Automatic Evaluation of Linguistic Validity in Japanese CCG Treebanks

Asa Tomita[1], Hitomi Yanaka [2,3], Daisuke Bekki [1]
[1] Ochanomizu University [2] The University of Tokyo [3] RIKEN

tomita.asa@is.ocha.ac.jp
https://morning85.github.io/

TLT, Syntax Fest
Ljubljana, 29 Aug (Fri.)

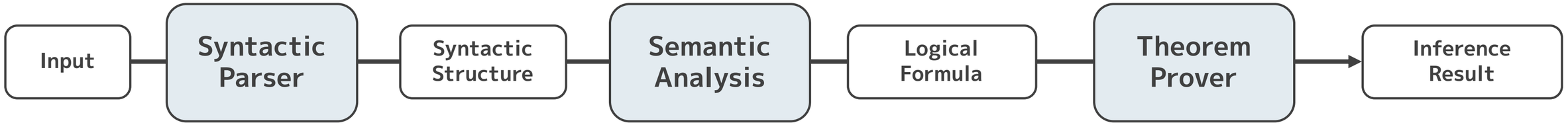
Natural Language Inference

Natural Language Inference (NLI) is a core task in NLP, requiring systems to determine inferential relationship (e.g., entailment, contradiction, or neutral) between premises and a hypothesis

Approach 1 : Using Large Language Models (LLMs)
- Achieve high accuracy on benchmark datasets
- But the decision process is often not transparent

Approach 2 : Using Compositional Semantics-Based Models
- Enables explicit tracing of the reasoning process
- Allows identification of where inference fails when an error occurs

NLI based on compositional semantics



- Inference accuracy is strongly influenced by the output of syntactic and semantic analysis, which serve as preprocessing for inference.
  - Syntactic and semantic analysis that contain errors can lead to incorrect inference result.

Parsing accuracy and validity

Accuracy	How well a parser trained on a dataset can reproduce the analyses in the evaluation dataset.
Validity	How well the parser's output conforms to the principles of linguistic theory.

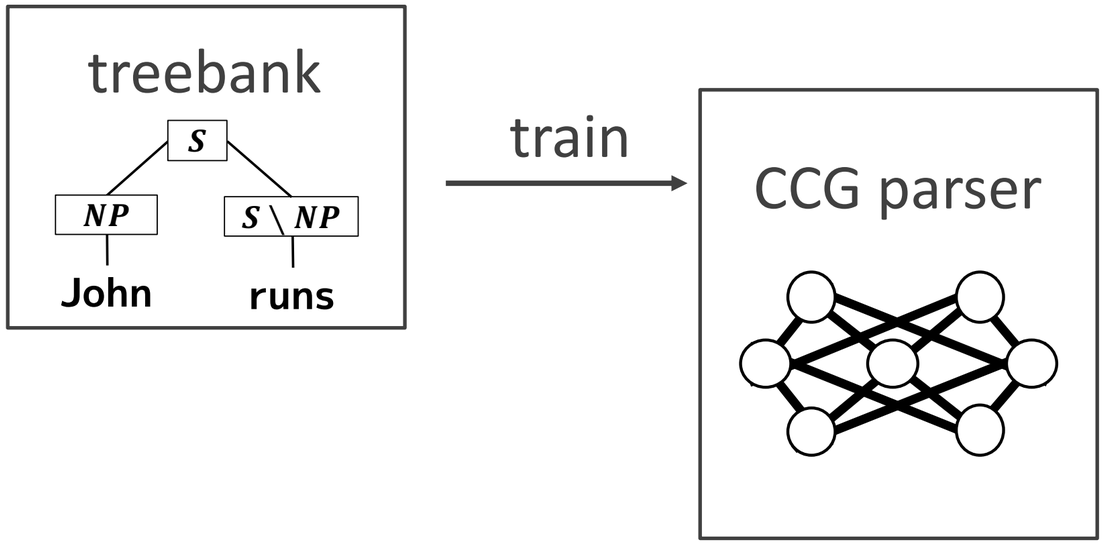
Parsing accuracy and validity

Accuracy

How well a parser trained on a dataset can reproduce the analyses in the evaluation dataset.

Validity

How well the parser's output conforms to the principles of linguistic theory.



