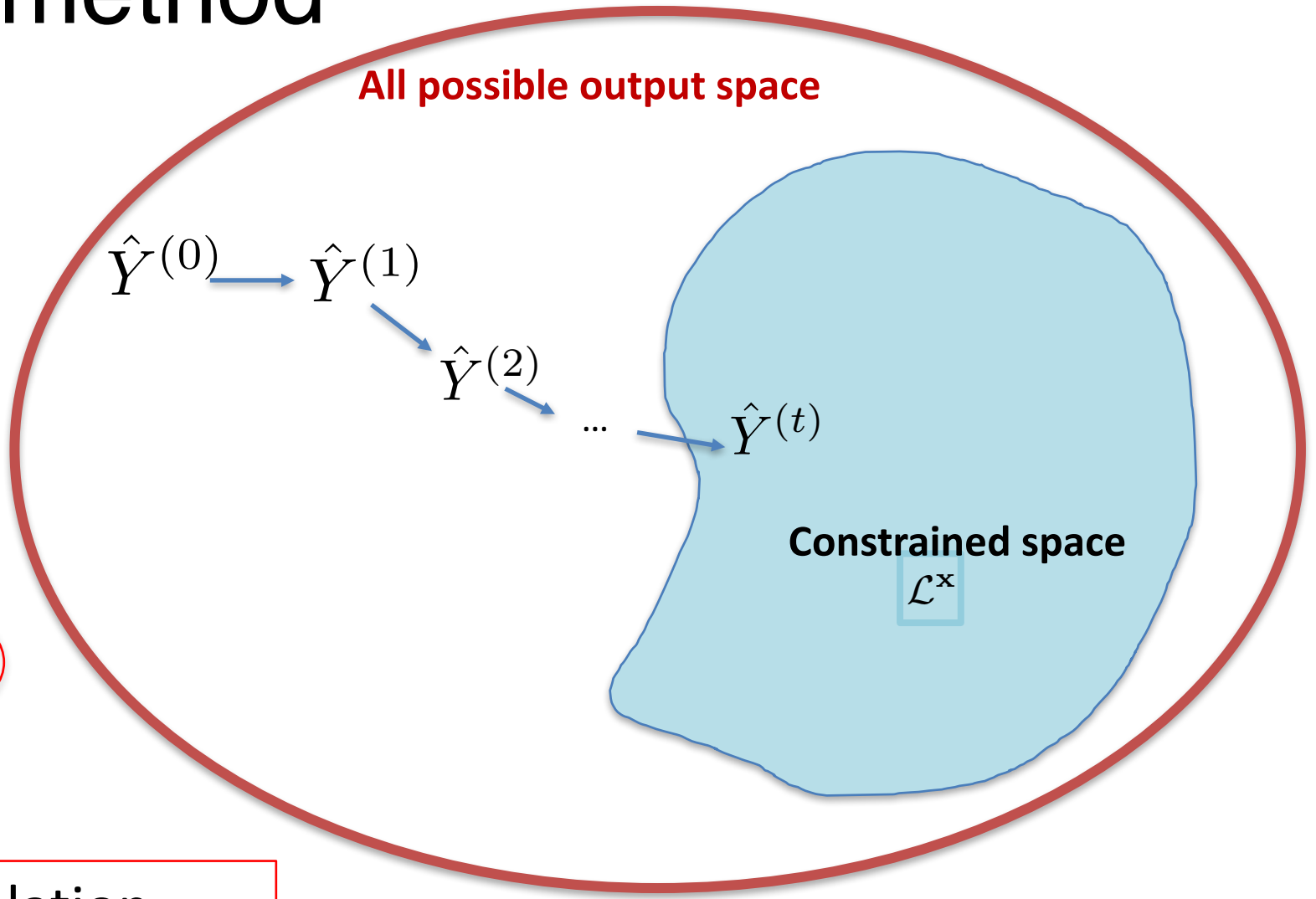
Key Idea for the method

Can we find constraint-satisfying output by updating the model parameter ?

$$\hat{Y}^{(k)} = \arg\max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W^{(k)})$$

$$-\nabla_W \Psi(\mathbf{x}, \mathbf{y}, W) g(\mathbf{y}, \mathcal{L}^{\mathbf{x}})$$

Expected constraint violation



Gradient-Based Inference (GBI)

- **Training**
  - Maximizing log-likelihood of (input, true output) pair using stochastic gradient descent.
    - x_L, y_L : labeled input, output.
    - Ψ: score (log probability, energy)

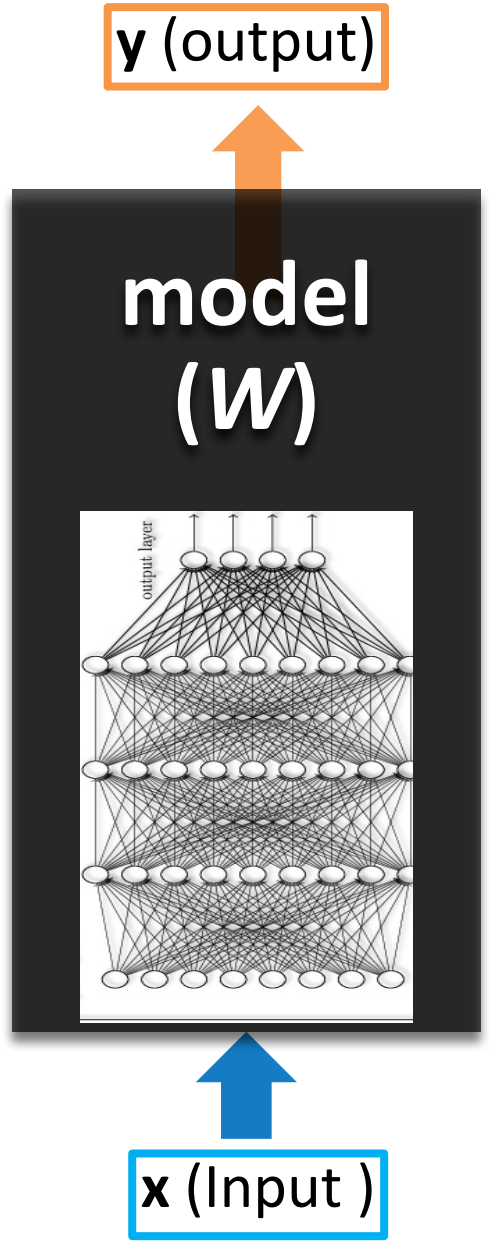
$$\max_{W} \quad \Psi(\mathbf{x_L}, \mathbf{y_L}, W)$$

- **Inference**
  - Finding the output **y** with the highest model score (probability).
    - x: a test input.
    - y: length T, vocab size V

$$\max_{\mathbf{y}} \quad \Psi(\mathbf{x}, \mathbf{y}, W)$$

Learned model

Can we do "continuous space" search to find constraint-satisfying output y? (like Training)



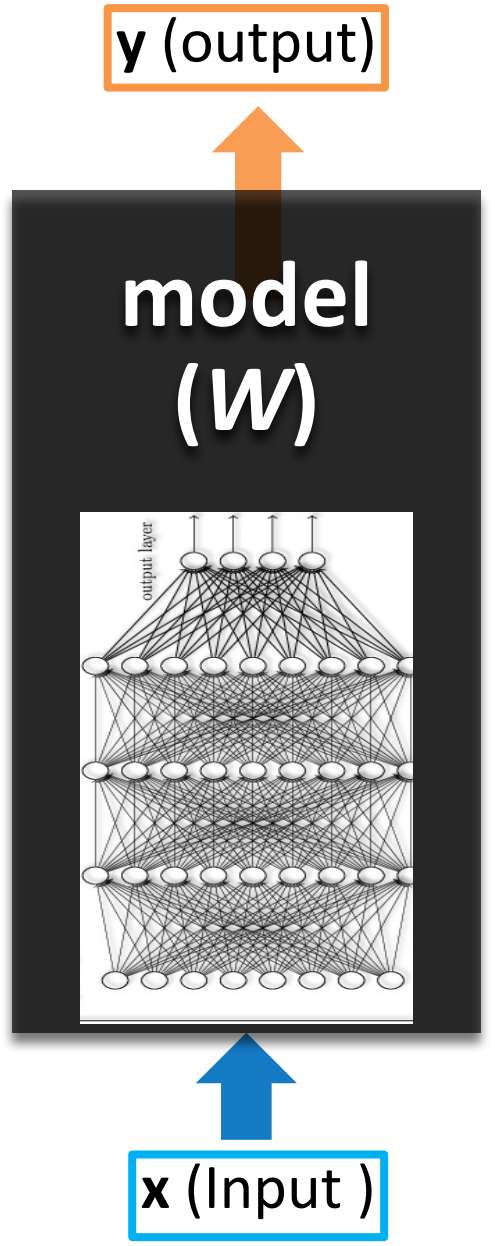
Gradient-Based Inference (GBI)

- **Training**
  - Maximizing log-likelihood of (input, true output) pair using stochastic gradient descent.
    - x_L, y_L : labeled input, output.
    - Ψ: score (log probability, energy)

$$\max_{W} \quad \Psi(\mathbf{x_L}, \mathbf{y_L}, W)$$

- **Gradient-Based Inference**
  - Finding the output **y** by updating the model **W** using stochastic gradient descent.

$$\min_{W_\lambda} \quad \Psi(\mathbf{x}, \hat{\mathbf{y}}, W_\lambda) g(\hat{\mathbf{y}}, \mathcal{L}) + \alpha \|W - W_\lambda\|_2$$
$$\text{where} \quad \hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}}\ \Psi(\mathbf{x}, \mathbf{y}, W_\lambda)$$



Gradient-Based Inference (GBI)

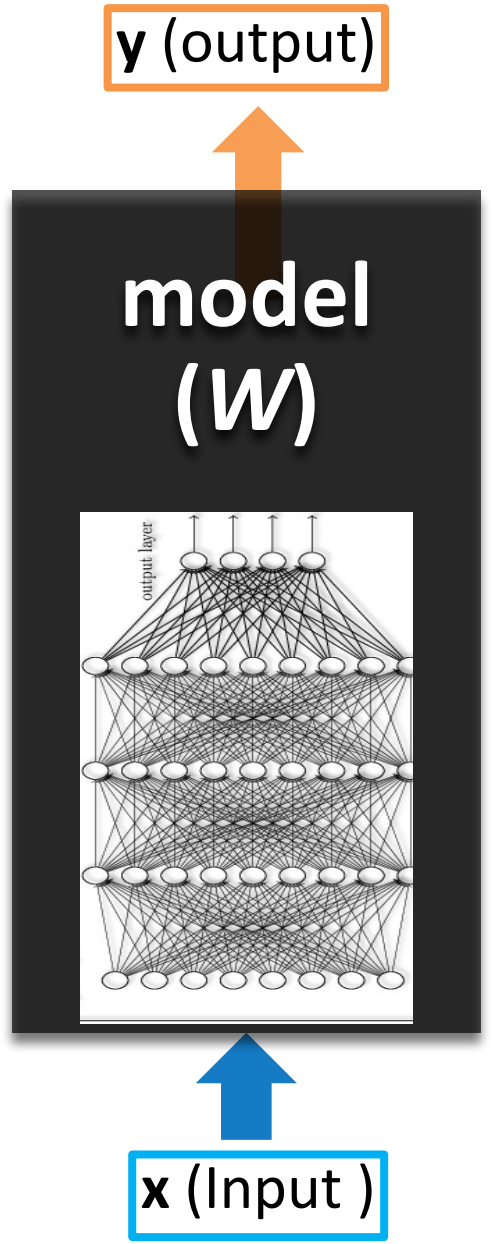
- **Gradient-Based Inference**
  - Finding the output **y** by updating the model **W** using stochastic gradient descent.

$$\min_{W_\lambda} \quad \Psi(\mathbf{x}, \hat{\mathbf{y}}, W_\lambda) g(\hat{\mathbf{y}}, \mathcal{L}) + \alpha \|W - W_\lambda\|_2$$
$$\text{where} \quad \hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}}\ \Psi(\mathbf{x}, \mathbf{y}, W_\lambda)$$

Learned model

Regularizer

- Details
  - Initialize the search parameter W_λ with learned model W.
  - Constraint can lead to "*catastrophic forgetting*".



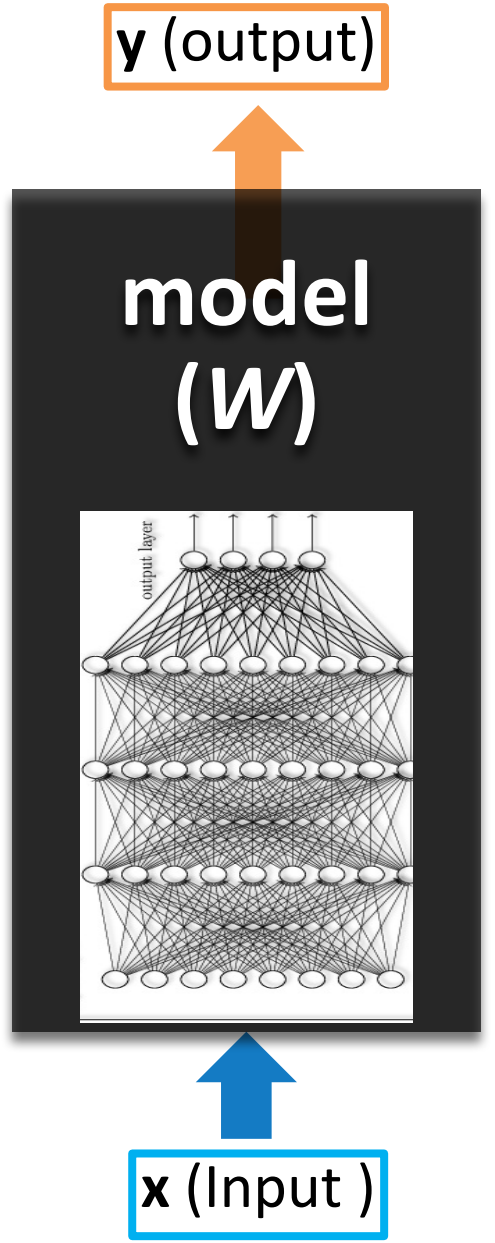
Gradient-Based Inference (GBI)

- **Gradient-Based Inference**
  - Finding the output **y** by updating the model **W** using stochastic gradient descent.

$$\min_{W_\lambda} \quad \Psi(\mathbf{x}, \hat{\mathbf{y}}, W_\lambda) g(\hat{\mathbf{y}}, \mathcal{L}) + \alpha \|W - W_\lambda\|_2$$

$$\text{where} \quad \hat{\mathbf{y}} = \underset{\mathbf{y}}{\text{argmax}}\ \Psi(\mathbf{x}, \mathbf{y}, W_\lambda)$$

- Details
  - Initialize the search parameter W_λ with learned model W.
  - Constraint can lead to "*catastrophic forgetting*".



Comparison of complexity

- Greedy inference: $O(TV)$
- 1^{st}-order inference: $O(TV^2)$, e.g. Viterbi decoding
- Combinatorial search $O(V^T)$

- GBI iteratively updates model parameter with simple inference.
  - Update model parameter: $O(TLh^2)$
  - L: number of layers, h: representation size

- e.g. SRL model with (T=7, V=130, h=100, L=10)

Comparison of complexity

- Greedy inference: $O(TV) \sim 900$
- 1st-order inference: $O(TV^2) \sim 1E+5$
- Combinatorial search $O(V^T) \sim 6E+14$

- GBI iteratively updates model parameter with simple inference.
  - Update model parameter: $O(TLh^2) \sim 7E+5$ $O(TV^2) \sim 1E+5$
  - L: number of layers, h: representation size

- e.g. SRL model with (T=7, V=130, h=100, L=10)

Experiments on SRL

- OntoNotes v5.0[1], CoNLL-2012 shared task
  (278k train, 38.3k dev, 25.6k test)

- Baseline architecture
  - multi-layer highway bi-directional LSTM (ELMo embedding)[2],[3]

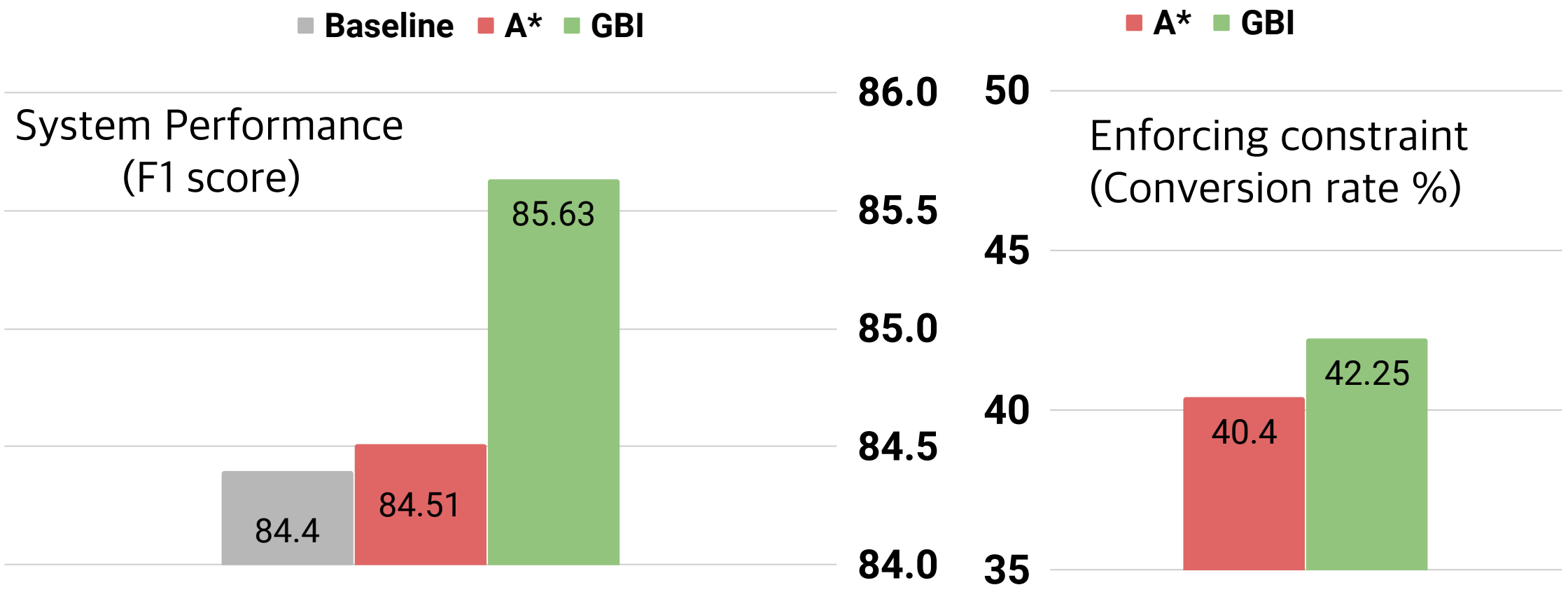
- Compared GBI with baseline and A* as used in He et al[2].

[1] Predhan et al., *Towards Robust Linguistic Analysis using OntoNotes*, CoNLL2013

[2] He et al., *Deep Semantic Role Labeling: What Works and What's Next*, ACL 2017

[3] Peters et al., *Deep contextualized word representations,* NAACL 2018

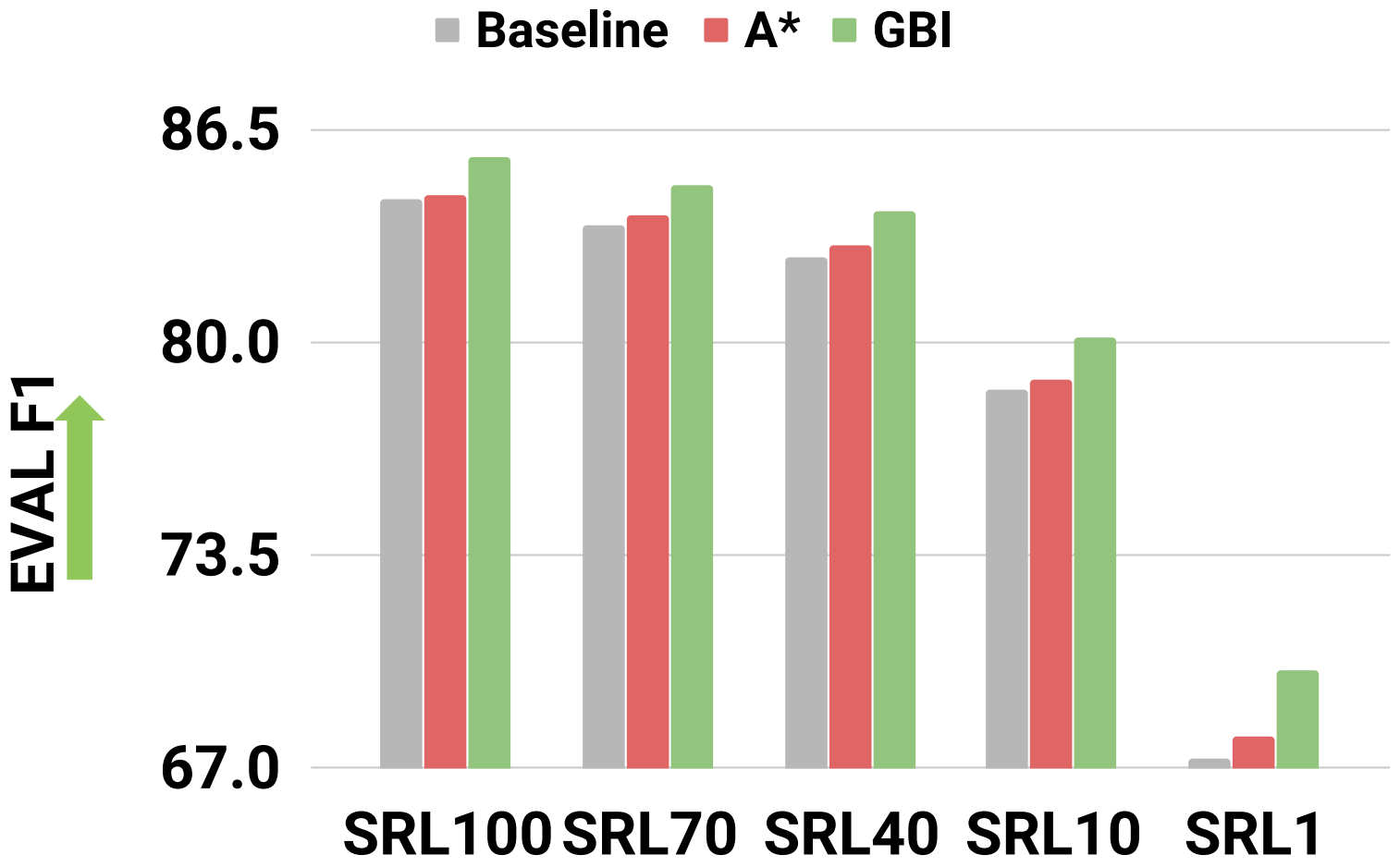
GBI enforces constraint + improves F1



SRL experiment result on CoNLL-2012

Effect of baseline model

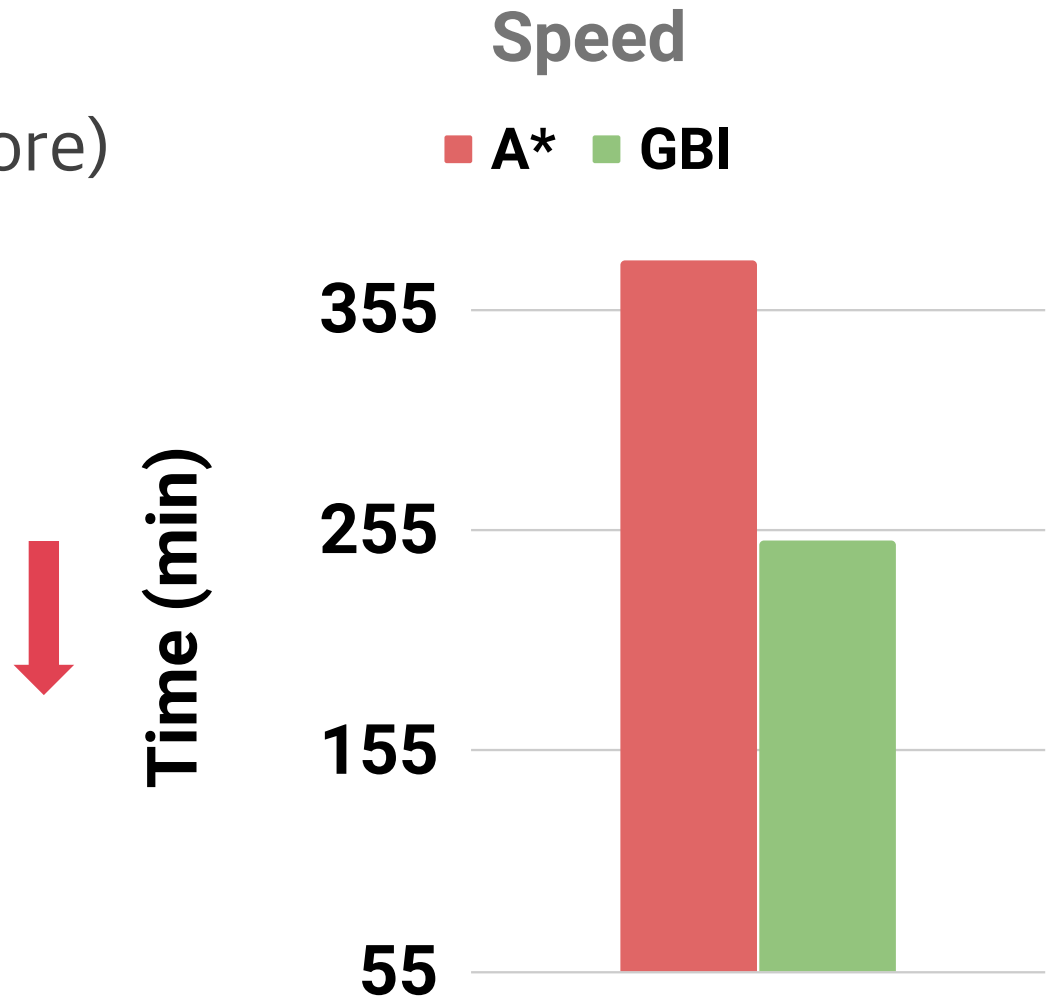
- GBI enforces constraint 42%-52% of the time.
- Enforces constraints **without hurting** the overall system.
- More gains on the lower-resource model.



