Transformers Learn Different Parsing Strategies across Languages

Anonymous ACL submission

Abstract

Transformer-based language models (LMs) are trained purely for next-word prediction, yet they exhibit sensitivity to syntax. However, little is known about how they internally parse syntactic structures. Recent work has probed autoregressive LMs via an arc-standard shift-reduce dependency parser, revealing incremental syntactic states in LM representations, but their methodology is limited to a single dependency parsing strategy and fails to give insight into which parsing strategy is most compatible with autoregressive LM representations among many possible parsing strategies. In this paper, we extend the incremental probing methodology to constituency structures and investigate which parsing strategy best explains the internal parsing process of autoregressive LMs among top-down, bottom-up, and left-corner strategies. Our empirical results suggest that LMs implicitly learn different parsing strategies for different languages, with top-down being most prevalent in English and left-corner in Japanese.

1 Introduction

Transformer-based language models (LMs) often exhibit strong sensitivity to syntax, even though they are trained purely for next-word prediction. Targeted syntactic evaluation (TSE) reveals that LMs are sensitive to certain syntactic constraints by examining their behavior on linguistically controlled datasets (Hu et al., 2020; Mueller et al., 2020; Someya and Oseki, 2023; Taktasheva et al., 2024). In parallel with these output-based evaluations, interpretability work probes hidden states to understand what structural information is encoded in internal representations. As a representative example, structural probing has decoded dependency structures from contextual representations in masked LMs (Hewitt and Manning, 2019), providing evidence that syntactic trees are recoverable from representation geometry. Eisape et al. (2022) extended probing to the incremental setting by decoding partial dependency structures from autoregressive LMs, under an arc-standard shift-reduce transition system, showing that LMs encode incremental parsing processes in their latent representations.

However, their methodology is limited to a single dependency parsing strategy and fails to give insight into which parsing strategy is most compatible with autoregressive LM representations among many possible parsing strategies. Additionally, while multilingual structural probing has begun to study cross-linguistic regularities in how masked LMs encode syntax (Chi et al., 2020), there has been no direct investigation of which incremental parsing strategy is most compatible with autoregressive LM representations across languages.

In this work, we extend the incremental probing methodology to constituency structures and investigate which parsing strategy best explains the internal parsing process of autoregressive LMs.[1] More concretely, we derive gold action sequences for three canonical constituency parsing strategies (top-down, bottom-up, and left-corner) within a unified shift-reduce action inventory and train separate structural probes to decode these actions from LM hidden states in two typologically distinct languages, English and Japanese. We then evaluate and compare the performance of these probes to determine which parsing strategy best explains the incremental parsing process of autoregressive LMs.

Our results suggest that LMs implicitly learn different incremental parsing strategies for different languages, with the top-down strategy being the most prevalent in English and the left-corner strategy being the most prevalent in Japanese.

[1] We focus on constituency structures because they represent richer syntactic information (Arps et al., 2022) and the differences between parsing strategies are more meaningful than dependency structures in terms of the cognitive plausibility of the strategies. See App. A for further discussion on this point.

2 Methods

2.1 Incremental-Parse Probe

Inspired by Eisape et al. (2022), we model incremental parsing as a sequence prediction problem over parser actions, conditioned on an LM's hidden representations, with auxiliary token prediction loss to facilitate training. Concretely, at each word boundary t (between w_t and w_{t+1}), the probe predicts the sequence of parser actions that occur *between* w_t and w_{t+1} based on (i) the action history and (ii) the LM representations of the prefix.

We instantiate this probe with the *No-Stack Action Probe* (NAP), which achieved the best action perplexity among the probes proposed by Eisape et al. (2022). Let $\mathrm{e}(a_j)$ be a learnable action embedding for the j-th action and let $\mathbf{h}_{\leq t}$ be the LM hidden representations for the prefix up to word boundary t. NAP parameterizes $P(a_{j+1} \mid a_{\leq j}, \mathbf{h}_{\leq t})$ by encoding the action history $a_{\leq j}$ with an LSTM and attention over LM hidden states $\mathbf{h}_{\leq t}$ for the prefix $w_{\leq t}$.

