A* CCG Parsing with a Supertag and Dependency Factored Model

Masashi Yoshikawa, Hiroshi Noji, Yuji Matsumoto
Nara Institute of Science and Technology, Japan

Today’s Talk: A* CCG Parsing

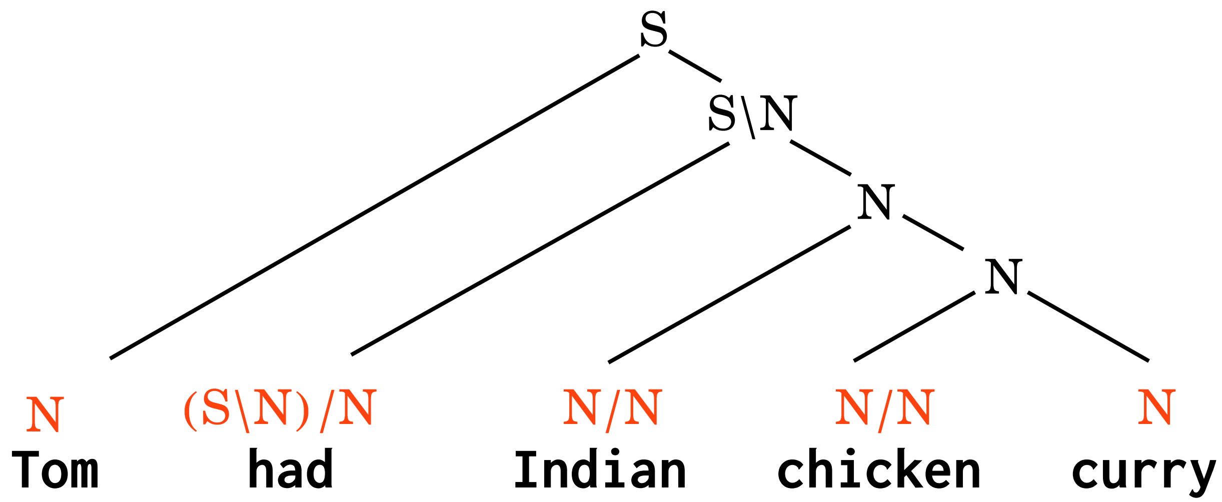
- Previous work: Supertag-factored Model (Lewis+, 2014, 2016)
  - Efficient & accurate
  - ISSUE: Use of a heuristic rule to resolve attachment ambiguities
- Our approach
  - Joint model of supertags and **syntactic dependencies**
  - LSTM-based simple dependency model allows efficient A*
- Result
  - **New state-of-the-art** on English & Japanese CCGbanks

Outline

- **Background: Supertag-factored Model**
- Proposed Method
- Experiments

Combinatory Categorial Grammar (CCG)

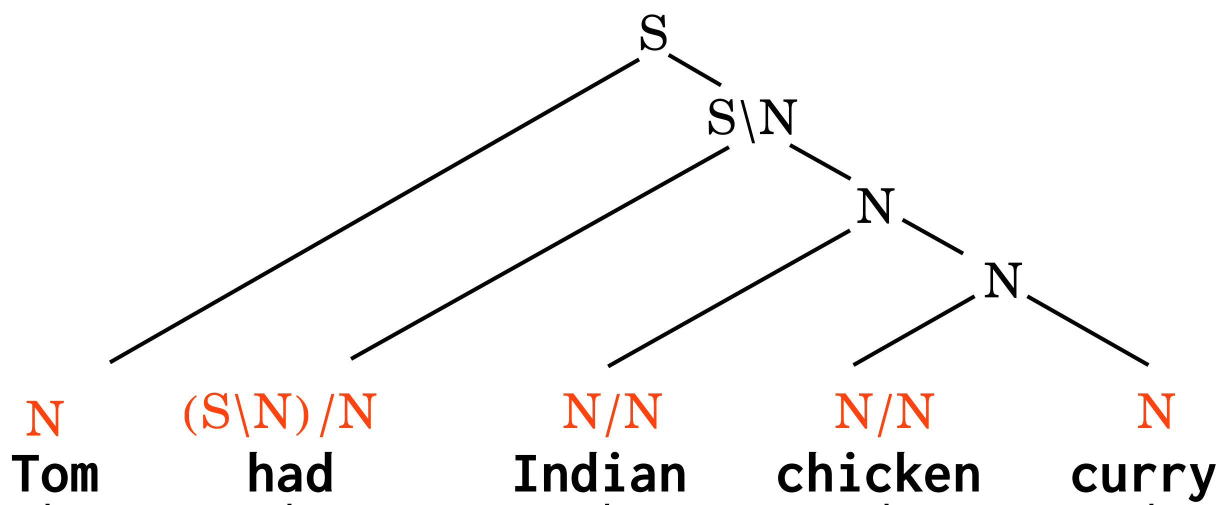
- Rich supertags, a small set of rules
- Supertagging is almost parsing (Bangalore and Joshi, 1999)
  - Given the supertags, the tree structure below is unique under normal form.