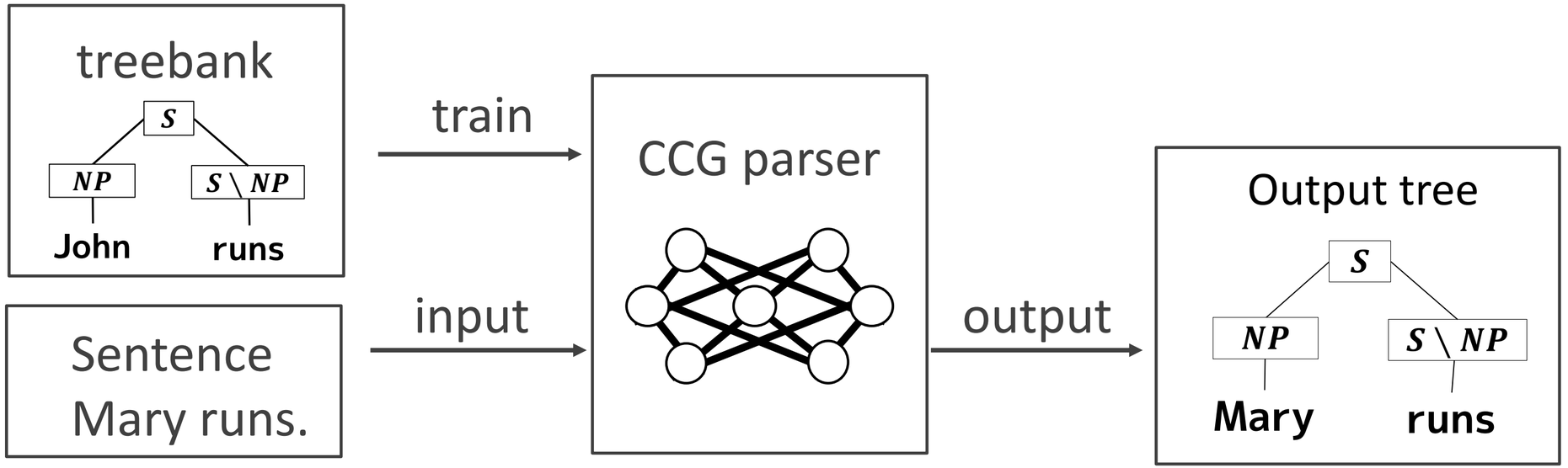
Parsing accuracy and validity

Accuracy

How well a parser trained on a dataset can reproduce the analyses in the evaluation dataset.

Validity

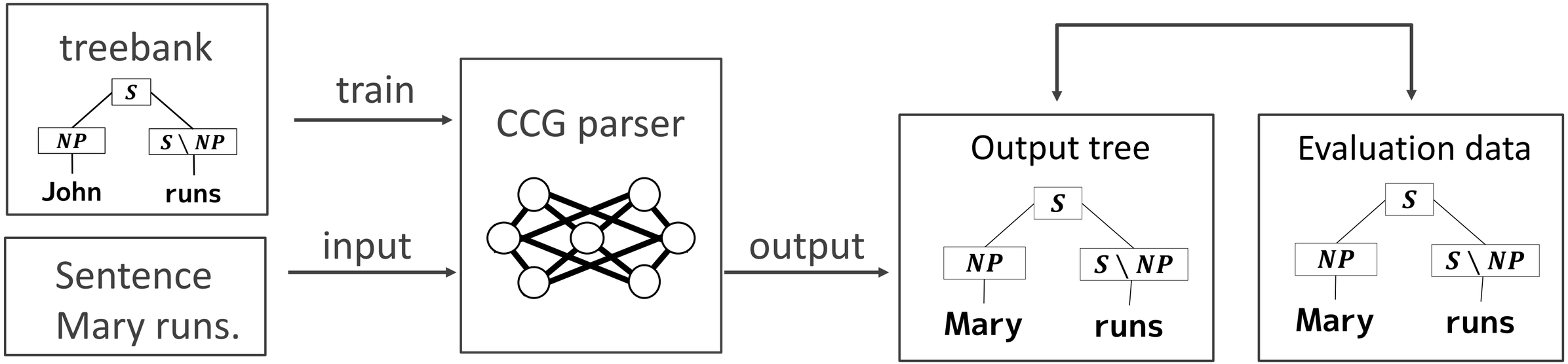
How well the parser's output conforms to the principles of linguistic theory.



Parsing accuracy and validity

Accuracy How well a parser trained on a dataset can reproduce the analyses in the evaluation dataset.

Validity How well the parser's output conforms to the principles of linguistic theory.



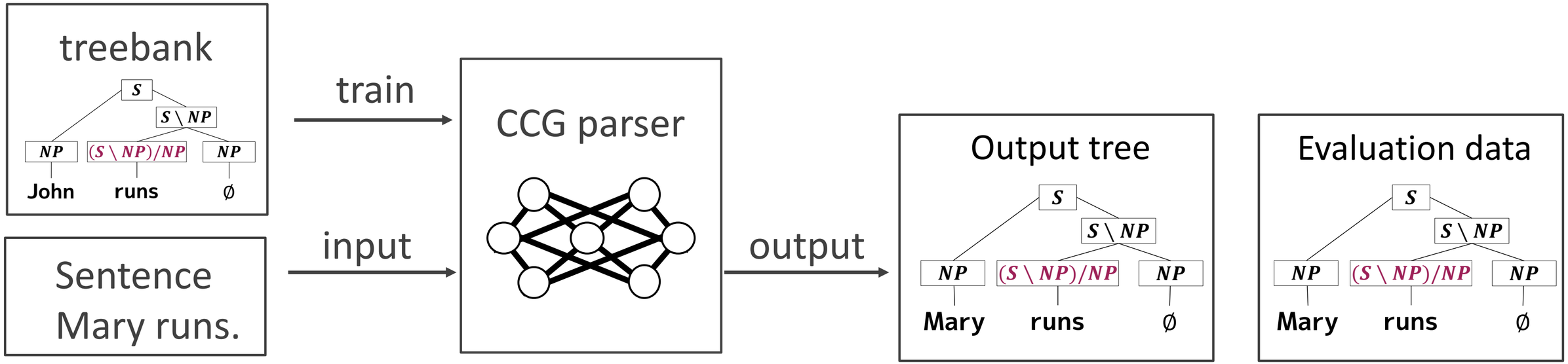
Parsing accuracy and validity

Accuracy	How well a parser trained on a dataset can reproduce the analyses in the evaluation dataset.
Validity	How well the parser’s output conforms to the principles of linguistic theory.