We compute the probe's action probabilities as follows:

$$\mathbf{v}_j = \text{Action-LSTM}(\mathbf{v}_{j-1}, \mathrm{e}(a_j)), \tag{1}$$

$$\tilde{\mathbf{h}}_j = \text{Attention}(\mathbf{h}_{\leq t}, \mathbf{v}_j), \tag{2}$$

$$P(a_{j+1} \mid a_{\leq j}, \mathbf{h}_{\leq t}) = \text{softmax}(\text{MLP}_\text{a}([\tilde{\mathbf{h}}_j, \mathbf{v}_j])). \tag{3}$$

We additionally optimize an auxiliary token prediction loss at `SHIFT` positions (cf. §2.2): we compute cross-entropy between token logits produced by MLP_t and the corresponding gold token IDs. The total objective is $\mathcal{L} = \mathcal{L}_\text{action} + \lambda \mathcal{L}_\text{token}$ with $\lambda = 0.1$. See App. E for hyperparameter details.

2.2 Probing Incremental Parsing Strategies

Eisape et al. (2022) focus on incremental *dependency* parsing under a fixed transition system (arc-standard), implicitly committing to one particular order of structure building. In contrast, we treat the parsing strategy itself as a testable hypothesis about the LM. To compare strategies within a unified framework, we use a *generalized* shift-reduce action inventory (Ishii and Miyao, 2025) in which different strategies correspond to different constraints on when structure is predicted. Our action inventory is $\mathcal{A} = \{\texttt{NT}(X, n), \texttt{SHIFT}, \texttt{REDUCE}\}$:

- $\texttt{NT}(X, n)$: open a constituent with label X (e.g., `S`, `NP`, `VP`), conceptually inserting an open nonterminal at the n-th position from the top in the implicit derivation,
- `SHIFT`: consume the next token from the input,
- `REDUCE`: close the topmost open constituent, completing it as a subtree,

where n controls *how speculative* the model is about upcoming structure: $n = 0$ corresponds to opening a constituent before any of its children are completed (more top-down/speculative), while larger n opens the constituent only after more material has already been processed (more bottom-up/less speculative). In this work, we focus on three canonical parsing strategies (Figure 1) and realize each as a constraint scheme over $\mathcal{A}$:

- **Top-down:** if the position to open a phrase is always $n = 0$ (i.e., before any children are completed), the strategy becomes equivalent to top-down.
- **Bottom-up:** if a `REDUCE` action is always performed immediately after an $\texttt{NT}(X, n)$ action, the strategy becomes equivalent to bottom-up, because a phrase with n children is only predicted after all of its children are completed.
- **Left-corner:** an intermediate strategy that opens a phrase immediately after its leftmost child (the *left corner*) is completed, but before the remaining children are processed ($n = 1$).

To keep the action inventory manageable, we cap the insertion parameter at $n \leq 10$ for all nonterminal actions (including $\texttt{NT}(\texttt{S}, n)$), following Ishii and Miyao (2025). For each strategy, we derive gold action sequences and train a separate probe to decode those actions from LM hidden states.

3 Experiments

3.1 Dataset

We use gold constituency trees from the Penn Treebank (PTB; Wall Street Journal portion) for English (Marcus et al., 1993) and the Kainoki Treebank (a parsed corpus of contemporary Japanese) for Japanese (Kainoki, 2022). We filter out sentences whose gold action sequences would require an insertion position $n > 10$ for $\texttt{NT}(X, n)$ (cf. §2.2) and use all available sentences in each split after applying a minimum-length filter of ≥ 10 words. After filtering, we retain ≈ 40K sentences for PTB and ≈ 56K sentences for Kainoki (see Table 1 for split sizes).