Supertag-factored Model (Lewis+, 2014, 2016)

- The probability of a tree is the product of **supertag** probabilities
- CCG Parsing:
  - Find the best supertag sequence that forms a tree

    → Efficient A* search is possible

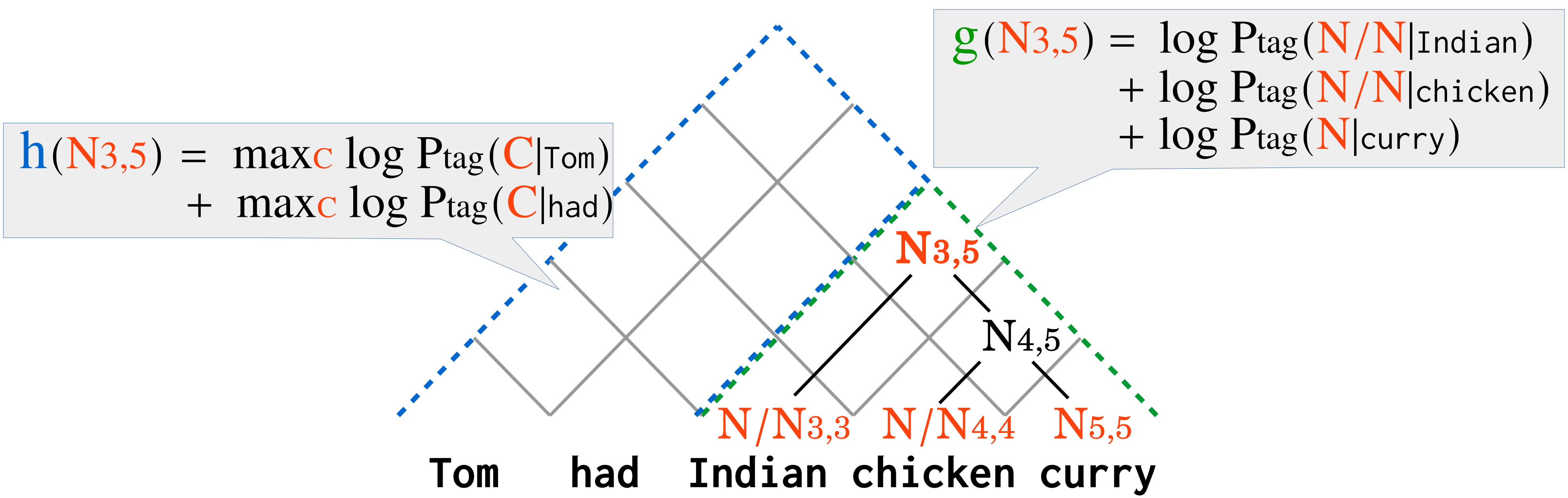


$$P(\mathbf{y}) = P_{\mathrm{tag}}(N)\ P_{\mathrm{tag}}(S\backslash N/N)\ P_{\mathrm{tag}}(N/N)\ P_{\mathrm{tag}}(N/N)\ P_{\mathrm{tag}}(N)$$

Efficient A* with Supertag-factored Model

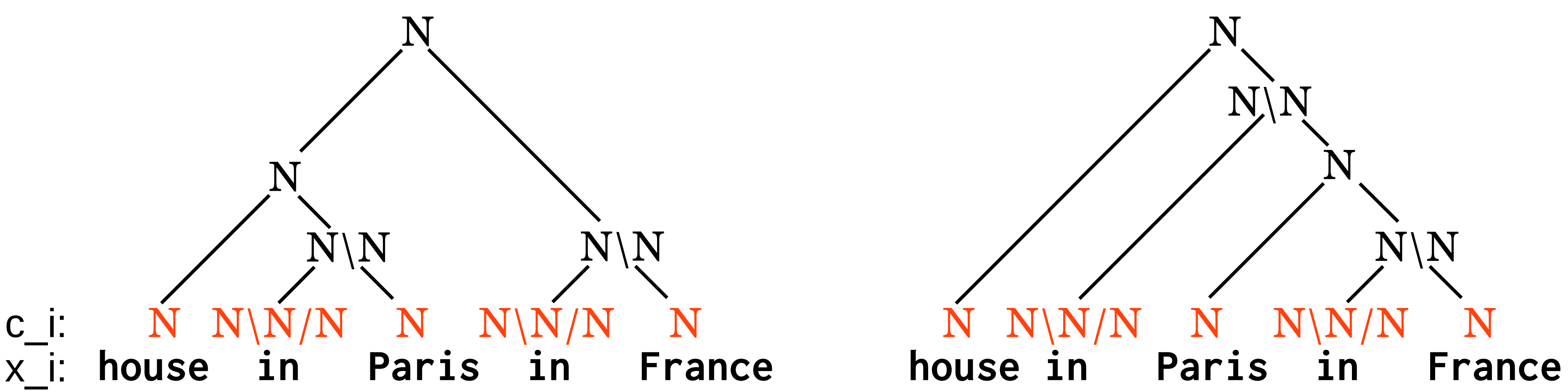
- A* parsing: populates chart with edges with the highest inside score (g) plus upper bound on outside score (h)

- Tight upper bound h can be easily obtained for this model
  - Just the sum of max scores for all outside words