Parsing accuracy and validity

Accuracy How well a parser trained on a dataset can reproduce the analyses in the evaluation dataset.

Validity How well the parser's output conforms to the principles of linguistic theory.



Parsing accuracy and validity

Accuracy	How well a parser trained on a dataset can reproduce the analyses in the evaluation dataset.
Validity	How well the parser's output conforms to the principles of linguistic theory.

The development of a valid parser requires …

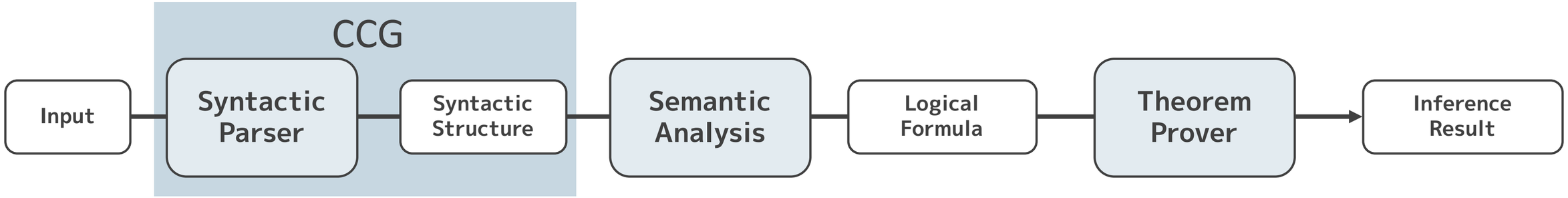
High Accuracy Parser × Linguistically Valid Treebank

Research Question

Question 1.
How can you construct linguistically valid treebank?

Question 2.
How can you automatically evaluate linguistic validity of the treebank?

Combinatory Categorial Grammar（CCG; Steedman 1996）



- CCG is a lexicalized grammar that describes syntactic structures using lexicon and combinatory rules

lexical items

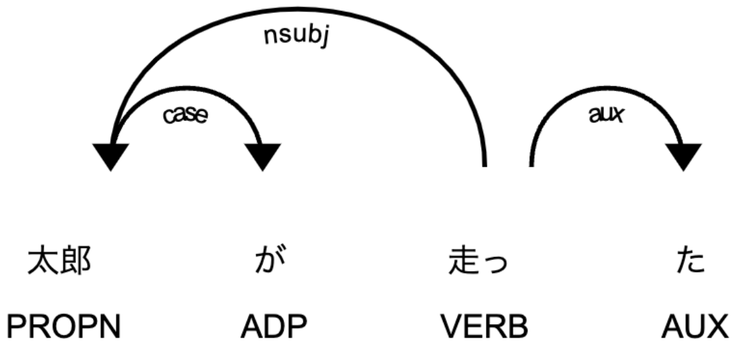
$$\begin{aligned} \text{Keats} &\vdash NP &&: keats \\ \text{eats} &\vdash (S \setminus NP)/NP &&: \lambda xy.eat(y,x) \\ \text{apples} &\vdash NP &&: apples \end{aligned}$$

CCG Syntactic Structure

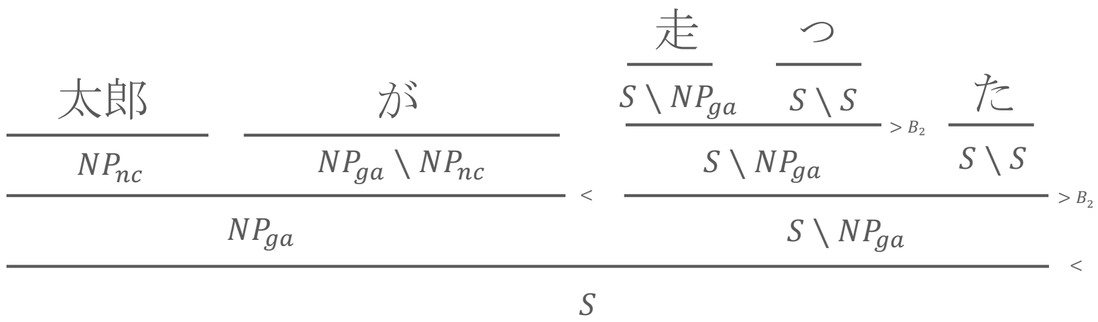
$$\dfrac{\text{Keats} \quad \dfrac{\text{eats} \quad \text{apples}}{(S \setminus NP) / NP : \lambda xy.eat(y,x) \quad NP : apples}\,<}{\dfrac{NP : keats \quad S \setminus NP : \lambda y.eat(y, apples)}{S : eat(keats, apples)}}$$

Japanese CCGbank (Uematsu et al. 2013, 2015)

- Japanese CCGbank is a representative CCG treebank for Japanese
- It was constructed via automatic conversion from a corpus of dependency structures
- It is widely used as training and evaluation data for Japanese CCG parsers