SRL x : SRL system trained on x % data.

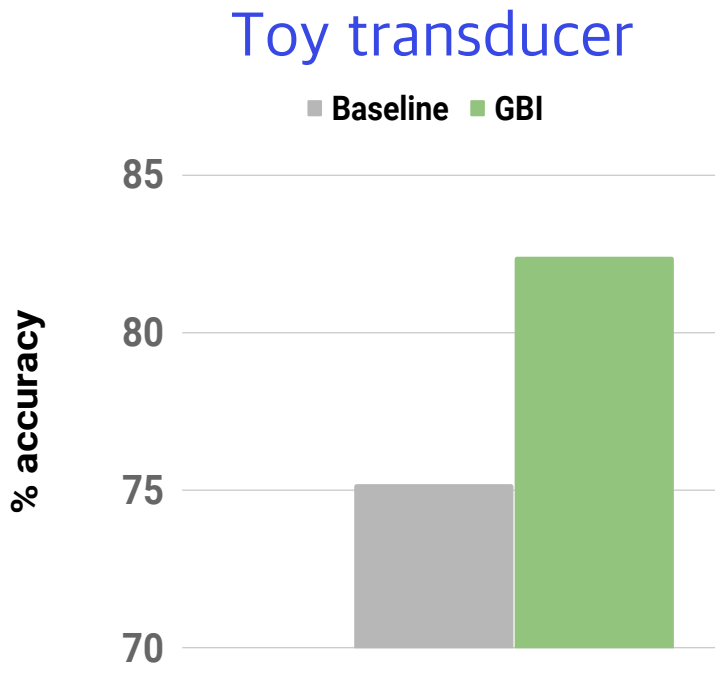
Speed experiment (SRL)

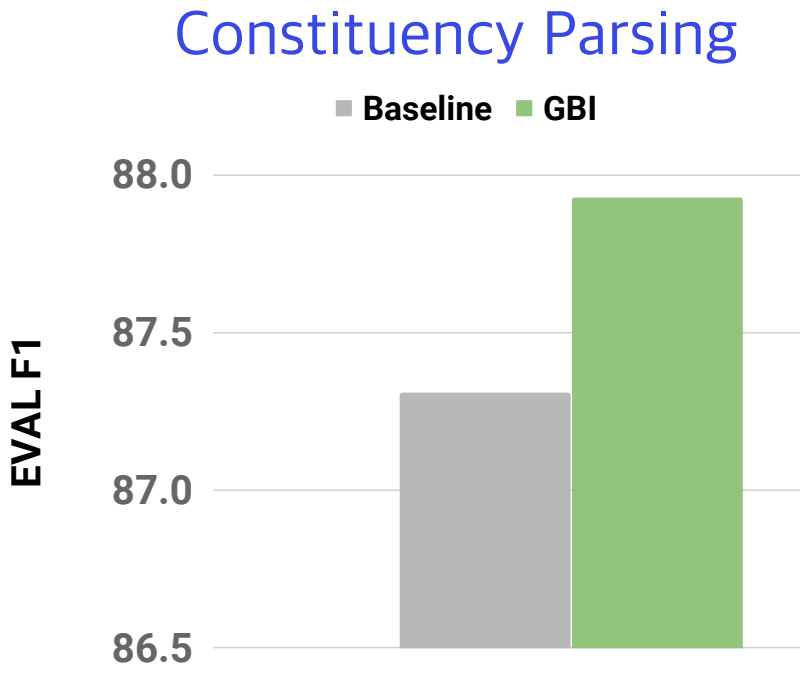
- Verified that GBI is faster than the A*
  (and better performance as seen before)

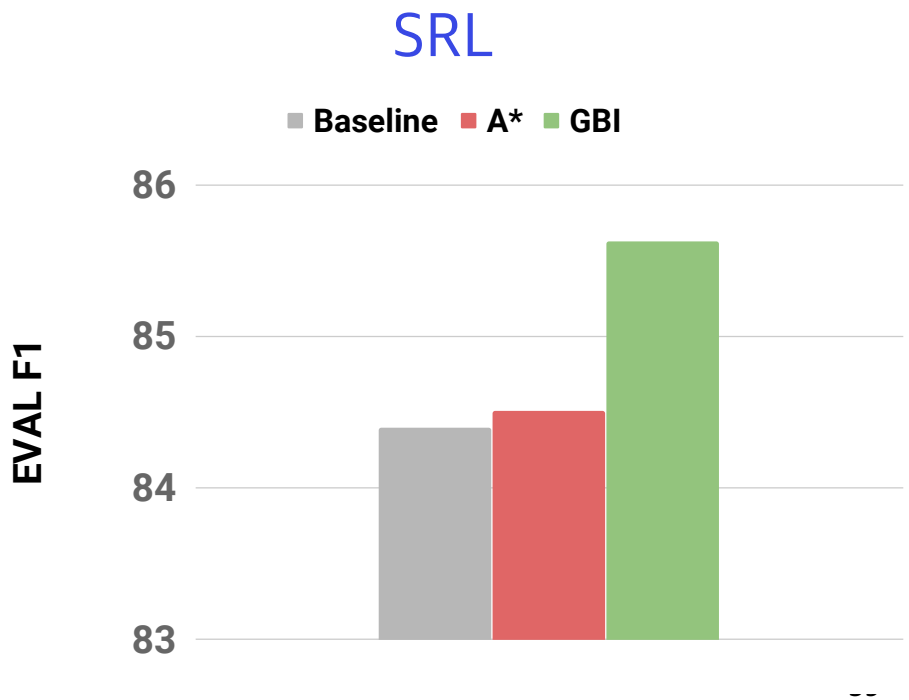


GBI was tested on

- Three applications
  - Sequence tagging model: SRL
  - Sequence-to-Sequence model: Toy transducer, Syntactic parsing



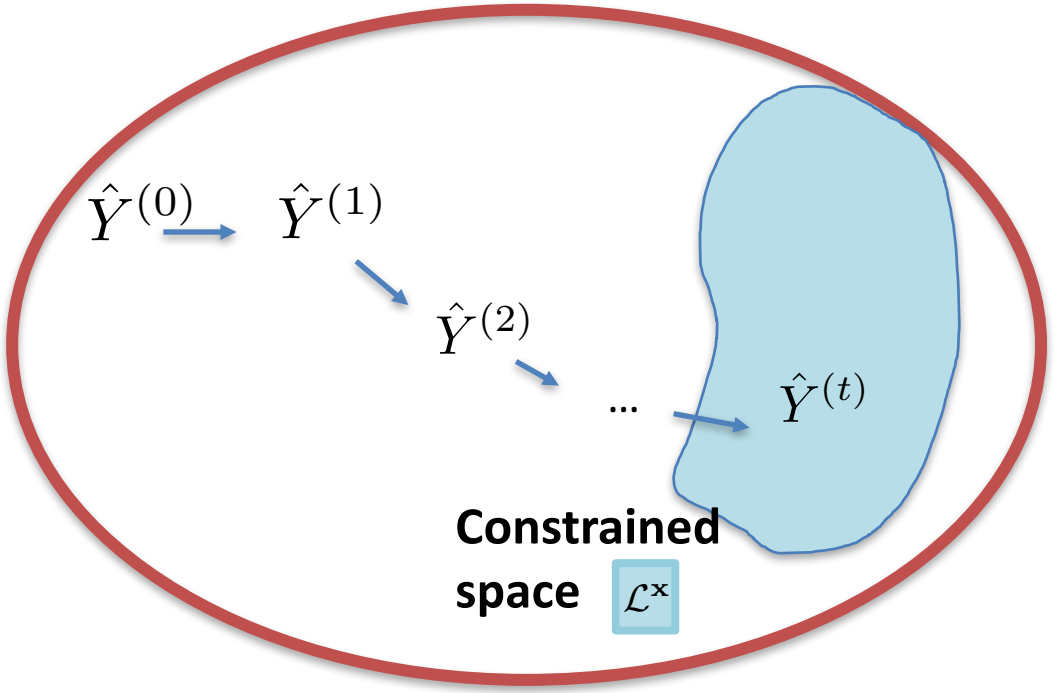




Conclusions on Inference with Constraints

- Enforce global constraints in the inference
  - using gradient signals from “constraint violations”.

- GBI is especially helpful
  - on low-resource,
  - on out-of-domain data,
  - when constraint is noisy.



Thesis goals

✔ **Provide** more logical output,
✔ **Improve** the model performance,
✔ **Overcome** the data **deficiency** problem.

Related Work

- Use of finite state automatons[1]
  - Dealt with similar constraint in syntactic parsing.
  - To reshape the probability distribution at each decoding step.

- New constraint injection work on learning time.
  - Key idea: Differentiable constraint loss using soft logic.
  - Applying this constraint loss for continuous space search.

[1] Deutsch et. al, General-purpose algorithm for constrained sequential inference. CoNLL 2019

Roadmap

- ☑ Introduction.
- ☑ Applications.
- ☑ Inference with constraints. (Ch.3) [Lee, Mehta, Wick, Tristan, Carbonell AAAI19]
- Learning with constraints. (Ch.4) [Mehta*, Lee*, Carbonell EMNLP18]
- Agreement as constraint. (Ch.5) [Lim*, Lee*, Carbonell, Poibeau AAAI20]

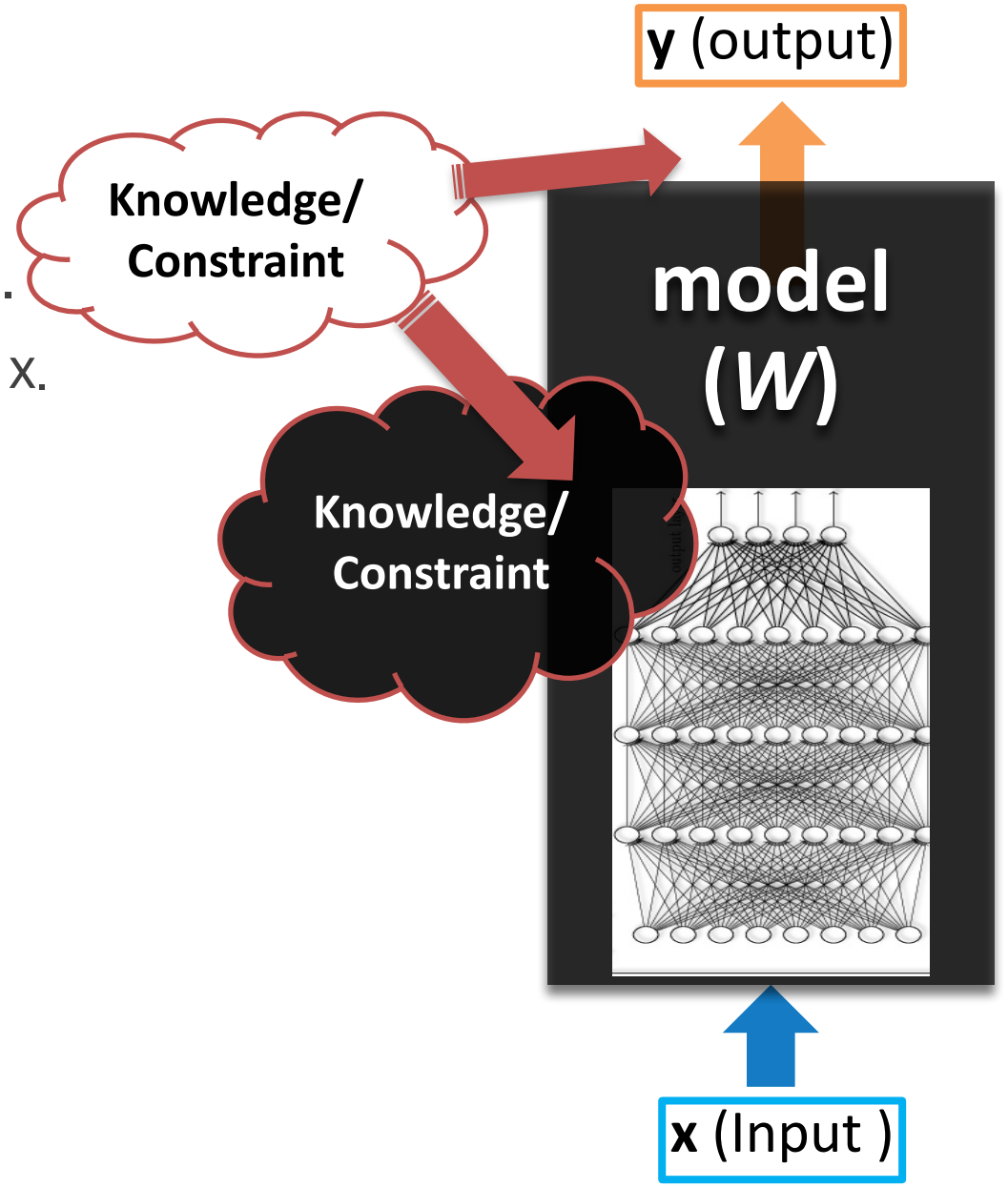
* co-first authorship with equal contribution.

Learning with constraints?

- GBI: injecting constraint in the inference.
  - Model parameter update per input instance x.

$$g(\mathbf{y}, \mathcal{L}^{\mathbf{x}})\Psi(\mathbf{x}, \mathbf{y}, W)$$

- Can we use the same update rule to inject constraint $\mathcal{L}^{\mathbf{x}}$ in train time?



Learning with constraints?

- GBI: injecting constraint in the inference.
  - Model parameter update per input instance x.

$$g(\mathbf{y}, \mathcal{L}^{\mathbf{x}})\Psi(\mathbf{x}, \mathbf{y}, W)$$

- Can we use the same update rule to inject constraint $\mathcal{L}^{\mathbf{x}}$ in train time?

Questions:

- Would the previous loss function be useful as is?
- Note that constraint is not the main objective of the task.
  - What kind of regularizer can be used in learning time?

Ranzanto et al., Sequence Level Training with Recurrent Neural Networks, ICLR16

Comparison of inference & learning

	Inference (GBI)	Learning
Loss function	$g(\hat{y}, \mathcal{L}^x) \log P(\hat{y}\|x; w)$	$\sum_{i=1}^{B} s(\hat{y}_i, \mathcal{L}^x) \log P(\hat{y}_i\|x; w)$ $s(\hat{y}, \mathcal{L}^x) = 2g(\hat{y}, \mathcal{L}^x) - 1$
Score range	$-1 \leq -g \leq 0$	$-1 \leq s \leq 1$
Loss applied on	Iteratively on a single test instance	Different batch at each update
Model parameter	Fixed learned model.	Constantly updated.

Overview for “learning with constraints”

Can we use how well the model satisfies the constraint as a signal to learn further?

- **Proposed method**
  - Proposing three variant loss functions using constraint information.
  - Semi-supervised Learning loss to help low-resource setting.

- **Take away**
  - Semi-supervised learning shows significant help on low-resource as well as on large resource.

Types of loss functions

- Enforcing constraints with the reward s(y,L^x)
  - Constraint loss (c-loss)

$$\sum_{x_i \in B} s(\hat{y}_i, \mathcal{L}^x) \log P\left(\hat{y}_i | x_i; w\right) + \|W - W_{\text{pre-train}}\|_2$$

Types of loss functions

- Enforcing constraints with the reward s(y,L^x)
  - Constraint loss (c-loss)

$$\sum_{x_i \in B} s(\hat{y}_i, \mathcal{L}^x) \log P(\hat{y}_i | x_i; w) + \|W - W_{\text{pre-train}}\|_2$$

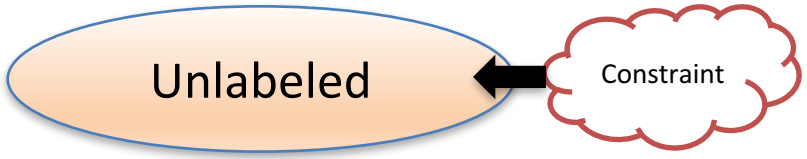
  - Joint loss

SRL-***labeled*** data

$$-\alpha_1 \sum_{x_i \in B_{label}} \log P(y_i | x_i; w) + \alpha_2 \sum_{x_j \in B_c} s(\hat{y}_j, \mathcal{L}^{x_j}) \log P(\hat{y}_j | x_j; w)$$

Types of loss functions

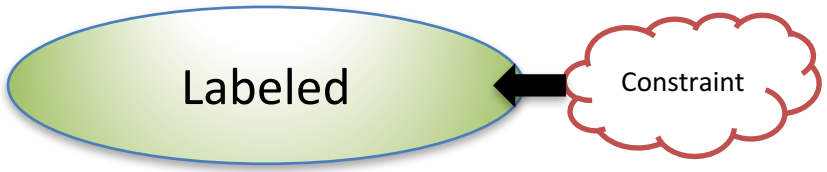
- Enforcing constraints with the reward s(y,L^x)
  - Constraint loss (c-loss)



$$\sum_{x_i \in B} s(\hat{y}_i, \mathcal{L}^x) \log P(\hat{y}_i | x_i; w) + \|W - W_{\text{pre-train}}\|_2$$

  - Joint loss

SRL-***labeled*** data ($B_{label}=B_c$)



$$-\alpha_1 \sum_{x_i \in B_{label}} \log P(y_i | x_i; w) + \alpha_2 \sum_{x_j \in B_c} s(\hat{y}_j, \mathcal{L}^{x_j}) \log P(\hat{y}_j | x_j; w)$$

Types of loss functions

- Enforcing constraints with the reward s(y,L^x)
  - Constraint loss (c-loss)

$$\sum_{x_i \in B} s(\hat{y}_i, \mathcal{L}^x) \log P(\hat{y}_i | x_i; w) + \|W - W_{\text{pre-train}}\|_2$$

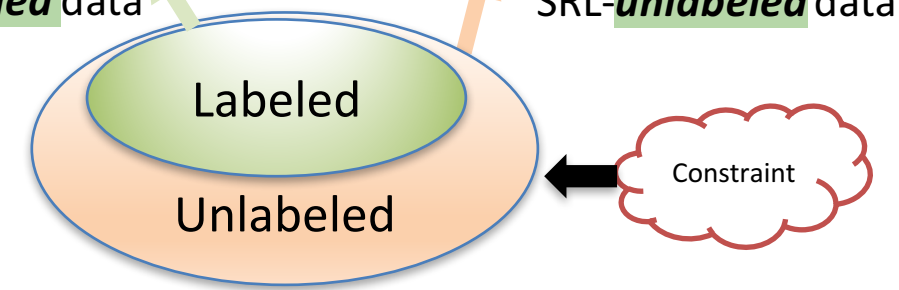
  - Joint loss

$$-\alpha_1 \sum_{x_i \in B_{label}} \log P(y_i | x_i; w) + \alpha_2 \sum_{x_j \in B_c} s(\hat{y}_j, \mathcal{L}^{x_j}) \log P(\hat{y}_j | x_j; w)$$

SRL-***labeled*** data

SRL-***unlabeled*** data $(B_{label} \subset B_c)$

  - Semi-supervised (SSL)



Chang, Ratinov, & Roth,
Structured learning with constrained conditional models, 2008

Types of loss functions

- Enforcing constraints with the reward s(y,L^x)
  - Constraint loss (c-loss)

$$\sum_{x_i \in B} s(\hat{y}_i, \mathcal{L}^x) \log P(\hat{y}_i | x_i; w) + \|W - W_{\text{pre-train}}\|_2$$

  - Joint loss

SRL-***labeled*** data ($B_{label}=B_c$)

$$-\alpha_1 \sum_{x_i \in B_{label}} \log P(y_i | x_i; w) + \alpha_2 \sum_{x_j \in B_c} s(\hat{y}_j, \mathcal{L}^{x_j}) \log P(\hat{y}_j | x_j; w)$$

SRL-***labeled*** data

SRL-***unlabeled*** data $(B_{label} \subset B_c)$

  - Semi-supervised (SSL)

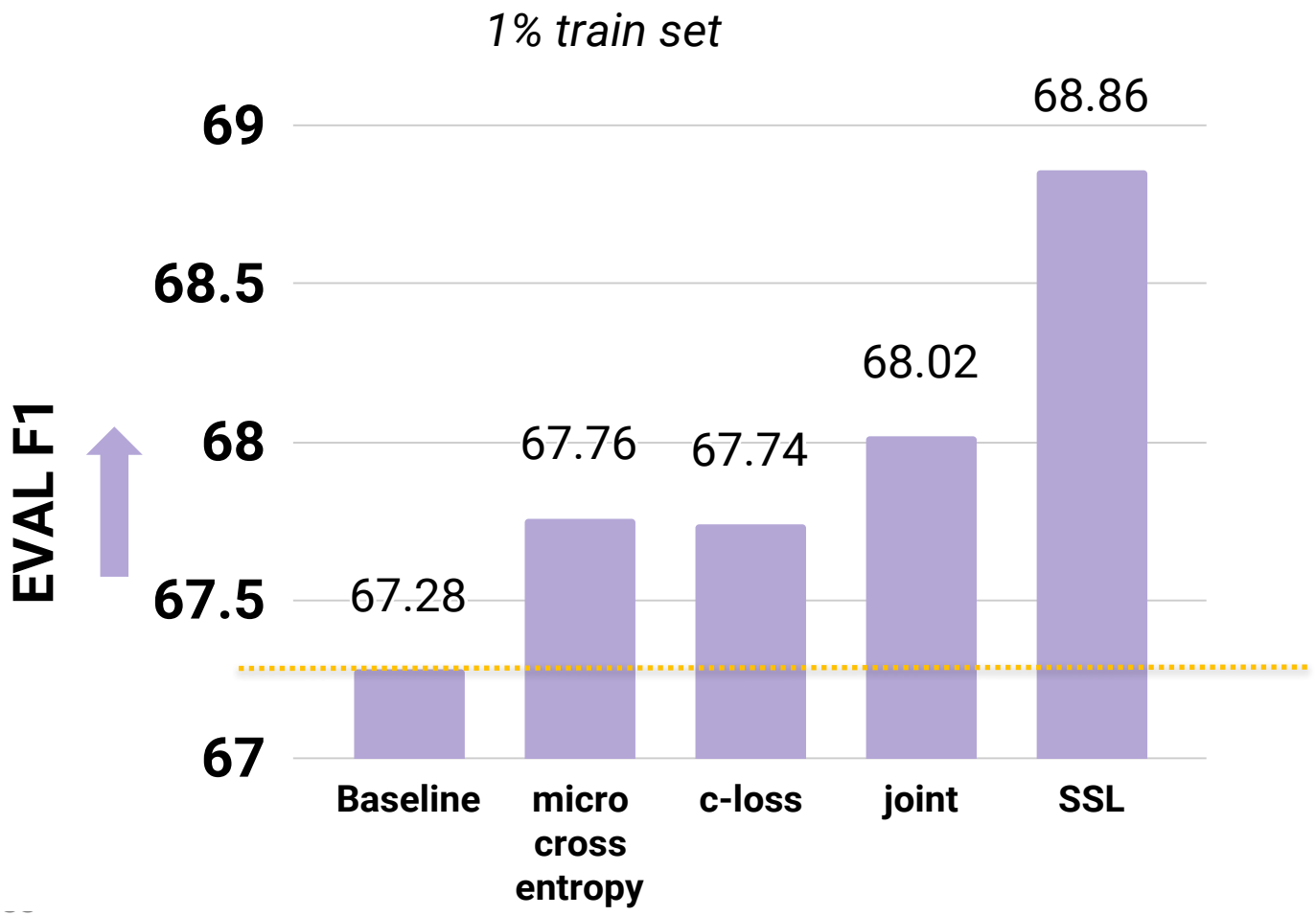
SRL experiment

- OntoNotes v5.0[1], CoNLL-2012 shared task
  (278k train, 38.3k dev, 25.6k test)

- The same experiment setup with GBI
  - Using 1%, 10% of training set to represent lower resource setting.
  - Using full resource as well.