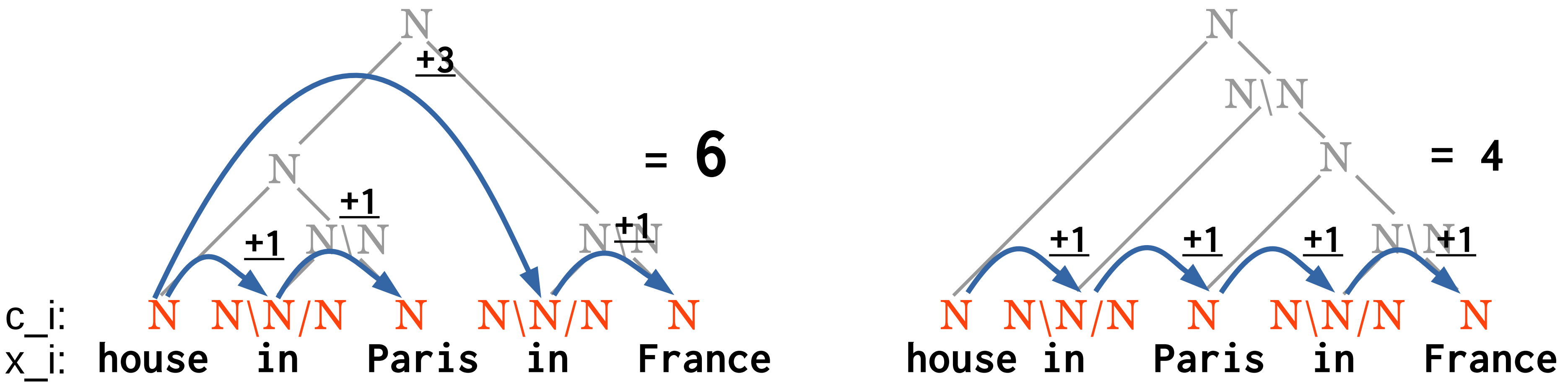
Limitation of Supertag-factored Model

- The same supertags can result in more than one tree.
  - → The model can't decide which one is better!



Limitation of Supertag-factored Model

- The same supertags can result in more than one tree.
  → The model can't decide which one is better!
- Dependency-based heuristics (Lewis+, 2014, 2016)
  – Choose one with longer dependencies
  – This does not always give the correct answer



Outline

- Background: Supertag-factored Model
- **Proposed Method**
- Experiments

Supertag & Dependency Factored Model

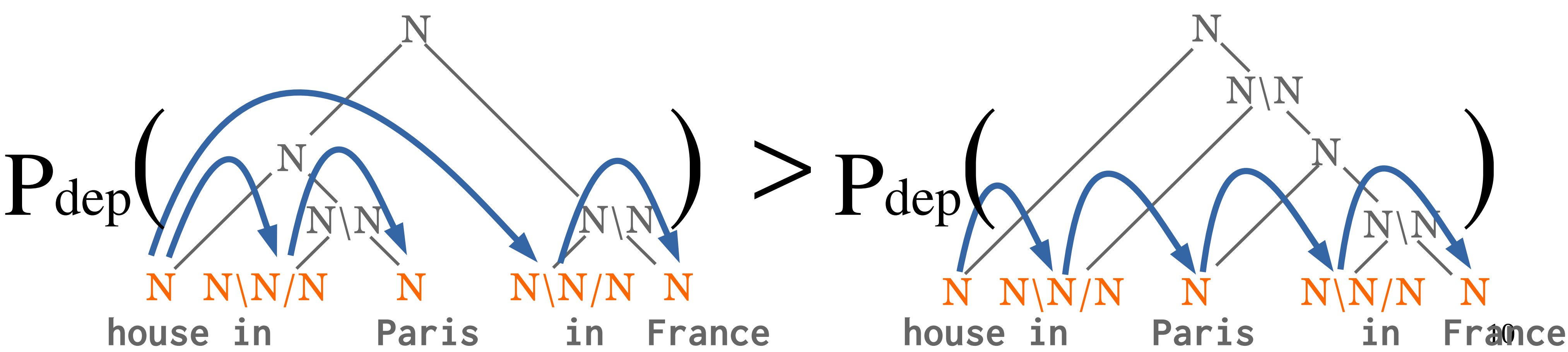
- The probability of a CCG tree is the product of the probabilities of the **supertags** and **dependency structure**

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{c_i \in \boldsymbol{y}} P_{tag}(c_i|x_i) \prod_{h_i \in \boldsymbol{y}} P_{dep}(h_i|x_i)$$

- What if there are two trees from the same supertags?