Dependency structure



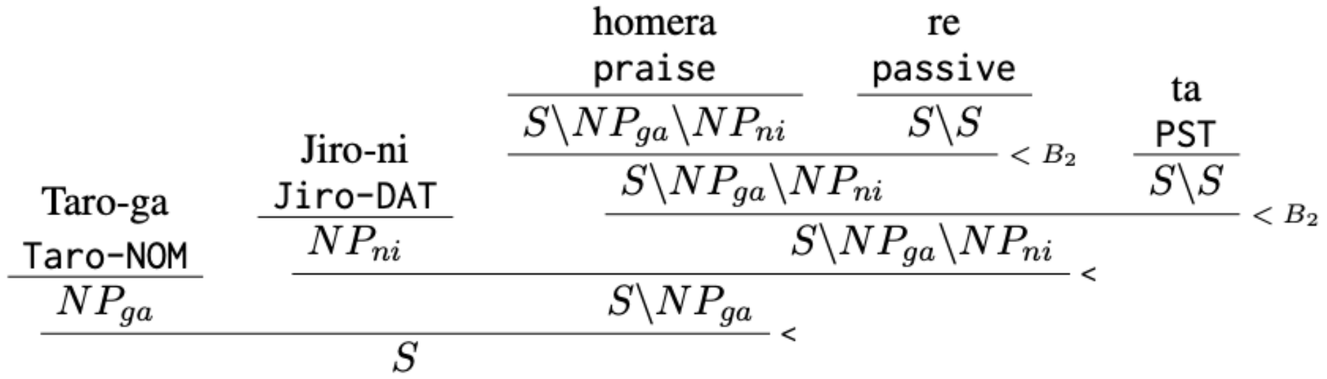
CCG syntactic structure

Japanese CCGbank (Uematsu et al. 2013, 2015)

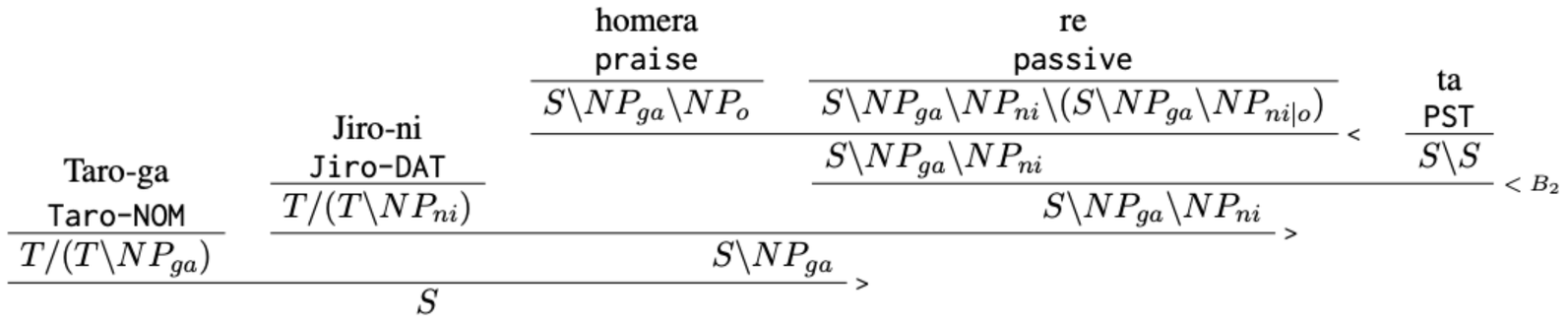
- Japanese CCGbank is a representative CCG treebank for Japanese
- It was constructed via automatic conversion from a corpus of dependency structures
- It is widely used as training and evaluation data for Japanese CCG parsers
- contains errors in the analysis of sentences involving case alternation such as passive and causative constructions (Bekki & Yanaka, 2023)

Japanese CCGbank (Uematsu et al. 2013, 2015)

Analysis of passive constructions in the Japanese CCGbank



Analysis of passive constructions in Japanese CCG (Bekki 2010)



Method to construct treebank

1. Automatic conversion from another corpus (ex: Japanese CCGbank

- ✓ It is possible to automatically construct large-scale corpora.
- ✗ Linguistic validity cannot be guaranteed.

Method to construct treebank

1. Automatic conversion from another corpus (ex: Japanese CCGbank
   - ✓ It is possible to automatically construct large-scale corpora.
   - ✗ Linguistic validity cannot be guaranteed.
2. Manual Annotation (ex: ABCTreebank (Kubota et al. 2017)
   - ✓ Manual annotation by linguists ensures high linguistic validity.
   - ✗ It requires specialized linguistic knowledge, making it highly costly.

Method to construct treebank

1. Automatic conversion from another corpus (ex: Japanese CCGbank

   ✓ It is possible to automatically construct large-scale corpora.

   ✗ Linguistic validity cannot be guaranteed.

2. Manual Annotation (ex: ABCTreebank (Kubota et al. 2017)

   ✓ Manual annotation by linguists ensures high linguistic validity.

   ✗ It requires specialized linguistic knowledge, making it highly costly.