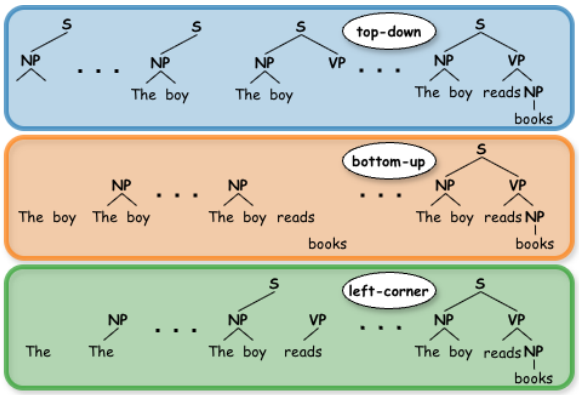

Figure 1: Three canonical parsing strategies: top-down, bottom-up and left-corner.

Preprocessing. We apply a merge-based token alignment procedure that edits each constituency tree to ensure compatibility with the target LM's subword tokenizer. Given a tree with word/morpheme leaves and the tokenizer's output tokens for the corresponding sentence, we detect the point where the tokenizer tokens cross the boundary of multiple constituencies. When we find such a case, we identify the lowest common ancestor (LCA) of the constituencies and directly attach the tokenizer tokens within those constituencies to the LCA node. We repeat this process until no tokenizer token crosses a constituent boundary, and the resulting tree has tokenizer tokens as leaf nodes. See Figure 3 for an illustrative example.

3.2 Model

Throughout this work, we use `llm-jp-3-150m` and `llm-jp-3-980m` (LLM-jp et al., 2024) models as the shared LM backbone. We choose this model because they are trained on the same public dataset and use the same tokenizer, allowing us to run the same probing pipeline across languages without changing tokenization.

3.3 Evaluation

We report *parse accuracy* for each parsing strategy: we decode a complete parse from the probe's action probabilities using word-synchronous beam search (Stern et al., 2017)[2] and compare the resulting tree to the gold treebank annotation using the labeled constituent F1 score. We evaluate decoding under beam sizes $k \in \{50, 100, 500, 1000, 2000\}$ and include a random-embedding baseline where LM embeddings are replaced with random vectors sampled from a standard normal distribution.

Regarding beam size, we theoretically expect that smaller beams may fail to fully capture the structural information the probe extracts from LM representations. As beam size increases, we expect F1 scores to saturate once the beam sufficiently explores the probe's probability distribution. Hence, we will mainly focus on the performance with the largest beam size ($k = 2000$).

4 Results

Figure 2 shows the parse accuracy for the NAP probe under top-down, bottom-up, and left-corner strategies for PTB (English) and Kainoki Treebank (Japanese) for `llm-jp-3-150m`.[3] The random-embedding baseline is included for reference as a dashed line. The results are averaged over three runs with different random seeds.

Overall trends. Across both languages and all strategies, intermediate layers achieve substantially higher F1 scores compared to the random-embedding baselines and pre-contextualized embeddings (`emb`), suggesting that the probes indeed capture the syntactic information encoded in the LM's hidden representations, rather than merely reflecting the inherent learnability of each parsing strategy for a given language.

Notably, the bottom-up strategy consistently underperforms the top-down and left-corner strategies across all beam sizes in both languages. Given the three strategies mainly differ in *when* they commit to higher-level structure (top-down earliest, left-corner intermediate, bottom-up latest; §2.2), all else equal, decoding should therefore be no harder for less speculative strategies, since they postpone commitments until more context is available. The poor performance of the bottom-up strategy likely stems from two factors: (1) the inherently larger action space[4] makes the action prediction task itself more difficult, and (2) a potential architectural mismatch with NAP—the attention mechanism of NAP (Eq. (2)) may struggle to retrieve the information needed for late constituent opening with the action-history embedding $\mathbf{v}_j$, which doesn't have

[2] During decoding, we do not impose additional next-action constraints based on parser state (e.g., disallowing actions that would be invalid under the current stack state). As a result, some beams fail to yield a well-formed, complete parse; we report parsing failure rates in Table 4.

