Probing for Incremental Parse States in Autoregressive Language Models

Tiwalayo Eisape[1] **Vineet Gangireddy**[2] **Roger P. Levy**[1] **Yoon Kim**[1]
MIT[1], Harvard University[2]
{eisape,rplevy,yoonkim}@mit.edu
vineetgangireddy@college.harvard.edu

Abstract

Next-word predictions from autoregressive neural language models show remarkable sensitivity to syntax. This work evaluates the extent to which this behavior arises as a result of a learned ability to maintain implicit representations of incremental syntactic structures. We extend work in syntactic probing to the incremental setting and present several probes for extracting incomplete syntactic structure (operationalized through parse states from a stack-based parser) from autoregressive language models. We find that our probes can be used to predict model preferences on ambiguous sentence prefixes and causally intervene on model representations and steer model behavior. This suggests implicit incremental syntactic inferences underlie next-word predictions in autoregressive neural language models.

1 Introduction

The behavior of large-scale autoregressive neural language models (ALMs) appears to demonstrate impressive command of syntax (Wilcox et al., 2019; Hu et al., 2020; Futrell et al., 2019; Wilcox et al., 2021; Arehalli and Linzen, 2020; Warstadt and Bowman, 2020). To what extent can we attribute this behavior to a model's maintaining and updating representations of incremental syntactic structures?

Interpretability work on *bidirectional* masked language models suggests that neural models of language may learn to encode syntactic structure through the geometry of their word embedding space. For example, Hewitt and Manning (2019) demonstrate that the dependency parse tree of a sentence can be decoded by finding the minimum spanning tree on pairwise syntactic distances regressed from linearly transformed contextualized word embeddings.

Unlike the models considered in Hewitt and Manning (2019), ALMs are *unidirectional*, and the decision problem of incrementally deriving global syntactic structures has several nuances not present in the bidirectional setting. Sentence prefixes (e.g. "I watched her duck ...") can be ambiguous in their intended meaning in ways that are disambiguated by their suffixes (e.g. "quack" vs. "under the table"). Thus, incremental processors must maintain a belief state of parses that can be flexibly updated on the basis of future input. Insofar as language in the real world (e.g., speech, text, sign) is processed sequentially, incremental disambiguation of structure and meaning is a task humans solve in everyday cognition, seemingly effortlessly (Jurafsky, 1996; Hale, 2001; Levy, 2008).

ALMs have been shown to recapitulate crucial features of human sentence processing (Futrell et al., 2019) and their representations moreover have been shown to align with language processing in the brain better than their bidirectional counterparts (Schrimpf et al., 2021; Caucheteux and King, 2022). We hypothesize that ALMs learn and maintain correlates of incremental syntactic structures which play an important role in mediating model behavior. We investigate this hypothesis through the lens of *counterfactual probing*, i.e., by learning classifiers over hidden states of pretrained ALMs to predict linguistic properties, and then using the probes to intervene on model representations.

We present a suite of probing architectures for decoding belief states of incremental structures from pretrained Transformer ALMs given the model's hidden state, each of which embodies a different hypothesis for how incremental parse states might be encoded. We validate (and adjudicate between) our probes by using them to predict and control model behavior. Our results suggest that ALMs, through pretraining at scale, learn stack-like representations of syntax and use them in interpretable and controllable ways.

Code and materials: https://github.com/eisape/incremental_parse_probe

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 2801 - 2813
December 7-11, 2022