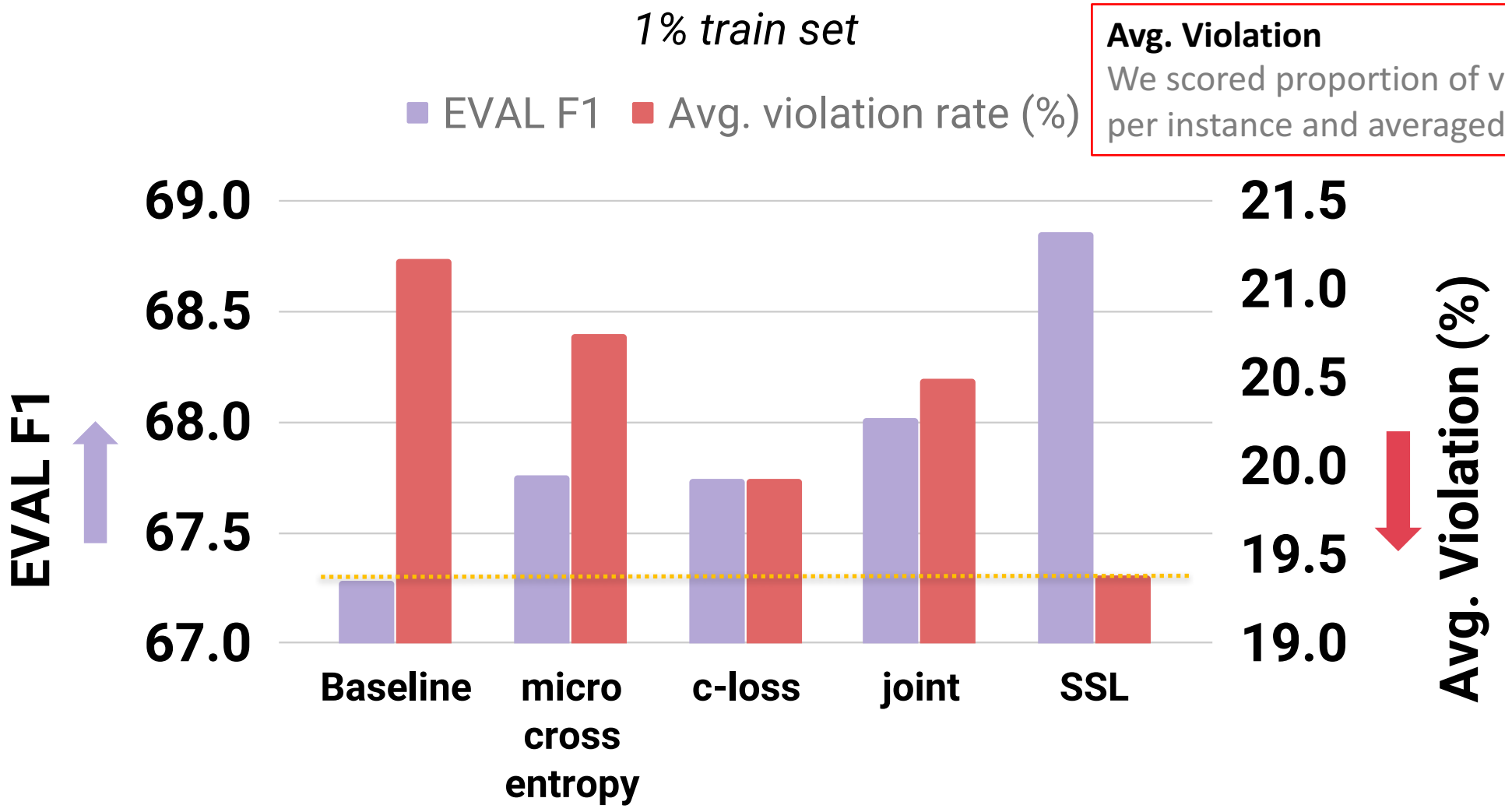
[1] Predhan et al., *Towards Robust Linguistic Analysis using OntoNotes*, CoNLL2013

Comparison of losses (on 1% training)



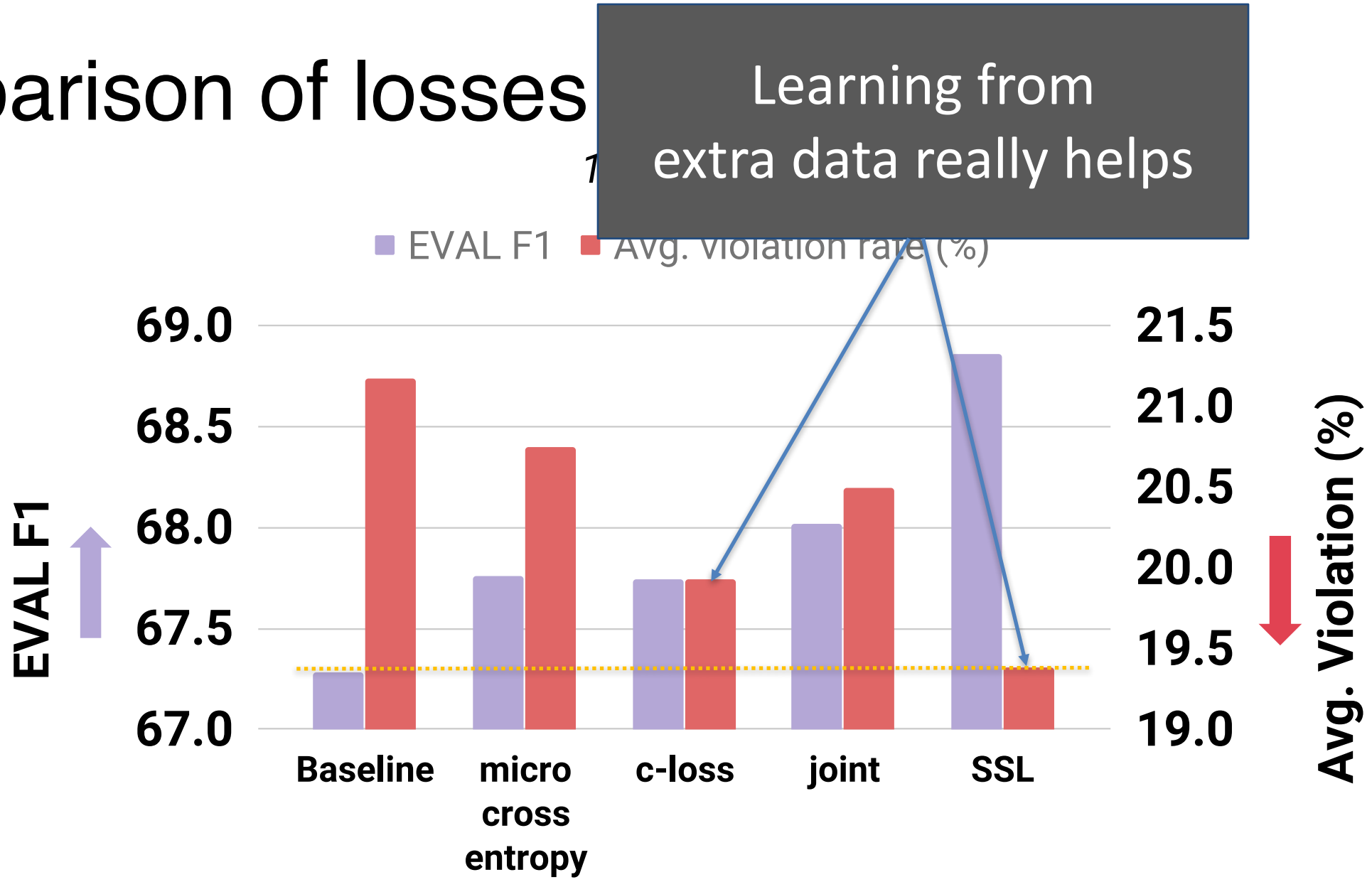
Micro cross entropy*
Fine-tuning with the sa
learning rate as the c-lc
With cross entropy.

Comparison of losses (on 1% training)



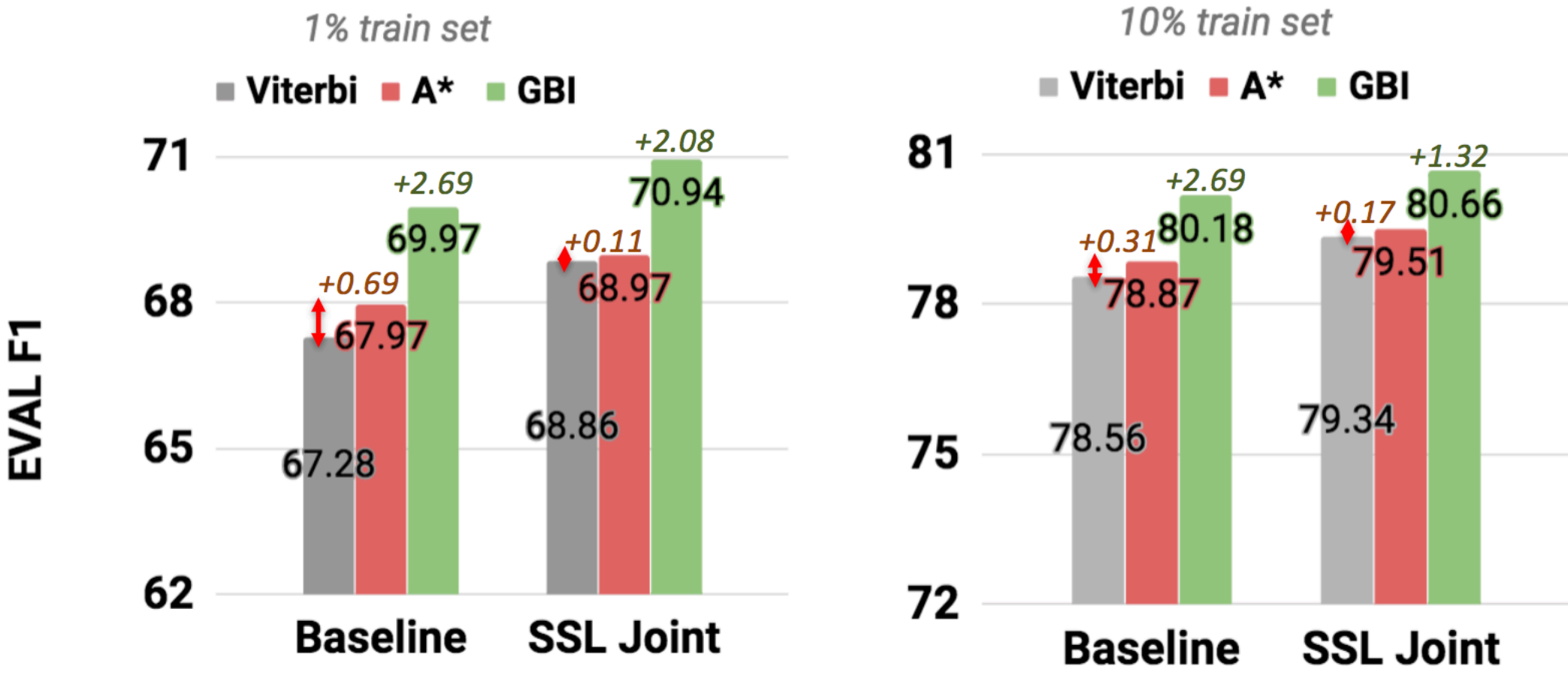
Avg. Violation
We scored proportion of violation per instance and averaged it across.

Comparison of losses



Q) Is inference with constraint still useful?

On **1%** dataset: GBI brings **additional gains on top of** train-time constraint injection.



Plain supervised learning

Learning with constraint

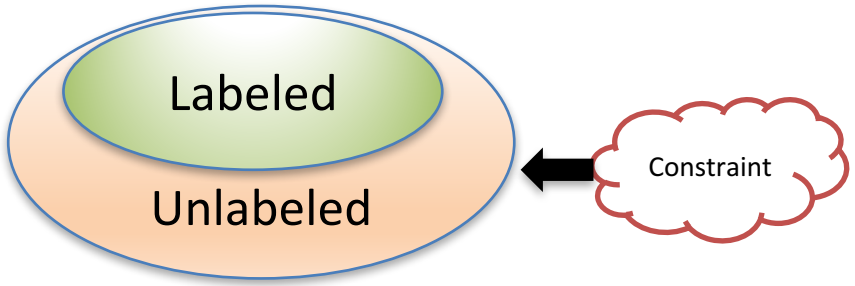
Plain supervised learning

Learning with constraint

Conclusions on Learning with Constraints

- Constraint information enabled training with unlabeled data.

- SSL with constraint is
  - Especially on lower-resource
  - Also significant in high-resource
  - Complementary with GBI as well.



Thesis goals

✔ **Provide** more logical output,
✔ **Improve** the model performance,
✔ **Overcome** the data **deficiency** problem.

Related work

- Modeling SRL with syntactic information.
  - Multi-task learning. (Swabha et al., 2018[1])
  - Guiding attention with dependency parse tree. (Strubell et al. 2018[2])

- Model-agnostic approach
  - Differentiable constraint function using soft logic.
  - Dual decomposition (Nandwani et al. 2019[3]) showed similar gains in SRL.
  - Regularizing the models away from inconsistency (Li et al., 2019[4])
  - Structured Tuning for Semantic Role Labeling (Li et al., 2019[5])

[1] Swayamdipta et al., Syntactic Scaffolds for Semantic Structures, EMNLP 18
[2] Strubell et al., Linguistically-Informed Self-Attention for Semantic Role Labeling, EMNLP18
[3] Nandwani et al., A Primal-Dual Formulation for Deep Learning with Constraints, NeurIPS 2019
[4] Li et al., A Logic-Driven Framework for Consistency of Neural Models, EMNLP 2019
[5] Li et al., Structured Tuning for Semantic Role Labeling, ACL 2020

Roadmap

- ☑ Introduction.
- ☑ Applications.
- ☑ Inference with constraints. (Ch.3) [Lee, Mehta, Wick, Tristan, Carbonell AAAI19]
- ☑ Learning with constraints. (Ch.4) [Mehta*, Lee*, Carbonell EMNLP18]
- Agreement as constraint. (Ch.5) [Lim*, Lee*, Carbonell, Poibeau AAAI20]

Other ways of using SSL with constraint

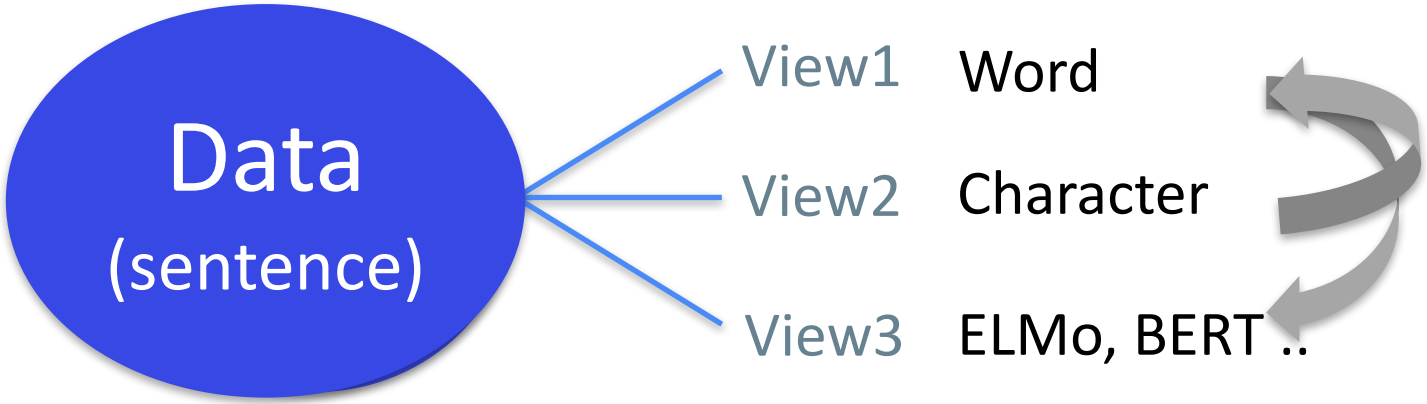
- Consider the classic semi-supervised learning methods that used unlabeled dataset.

- Many work have used *multi-view learning* to use unlabeled dataset.
  - Co-training[1], co-regularization[2] and many others[3] [4].

[1] Blum and Mitchell, *Combining labeled and un- labeled data with co-training,* COLT 1998
[2] Sindhwani and Belkin, *A co-regularization approach to semi-supervised learning with multiple views*. In ICML 2005
[3] Muslea et al., *Active+ semi-supervised learning= robust multi-view learning*. ICML 2002.
[4] Yu et al., *Bayesian Co-Training,* JMLR 2011

Multi-view learning?

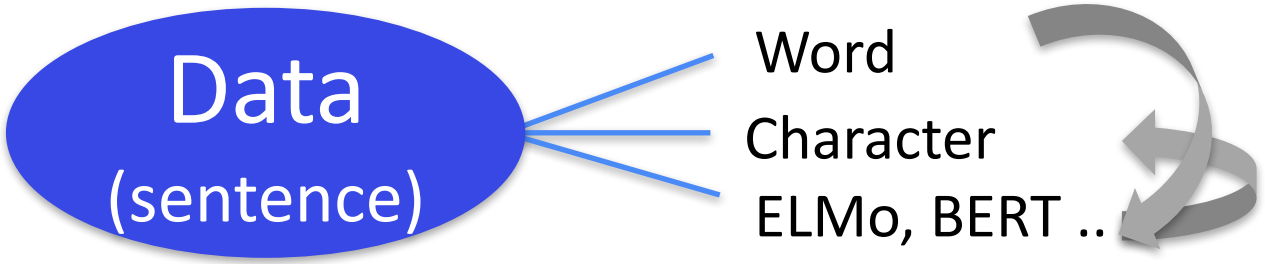
Multi-view means different representation (view) of the same data.



Can we enhance multi-view models by ***promoting agreement*** on the **unlabeled data**?

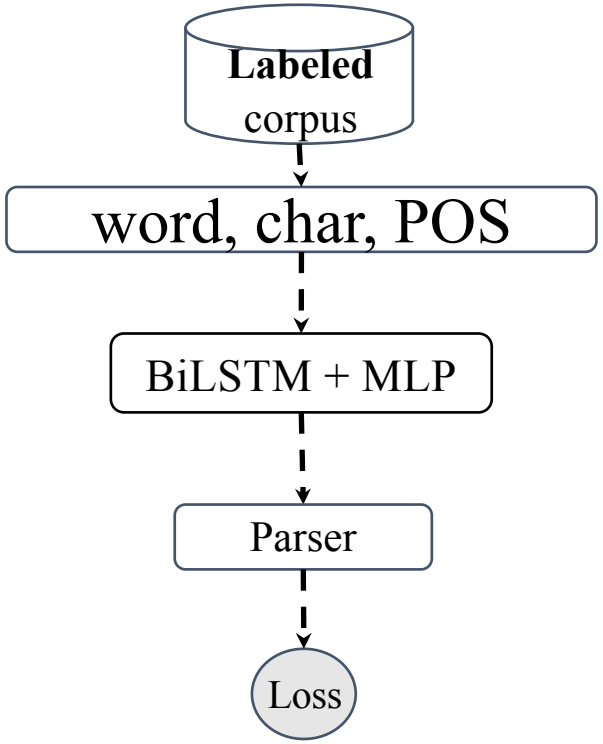
Overview for “Agreement as constraints”

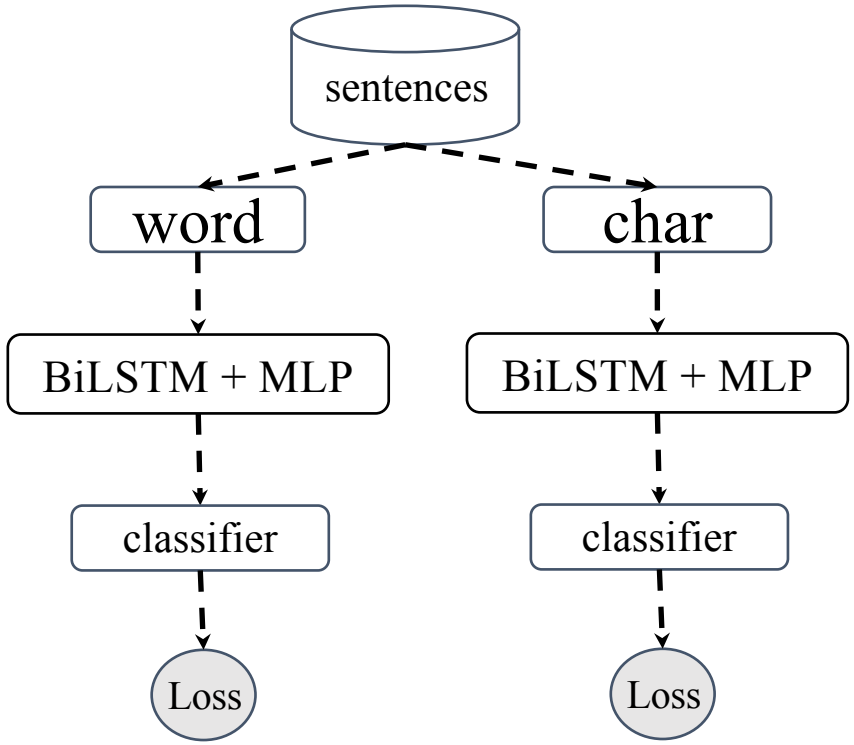
Can we enhance multi-view models by ***promoting agreement on the unlabeled data***?



- **Proposed method**
  - Applying previous SSL method with *agreement score* across the views.
- Take away
  - SSL significantly improves extremely low-resource setting.
  - SSL makes views to agree more on unlabeled data.