  → Choose one with **the higher scoring dep. structure**

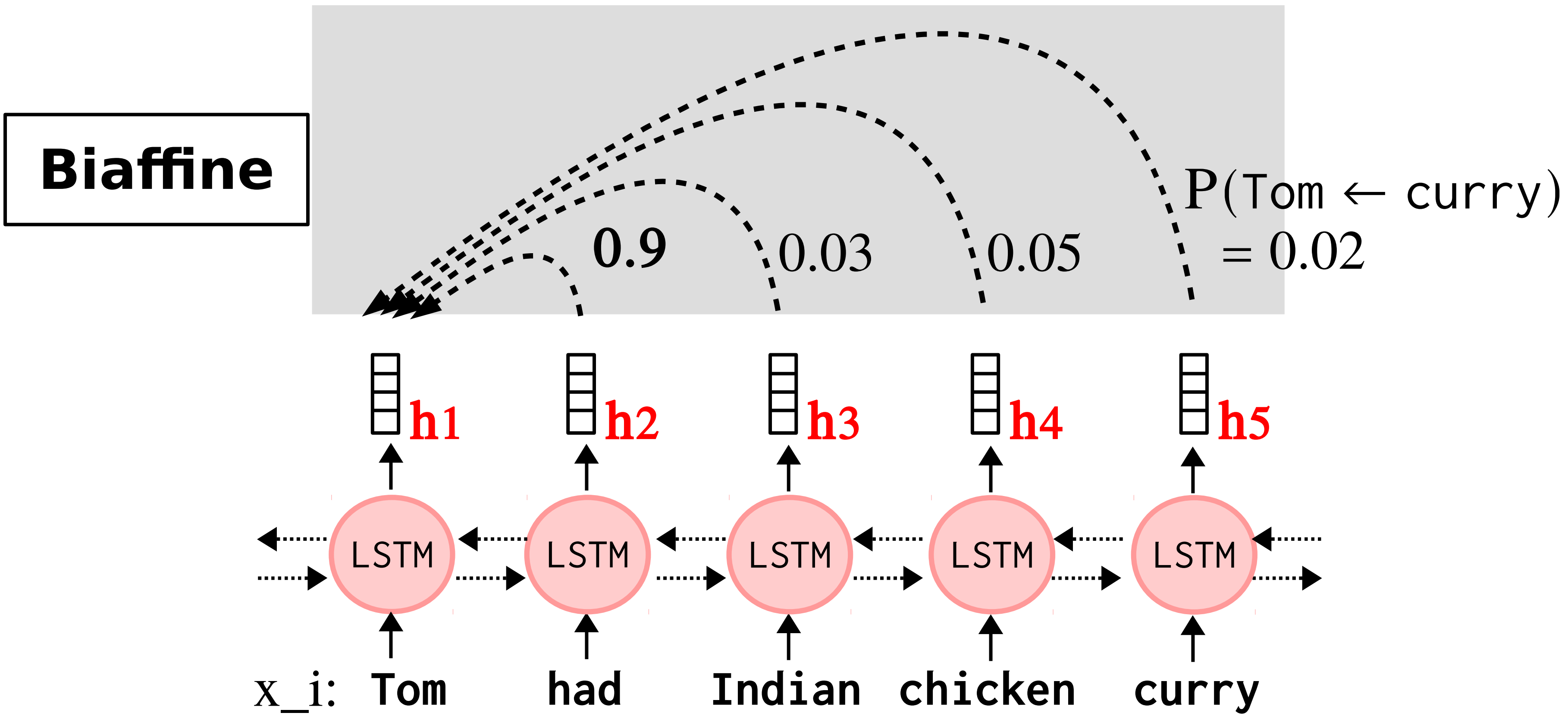
- **KEY**: a simpler dependency model still allows efficient A* decoding



LSTM-based Dependency Parsing

(Kiperwasser+, 2016, Dozat+, 2017)

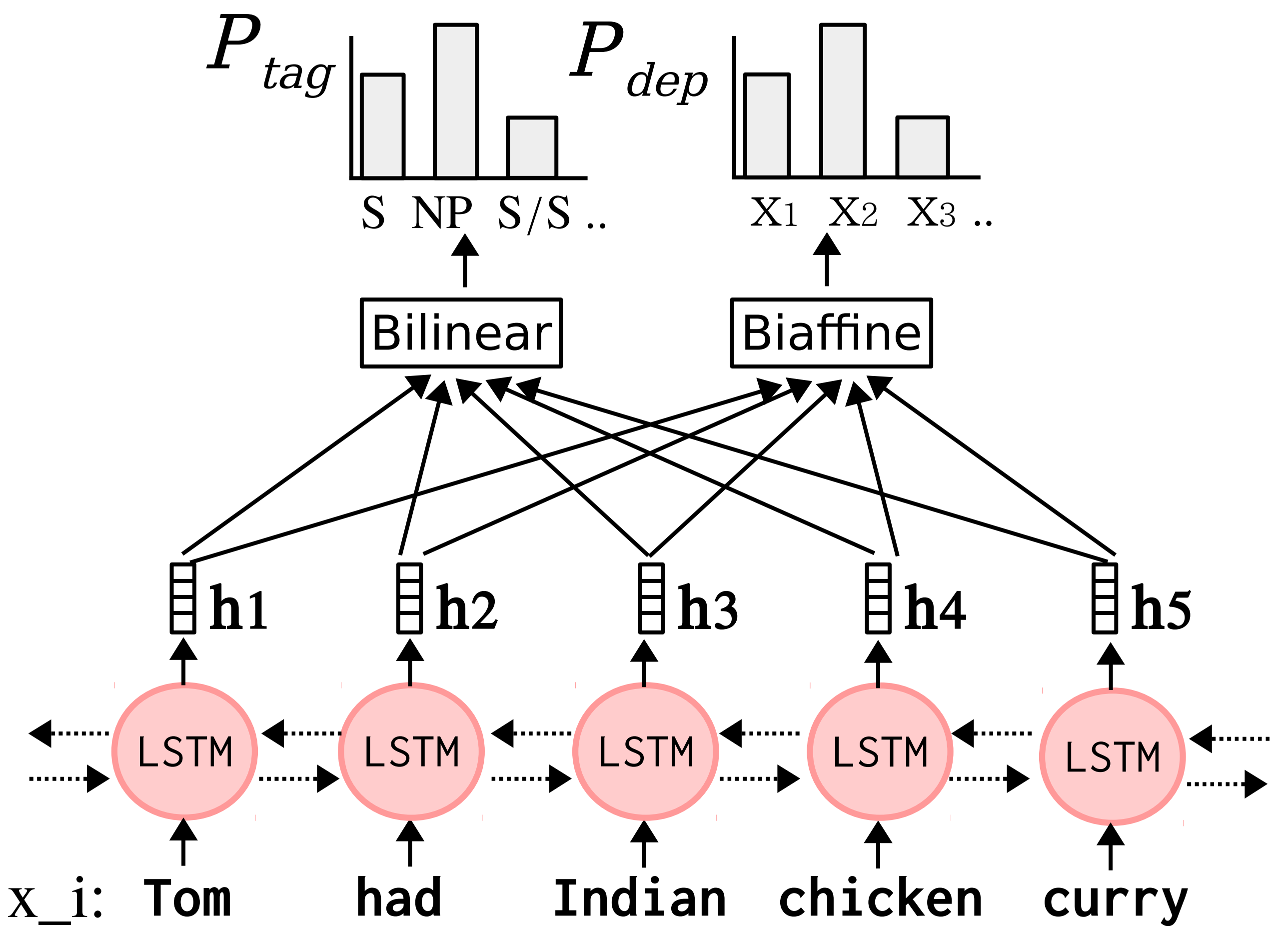
- <u>Independently</u> assigns a head to every word
- We use “Biaffine” layer (Dozat+, 2017)
  - $P(xj \rightarrow xi) \propto \text{Biaffine}(\mathbf{hi}, \mathbf{hj})$