[3] See Figure 4 for the results for `llm-jp-3-980m`. We observe similar trends for `llm-jp-3-980m` as well.

[4] This is because while the n in $\mathtt{NT}(\mathtt{S}, n)$ is constant for top-down ($n = 0$) and left-corner ($n = 1$), multiple ns are allowed in bottom-up strategy.

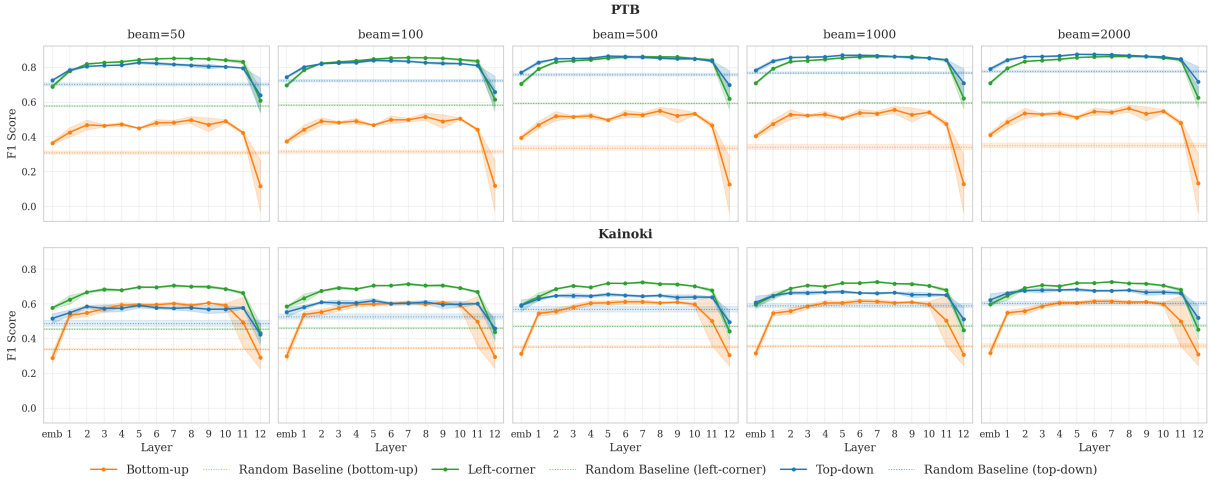

Figure 2: Labeled constituent F1 for the NAP probe under different parsing strategies for PTB (top) and Kainoki Treebank (bottom) for $k \in \{50, 100, 500, 1000, 2000\}$. The random-embedding baseline is shown as a dashed line for reference.

terminal information, weakening structural cues before the final action classifier in Eq. (3). Given these confounds, we focus our analysis on comparing top-down and left-corner strategies in the following part of this section.

English (PTB). In English, top-down and left-corner strategies achieve comparable parse accuracy at small beam sizes, but as beam size increases and F1 scores saturate, top-down consistently reaches slightly higher performance. This suggests that LMs prioritize a relatively eager, top-down-like structure when processing English. This finding aligns naturally with results from syntactic parsing research showing that eager strategies are well-suited for right-branching languages like English (Kuncoro et al., 2018; Ishii and Miyao, 2025), suggesting that even LMs trained purely for next-word prediction converge on an optimal strategy.

Japanese (Kainoki Treebank). In Japanese, the pattern differs markedly: left-corner consistently outperforms top-down at small beam sizes, and crucially, even when F1 scores saturate at larger beam sizes, left-corner maintains a clear advantage. This suggests that LMs processing Japanese encode structural information that is better captured by a moderately eager left-corner strategy rather than the maximally eager top-down approach. This result aligns with psycholinguistic findings that humans process left-branching languages like Japanese with the left-corner parsing strategy rather than the top-down parsing strategy (Yoshida et al., 2021; Sugimoto et al., 2024). Notably, LMs, despite being trained on the inherently speculative task of next-word prediction, appear to naturally learn a strategy with reduced structural speculation for Japanese compared to English, aligning with psycholinguistic findings that this may be more human-like for left-branching languages.