Single-view vs. Multi-view

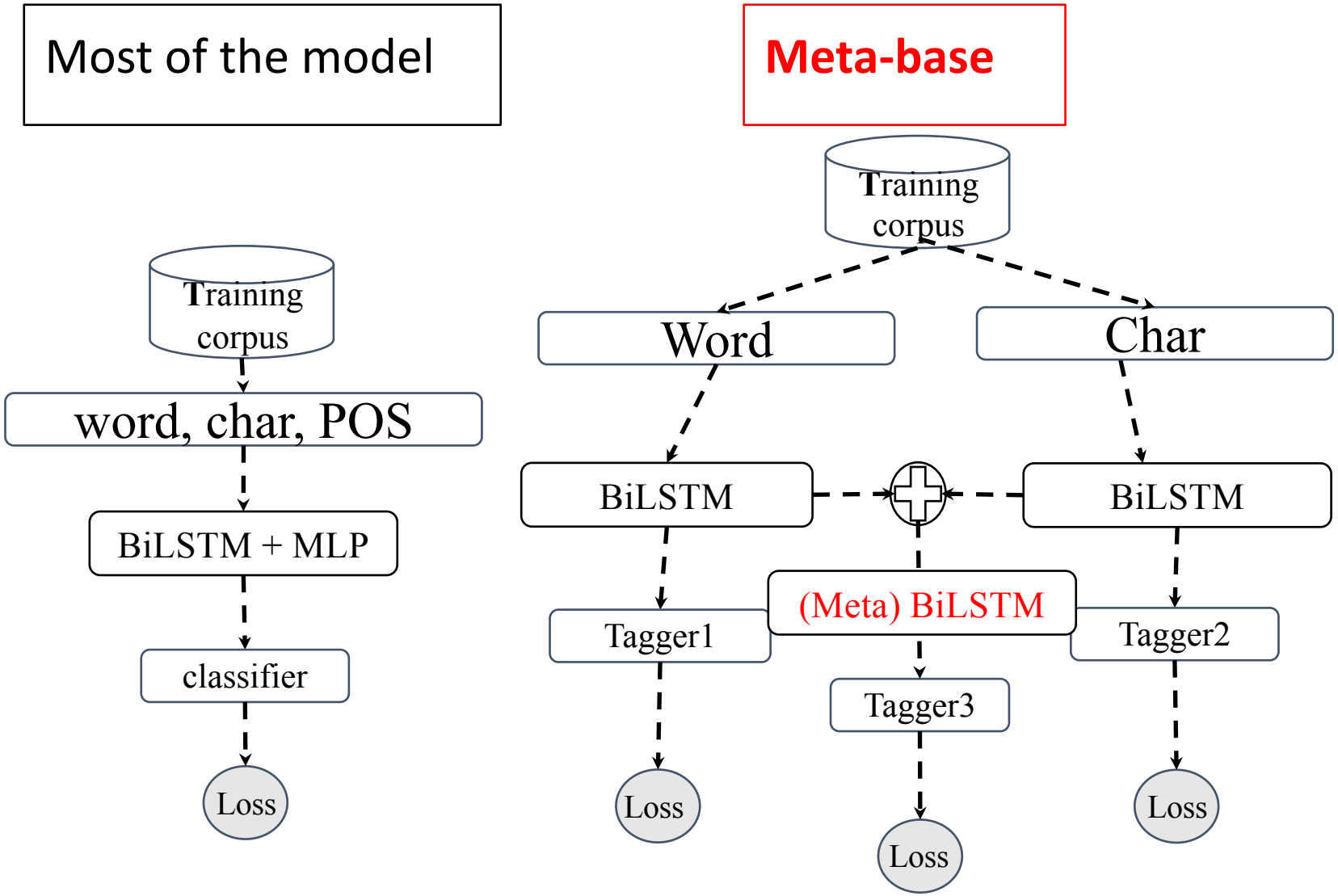




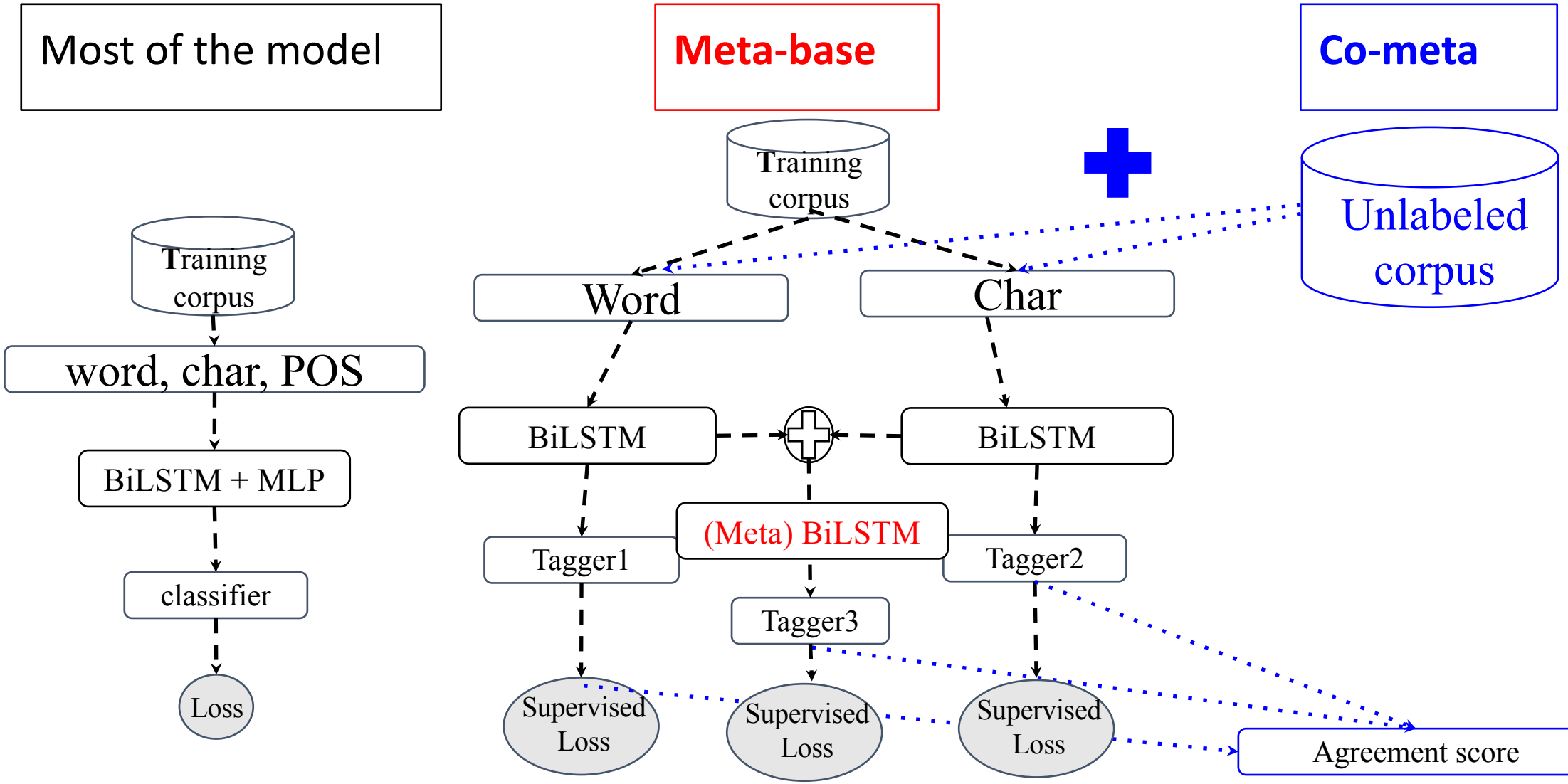
9 out of top 10 results in CoNLL 2018 challenge uses concatenated feature representation with single view.
Zeman et al., *CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, CoNLL 2018

Meta-BiLSTM model

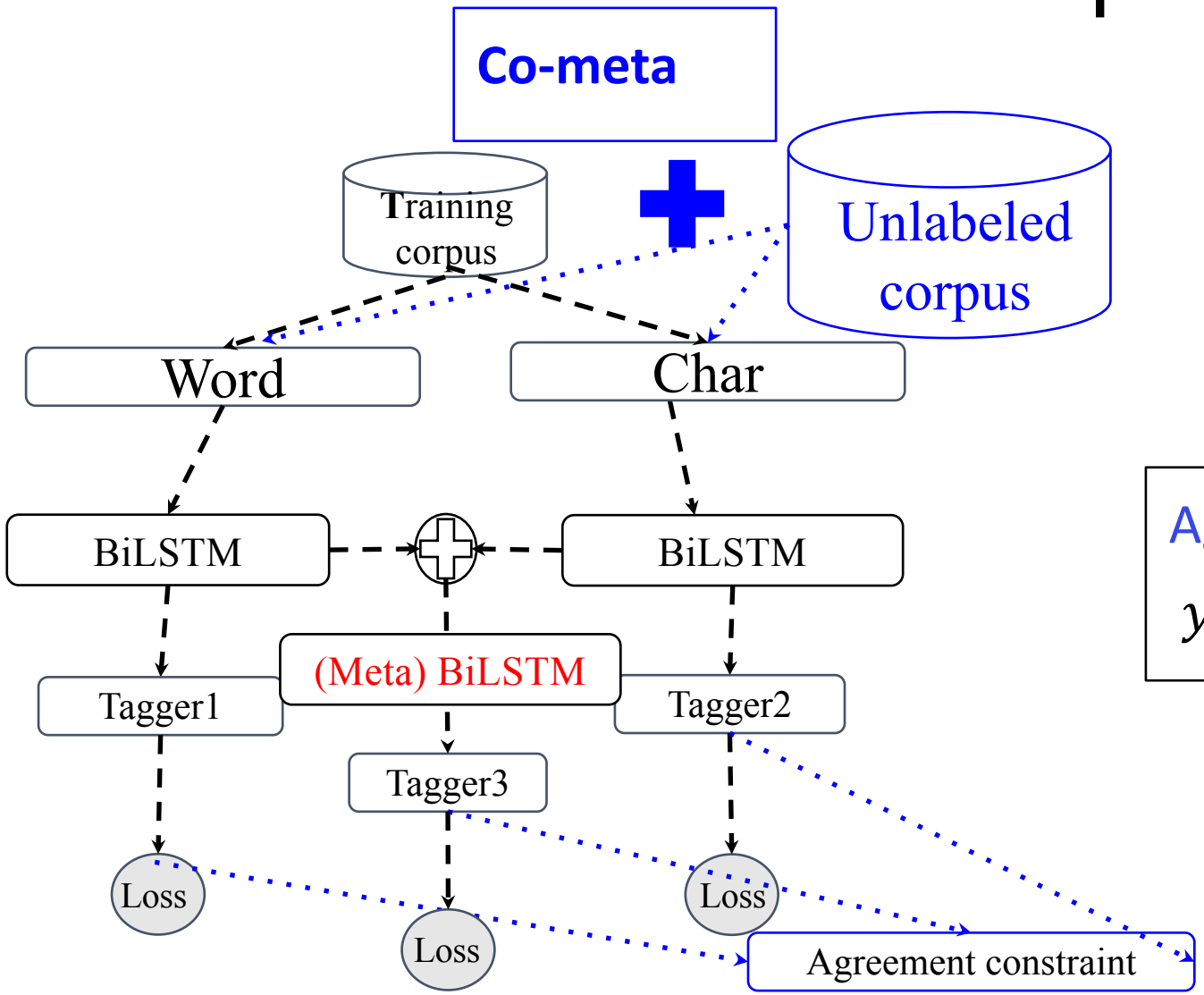
Bohnet et al., *Morphosyntactic tagging with a meta-bilstm model over context sensitive token encodings*, ACL18



Meta-BiLSTM + Semi-supervised model



Meta-BiLSTM + Semi-supervised model

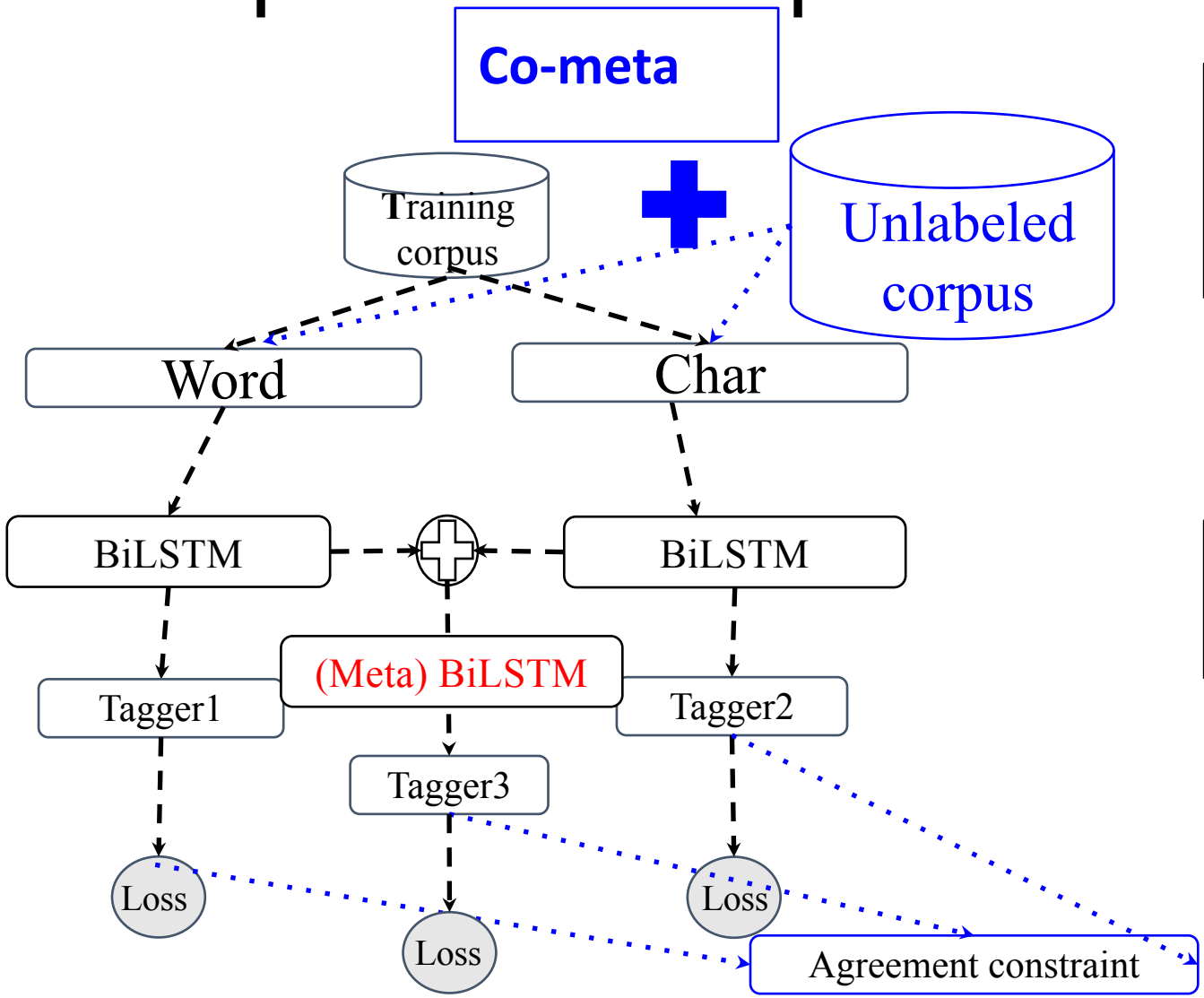


Agreement promotion:

$$y_{word} = y_{char} \iff g(y_{word}, y_{char}) = 1$$

$$g(\hat{y}^{vi}, \hat{y}^{vj}) = \sum_{t=1}^{n} \frac{I(y_t^{vi}, y_t^{vj})}{n}$$

Comparison with previous chapters



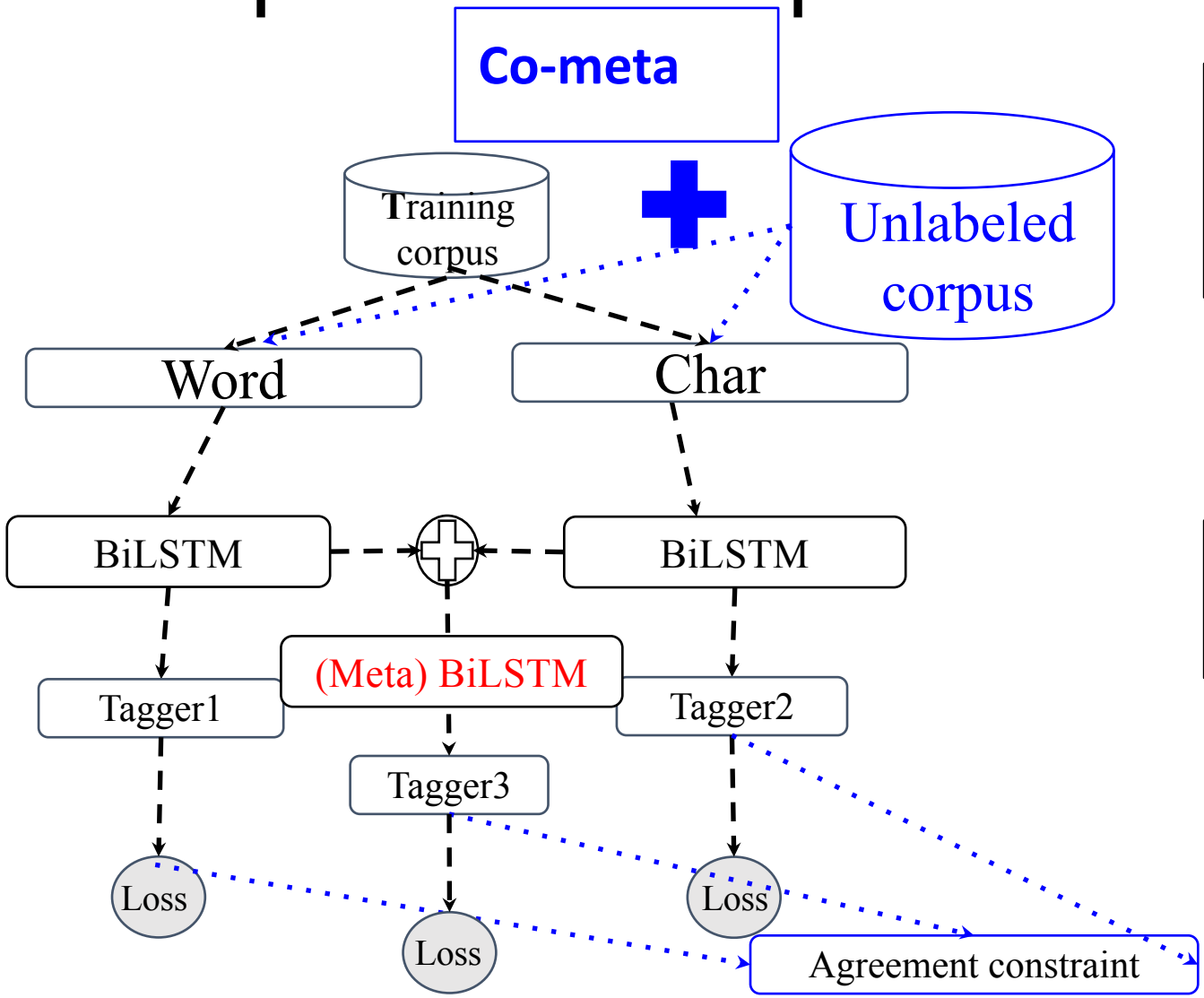
Previous Chapters:
Checking output on the explicit rule $\mathcal{L}^{\mathbf{x}}$

$$\mathbf{y} \in \mathcal{L}^{\mathbf{x}} \Longleftrightarrow g(\mathbf{y}, \mathcal{L}^{\mathbf{x}}) = 0$$

Agreement promotion:
Comparison of pair of output.

$$y_{word} = y_{char} \Leftrightarrow g(y_{word}, y_{char}) = 1$$

Comparison with previous chapters



Previous Chapters:
Checking output on the explicit rule $\mathcal{L}^{\mathbf{x}}$

$$\mathbf{y} \in \mathcal{L}^{\mathbf{x}} \Longleftrightarrow g(\mathbf{y}, \mathcal{L}^{\mathbf{x}}) = 0$$

Agreement promotion:
Comparison of pair of output.

$$y_{word} = y_{char} \Leftrightarrow g(y_{word}, y_{char}) = 1$$

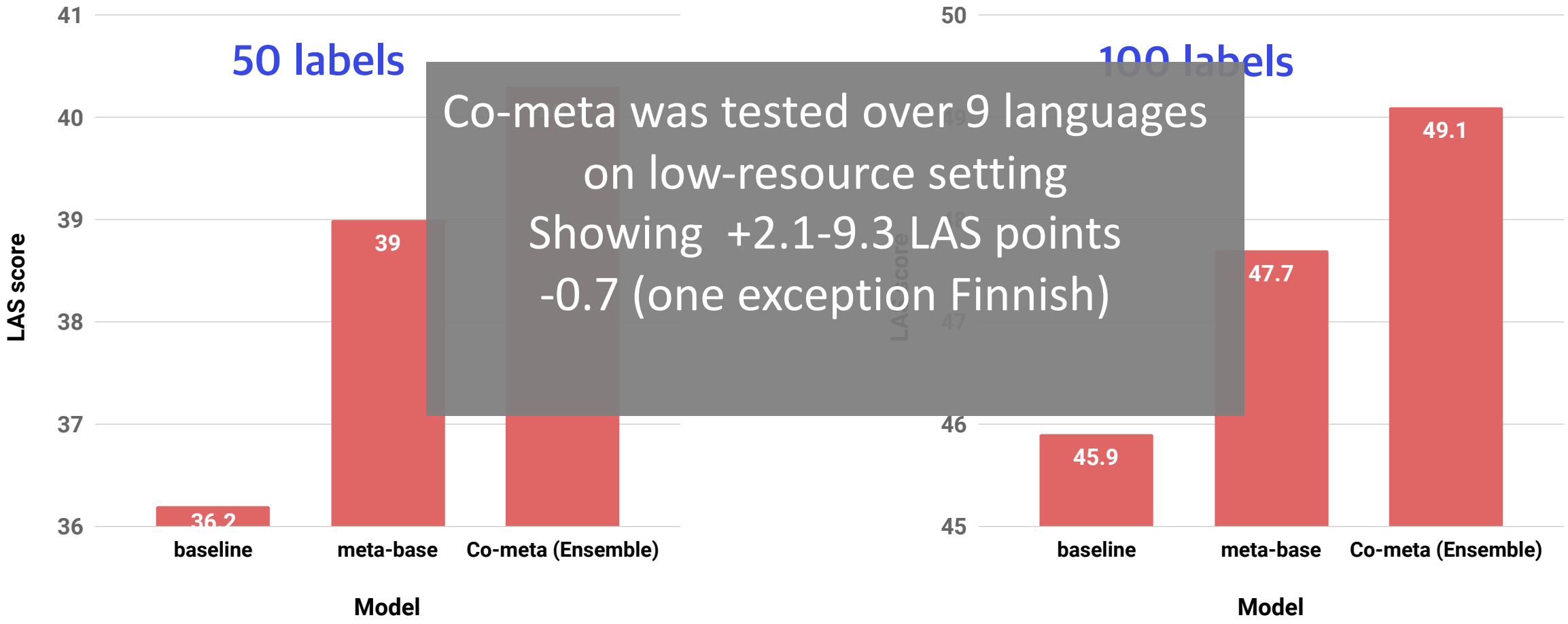
Experiment setup

- Multi-task of dependency parsing & POS tagging.
  - *LAS score* for parsing, *UPOS score* for POS tagging

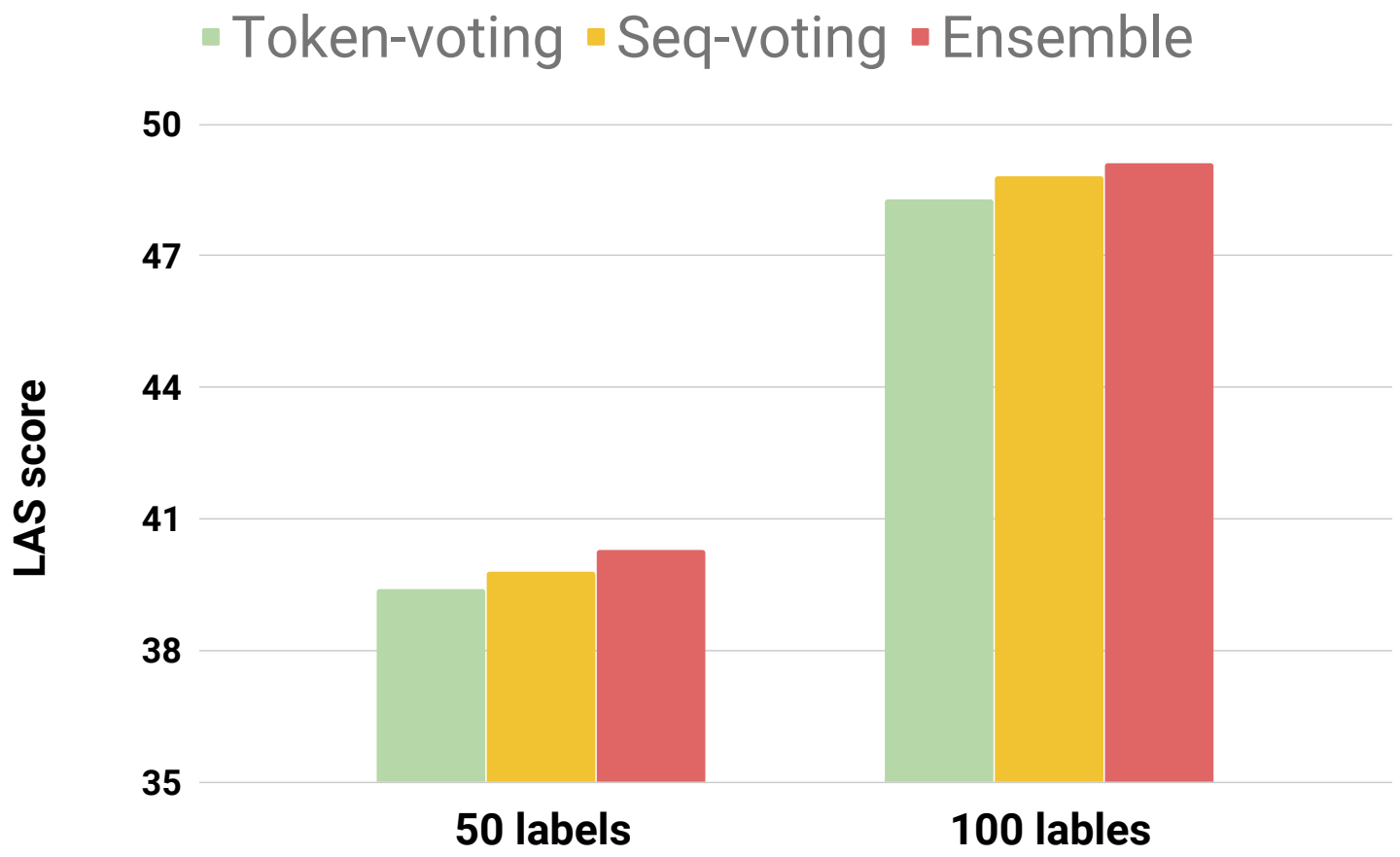
- Using CoNLL2018 dataset.
  - Extremely low-resource setting
  - Plain texts in the same corpus.
  - Tested over 9 languages.