Joint Supertag & Dependency Prediction

- Two different layers for supertags and dependencies
- Our model is the product of independent factors

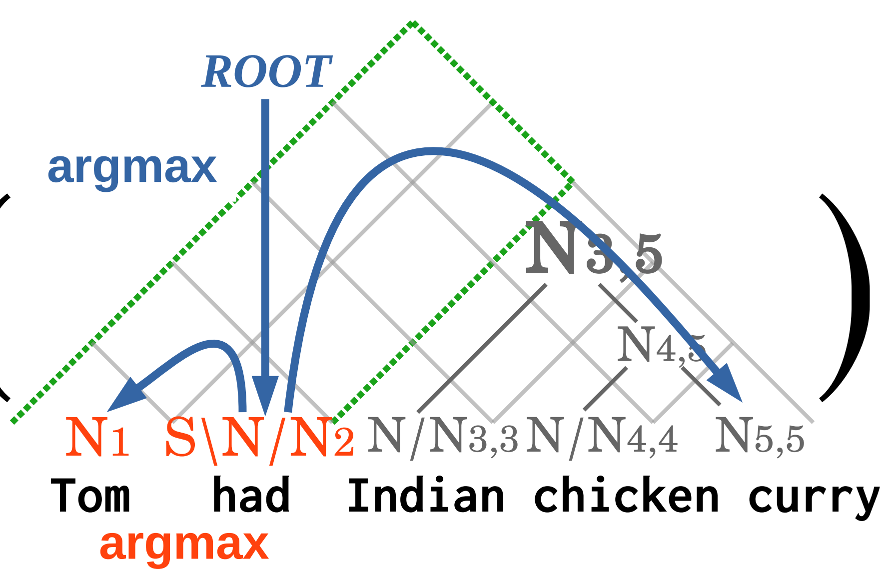
  → We can obtain all the scores before A* search!



A* Parsing with Our Model

- Scores in A* parsing can be extended naïvely.
- Upper bound on the outside score (h):
  - Sum of the max of supertag and dependency scores