5 Conclusion

In this paper, we extended the incremental probing methodology to constituency structures and investigated which parsing strategy best explains the internal parsing process of autoregressive LMs. Our results suggested that LMs learn different parsing strategies for different languages, with the top-down strategy being more prevalent than the left-corner in English and the left-corner strategy being more prevalent than the top-down in Japanese.

Limitations

Our study compares incremental parsing strategies across two languages. While we believe that this two-language comparison is already informative, it remains unclear how the observed cross-linguistic differences generalize to other languages and domains.

Our probing method also has architectural limitations. As observed with bottom-up parsing (§4), the NAP probe's attention mechanism may struggle to retrieve information needed for late constituent opening. This makes it difficult to distinguish whether poor probe performance reflects absent information in LM representations or limitations in

the probe's extraction capacity. Alternative probing architectures may be needed to better evaluate different parsing strategies.

We also compared 150M- and 1B-parameter LMs and observed similar trends, but it is unknown whether the same patterns hold for substantially larger models.

Finally, probing establishes which parsing strategy is implicitly learned by the LM, but does not by itself show that the LM *uses* that information during next-word prediction. Future work should test whether the LM actually uses the information by conducting controlled interventions on internal representations.

References

Steven P. Abney and Mark Johnson. 1991. Memory requirements and local ambiguities of parsing strategies. *Journal of Psycholinguistic Research*, 20(3):233–250.

David Arps, Younes Samih, Laura Kallmeyer, and Hassan Sajjad. 2022. Probing for constituency structure in neural language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6738–6757, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577, Online. Association for Computational Linguistics.

Tiwalayo Eisape, Vineet Gangireddy, Roger Levy, and Yoon Kim. 2022. Probing for incremental parse states in autoregressive language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2801–2813, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Jennifer Hu, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger P Levy. 2020. A systematic assessment of syntactic generalization in neural language models.

Taiga Ishii and Yusuke Miyao. 2025. Is incremental structure prediction process universal across languages?: Revisiting parsing strategy through speculation. In *Proceedings of the 29th Conference on Computational Natural Language Learning*, pages 437–451, Vienna, Austria. Association for Computational Linguistics.

Ed Kainoki. The kainoki treebank – a parsed corpus of contemporary japanese [online]. 2022.

Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. 2018. LSTMs can learn syntax-sensitive dependencies well, but modeling structure makes them better. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436, Melbourne, Australia. Association for Computational Linguistics.

Tal Linzen and Marco Baroni. 2021. Syntactic structure from deep learning. *Annu. Rev. Linguist.*, 7(1):195–212.