corpus	unlabeled
cs_cac (Czech)	23478
fi_ftb (Finnish)	14981
en_ewt (English)	12543
grc_perseus (Ancient Greek)	11460
he_htb (Hebrew)	5240
zh_gsd (Chinese)	3997
el_bdt (Greek)	1162
ta_ttb (Tamil)	400
kk_ktb (Kazakh)*	12000*

Experiment: comparison to baseline



Experiment: Co-meta variant comparison



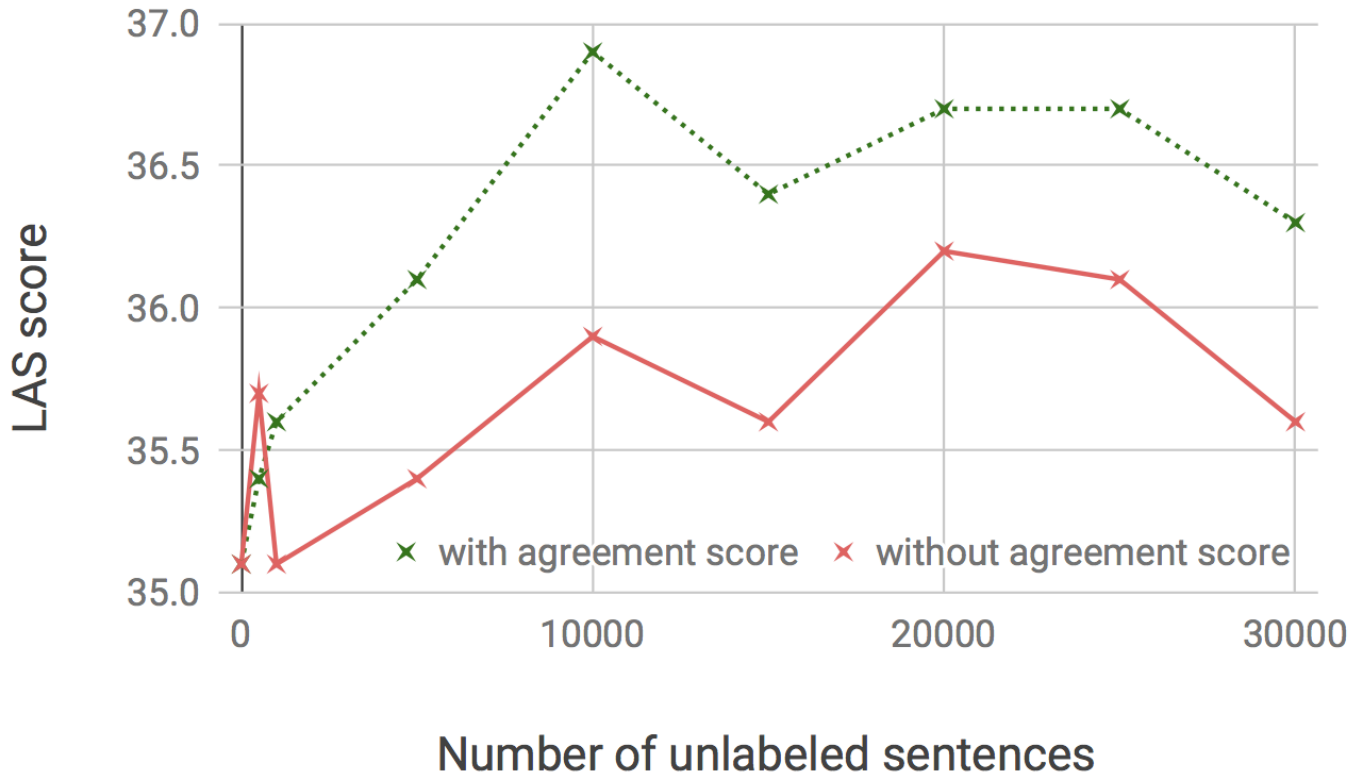
Experiment: Co-meta variant comparison

- Which works the best depends on the language.
  - In average, ensemble works the best.

corpus name	number of unlabeled	CO-META TOKEN VOTING LAS	TOKEN VOTING POS	SEQUENCE VOTING LAS	SEQUENCE VOTING POS	ENSEMBLE LAS	ENSEMBLE POS
cs_cac (Czech)	23478	47.4	79.4	47.4	79.7	**48.7**	**81.4**
grc_perseus (Ancient Greek)	11460	30.8	70.1	**31.7**	**70.9**	31.3	70.7
ta_ttb (Tamil)	400	38.1	69.1	39.0	**69.7**	**40.0**	69.3
kk_ktb[8] (Kazakh)	12000	27.6	56.9	27.9	57.0	**28.7**	57.1

Effect of the agreement score

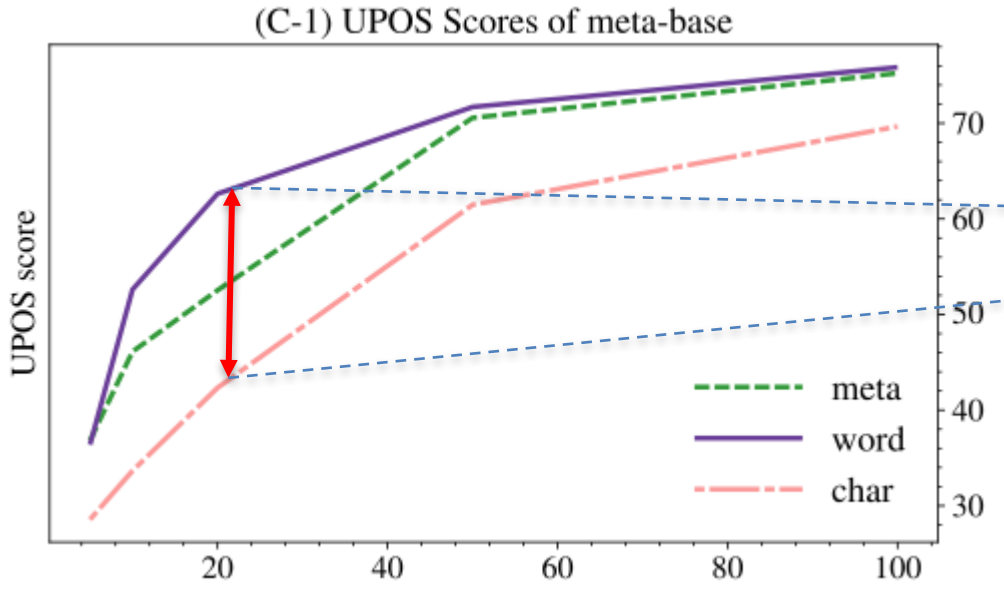
- Without agreement score: simple self-training (red)
- With agreement score: co-meta (green)



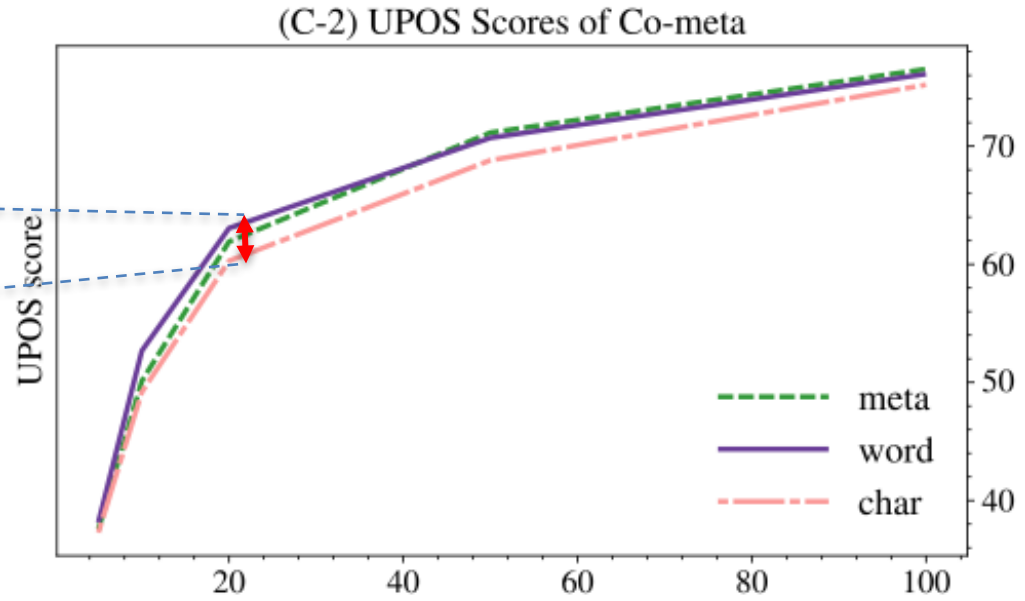
Effect of co-meta on agreement?

Ablation study done on Chinese with increasing training data

Meta-Base
(No use of unlabeled data)

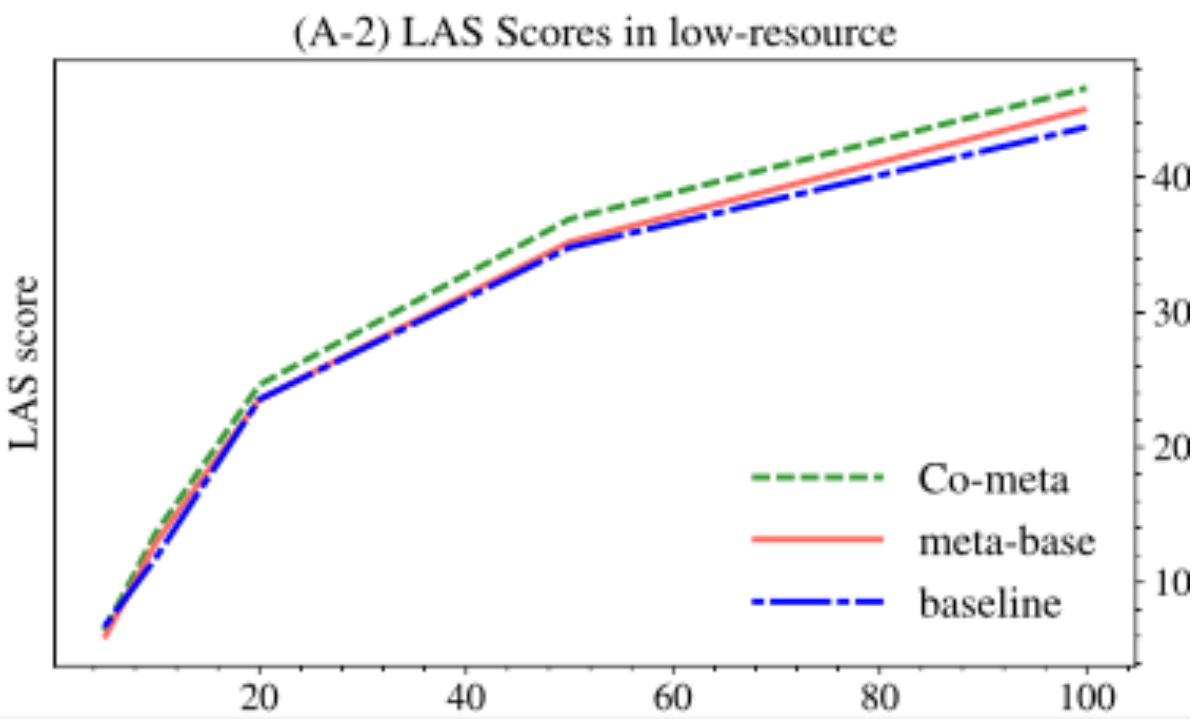


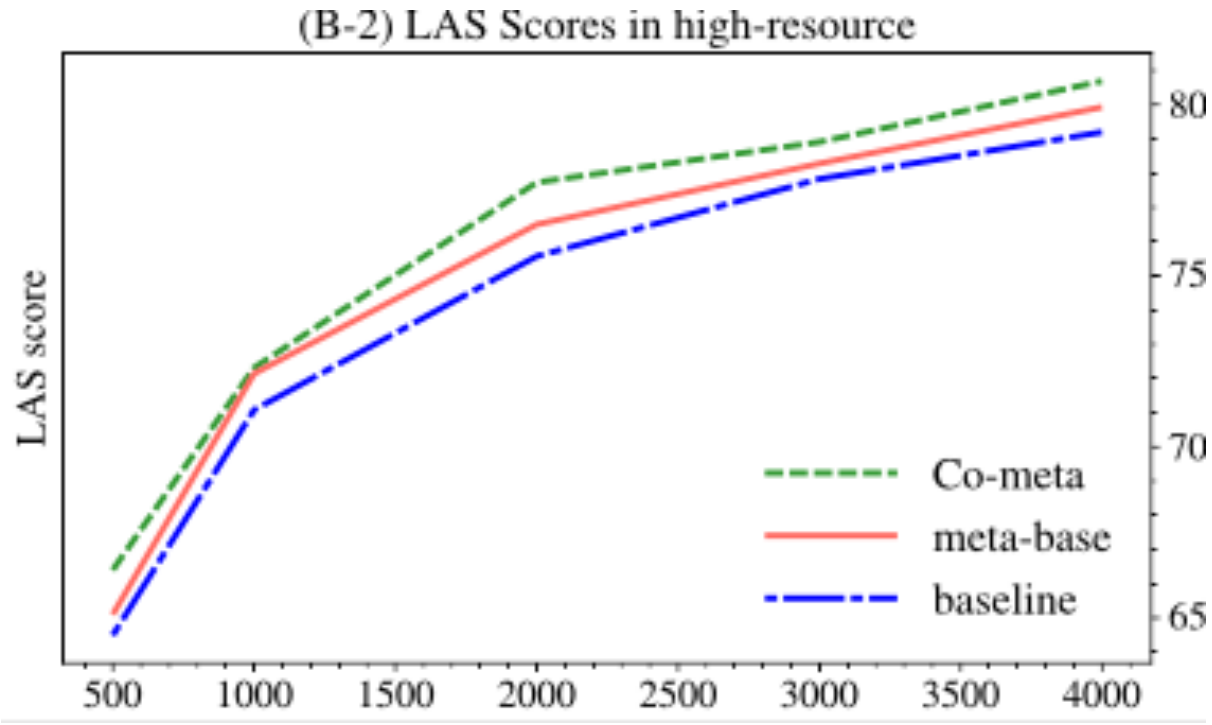
Co-meta
(with agreement score)



Increasing labels.

- Co-meta (green) vs. Meta-base (orange) vs. Baseline (blue)
- Co-meta outperforms all with

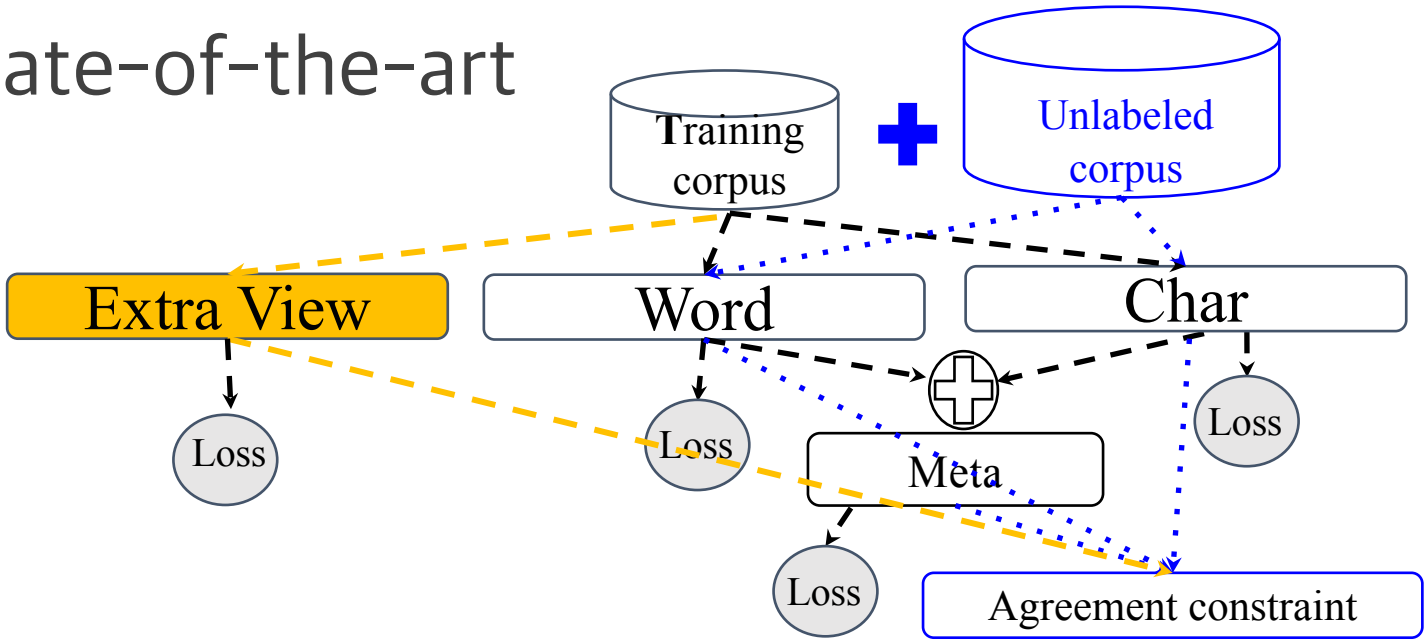




Extra view: ELMo, BERT

- For English, marginal improvement over Meta-Base.
  - But, significant improvement over competing models.
  - (+1.02 LAS, 0.6 POS)
- For Chinese, records the state-of-the-art

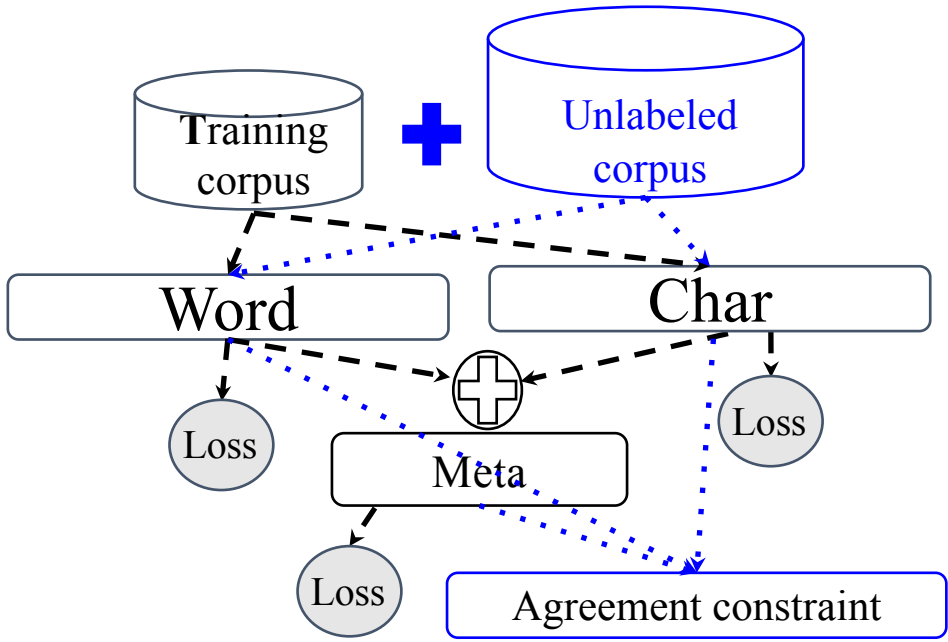
Model	LM	LAS	UAS	POS
UDPIPE	-	80.50	84.64	**94.88**
BASELINE	-	79.70	84.28	94.41
METABASE	-	80.32	84.58	94.72
CO-META	-	**80.71**	**84.99**	94.81
UDIFY	BERT-MULTI	83.75	87.93	95.35
UUPARSER	BERT-MULTI	83.7	-	-
METABASE	BERT-MULTI	83.90	88.07	**96.07**
CO-META	BERT-MULTI	**84.21**	**88.39**	**96.07**



Conclusion on SSL with agreement score

- Agreement score can help multiple views co-learn each other on unlabeled data.

- Co-meta
  - can help extremely low-resource,
  - can expand views
  - incorporate ELMo, BERT



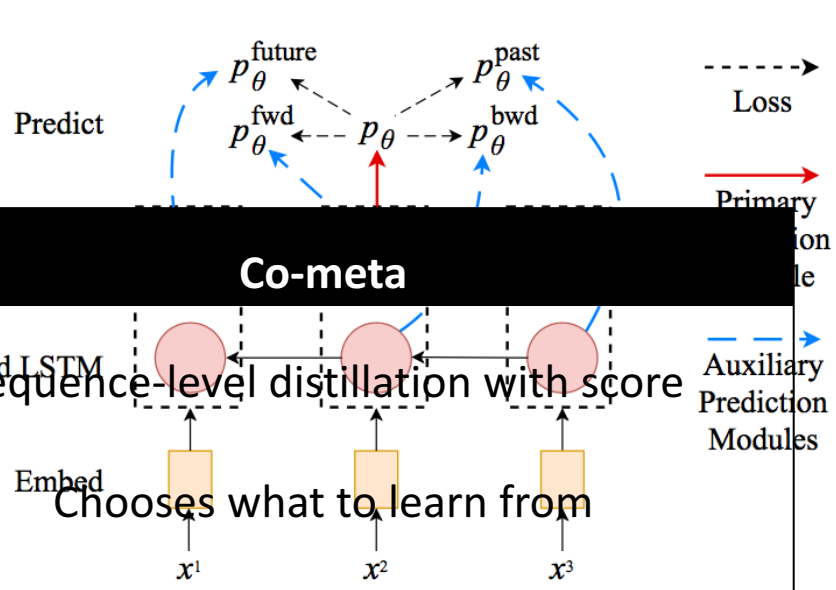
Goals

✔ **Improve** the model performance,

✔ **Overcome** the data **deficiency** problem.

Related work

- Cross-View Training (CVT)[1]
  - Semi-supervised learning on multi-view learning
  - "Primary prediction" from full view [h→,h←] teaches "partial views" [h→] and [h←].



	CVT	Co-meta
Technique	Token-level distillation	Sequence-level distillation with score
Direction of teaching	Primary → Auxiliary	Chooses what to learn from
Multi-view	Limit of time	Different feature, embeddings, structures

[1] Clark et al., Semi-Supervised Semantic Role Labeling with Cross-View Training, EMNLP18

Roadmap

- ☑ Introduction.
- ☑ Applications.
- ☑ Inference with constraints. (Ch.3) [Lee, Mehta, Wick, Tristan, Carbonell AAAI19]
- ☑ Learning with constraints. (Ch.4) [Mehta*, Lee*, Carbonell EMNLP18]
- ☑ Agreement as constraint. (Ch.5) [Lim*, Lee*, Carbonell, Poibeau AAAI20]

Roadmap

- ☑ Applications.
- ☑ Inference with constraints. (Ch.3) [Lee, Mehta, Wick, Tristan, Carbonell AAAI19]
- ☑ Learning with constraints. (Ch.4) [Mehta*, Lee*, Carbonell EMNLP18]
- ☑ Agreement as constraint. (Ch.5) [Lim*, Lee*, Carbonell, Poibeau AAAI20]
- Conclusion & Future work

Conclusions and Discussions

Can we use output constraint as an extra signal to improve?