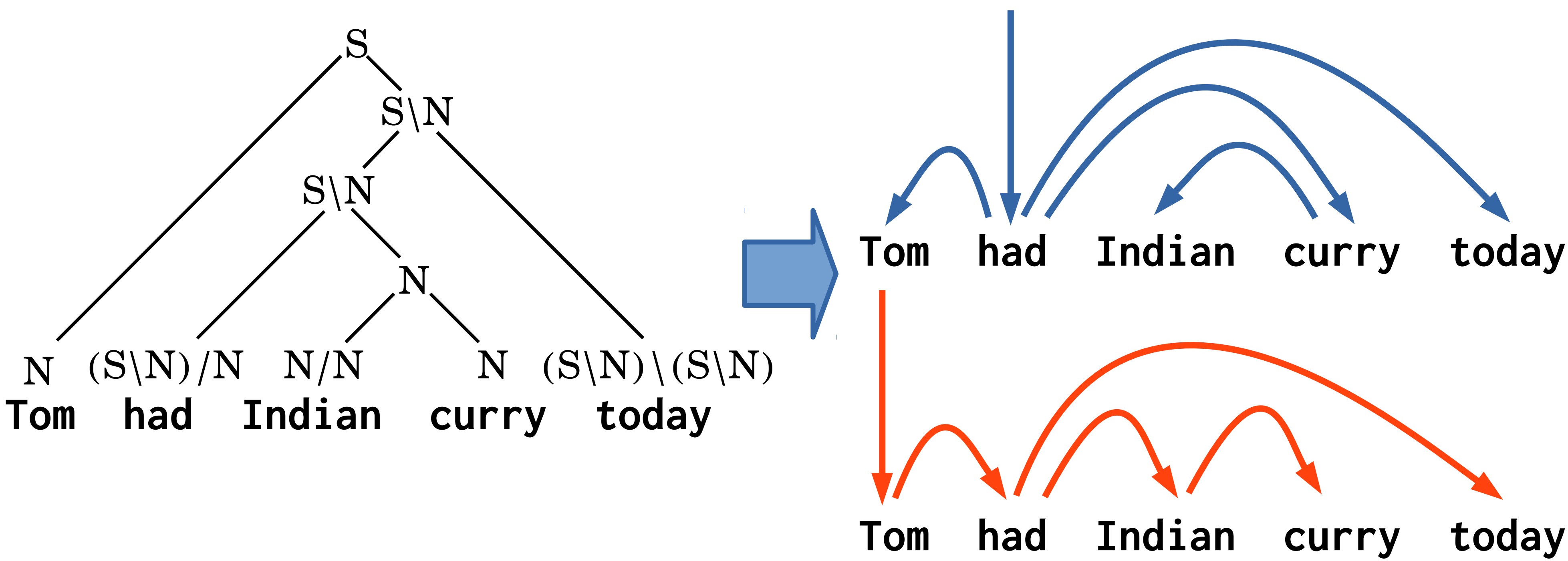
$h\Big($



$$\Big) = \max_{c} \log P_{tag}(c \mid \text{Tom}) + \max_{c} \log P_{tag}(c \mid \text{had}) + \max_{h} \log P_{dep}(h \rightarrow \text{Tom}) + \max_{h} \log P_{dep}(h \rightarrow \text{had}) + \max_{h} \log P_{dep}(h \rightarrow \text{curry})$$

CCG to Dependencies

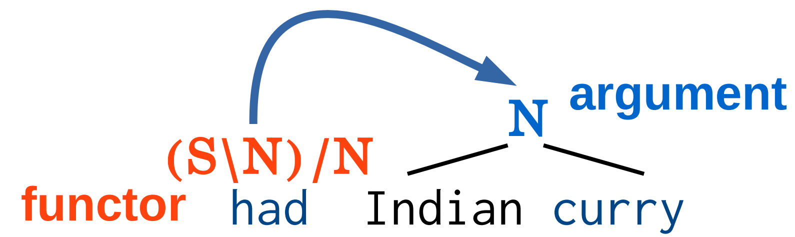
- We need to map a CCG tree to a dependency one
- We tried two approaches

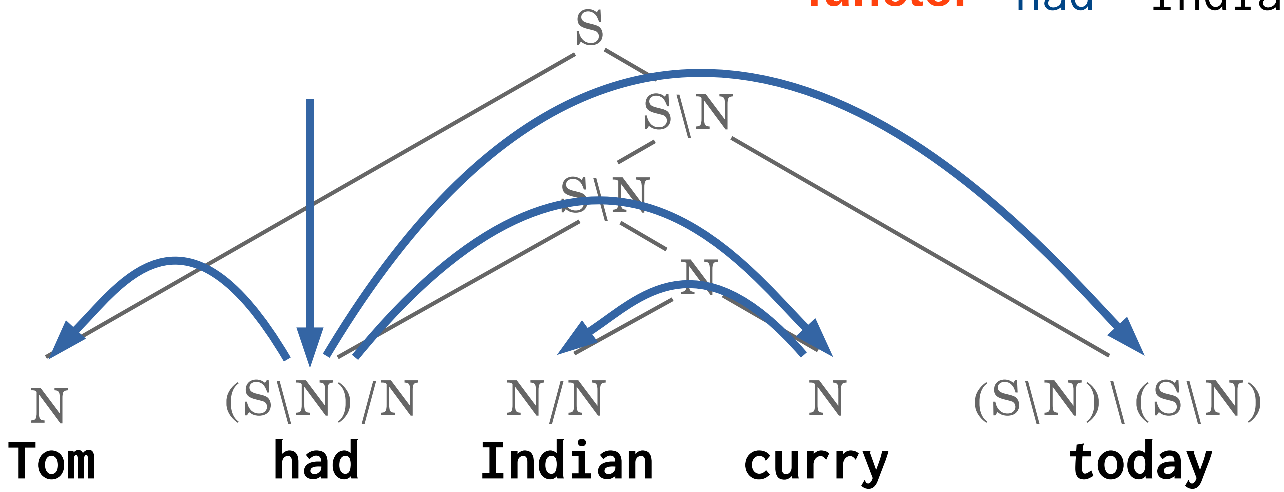


•Lewis et al.'s rule (LewisRule)