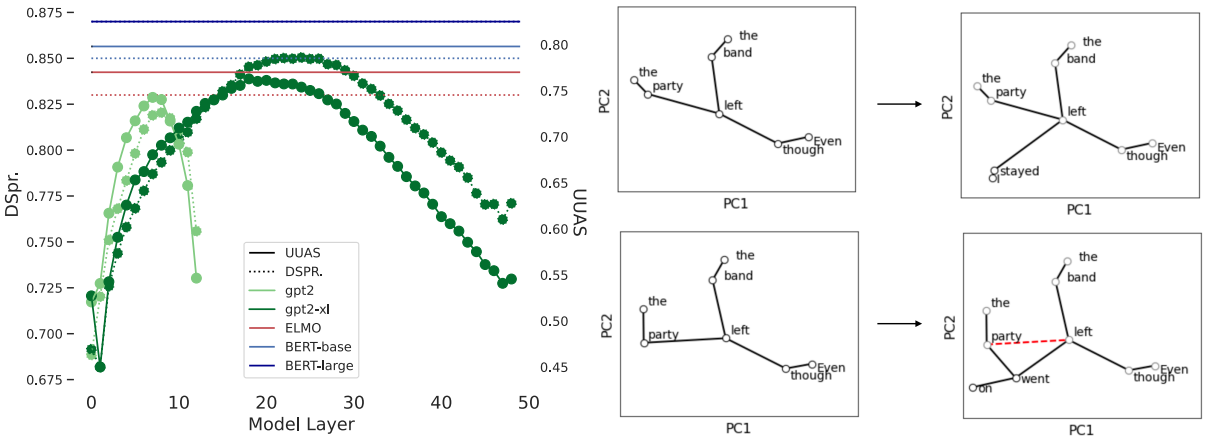

Figure 1: (Left) Results of applying the structural probe of Hewitt and Manning (2019) to GPT2. Displayed are the performances of the distance as quantified in both unlabeled undirected attachment score (UUAS) and Spearman correlation (Dspr.) across layers. Layer 0 denotes uncontextualized embeddings. The best performing BERT and ELMo layers as reported in Hewitt and Manning (2019) are shown as horizontal lines. (Right) Dependency graphs elicited from GPT2 small for the sentences in Figure 2 via Hewitt and Manning (2019)'s structural probe before (left) and after (right) disambiguation, and then decoding with minimum spanning tree. Visualization is done by projecting the predicted pairwise distances into two dimensions via PCA. In the zero-complement condition (bottom), the "went" embedding is emitted between an existent dependency resulting in a new parse, effectively 'deleting' the arc in red.

2 Motivating Study

Prior work has developed methods for extracting syntactic parses from neural language models given global context. In particular, Hewitt and Manning (2019) learn a distance function parameterized by a linear transformation (B) of the word embedding space of BERT,

$$\delta_B\left(\mathbf{h}_i, \mathbf{h}_j\right)^2 = \left(B\left(\mathbf{h}_i - \mathbf{h}_j\right)\right)^T \left(B\left(\mathbf{h}_i - \mathbf{h}_j\right)\right),$$

such that this distance approximates the distance of words w_i, w_j in a dependency tree. After learning B via regression, i.e.,

$$\min_B \sum_\ell \frac{1}{|s^\ell|^2} \sum_{i,j} \left| \delta_{T^\ell}\left(w_i^\ell, w_j^\ell\right) - \delta_B\left(\mathbf{h}_i^\ell, \mathbf{h}_j^\ell\right)^2 \right|,$$

(here ℓ indexes the sentences in the training set, $|s^\ell|$ is the sentence length, and $\delta_{T^\ell}(w_i, w_j)$ is the tree distance between w_i and w_j), the authors find that the minimum spanning tree obtained from $\delta_B(\cdot,\cdot)$ well-approximates the gold dependency tree in many cases, indicating that the geometry of contextualized word embeddings captures aspects of syntax.

In an initial study, we run Hewitt and Manning (2019)'s structural probe on the hidden states of GPT2 (Radford et al., 2019), a unidirectional ALM, and find that GPT2's hidden states recover gold tree structures almost as well as similarly-sized bidirectional models (Figure 1). For example GPT2-XL

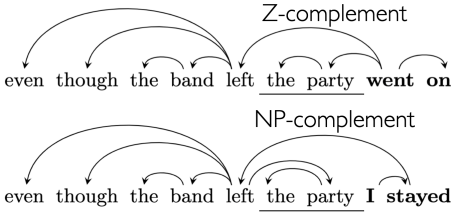

Figure 2: NP/Z ambiguous sentences and their parses.

matches BERT-base in correlation with syntactic distances (Dspr.; .85 for both models). We further observe that this probe can recover the correct structures even on sentences with ambiguous prefixes, such as the 'NP/Z' ambiguity shown in Figure 2. In this ambiguity, the shared prefix ("even though the band left the party") is consistent with at least two interpretations: one in which the underlined span ("the party") is a direct object noun phrase (NP) of the verb "left", and one in which the underlined content is the subject of a new noun phrase and "left" has a zero (Z) complement. This is a well-studied phenomenon in psycholinguistics, and humans make rapid and accurate inferences in this setting (Bever, 1970).

It is not at all obvious how a strictly left-to-right model, whose representations of the words in context are *static* (i.e., not affected by future words), can maintain a prefix representation that are *dynamic* enough to encode global syntactic distances with high accuracy. We visualize the evolution of pairwise syntactic distances for the two sentences in Figure 2 along with the minimum spanning trees

in Figure 1 (right). We observe that parses probed from sentence prefixes up until the disambiguators are consistent with the NP-complement parse (i.e., where "party" is attached to "left"), which is in line with previous work that analyzes GPT2's behavior on the NP/Z ambiguity (Futrell et al., 2019). After observing "went," GPT2 emits a representation whose distance is interleaved between "left" and "party" (i.e., $\delta_B(\mathbf{h}_{\text{party}}, \mathbf{h}_{\text{went}}) < \delta_B(\mathbf{h}_{\text{party}}, \mathbf{h}_{\text{left}})$), thus altering the minimum spanning tree to be consistent with the zero-complement parse.