- GBI: combinatorial search to continuous-space search using constraint loss.
- SSL with constraint loss significantly helped lower-resource models.
- Applied SSL in multi-view learning by introducing agreement score.
  - Showed benefits in extremely low resource.

Interpretable constraint can decompose the end-to-end learning

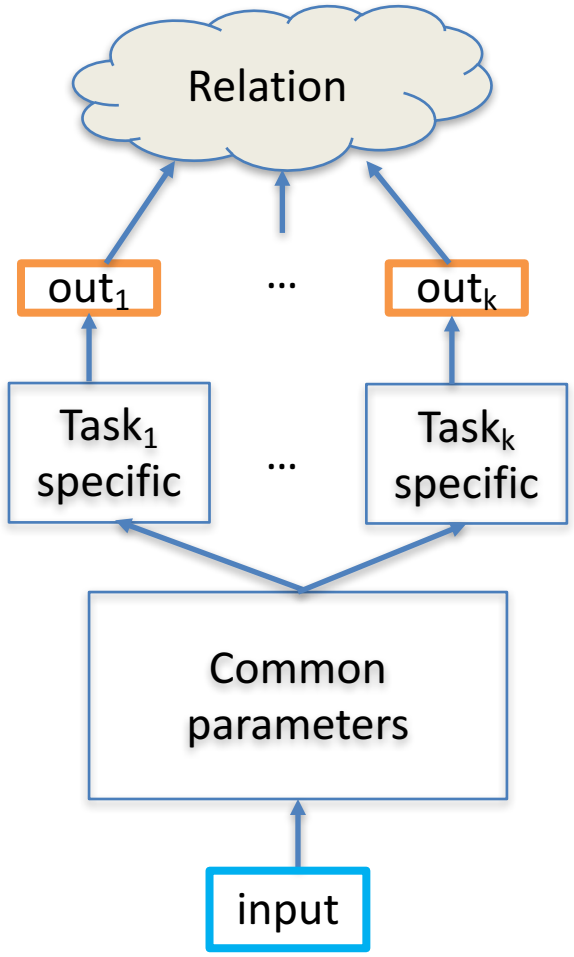
- Leading to more consistent and better-performing model.

Future Work

- Extending to logical constraints.
  (e.g. QA, multi-turn Dialogue)

- Using the *agreement score* over
  Multi-task outputs.

- Toward neuro-symbolic models
  by further modeling inter-task relations.



Thank You!

Questions?

jaylee@cs.cmu.edu

[backup]

- SRL full results (p.115)
  - Off-the-shelf parser (p.116)
  - Time (p.118)
  - Robustness (p.120)
- Other experiments
  - Transducer (p.122)
  - Syntactic Parsing (p.125)
- Constraint (g) definition (p.128)

[backup] SRL full result

Network	Failure rate(%)	Inference	Conv rate(%)	Failure set						Test set	
				Average (%) Disagreement		F1		Exact Match (%)		F1	
				before	after	before	after	before	after	before	after
SRL-100	9.82	GBI A*	42.25 40.40	44.85	24.92 33.91	48.00	59.70 (+11.7) 48.83 (+0.83)	0.0	19.90 13.79	84.40	85.63 (+1.23) 84.51 (+0.11)
SRL-70	10.54	GBI A*	46.22 44.42	45.54	23.02 32.32	47.81	59.37 (+11.56) 50.49 (+2.68)	0.0	19.57 16.12	83.55	84.83 (+1.28) 83.90 (+0.35)
SRL-40	11.06	GBI A*	47.89 44.74	45.71	22.42 32.17	46.53	58.83 (+12.3) 46.53 (+2.88)	0.0	19.45 15.15	82.57	84.03 (+1.46) 82.98 (+0.41)
SRL-10	14.15	GBI A*	44.28 43.66	47.14	24.88 32.80	44.19	54.78 (+10.59) 45.93 (+1.74)	0.0	15.28 12.28	78.56	80.18 (+1.62) 78.87 (+0.31)
SRL-1	21.90	GBI A*	52.85 48.96	50.38	21.45 30.28	37.90	49.00 (+11.10) 41.59 (+3.69)	0.0	12.83 11.25	67.28	69.97 (+2.69) 67.97 (+0.69)

Table 1: Comparison of the GBI vs. A* inference procedure for SRL. We report the avg. disagreement rate, F1-scores and exact match for the *failure set* (columns 5-10) and F1-score for the whole test set (last 2 columns). Also, we report performances on a wide range of reference models SRL-X, where X denotes % of dataset used for training. We employ Viterbi decoding as a base inference strategy (before) and apply GBI (after) in combination with Viterbi.

[backup] SRL using off-the-shelf parser

Parse Tree & Predicate	B1-J5x		B10-J10x		J100-J3x	
	EVAL F1	Avg. Disagreement	EVAL F1	Avg. Disagreement	EVAL F1	Avg. Disagreement
Gold	68.86 (+1.58)	19.38 (-1.79)	79.34 (+0.78)	15.88 (-1.13)	-	-
System predicted	68.57 (+1.29)	19.73 (-1.44)	79.01 (+0.45)	16.39 (-0.62)	84.87 (+0.47)	14.23 (-0.46)

[backup] SRL full result

Genre \Task	Failure rate (%)	Conversion rate (%)	F1 on failure set	
			before	after
	Syntactic Parsing			
Broadcast Conversation (BC)	19.3	98.8	56.4	59.0 (+2.6)
Broadcast News (BN)	11.7	98.1	63.2	68.8 (+5.6)
Pivot Corpus (PT)	9.8	97.8	71.4	75.8 (+4.4)
Telephone Conversation (TC)	10.1	86.2	56.9	57.6. (+0.7)
Weblogs (WB)	17.6	95.3	62.0	63.2 (+1.2)
Genre \Task	***SRL***			
Broadcast Conversation (BC)	26.86	53.88	39.72	52.4 (+12.68)
Broadcast News (BN)	18.51	55.19	39.28	50.58 (+11.3)
Pivot Corpus (PT)	10.01	62.34	47.19	63.69 (+16.5)
Telephone Conversation (TC)	19.09	54.62	47.7	58.04 (+10.34)
Weblogs (WB)	20.32	44.13	47.6	57.39 (+9.39)

[backup] Time

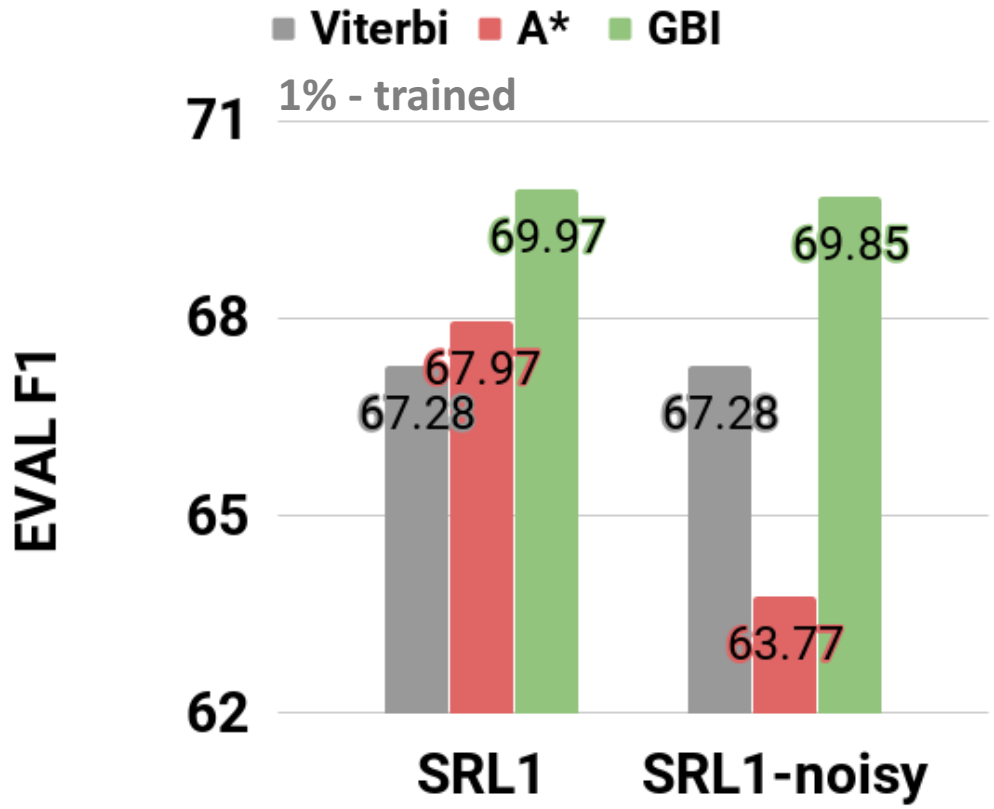
Network	Genre(s)	No. of examples	Failure rate (%)	Inference time (approx. mins)		
				Viterbi	GBI	A*
SRL-100	All	25.6k	9.82	109	288	377
SRL-NW	BC	4.9k	26.86	23	110	117
	BN	3.9K	18.51	18	64	100
	PT	2.8k	10.01	8	19	15
	TC	2.2k	19.01	5	23	20
	WB	2.3k	20.32	12	49	69

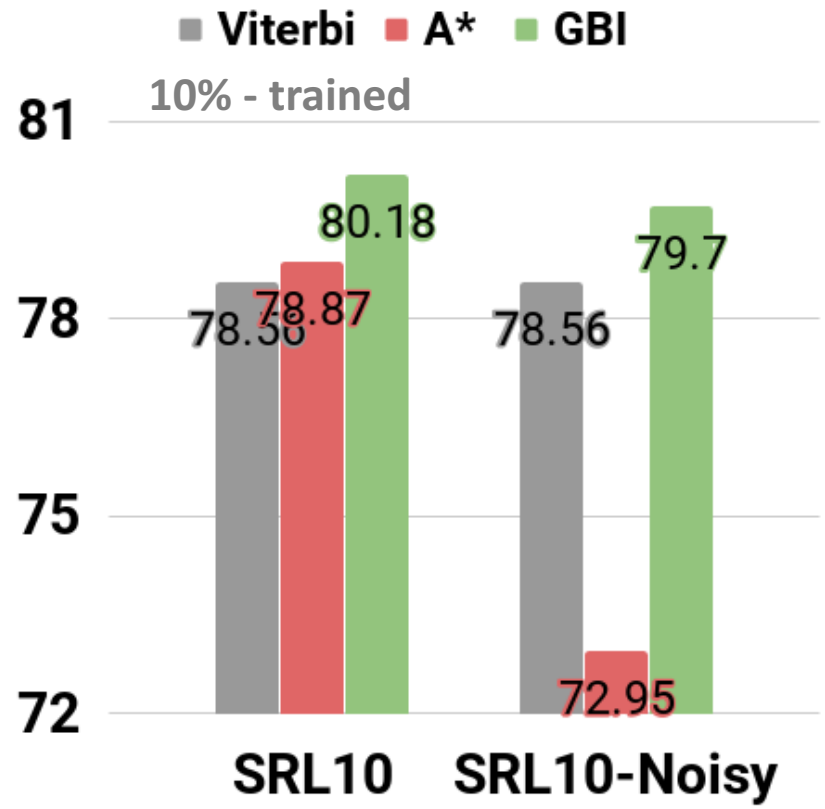
Robustness of GBI

- SRL has about 10% of the span mismatch between annotated SRL output (propbank) & parse tree (treebank)

Robustness of GBI

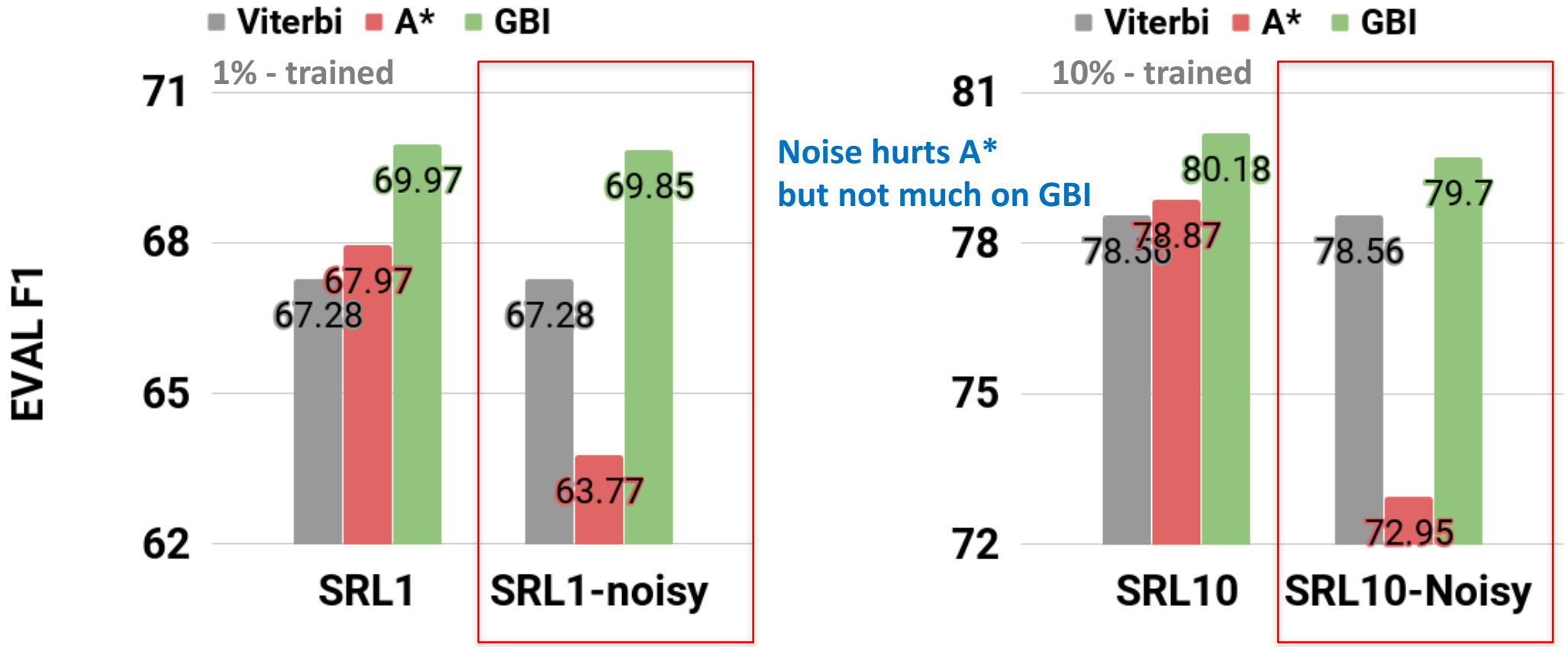
- SRL has about 10% of the span mismatch between annotated SRL output (propbank) & parse tree (treebank)





Robustness of GBI

- SRL has about 10% of the span mismatch between annotated SRL output (propbank) & parse tree (treebank)



Key Idea for the method

Can we nudge the model parameter W to search constraint-satisfying output y?

- Model parameter W is in continuous space.
  - W can be updated with “some” loss function.
- Change of thought:
  - Model parameter update are done in training.
  - But, can we update model parameter to “search” in inference time?

Category	Method	Complexity	e.g.) SRL (T=7, V=130)
Train	W update	$O(TLh^2)$	700k
Inference	Combinatorial search	$O(V^T)$	6.27E+14
Proposed inference	Search Y by updating W (with greedy)	$O(TV) + O(TLh^2)$	0.9k + 700k (E+5)
	Search Y by updating W (with Viterbi)	$O(TV^2) + O(TLh^2)$	110k + 700k (E+5)

GBI on transducer (sequence-to-sequence)

- *Transduction (toy)*
  - A transducer (T) is a function from a 'source language' (L_s) to a 'target language' (L_T).

$$T : \mathcal{L}_S \rightarrow \mathcal{L}_T$$

T: az → aaa
bz → zb

S: azazbzazbzbazbzbzbzbzbz

T: aaaaaazbaaazbzbaaazbzbzbzbzb

constraint:
3 x (# a in **S**) = # a in **T**

GBI on transducer (sequence-to-sequence)

- *Transduction (toy)*
  - A transducer (T) is a function from a 'source language' (L_s) to a 'target language' (L_T).

Experiment Result
- 65.2% conversion rate
- On Failure set (accuracy)
  - Before-GBI: 75.2%
  - After-GBI: 82.4%

$$T : \mathcal{L}_S \rightarrow \mathcal{L}_T$$

T: az → aaa
bz → zb

S: azazbzazbzbazbzbzbzbzbz

T: aaaaaazbaaazbzbaaazbzbzbzbzb

constraint:
3 x (# a in **S**) = # a in **T**

GBI on transducer (sequence-to-sequence)

- *Transduction (toy)*
  - A transducer (T) is a function from a 'source language' (L_s) to a 'target language' (L_T).

Experiment Result
- 65.2% conversion rate
- On Failure set (accuracy)
  - Before-GBI: 75.2%
  - After-GBI: 82.4%

$$T : \mathcal{L}_S \rightarrow \mathcal{L}_T$$

T: az → aaa
bz → zb

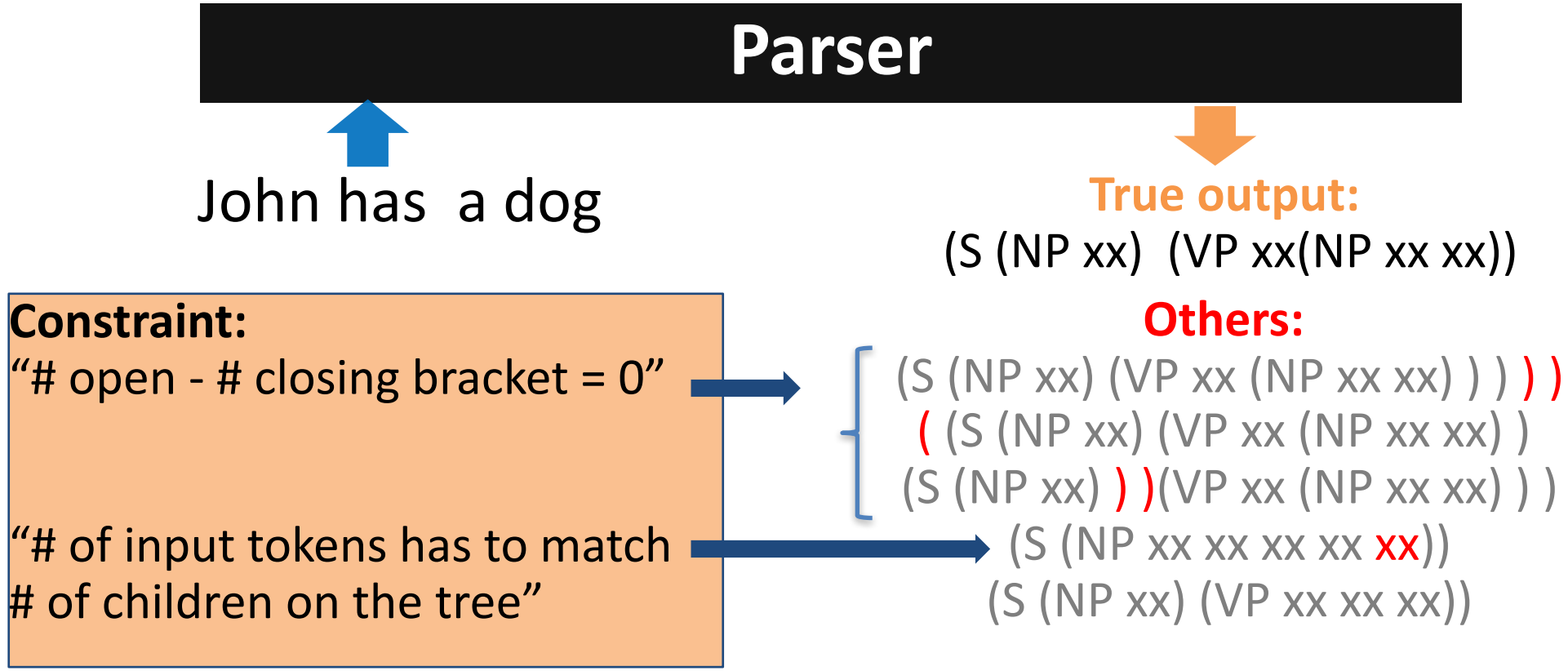
S: azazbzazbzbazbzbzbzbzbz

T: aaaaaazbaaazbzbaaazbzbzbzbzb

constraint:
3 x (# a in **S**) = # a in **T**

GBI on parsing (sequence-to-sequence)

- *Syntactic Parsing*
  - Finding structure of the provided (input) sentence.



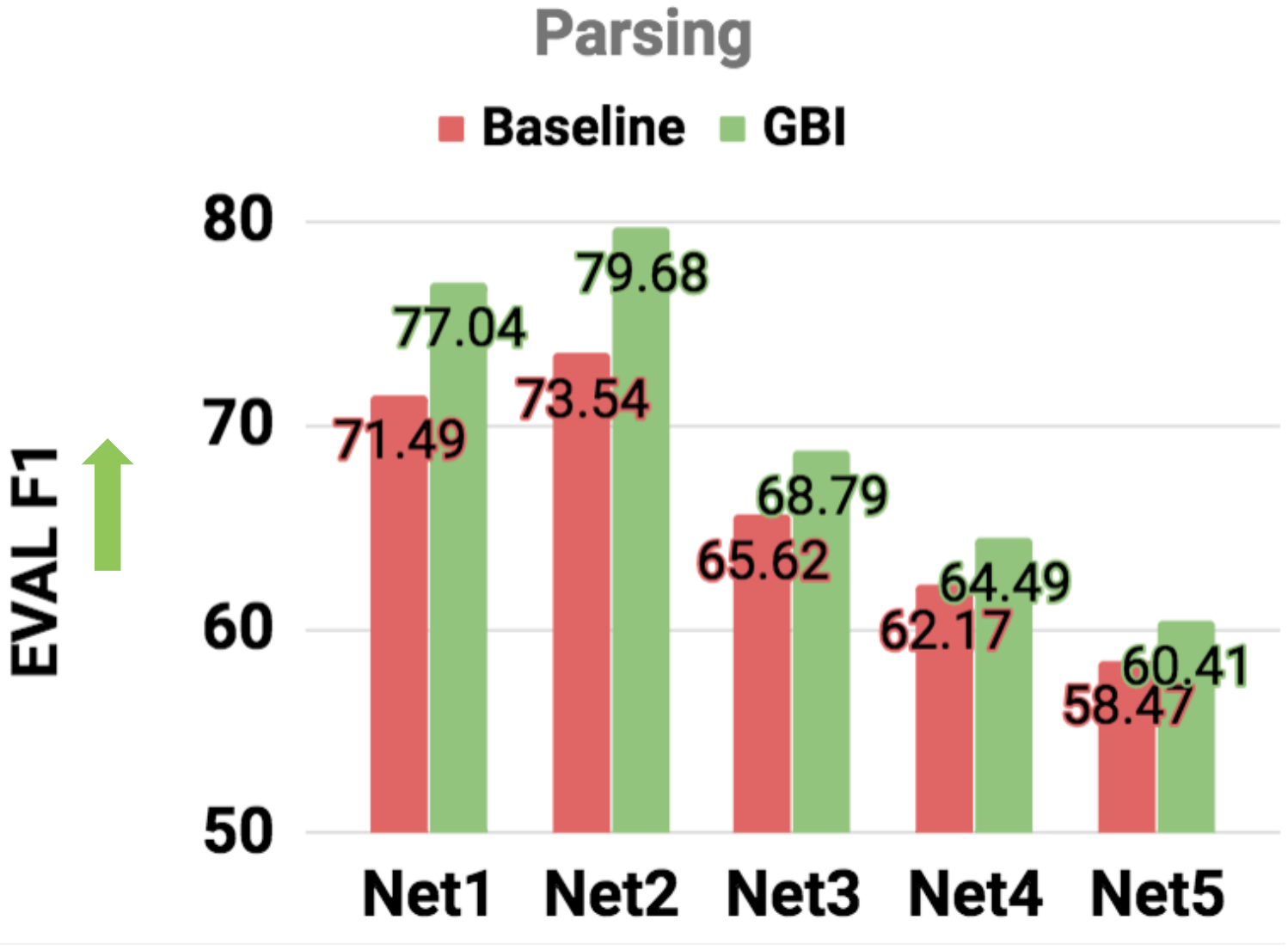
GBI on parsing: Experiments

- Trained 5 parsers to answer Q1-4 (especially Q3)
  - Net1-2: GNMT (bi-directional encoder, uni-directional decoder)
  - Net3-5: uni-directional encoder, decoder.
  - Different dropout, used different portion of the dataset.

name	F1		hyper-parameters			data
	BS-9	greedy	hidden	layers	dropout	(%)
Net1	87.58	87.31	128	3	0.5	100%
Net2	86.63	86.54	128	3	0.2	100%
Net3	81.26	78.32	172	3	no	100%
Net4	78.14	74.53	128	3	no	75%
Net5	71.54	67.80	128	3	no	25%

GNMT: Wu et al, Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, 2016

GBI on parsing (sequence-to-sequence)



Constraints function

- **Rules**
  - # of input tokens = # of children on output tree
  - # opening brackets = # of closing brackets

- **Constraint function definition (x:input, y: output)**
  - **Simple count of errors normalized by** sum of input, output **length**
  - Roughly speaking:

$$g(y, \mathcal{L}^x) = \frac{1}{|x| + |y|} \{abs(|x| - |y_{children}|) + |\text{count}_{open} - \text{count}_{close}|\}$$

Constraints function

- **Rules**
  - # of input tokens = # of children on output tree
  - # opening brackets = # of closing brackets

- **Constraint function definition (x:input, y: output)**
  - **Simple count of errors normalized by** sum of input, output **length**.
  - Roughly speaking:

$$g(y, \mathcal{L}^x) = \frac{1}{|x| + |y|} \{abs(|x| - |y_{children}|) + |\text{count}_{open} - \text{count}_{close}|\}$$

Normalizaiton

Preservation of # of tokens between input & output.