- Define the head direction for each combinatory rule
- Linguistically intuitive

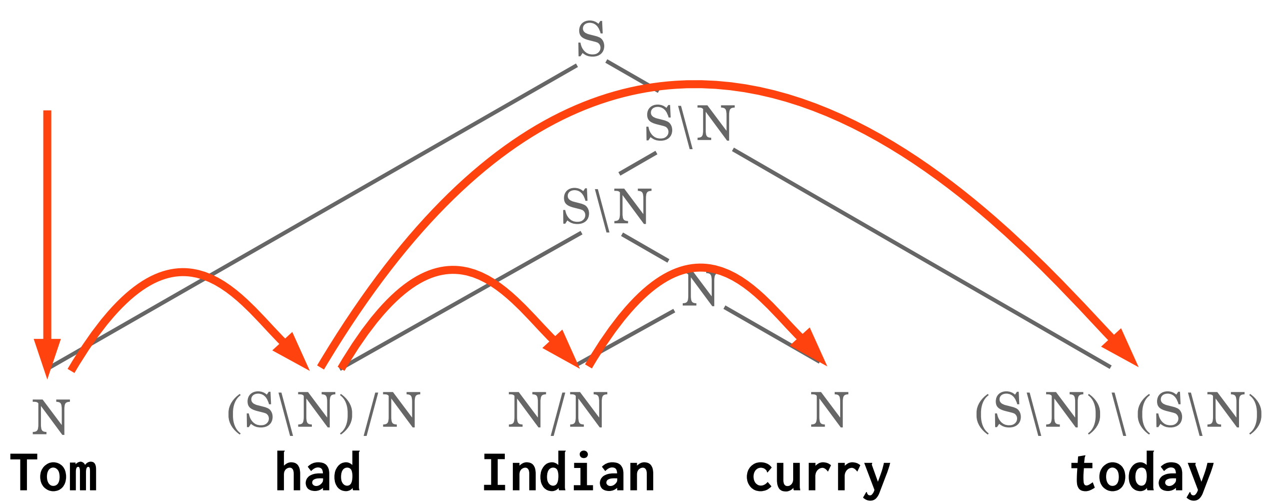
ex. forward application





Simpler “HeadFirst” Conversion Rule

- Always choose the left child as a head
  - Simple but linguistically odd
  - Easier to predict
    - 94.9 vs. 92.5 (UAS on dev, lstm-parser (Dyer+, 2015))



Semi-supervised Training (Tri-training)

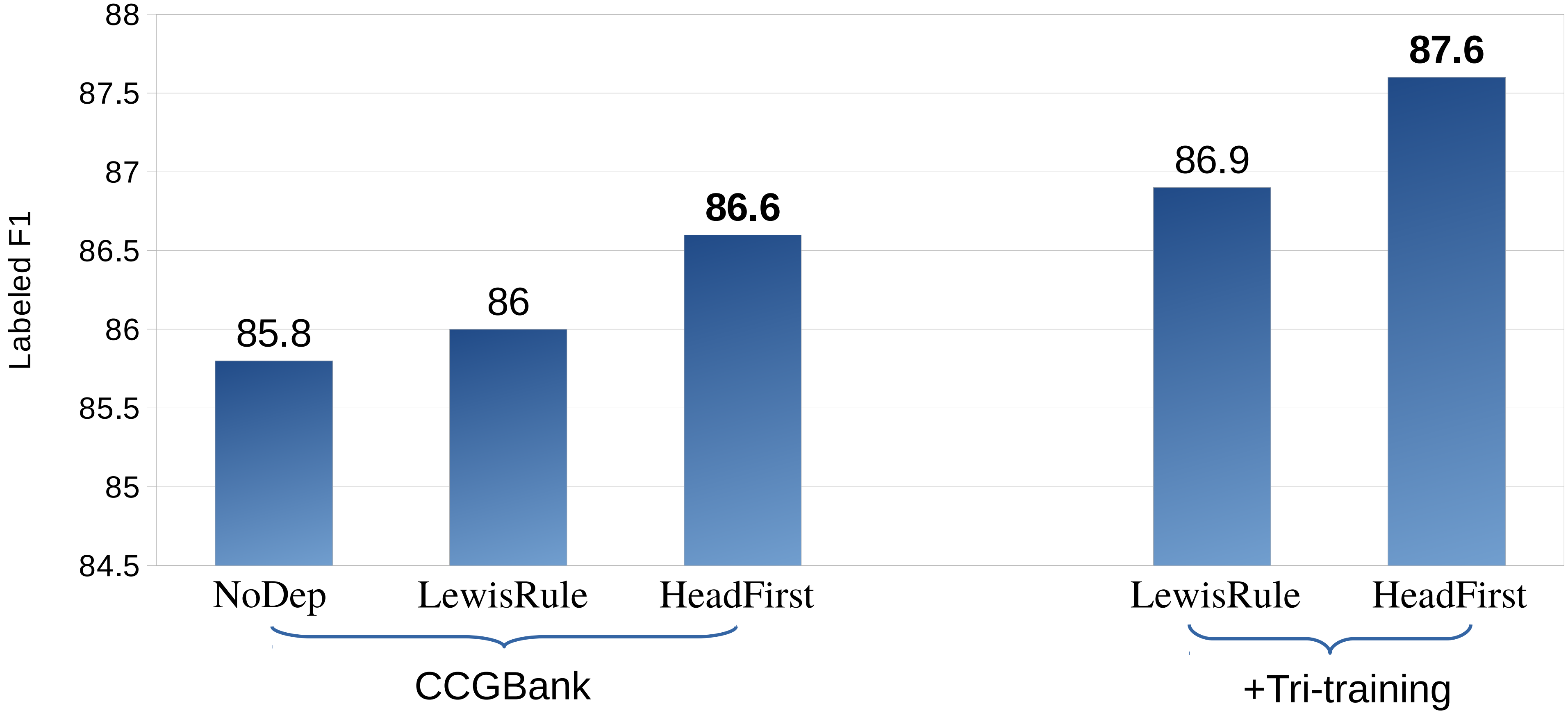
- Create a training data by taking the intersection of two existing parsers' predictions on an unlabeled corpus