LLM-jp, :, Akiko Aizawa, Eiji Aramaki, Bowen Chen, Fei Cheng, Hiroyuki Deguchi, Rintaro Enomoto, Kazuki Fujii, Kensuke Fukumoto, Takuya Fukushima, Namgi Han, Yuto Harada, Chikara Hashimoto, Tatsuya Hiraoka, Shohei Hisada, Sosuke Hosokawa, Lu Jie, Keisuke Kamata, Teruhito Kanazawa, Hiroki Kanezashi, Hiroshi Kataoka, Satoru Katsumata, Daisuke Kawahara, Seiya Kawano, Atsushi Keyaki, Keisuke Kiryu, Hirokazu Kiyomaru, Takashi Kodama, Takahiro Kubo, Yohei Kuga, Ryoma Kumon, Shuhei Kurita, Sadao Kurohashi, Conglong Li, Taiki Maekawa, Hiroshi Matsuda, Yusuke Miyao, Kentaro Mizuki, Sakae Mizuki, Yugo Murawaki, Akim Mousterou, Ryo Nakamura, Taishi Nakamura, Kouta Nakayama, Tomoka Nakazato, Takuro Niitsuma, Jiro Nishitoba, Yusuke Oda, Hayato Ogawa, Takumi Okamoto, Naoaki Okazaki, Yohei Oseki, Shintaro Ozaki, Koki Ryu, Rafal Rzepka, Keisuke Sakaguchi, Shota Sasaki, Satoshi Sekine, Kohei Suda, Saku Sugawara, Issa Sugiura, Hiroaki Sugiyama, Hisami Suzuki, Jun Suzuki, Toyotaro Suzumura, Kensuke Tachibana, Yu Takagi, Kyosuke Takami, Koichi Takeda, Masashi Takeshita, Masahiro Tanaka, Kenjiro Taura, Arseny Tolmachev, Nobuhiro Ueda, Zhen Wan, Shuntaro Yada, Sakiko Yahata, Yuya Yamamoto, Yusuke Yamauchi, Hitomi Yanaka, Rio Yokota, and Koichiro Yoshino. 2024. Llm-jp: A cross-organizational project for the research and development of fully open japanese llms. *Preprint*, arXiv:2407.03963.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Rowan Hall Maudslay and Ryan Cotterell. 2021. Do syntactic probes probe syntax? experiments with jabberwocky probing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 124–131, Online. Association for Computational Linguistics.

Aaron Mueller, Garrett Nicolai, Panayiota Petrou-Zeniou, Natalia Talmina, and Tal Linzen. 2020.

Cross-linguistic syntactic evaluation of word prediction models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5523–5539, Online. Association for Computational Linguistics.

Philip Resnik. 1992. Left-corner parsing and psychological plausibility. In *COLING 1992 Volume 1: The 14th International Conference on Computational Linguistics*.

Taiga Someya and Yohei Oseki. 2023. JBLiMP: Japanese benchmark of linguistic minimal pairs. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1581–1594, Dubrovnik, Croatia. Association for Computational Linguistics.

Taiga Someya, Ryo Yoshida, Hitomi Yanaka, and Yohei Oseki. 2025. Derivational probing: Unveiling the layer-wise derivation of syntactic structures in neural language models. In *Proceedings of the 29th Conference on Computational Natural Language Learning*, pages 93–104, Vienna, Austria. Association for Computational Linguistics.

Mitchell Stern, Daniel Fried, and Dan Klein. 2017. Effective inference for generative neural parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1695–1700, Copenhagen, Denmark. Association for Computational Linguistics.

Yushi Sugimoto, Ryo Yoshida, Hyeonjeong Jeong, Masatoshi Koizumi, Jonathan R. Brennan, and Yohei Oseki. 2024. Localizing Syntactic Composition with Left-Corner Recurrent Neural Network Grammars. *Neurobiology of Language*, 5(1):201–224.

Ekaterina Taktasheva, Maxim Bazhukov, Kirill Koncha, Alena Fenogenova, Ekaterina Artemova, and Vladislav Mikhailov. 2024. RuBLiMP: Russian benchmark of linguistic minimal pairs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9268–9299, Miami, Florida, USA. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Jennifer C. White, Tiago Pimentel, Naomi Saphra, and Ryan Cotterell. 2021. A non-linear structural probe. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 132–138, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ryo Yoshida, Hiroshi Noji, and Yohei Oseki. 2021. Modeling human sentence processing with left-corner recurrent neural network grammars. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2964–2973, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Related Work

Syntactic evaluation and representational analyses. The syntactic knowledge of neural LMs has been investigated through both behavioral evaluations and analyses of internal representations. Behavioral-style evaluations measure model preferences on controlled stimuli, including acceptability-judgment benchmarks (Warstadt et al., 2018) and broader targeted syntactic evaluations (Linzen and Baroni, 2021). Complementarily, representational probing methods ask whether syntactic structure is decodable from hidden states. For example, Hewitt and Manning (2019) propose a structural probe that learns a linear transformation of token representations so that pairwise distances in the transformed space match the distances between words in the gold dependency tree (i.e., the number of edges along the path connecting two words).