Balancing brackets

Constraint function g on SRL

- srl-spans(**y**): spans of predicted SRL output **y**
- parse-spans(**x**): parsing constituents without labels.
- disagreeing-spans(**x**,**y**) = $\{span_i \in \text{srl-spans}(\mathbf{y}) \mid span_i \notin \text{parse-spans}(\mathbf{x})\}$
- constraint function $g(\mathbf{x},\mathbf{y}) \geq 0$ on SRL

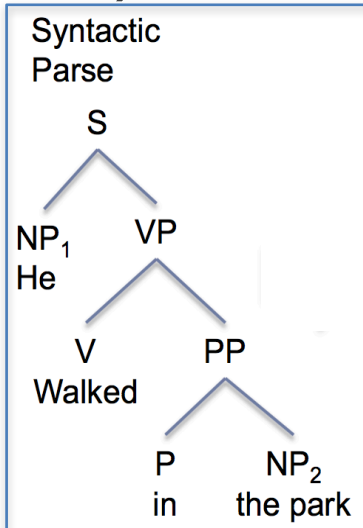
$$g(\mathbf{x}, \mathbf{y}) = \frac{|\text{disagreeing-spans}(\mathbf{x}, \mathbf{y})|}{|\text{srl-spans}(\mathbf{x}, \mathbf{y})|}$$

Constraint on SRL

- Constraint (violation) scores on previous exmaple

X (Input)

"He walked in the park"



Possible span candidate

(He), (walked), (in the park), (walked in the park), (He walked in the park)

Assign scores to the constraint-violating output

g(x,y)	
0.0	(He) (walked) (in the park)
0.33	(He) ~~(walked in)~~(the park)
0.5	~~(He walked)~~ (in the park)
1.0	~~(He walked in the)(park)~~
0.0	(He walked in the park)

Y (output)

VERB – walked
ARG0 – He
LOC– in the park

[Backup] A* algorithm

- If we use “admissible heuristic function”:
  - Admissible: never overestimates the cost of reaching the goal
  - g(n) <= g*(n), where
    - f(n) = g(n) + h(n)
    - f: evaluation function
    - g(n): heuristic function (estimated cost from current node n to goal)
    - g*(n): optimal **cost to reach a goal** from node n.
    - h: the cost **from the start node** to the current node.
- Then A* is
  - Complete: Finds a path if one exists
  - Optimal: Finds the shortest.

[Backup] A* algorithm

For max cost searching

- If we use "admissible heuristic
  - Admissible: never overestimates
  - g(n) >= g*(n), where
    - f(n) = g(n) + h(n)
    - f: evaluation function
    - g(n): heuristic function (estimated
    - g*(n): optimal **cost to reach a goa**
    - h: the cost **from the start node** to
- Then A* is
  - Complete: Finds a path if one ex
  - Optimal: Finds the shortest.

tag sequence is built from left to right. The score for a partial sequence with length t is defined as:

$$f(\boldsymbol{w}, y_{1:t}) = \sum_{i=1}^{t} \log p(y_i \mid \boldsymbol{w}) - \sum_{c \in \mathcal{C}} c(\boldsymbol{w}, y_{1:i}) \quad (17)$$

An admissible A* heuristic can be computed efficiently by summing over the best possible tags for all timesteps after t:

$$g(\boldsymbol{w}, y_{1:t}) = \sum_{i=t+1}^{n} \max_{y_i \in \boldsymbol{T}} \log p(y_i \mid \boldsymbol{w}) \quad (18)$$

If there is no constraint violation, g(n)=g*(n)

If there is constraint violation, g(n)>g*(n)

→ *g(n) is admissible!*

[Backup] A* algorithm

- If we use “admissible heuristic function”
  - Admissible: never overestimates the cost o
  - h(n) <= h*(n), where
    - f(n) = h(n) + g(n)
    - f: evaluation function
    - h(n): heuristic function (estimated cost from cu
    - h*(n): optimal **cost to reach a goal** from node $
    - g: the cost **from the start node** to the current
- If h(x)<=d(x,y)+h(y)
- Then A* is
  - Complete: Finds a path if one exists
  - Optimal: Finds the shortest.

2.2 Constrained A* Decoding

The approach described so far does not model any dependencies between the output tags. To incorporate constraints on the output structure at decoding time, we use A* search over tag prefixes for decoding. Starting with an empty sequence, the tag sequence is built from left to right. The score for a partial sequence with length t is defined as:

$$f(\boldsymbol{w}, y_{1:t}) = \sum_{i=1}^{t} \log p(y_i \mid \boldsymbol{w}) - \sum_{c \in \mathcal{C}} c(\boldsymbol{w}, y_{1:i}) \quad (17)$$

An admissible A* heuristic can be computed efficiently by summing over the best possible tags for all timesteps after t:

$$g(\boldsymbol{w}, y_{1:t}) = \sum_{i=t+1}^{n} \max_{y_i \in \boldsymbol{T}} \log p(y_i \mid \boldsymbol{w}) \quad (18)$$

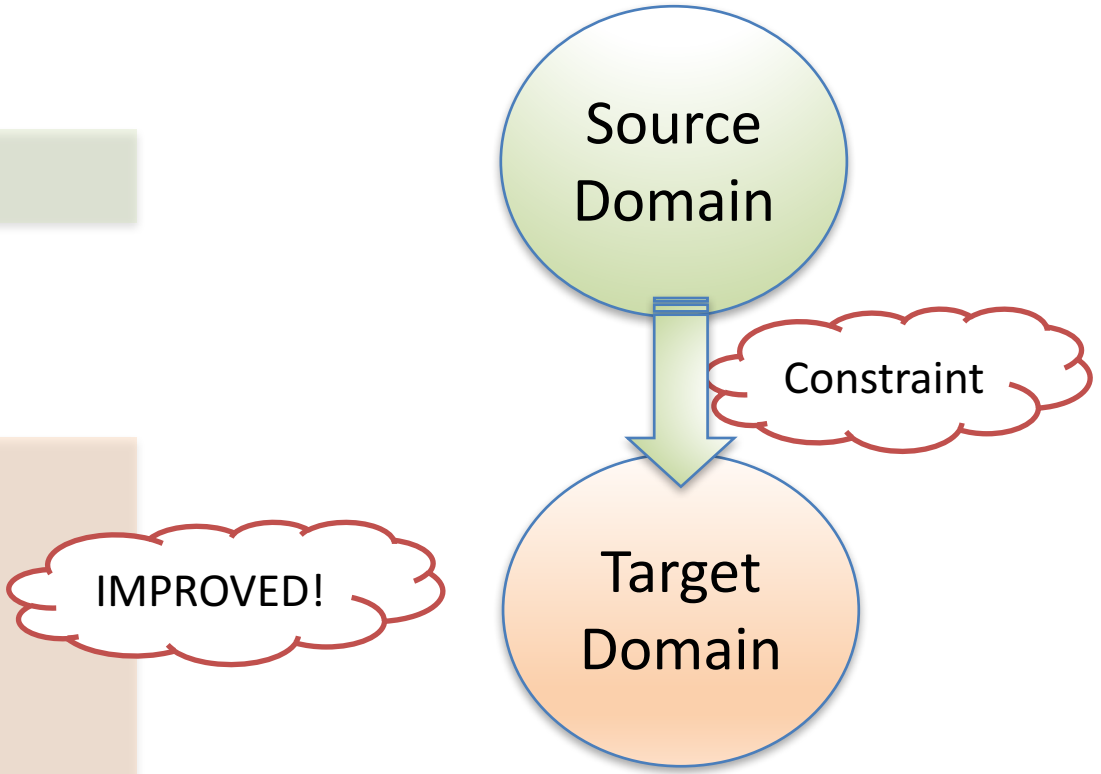
Exploration of the prefixes is determined by an agenda $\mathcal{A}$ which is sorted by $f(\boldsymbol{w}, y_{1:t}) + g(\boldsymbol{w}, y_{1:t})$. In the worst case, A* explores exponentially many prefixes, but because the distribution $p(y_t \mid \boldsymbol{w})$ learned by the BiLSTM models is very peaked, the algorithm is efficient in practice.

GBI on out-of-domain data (Q5)

- CoNLL2012
  - NewsWire (NW)
  - Broadcast Conversation (BC)
  - Broadcast News (BN)
  - Pivot Corpus (PT)
  - Telephone Conversation (TC)
  - Weblogs (WB)

GBI on out-of-domain data

- CoNLL2012
  - NewsWire (NW) which includes WSJ
  - Broadcast Conversation (BC)
  - Broadcast News (BN)
  - Pivot Corpus (PT)
  - Telephone Conversation (TC)
  - Weblogs (WB)



Can constraint injection transfer to other domains?

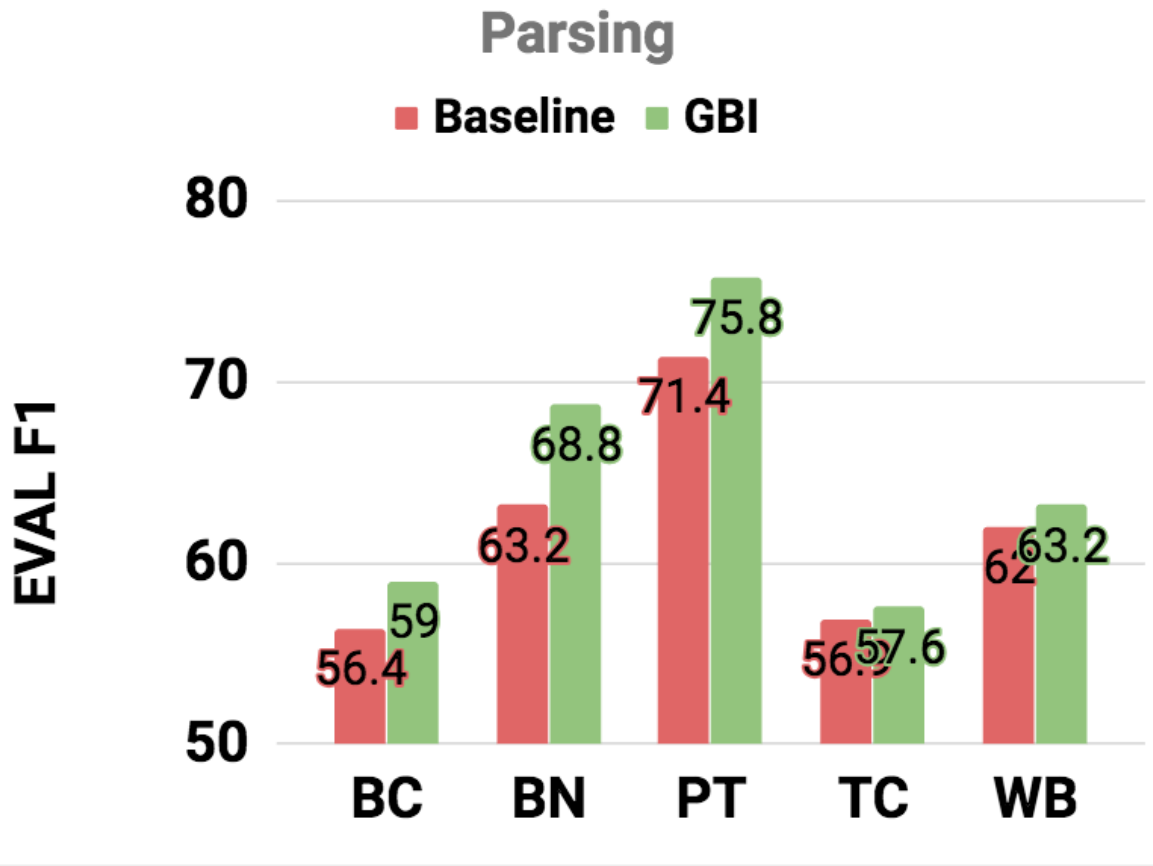
GBI on out-of-domain data

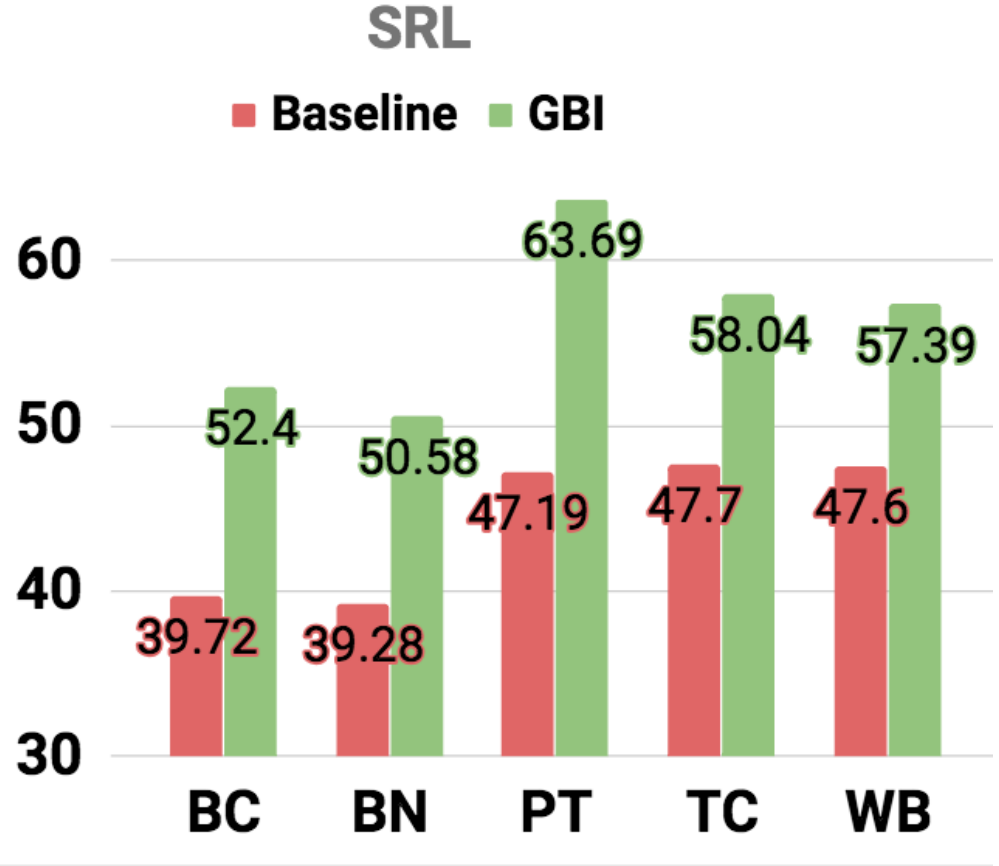
- Generally higher failure rate on out-of-domain evaluation compared to in-domain (parsing: 11.9 %, SRL: 18.1%).

Genre \Task	Failure rate (%)	Conversion rate (%)	F1 on failure set	
			before	after
		Syntactic Parsing		
Broadcast Conversation (BC)	19.3	98.8	56.4	59.0 (+2.6)
Broadcast News (BN)	11.7	98.1	63.2	68.8 (+5.6)
Pivot Corpus (PT)	9.8	97.8	71.4	75.8 (+4.4)
Telephone Conversation (TC)	10.1	86.2	56.9	57.6. (+0.7)
Weblogs (WB)	17.6	95.3	62.0	63.2 (+1.2)
Genre \Task		***SRL***		
Broadcast Conversation (BC)	26.86	53.88	39.72	52.4 (+12.68)
Broadcast News (BN)	18.51	55.19	39.28	50.58 (+11.3)
Pivot Corpus (PT)	10.01	62.34	47.19	63.69 (+16.5)
Telephone Conversation (TC)	19.09	54.62	47.7	58.04 (+10.34)
Weblogs (WB)	20.32	44.13	47.6	57.39 (+9.39)

GBI on out-of-domain data

- In summary, improves in all cases (reporting on failure set)
  - While parsing is more sensitive to domains.



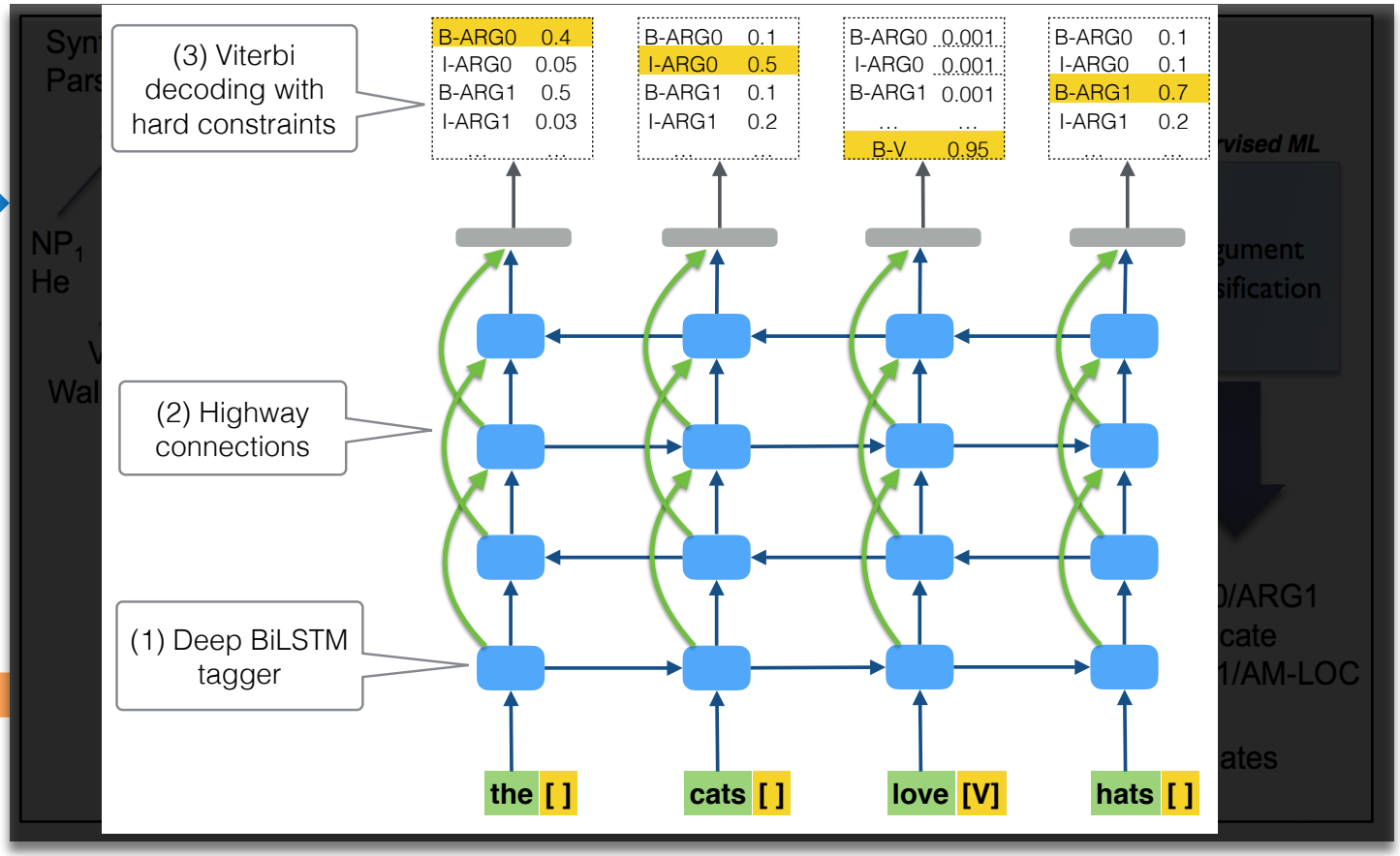


[Revisit] SRL architecture

- Semantic Role Labeling

X (Input)

The cat loves hats



Y (output)

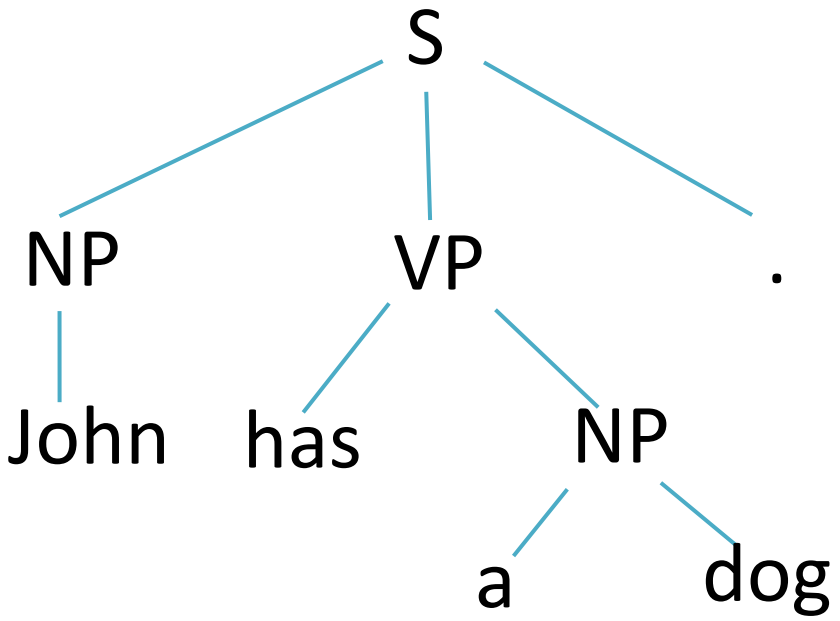
VERB – love
ARG0 – the cats
ARG1– hats

Slide credits: Luheng He

How to express the Grammar Tree as a sequence?

Proposed model

- "As a command" to produce the tree.

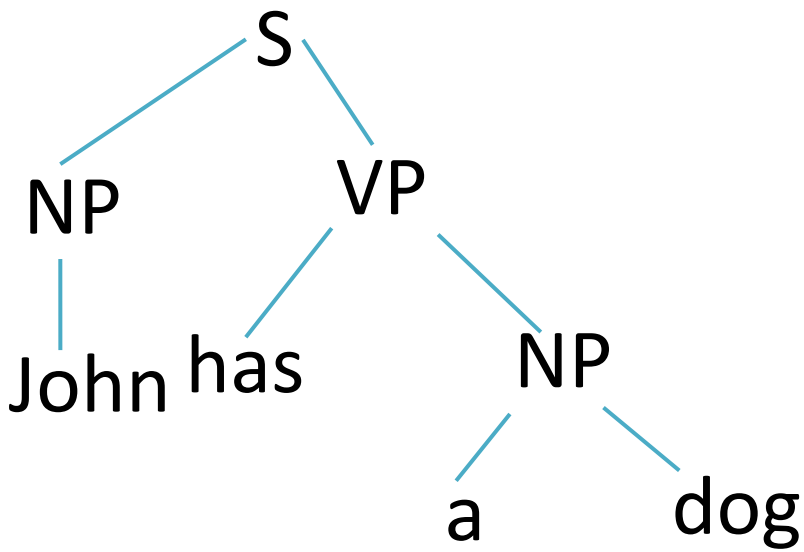


- Shift-Reduce syntax
  - Bottom-up approach
  - Similar to how human minds work.