This analysis provides an initial hypothesis for how incremental parsing arises in ALMs from the perspective of syntactic distance—ALMs emit word representations that maintain syntactic uncertainty that can be exploited by later emissions, effectively editing inferred dependencies and bypassing the need to re-position past emissions. However, this analysis also reveals several limitations that preclude it as a viable model of incremental parsing in ALMs: (1) this method assumes a *spanning-tree* and therefore cannot represent parses with open nodes, and (2) GPT2's behavior is consistent with a probabilistic parallel parser as it seems to entertain both possible parses until seeing the disambiguating words, but such a pure distance-based probe is not inherently probabilistic.[1] In the following section, we develop several parameterizations of a probabilistic incremental parser that can represent incomplete syntactic structures.

3 An Incremental-Parse Probe for Autoregressive Language Models

We argue that the stack representation of shift-reduce parsers provides a natural way to represent incomplete tree structures. In this paper, we work with the generative arc-standard dependency formalism (Nivre, 2004), which maintains a stack of generated subtrees $S = [s_1, s_2, s_3, ...]$ (where the root of subtree s_i is a word), and makes one of the following actions at each time step:

> `LEFT-ARC`: pop the top two nodes, create a new subtree by adding an arc $s_1 \rightarrow s_2$, and push the new subtree onto the stack,
>
> `RIGHT-ARC`: pop the top two nodes, create a new subtree by adding an arc $s_2 \rightarrow s_1$, and push the new subtree onto the stack,
>
> `GEN`: generate the next token.

[1] Though it is possible to derive a probabilistic parser by interpreting the summed distances as the energy of a globally normalized model.

Generation starts with only `ROOT` (an embedding learned independently for each probe) on the stack, $S = [\texttt{ROOT}]$, and terminates when only `ROOT` is left on the stack and the period token has been generated. Note that an action sequence $\mathbf{a}$ fully specifies a projective dependency structure.[2] Hence, letting $\mathbf{a}_{\leq n(t)}$ be the sequence of actions up to (and including) the generation of w_t, we use $\mathbf{a}_{\leq n(t)}$ to represent the incremental (i.e., incomplete) tree structure state after emitting w_t.[3] We design several probes that place distributions over action trajectories - $P\left(\mathbf{a}_{\leq n(w_t)} \mid w_{<t}\right)$ given ALM embeddings $\mathbf{h}_{<t}$.

3.1 Probe architectures

We explore three probing architectures for parameterizing the action probabilities, each of which embodies a different hypothesis for how incremental parse states might be represented in ALMs.

Geometric Action Probe (GAP). Our first architecture leverages the geometry of the embeddings and links the syntactic distances and depths derived from Hewitt and Manning (2019) to action probabilities (Figure 3, left). The action probabilities are parameterized as below (where for brevity we omit the conditioning variables $w_{<t}$ and the previously generated actions):

$$P\left(\texttt{GEN}\right) = \sigma\left(\frac{\delta_B\left(\mathbf{h}_{s_1}, \mathbf{h}_{s_2}\right) - \tau}{\beta}\right),$$

$$P\left(\texttt{LEFT-ARC}\right) = \left(1 - P\left(\texttt{GEN}\right)\right) \times \sigma\left(\frac{\|\mathbf{h}_{s_1}\|_B - \|\mathbf{h}_{s_2}\|_B}{\beta}\right),$$

$$P\left(\texttt{RIGHT-ARC}\right) = \left(1 - P\left(\texttt{GEN}\right)\right) \times \sigma\left(\frac{\|\mathbf{h}_{s_2}\|_B - \|\mathbf{h}_{s_1}\|_B}{\beta}\right),$$

where τ is a threshold parameter, β is a temperature term, $\sigma(\cdot)$ is the sigmoid to turn distances into probabilities, and $\mathbf{h}_{s_i}$ is the ALM's representation for the root word of subtree s_i. Note that the probability of `GEN` increases as the predicted syntactic distance between (the root words of) s_1 and s_2 increases, which intuitively captures the notion that there is less likely to be a link between s_1 and s_2 if their predicted distance is large. The action

[2] Because ArcStandard can only recognize projective dependency trees, we exclude non-projective structures from training and evaluation in the sections to follow.