3. Use parser for Annotation (ex: Keyaki Treebank (Butler et al. 2018)

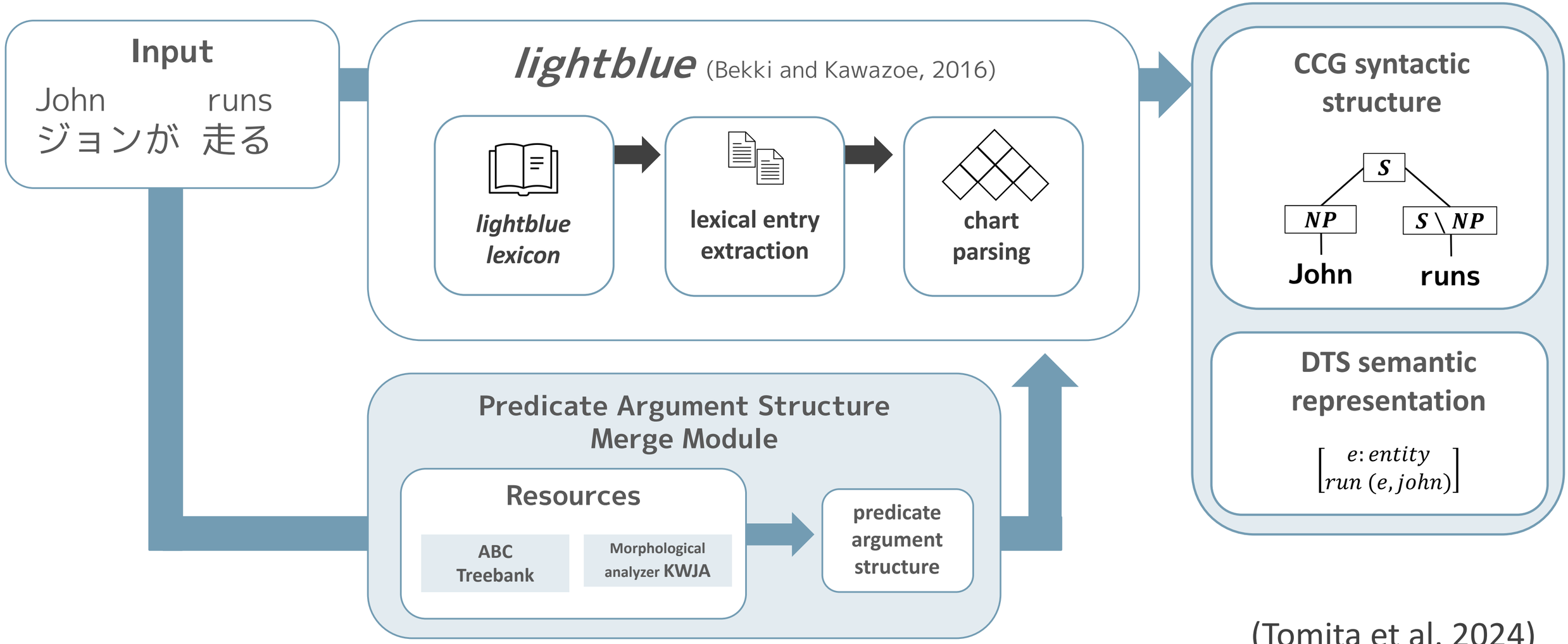
   Automatic annotation is done mechanically, without expert intervention

Bootstrapping Problem

3. Use parser for Annotation (ex: Keyaki Treebank (Butler et al. 2018)
Automatic annotation is done mechanically, without expert intervention

Use a non-neural parser (lexicon-based parser) that does not require a treebank to construct a treebank

Constructing linguistically valid Japanese CCG treebank



(Tomita et al. 2024)

Dependent Type Semantics (DTS; Bekki 2014, Bekki and Mineshima 2017)

DTS is a type-theoretical semantic framework based on Dependent Type Theory (DTT; Martin-Löf, 1984).

- Allows types (= propositions) to depend on terms (= proofs)
- Handles anaphora and presupposition via **proof search**
- **Type checking** ensures well-formedness of semantic representation

$$\left[\begin{array}{l} u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \\ \mathbf{walk}(\pi_1 u) \end{array}\right]$$

Semantic representation of “A man walks” in DTS

Linguistically valid Japanese CCG Treebank

lightblue CCGbank contains **13653** sentences categorized into 14 genres

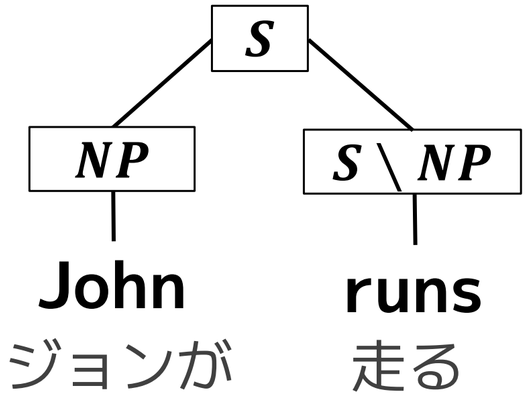
Genre	Sentences	Len-50+ sentences	Reforged trees
aozora	1773	590	1183
bible	1652	220	1430
book_expert	50	4	41
dict_lexicon	2640	4	2636
diet_kaigiroku	486	112	374
fiction	921	44	877
law	337	128	209
misc	335	59	276
news	443	103	340
non-fiction	223	87	126
spoken	570	11	559
ted_talk	605	54	551
text-book	4880	10	4870
wikipedia	222	51	171
Total	15137	1482	13653