Building on this, later work refined the approach by incorporating non-linear mappings (e.g., White et al., 2021), using a controlled corpus to isolate the effect of syntax (Maudslay and Cotterell, 2021) and applying it layer by layer to investigate the layer-wide development of syntactic structure (Someya et al., 2025)

Incremental probing and parsing strategies. Most closely related to our setting, Eisape et al. (2022) extend probing to the *incremental* case by learning probes that predict parse actions/states for autoregressive LMs under a fixed dependency transition system (arc-standard), and show that inferred states can be used for causal interventions. However, the incremental parsing process is not uniquely specified: another syntactic formalism, constituency structures, admits multiple parsing strategies that differ in *when* structure is committed (e.g., top-down vs. bottom-up vs. left-corner), and comparisons of such strategies have been central in psycholinguistics (Abney and Johnson, 1991;

Resnik, 1992) and in structured language modeling (Kuncoro et al., 2018; Yoshida et al., 2021).

B Dataset Statistics

Table 1 summarizes the number of sentences used in each dataset split after filtering.

Dataset	Split	Sentences
PTB	train	36,542
	dev	1,569
	test	2,191
Kainoki	train	50,883
	dev	2,821
	test	2,829

Table 1: Number of sentences per split after filtering.

C Merge-Based Token Alignment

Figure 3 shows the schematic examples of merge-based token alignment via LCA flattening.

Example A (No boundary violation; A leaf can consists of multiple tokens)
Tree (before): `(X (Y A B) (Z C))`
Tokenizer output: `[<sp>A][B1][B2][C]`
Tree (after): `(X (Y A B1 B2) (Z C))`

Example B (LCA merge within a local phrase)
Tree (before): `(X (Y A B) (Z C))`
Tokenizer output: `[<sp>AB][C]`
LCA: `Y`
Flatten at LCA and merge: `(Y A B)` ↦ `(Y AB)`
Tree (after): `(X (Y AB) (Z C))`

Example C (LCA spans a larger subtree; flatten then merge)
Tree (before): `(X (Y A) (Z (U B) (V C)))`
Tokenizer output: `[<sp>AB][C]`
LCA: `X`
Flatten at LCA and merge: `(X A B C)` ↦ `(X AB C)`

Figure 3: Schematic examples of merge-based token alignment via LCA flattening.

D License of the data/tools

We summarize the license of the data/tools employed in this paper in Table 2. All data and tools were used under their respective license terms.

E Hyperparameters

Table 3 shows the hyperparameters for the NAP probe training and inference.

Tool	License
`transformers` (Wolf et al., 2020)	Apache 2.0
`llm-jp-3-150m` (LLM-jp et al., 2024)	Apache 2.0
`llm-jp-3-980m` (LLM-jp et al., 2024)	Apache 2.0
Dataset	**License**
PTB (Marcus et al., 1993)	LDC User Agreement
Kainoki Treebank (Kainoki, 2022)	CC BY 4.0

Table 2: License of the data/tools

F Parsing Failure Rates

Table 4 shows the parsing failure rates for the NAP probe under different parsing strategies for PTB (English) and Kainoki Treebank (Japanese). The results are averaged over three runs with different random seeds. We observe higher error rates for the bottom-up strategy across all beam sizes and datasets.

G Additional Results

Figure 4 shows the parse accuracy for the NAP probe under different parsing strategies for PTB (English) and Kainoki Treebank (Japanese) for `llm-jp-3-980m`. The random-embedding baseline is included for reference as a dashed line. We observe similar trends as in `llm-jp-3-150m`, suggesting that the observed parsing preferences are shared across model sizes. Due to computational resource constraints, we conduct experiments only with $k \in \{50, 100, 500, 1000\}$.