[3] We omit the generation of the next word (from GPT2's next word distribution) after predicting `GEN`, as we assume that sentences are given for probing purposes.

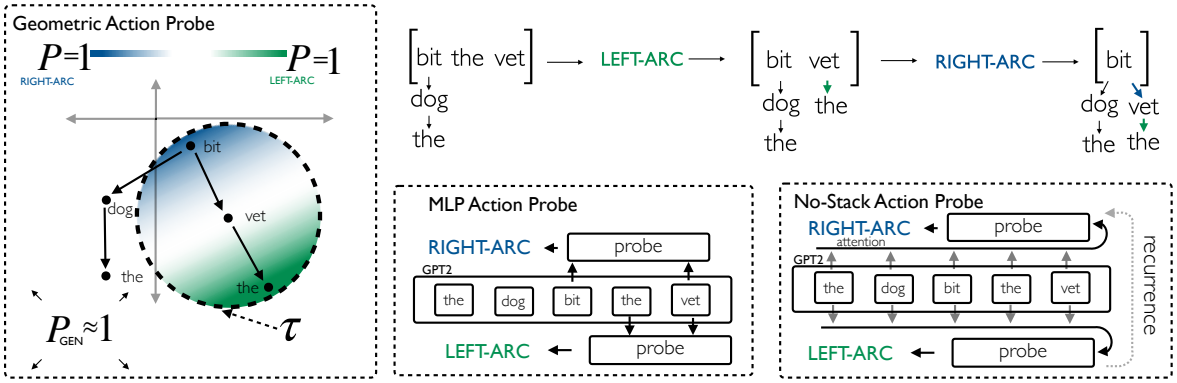

Figure 3: Schematics of each of the incremental parse probes parsing the sentence "the dog bit the vet". (Top) Incremental parse states (i.e., stacks of subtrees rooted by head words) are shown between the action transitions (LEFT-ARC, RIGHT-ARC) that connect the parse states. (Dotted) Visualizations of how each probe decides on the next actions. (Left) Geometric action probe (GAP) links action probabilities with its learned distance function δ_B. The distance from the top node on the stack ("vet" and "the" in the LEFT-ARC case) affects whether an arc is chosen. If the distance between the top two stack nodes is above a threshold τ, then then we GEN with increasing probability. Relative depth (blue = shallow relative to "vet", green = deeper) predicts arc directionality probability. (Center) MLP Action probe (MAP) directly classifies each action by running the contextualized word embeddings for the top two nodes on the stack through an MLP. (Right) No-Stack Action Probe (NAP) classifies by attending over the entire prefix without maintaining an explicit stack.

probabilities for LEFT-ARC and RIGHT-ARC are based on comparing the predicted *depths*, which we define to be the norm of the word embedding after a linear transformation as in Hewitt and Manning (2019). For example, the probability of the arc $s_1 \rightarrow s_2$ increases as s_2 is predicted to be further away from the root than s_1 (i.e., relatively deeper nodes are the dependants of shallower nodes; see Figure 3). We initialize B by pretraining on the distance and depth regression task from Hewitt and Manning (2019). Because we also use a linear projection our probe can also be interpreted as a learned distance function on word representations.

MLP Action Probe (MAP). The geometric action probe makes strong assumptions about the underlying geometry of the representation space, i.e., that syntactic distances and depths are well-captured by linear transformations of the ALM's representations and that these measures can moreover be monotonically transformed to action probabilities via a sigmoid link function. This next variant relaxes this assumption and replaces the distance and depth-based linking function in GAP with a learned multilayer perceptron,

$$P(a) \propto \exp\left(e_a^\top \text{MLP}\left([\mathbf{h}_{s_1}, \mathbf{h}_{s_2}]\right) + b_a\right)$$

where $a \in \{\text{GEN}, \text{LEFT-ARC}, \text{RIGHT-ARC}\}$. As in the geometric version, we still make use of an explicit stack, but this variant allows for an arbitrary link function from features of the model's representations to action probabilities.

No-Stack Action Probe (NAP). Our final variant relaxes assumptions about both the linking function and the existence of an explicit stack. This approach, which is closely related to Qian et al. (2021), simply predicts the sequence of actions *between* two words w_t and w_{t+1} using the action history and the hidden representations $\mathbf{h}_{<t}$,

$$\mathbf{v}_j = \text{Action-LSTM}(\mathbf{v}_{j-1}, a_j),$$
$$\tilde{\mathbf{h}} = \text{Attention}(\mathbf{h}_{<t}, \mathbf{v}_j),$$
$$P(a) \propto \exp\left(e_a^\top \text{MLP}\left([\tilde{\mathbf{h}}, \mathbf{v}_j]\right) + b_a\right).$$

Here the action LSTM's hidden state $\mathbf{v}_j$ is used to attend over the previous word representations $\mathbf{h}_{<t}$ to obtain a context vector $\tilde{\mathbf{h}}$, which is combined with the hidden state to produce a distribution over the following action.