Conventional Evaluation Metrics

1. Number of Lexical Entries (lexical extraction)
2. Coverage Rate
3. Parsing Accuracy

Conventional Evaluation Metrics

1. Number of Lexical Entries (lexical extraction)
2. Coverage Rate
3. Parsing Accuracy



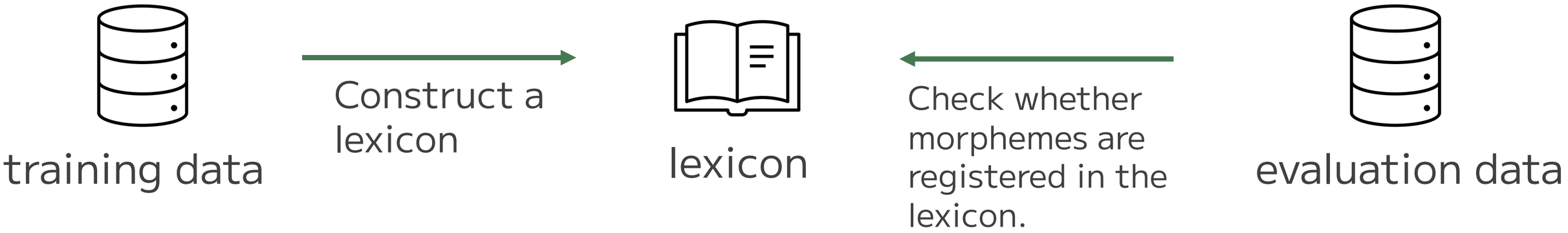
lexicon

$$\left[\begin{pmatrix} John, \\ noun, \\ -, \\ NP \end{pmatrix}, \begin{pmatrix} run, \\ verb, \\ terminal\ form, \\ S \setminus NP \end{pmatrix}\right]$$

Conventional Evaluation Metrics

1. Number of Lexical Entries (lexical extraction)
2. Coverage Rate
3. Parsing Accuracy

Lexical coverage : the proportion of categories that are registered in the lexicon for morphemes appearing in unseen sentences.



Conventional Evaluation Metrics

1. Number of Lexical Entries (lexical extraction)
2. Coverage Rate
3. Parsing Accuracy

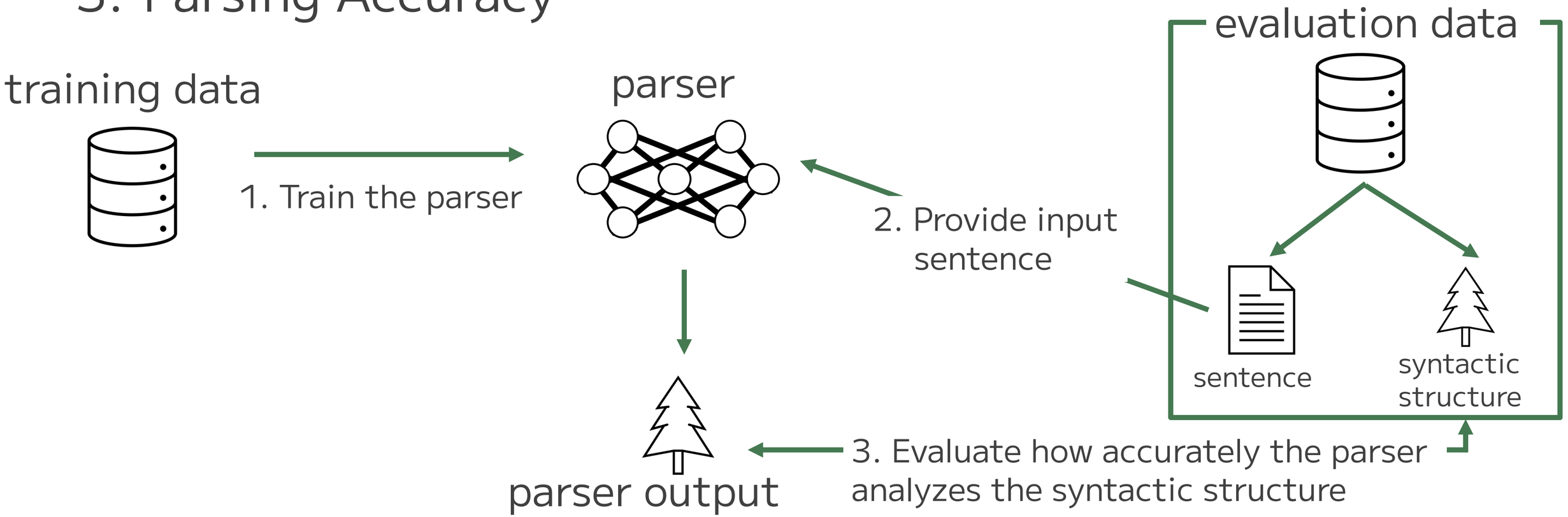
Limitations :

These metrics merely indicate the coverage of the data, and how extensively the lexicon can assign some category to encountered words.

→ high coverage rate does not ensure the quality or validity of the data

Conventional Evaluation Metrics

1. Number of Lexical Entries (lexical extraction)
2. Coverage Rate
3. Parsing Accuracy



Conventional Evaluation Metrics

1. Number of Lexical Entries (lexical extraction)
2. Coverage Rate
3. Parsing Accuracy

Limitations :

A parser can achieve high accuracy even when trained on erroneous treebank data.