Probe (NAP) architecture	
Action-LSTM hidden size	256
Action-LSTM layers	1
Attention hidden size	256
MLP hidden sizes	(512, 256)
Dropout	0.1
Token loss weight	0.1
Training / evaluation	
Optimizer	Adam (weight decay 10^{-5})
Learning rate	0.001
Batch size	32
Max epochs	100
Early stopping patience	5
LR scheduler	ReduceLROnPlateau (factor 0.5, patience 2)
Max word length	128
Beam sizes (evaluation)	$\{50, 100, 500, 1000, 2000\}$

Table 3: Model and probe hyperparameters.

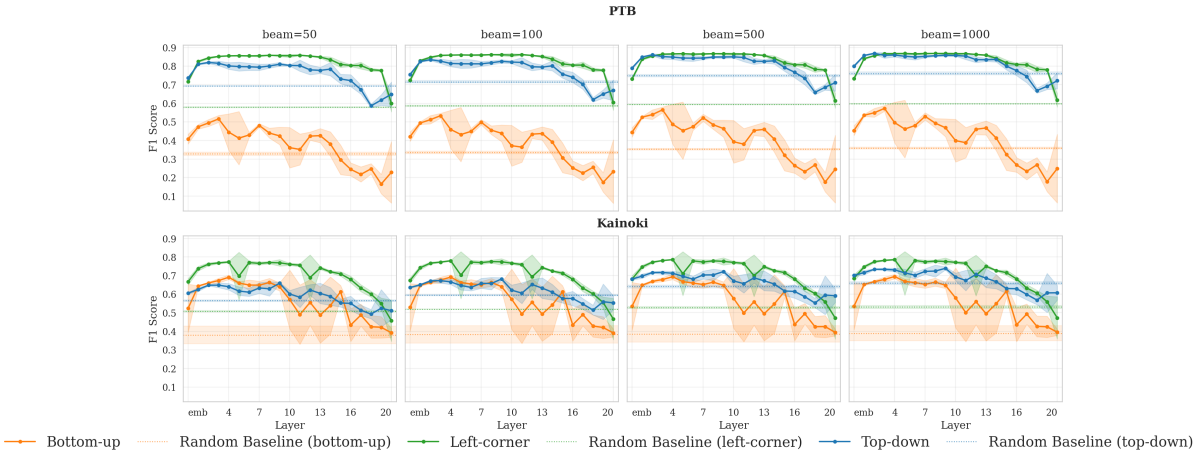

Figure 4: Parse accuracy for the NAP probe under different parsing strategies for PTB (English) and Kainoki Treebank (Japanese) for `llm-jp-3-980m`.

Dataset	Strategy	Beam	Failure rate (%)
PTB	top-down	50	4.87 ± 5.61
		100	4.24 ± 5.14
		500	2.98 ± 4.47
		1000	2.51 ± 4.30
	bottom-up	50	41.42 ± 13.67
		100	38.98 ± 14.19
		500	34.25 ± 15.43
		1000	32.83 ± 15.81
		2000	31.58 ± 16.18
	left-corner	50	2.00 ± 0.86
		100	1.63 ± 0.89
		500	1.04 ± 0.89
		1000	0.86 ± 0.85
Kainoki	top-down	50	3.91 ± 3.00
		100	3.07 ± 2.52
		500	2.16 ± 2.08
		1000	1.93 ± 2.00
	bottom-up	50	19.01 ± 9.80
		100	18.45 ± 9.65
		500	17.42 ± 9.28
		1000	17.19 ± 9.25
	left-corner	50	1.82 ± 2.42
		100	1.62 ± 2.30
		500	1.39 ± 2.24
		1000	1.31 ± 2.12

Table 4: Parsing failure rates for the NAP probe under different parsing strategies for PTB (English) and Kainoki Treebank (Japanese) for `llm-jp-3-150m`.