- We assigned dependency structures on the supertag-labeled dataset prepared by (Lewis+, 2016)
  - More than 1.7 million sentences labeled with both LewisRule and HeadFirst dependencies

Outline

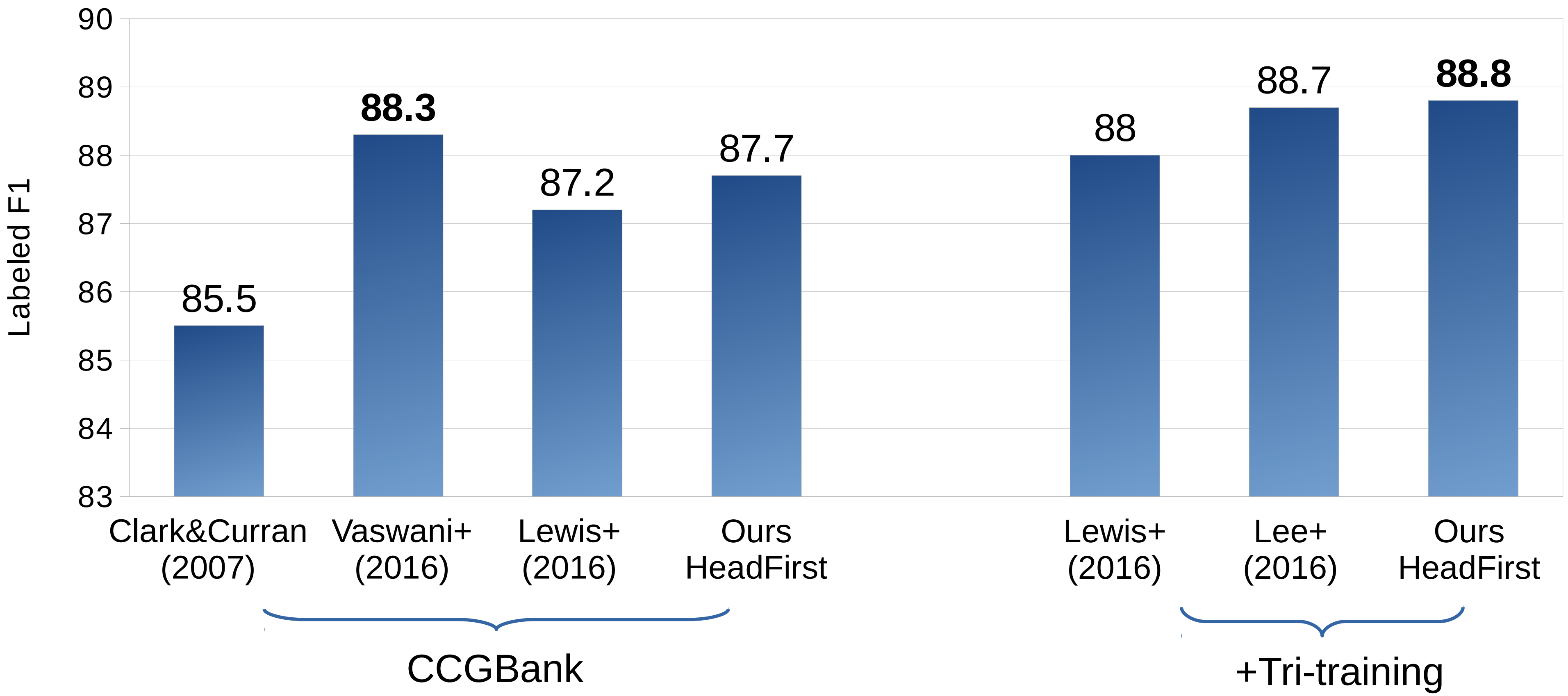
- Background: Supertag-factored Model
- Proposed Method
- **Experiments**

CCGBank Experiment (Dev)



- NoDep = discard dep. probs, use the heuristics in Lewis+, 2014
- Dependency probabilities contribute to performance gain.
- HeadFirst performs better.

CCGBank Experiment (Test)



- HeadFirst + Tri-training achieves the best result
- We also achieved the state-of-the-art on Japanese CCGBank!
  - 4.0 point up from previous work (Noji and Miyao, 2016)

Contributions

- Modeling syntactic dependencies behind a CCG tree
  - Local factorization allows efficient A* decoding
- NN architecture for supertags and dependencies
- Simpler HeadFirst conversion rules
- Semi-supervised Tri-training
  - State-of-the-art on English CCGBank

- **Codes and models (En, Ja) are available at**:
  - https://github.com/masashi-y/depccg