How to express the Grammar Tree as a sequence?

Proposed model

- "As a command" to produce the tree
  s R-1-NP s s s R-2-NP R-2-VP R-2-S



- Shift-Reduce syntax
  - **s** corresponds to "push" in stack.
  - **R-#-XX** corresponds to "pop"-ing that **#** of elements , merging them together to **XX**, and pushing it back to the stack.

GBI case study

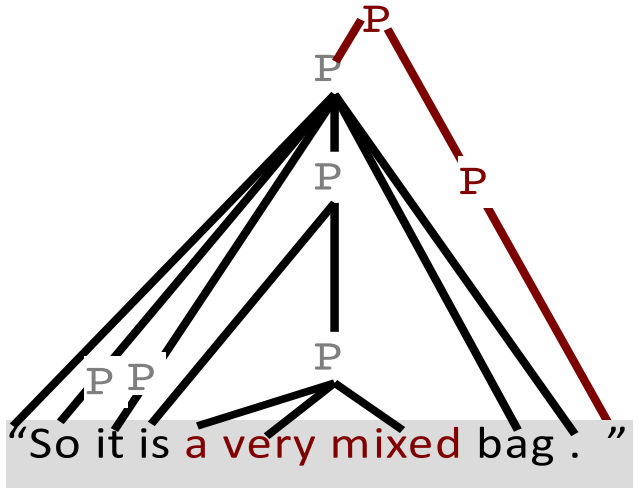
Case study of GBI on parsing

- Comparison to decoder with constraints on each step.
- **Unconstrained decoding + Error signal propagation** can be more powerful.

Constraints in decoder

`ssr!sr!ssssrrr!rr!ssrrrrrr!sr!rr!`

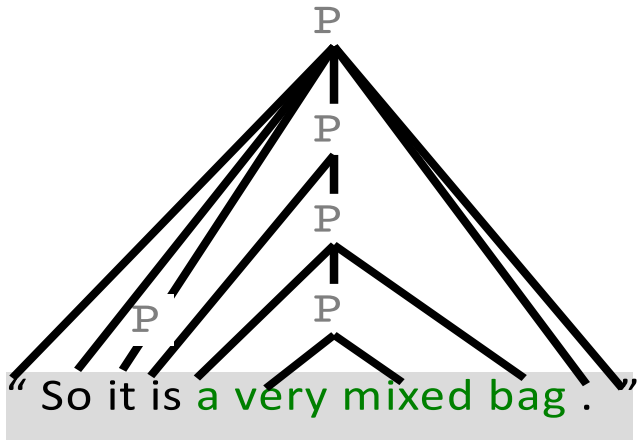
(“(so) (it) (is (a very mixed)) bag .) (”))



Our method
(recovered true)

`sssr!ssssrr!srrr!rr!ssrrrrrr!`

(“ so (it) (is (a (very mixed) bag)) . ”)



GBI extra experiment

- SRL has the following constraints
  - BIO - captured by vitrebi
  - Syntactic consistency
  - U,C,R violation
    - Unique core (U) / Continuation (C) / Reference (R) roles

- SOTA models have UCR violations
- GBI successfully reduces U,C,R violations.
  - No F1 score improvement, but the side effect of U,C,R is still unstudied as SRL output is used as an input to the following system.

GBI on Transduction

- Case study of GBI on transduction problem.

azazbzazbzbzazbzbzbzbzbz ⟶ aaaaaazbaaazbzbaaazbzbzbzbzb

iteration	output	loss	accuracy
0	aaaaaazbaaazbaazbzbzbzbaaazb	0.2472	66.7
1	aaaaaazbaaazbaazbzbzbzbaaazb	0.2467	66.7
2	aaaaaazbaaazbaazbzbzbzbaaazb	0.2462	66.7
3	aaaaaazbaaazbzbaaazbzbzbzbzb	0.0	100.0

Table 1: An example for which enforcing the constraints improves accuracy. Red indicates errors.

bzbzbzbzazbzbzazazazazbz⟶zbzbzbzbaaazbzbaaaaaaaaaaaaazb

iteration	output	loss	accuracy
0	zbzbzbzbaaazbaaaaaaaaaaazbaaa	0.2954	74.2
4	zbzbzbzbzbaaaaaaaaazbzbaaaaaa	0.0	60.0

Table 2: An example for which enforcing the constraints degrades accuracy. Errors in red.

GBI in detail

Gradient-Based Inference (GBI)

- Optimization with constraints

$$\begin{aligned} \max_{\mathbf{y}} \quad & \Psi(\mathbf{x}, \mathbf{y}, W) \\ \text{s.t.} \quad & g(\mathbf{y}, \mathcal{L}^{\mathbf{x}}) = 0 \end{aligned}$$

- Standard Lagrangian Relaxation

$$\min_{\lambda} \max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W) + \lambda g(\mathbf{y}, \mathcal{L})$$

- Modified Lagrangian Relaxation

$$\min_{W_\lambda} \max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W) + \Psi(\mathbf{x}, \mathbf{y}, W_\lambda) g(\mathbf{y}, \mathcal{L})$$

Likelihood

Constraint violation

Gradient-Based Inference (GBI)

- Standard Lagrangian Relaxation

$$\min_{\lambda} \max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W) + \lambda g(\mathbf{y}, \mathcal{L})$$

- Gradient-Based Inference

$$\begin{aligned} \min_{W_\lambda} \quad & \Psi(\mathbf{x}, \hat{\mathbf{y}}, W_\lambda) g(\hat{\mathbf{y}}, \mathcal{L}) + \alpha \|W - W_\lambda\|_2 \\ \text{where} \quad & \hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W_\lambda) \end{aligned}$$

Gradient-Based Inference (GBI)

- Modified Lagrangian Relaxation

$$\min_{W_\lambda} \max_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W) + \Psi(\mathbf{x}, \mathbf{y}, W_\lambda) g(\mathbf{y}, \mathcal{L})$$

- Gradient-Based Inference

$$\begin{aligned} \min_{W_\lambda} \quad & \Psi(\mathbf{x}, \hat{\mathbf{y}}, W_\lambda) g(\hat{\mathbf{y}}, \mathcal{L}) + \alpha \|W - W_\lambda\|_2 \\ \text{where} \quad & \hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}} \Psi(\mathbf{x}, \mathbf{y}, W_\lambda) \end{aligned}$$

Training & Inference of a neural model

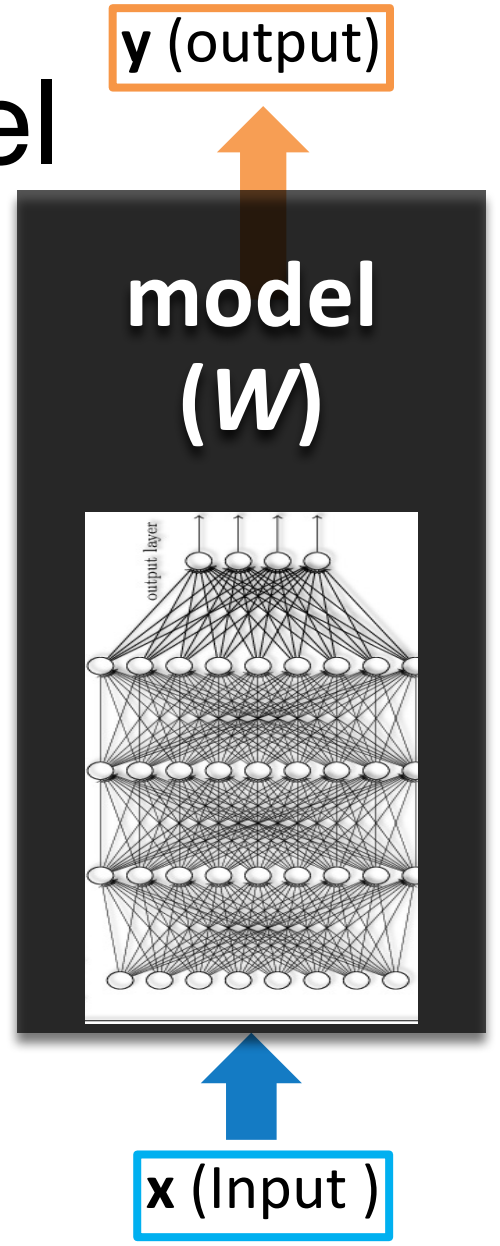
- **Training**
  - Maximizing log-likelihood of (input, true output) pair.
    - x_L, y_L : labeled input, output.
    - Ψ: score (log probability, energy)

$$\max_{W} \quad \Psi(\mathbf{x_L}, \mathbf{y_L}, W)$$

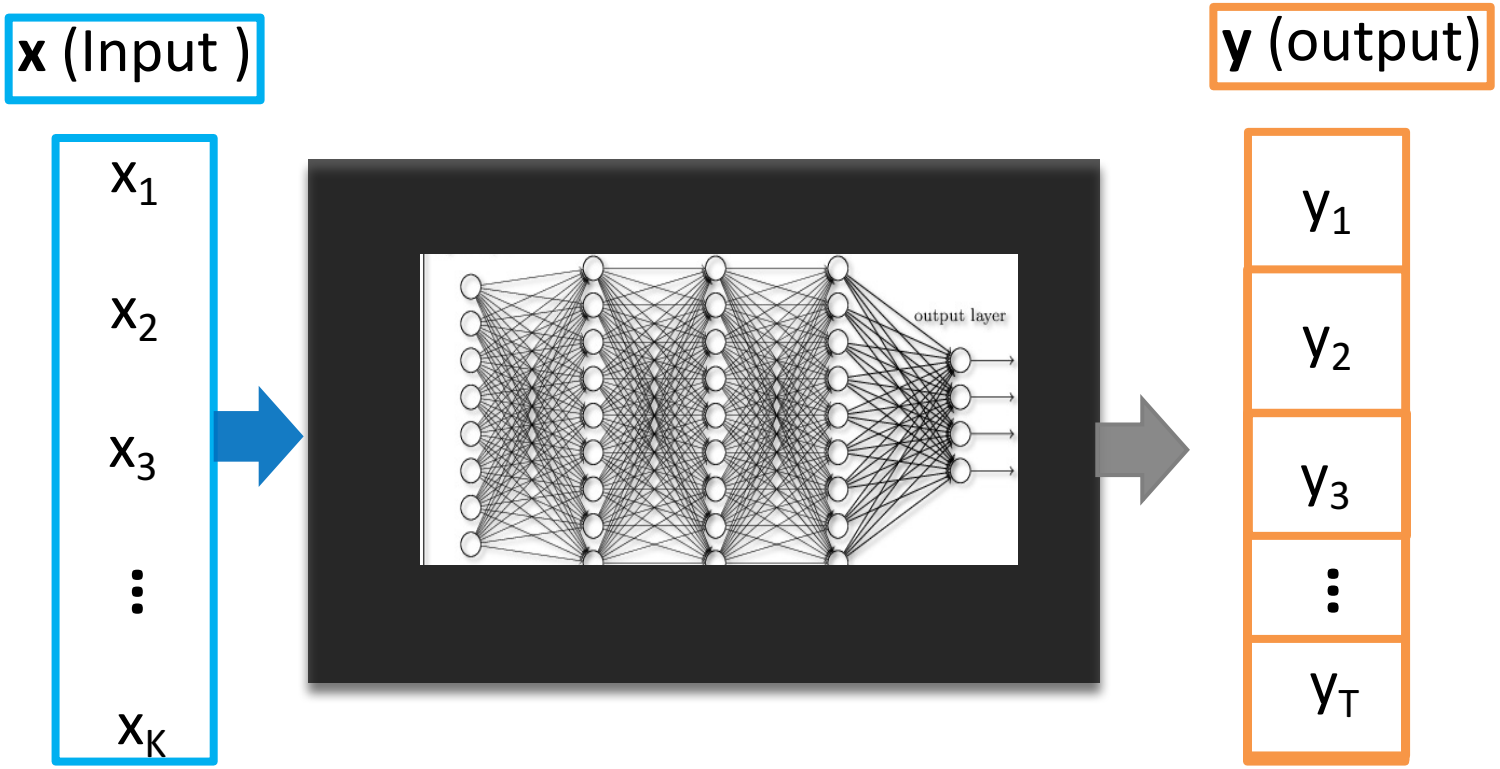
- **Inference**
  - Finding the output **y** with the highest model score (probability).
    - x: a test input.

$$\max_{\mathbf{y}} \quad \Psi(\mathbf{x}, \mathbf{y}, W)$$

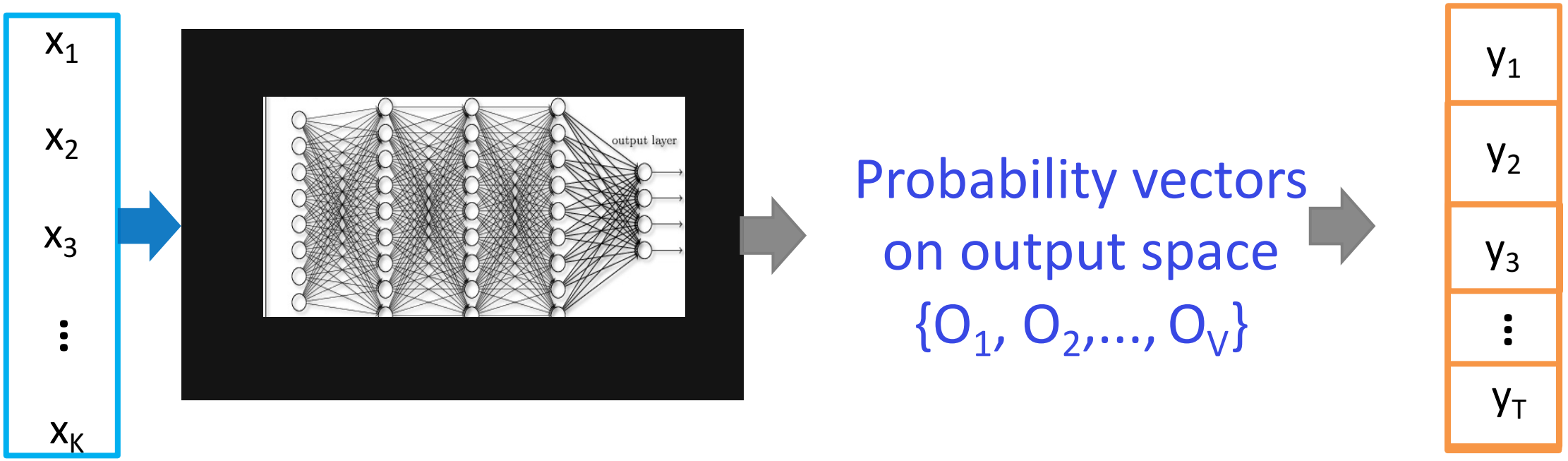
Learned model



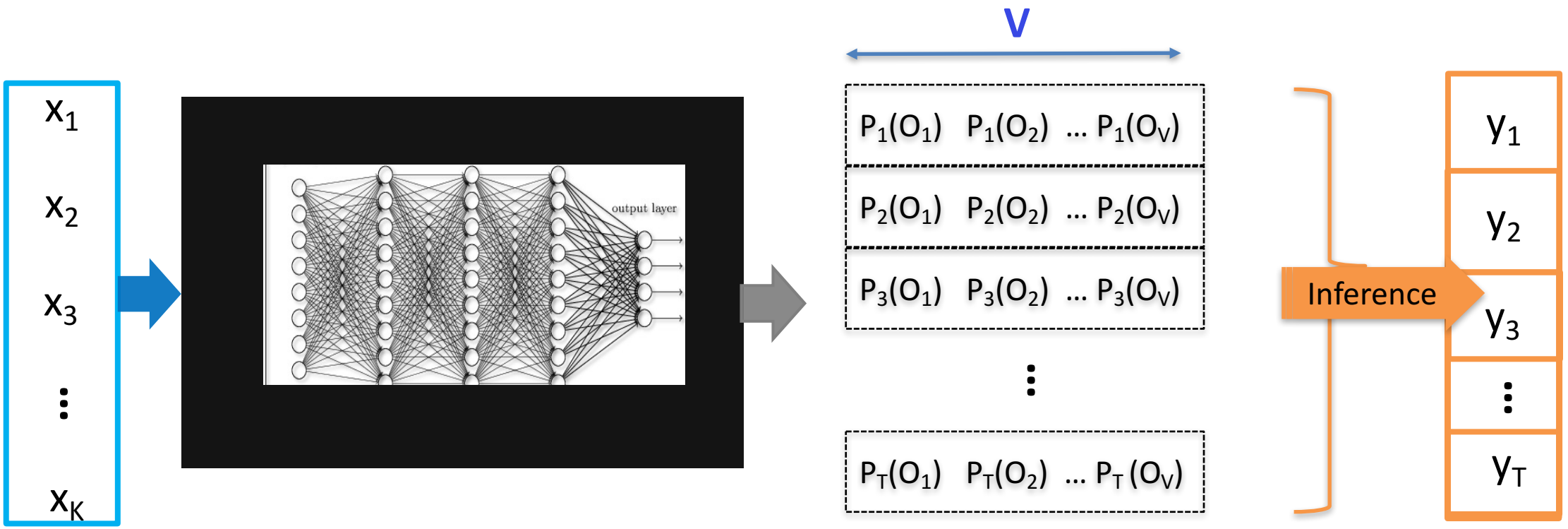
Inference on sequence output



Inference on sequence output

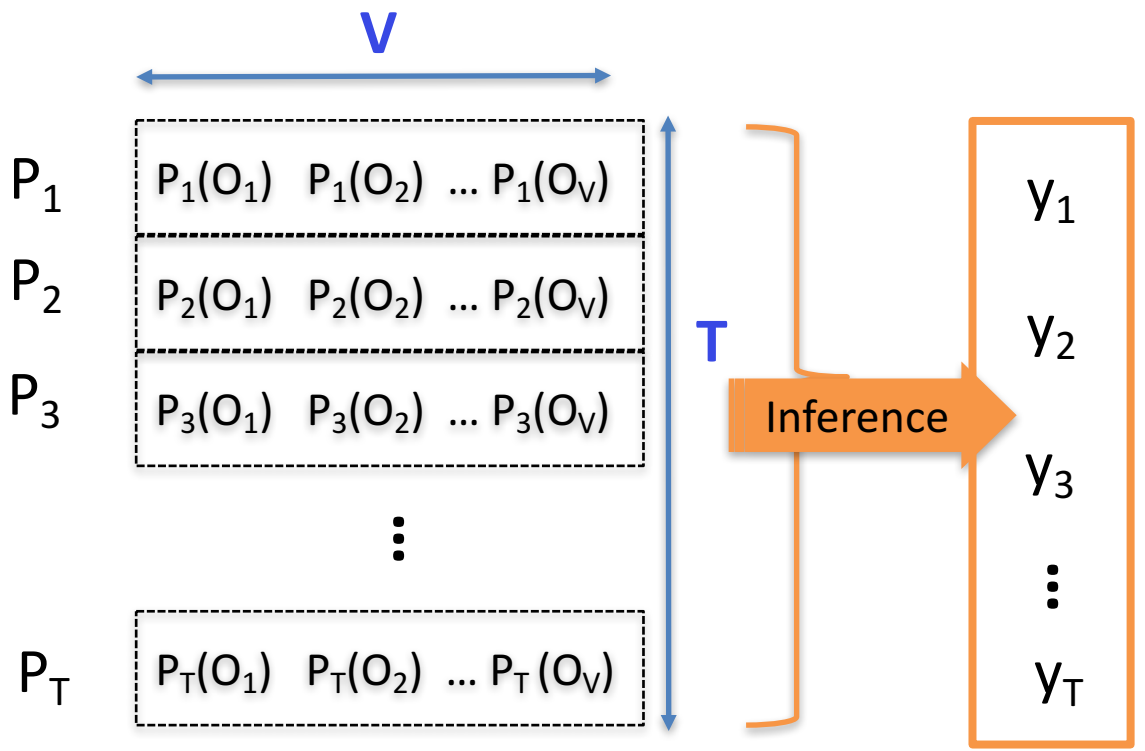


Inference on sequence output



Inference on sequence output

- The complexity of finding max scoring **Y**:
  - no dependence between output.
    - Greedy: ***O(TV)***
  - 1st-order constraint.
    - ***O(TV²)***
  - Global constraint e.g. sum(y_i , y_{i+1} ,⋯, y_{i+k})≤ k
    - ***O(V^T) (A* algorithm is often used)***



Inference on sequence output

- The complexity of finding max scoring **Y**:
  - no dependence between output.
    - Greedy: ***O(TV)***
  - 1st-order constraint.
    - ***O(TV²)***
  - Global constraint e.g. sum(y_i , y_{i+1} ,⋯, y_{i+k})≤ k
    - ***O(V^T) (A* algorithm is often used)***

Post-processing is often used instead:

Get multiple candidates and filter out constraint-violating candidates.

Change of perspective

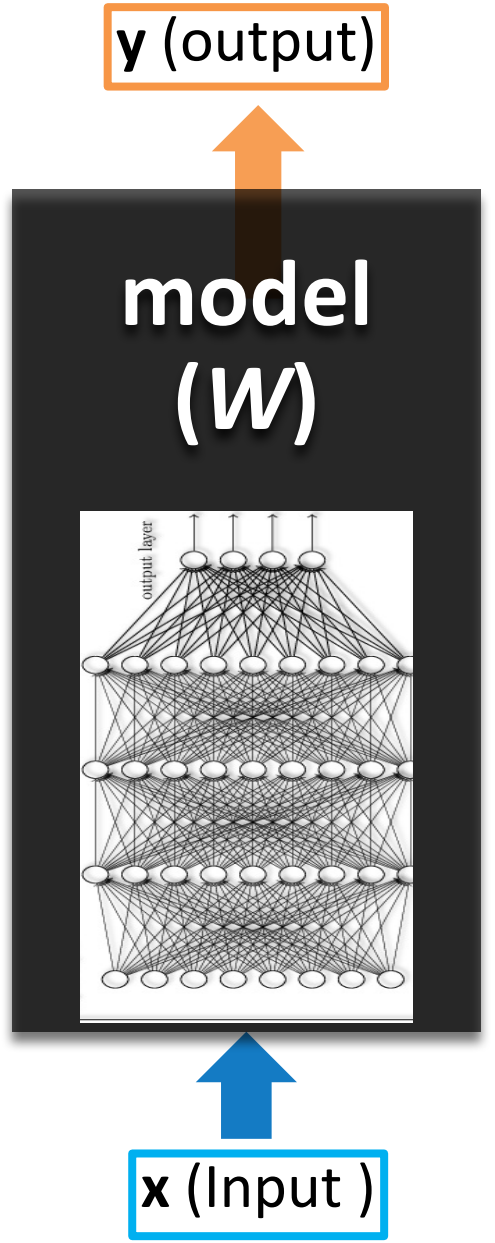
- **Training**
  - Maximizing log-likelihood of (input, true output) pair using stochastic gradient descent.
    - x_L, y_L : labeled input, output.
    - Ψ: score (log probability, energy)

$$\max_{W} \quad \Psi(\mathbf{x_L}, \mathbf{y_L}, W)$$

- **Inference**
  - Finding the output **y** with the highest model score (probability).
    - x: a test input.

$$\max_{\mathbf{y}} \quad \Psi(\mathbf{x}, \mathbf{y}, W)$$

  - Global constraint
    - *$O(V^T)$ (A* algorithm is often used)*



Agreement constraint to promote coherence

- Similar to constraint loss definition.
- However, rather than comparing to the definitive rule, now we compare each other.

$$- \sum_{vi \in V \setminus \{\text{meta}\}} \sum_{x \in U} g(\hat{y}^*, \hat{y}^{vi}) \log P(\hat{y}^* | x, \theta^{vi})$$

$V = \{\text{word,char,meta}\}$

- Predicted out for each model
- The best possible out given multi-view (ensemble, voting)
- A score function to measure “How similar the predictions are” between y^* *and* y^{vi}

(If the output is similar then learn more)

Co-meta overall objective (SSL)

- Similar to SSL work with SRL (syntactic constraint injection)
  - When we have Training set T and Unlabeled set U, our Joint-loss is as:

$$\text{J\_loss} = \sum_{(x_j, y_j) \in T} -\log P(y_j | x_j, \theta)$$

- Supervised

$$-\sum_{vi \in V} \sum_{x_k \in U} g(\hat{y}_k^*, \hat{y}_k^{vi}) \log P(\hat{y}_k^* | x_k, \theta^{vi})$$

- Unsupervised

where U, T might be $T \subseteq U$ when using T without labels. We call training with J_loss on the meta-LSTM structure as `Co-meta`.