→ high accuracy alone cannot be taken as evidence of a linguistically valid treebank

Conventional Evaluation Metrics

1. Number of Lexical Entries (lexical extraction)
2. Coverage Rate
3. Parsing Accuracy
4. Manual Evaluation

Limitations :

Evaluating CCG syntactic structures requires advanced knowledge of computational linguistics

→ manual evaluation is costly and impractical for large-scale treebank validation

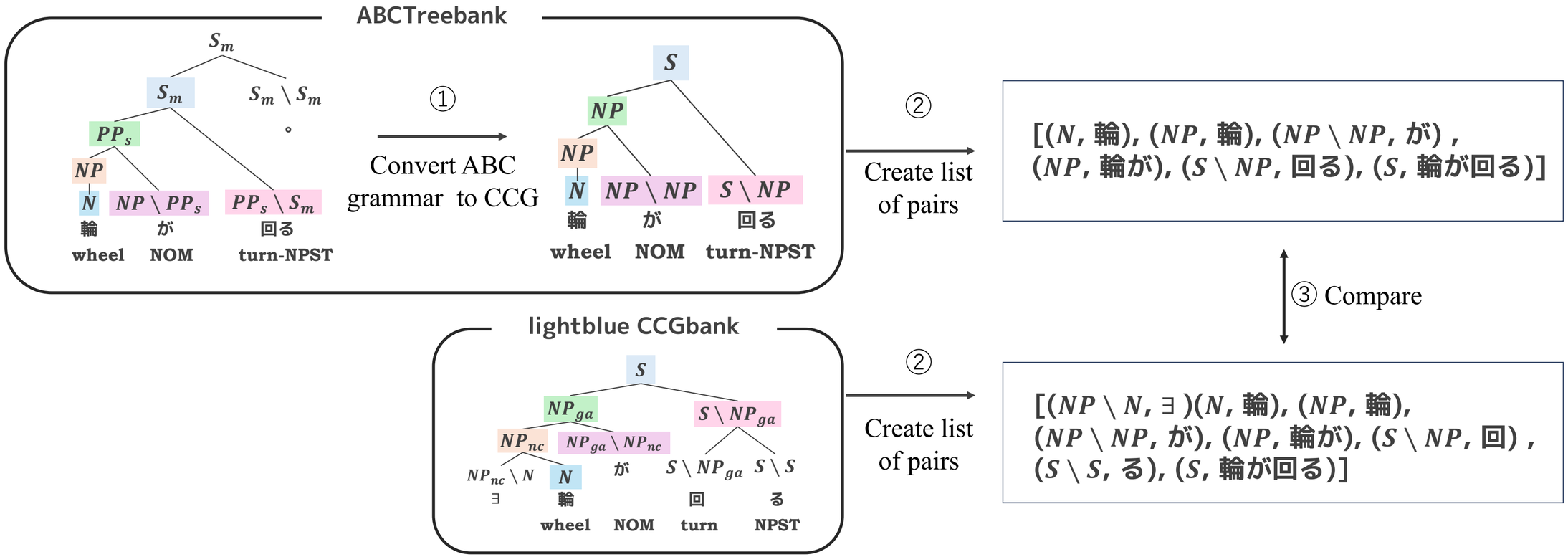
Our Solution

Combine two evaluation metrics to evaluate linguistic validity of the treebank

1. Syntax-based evaluation
2. Semantics-based evaluation

Syntax-Based Evaluation

We evaluate syntactic structures by measuring an alignment with another categorial grammar-based treebank, ABCTreebank (Kubota 2019)



Limitations of Syntax-Based Evaluation

- ✓ It can compare empty categories in CCG with unary rules in ABCTreebank
- ✓ It can accommodate differences in predicate analysis
- − It assumes that ABCTreebank is entirely correct, which may not necessarily be the case
- − It cannot evaluate syntactic features which are not annotated in ABCTreebank

Semantic-Based Evaluation

All syntactic structures in lightblue CCGbank are assigned DTS semantic representations

→ we can evaluate the well-formedness of the DTS semantic representation using type-theoretic verification "**type checking**"

$$\vdash \begin{bmatrix} u_0 : \begin{bmatrix} x_o : entity \\ \begin{bmatrix} x_1 : entity \\ wheel\ (x_1 . x_0) \end{bmatrix} \end{bmatrix} \\ \begin{bmatrix} x_2 : entity \\ turn(x_2, \pi_1(u_0)) \end{bmatrix} \end{bmatrix} : type?$$

$$\frac{\vdots \qquad \vdots}{\vdash \begin{bmatrix} u_0 : \begin{bmatrix} x_o : entity \\ \begin{bmatrix} x_1 : entity \\ 輪\ /\ わ\ (x_1 . x_0) \end{bmatrix} \end{bmatrix} \\ \begin{bmatrix} x_2 : entity \\ 回る\ /\ ガ(x_2, \pi_1(u_0)) \end{bmatrix} \end{bmatrix} : type} (\Sigma F)$$

$$\vdash \begin{bmatrix} x_0 : entity \\ \begin{bmatrix} x_1 : entity \\ 輪/\ わ\ (x_1, x_0) \end{bmatrix} \end{bmatrix} : type \qquad s_0 : \begin{bmatrix} x_0 : entity \\ \begin{bmatrix} x_1 : entity \\ 輪/\ わ\ (x_1, x_0) \end{bmatrix} \end{bmatrix} \vdash \begin{bmatrix} x_2 : entity \\ 回る/ガ(x_2, \pi_1(s_0)) \end{bmatrix} : type$$

- Type check is considered successful if the representation can be proven to have the type **type**

Semantic-Based Evaluation

- Type checking fails when the semantic representation is ill-formed.
- Under the combination of CCG and DTS all semantic representations are theoretically guaranteed to be well-typed (Bekki, forthcoming).
- Therefore, a type checking failure implies parsing errors

→ Semantic representations that are ill-typed are not linguistically valid in this system.

Limitations of Semantic-Based Evaluation

- ✓ It evaluates syntactic structures at the semantic level based on type theory
- − Passing type checking does not necessarily imply linguistic validity of the associated syntactic structure

Syntactic scores and type-theoretic verification serve complementary functions, and their combined use is essential for a comprehensive assessment of treebank quality.

Evaluation Setup

760 sentences sampled from lightblue CCGbank

Evaluated using three metrics

1. **Syntactic Structure Score Average**
   - Measures the percentage of matching (surface form, syntactic category) pairs between lightblue and ABCTreebank and computed averages by genre

2. **Type Checking Passage Rate**
   - Proportion of sentences with well-typed DTS semantic representations

3. **Overall Evaluation**
   - Sentences that scored ≥ 50 in syntax and passed type checking

Evaluation Result

Genre	Number of Data	Average Score	Type Checking Pass Rate	Overall Score
Aozora Bunko	125	42.4	63.2	20.89
Bible	40	49.1	57.5	21.57
Books	10	49.8	60.0	33.33
Dictionary	100	55.59	57.0	28.57
Proceedings	35	41.8	77.1	23.91
Fiction	30	51.1	66.7	31.82
Law	10	33.4	80.0	28.57
Other	50	50.2	64.0	26.47
News	50	40.4	78.0	21.88
Non-fiction	10	53.4	100.0	33.33
Spoken Language	50	36.98	88.0	25.37
TED Talks	25	41.68	64.0	21.88
Textbooks	200	49.59	60.0	27.54
Wikipedia	25	45.88	88.0	32.43
Total	760	46.0	66.2	26.00

Key Findings

No clear correlation between syntactic score and type checking rate

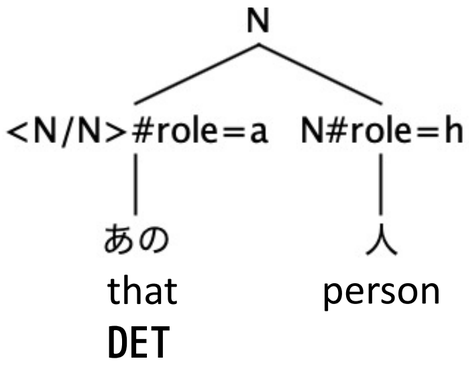
→ The two metrics capture different aspects of linguistic validity

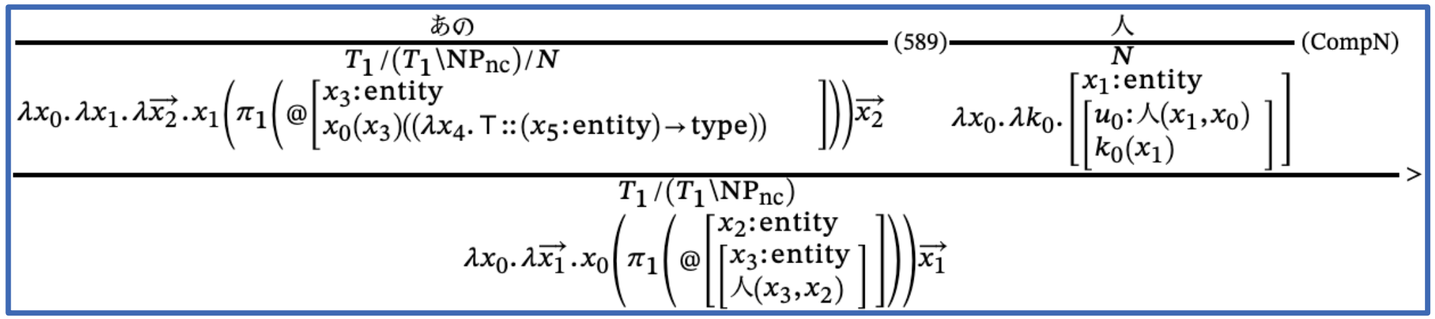
Why is score relatively low?

One reason: **annotation errors** in ABCTreebank (used as gold standard)

The evaluation **assumes ABCTreebank is linguistically valid**
→ Any errors in its annotations **directly lower the score**

Example:





- Determiners are annotated as **N/N** (returns a noun) but should be **NP/N** (yields a noun phrase)
- **Such category mismatches reduce scores,** even if the structure is in lightblue CCGbank is linguistically correct

Manual Evaluation

To assess the reliability of our syntax-based evalation metric, we compared its results against a manually annotated subset of 152 sentences from the lightblue CCGbank.

True : syntactic structure is linguistically valid

False : syntactic structure contains error

		Manual Evaluation	
		True	False
Score > 50	True	48	27
	False	13	64
Accuracy		0.739	
Precision		0.640	
Recall		0.787	
F1		0.706	

High recall: Most linguistically valid structures were correctly identified
Lower precision: Some false positives
→ likely caused by overpermissive category matching

Conclusion:
The syntax-based metric is a **useful proxy** for linguistic validity
However, it may require **refinement to improve precision**

Conclusion

Linguistically valid treebanks are essential for inference based on compositional semantics

- Proposed **metrics** for evaluating linguistic validity of Japanese CCG treebanks
  - **Syntactic alignment**
  - **Semantic well-formedness** (via type checking)