We train each of these architectures[4] on the ground truth action trajectories in the Penn Treebank (PTB) (Marcus et al., 1993) and use gpt2 and gpt2-xl model checkpoints from HuggingFace (Wolf et al., 2019) as our ALM text encoders. All hyperparameters are reported in Appendix A.1.

4 Experiments

We evaluate each of our probes on their parsing performance over the PTB test split as well as their ability to predict and control model behavior.

[4]Due to resource constraints we only train NAP on GPT-2 small.

Algorithm 1: Probe-Based Word-Synchronous Beam Search

Input: $\mathbf{w}$: Words in sentence.
P_{probe}: Incremental parse probe.
k_{action}: Number of action n-grams to consider between words.
k_{word}: Max number of parses to consider for each word.
k_{out}: Number of parses to output.
Output: $\mathbf{B}^{\text{out}}$: beam of terminal action sequences

1 $\mathbf{B}, \mathbf{B}^{\text{out}} \leftarrow [[\texttt{ROOT}]], [\,]$
2 **while** $|\mathbf{B}^{\text{out}}| \leq k_{\text{out}}$ **do**
3 **for** $n \in [1, ..., |\mathbf{w}|]$ **do**
4 $\mathbf{B}^{\text{word}} \leftarrow [\,]$
5 $\mathbf{h}_{<n} \leftarrow \text{GPT2}\,(\mathbf{w}_{<n})$ // get hidden states from GPT2
6 **while** $\mathbf{B}$ not empty **do**
7 $\mathbf{a} \leftarrow \text{pop}\,(\mathbf{B})$
8 $\{\mathbf{a}^1, \mathbf{a}^2, ..., \mathbf{a}^{k_{\text{action}}}\} \leftarrow \text{BeamSearch}\,(P_{\text{probe}}, \mathbf{a}, \mathbf{h}_{<n}, k_{\text{action}})$ // search until GEN or termination
9 **for** $\mathbf{a}_{\text{new}} \in \{\mathbf{a}^1, \mathbf{a}^2, ..., \mathbf{a}^{k_{\text{action}}}\}$ **do**
10 **if** $n = |\mathbf{w}|$: $\text{push}\,(\mathbf{B}^{\text{out}}, \mathbf{a}.\text{append}\,(\mathbf{a}_{\text{new}}))$
11 **else** : $\text{push}\,\big(\mathbf{B}^{\text{word}}, \mathbf{a}.\text{append}\,(\mathbf{a}_{\text{new}})\big)$
12 $\mathbf{B} \leftarrow \text{top-k}\,\big(\mathbf{B}^{\text{word}}, k_{\text{word}}\big)$ // synchronize beam at word-level
13 **return** $\mathbf{B}^{\text{out}}$ // return beam with complete parse trees

Algorithm 1: Probe-based word-synchronous beam search. We use beam search (line 8) as a subroutine to extend action sequences in the beam with action n-gram continuations. After the generation of each word the beam is pruned to the top-k_{word} best parses (line 12).

4.1 Parsing Performance

For parsing performance, we evaluate with action perplexity (PPL) and unlabeled attachment score on the PTB test set. For the latter, we extend the word-synchronous beam search algorithm for decoding from generative neural parsers (Stern et al., 2017) to the incremental probe setting. This *probe-based* word-synchronous beam search algorithm uses an incremental parse probe to decode action sequences *between* word emissions with beam search. Because the word emissions from the ALMs we consider are contextualized, this quantity implicitly conditions on the previous action sequence including all of the words generated in the prefix so far[5] thus allowing us to predict the likelihood: $\prod_{a \in \pi \backslash \mathbf{w}} P(a)$ (where $\pi \backslash \mathbf{w}$ are the actions in any parse π excluding the likelihoods of words conditioned on actions).

This algorithm is an interesting testbed for ALMs because it relies *only* on the syntactic disambiguation implicit in the ALMs hidden states to update its beam of parses (compared to models such as RNNGs (Dyer et al., 2016) whose next-word distribution is explicitly conditioned on the stack states). This decoding scheme is shown in Algorithm 1 where k_{action} is the the number of action n-grams to consider between model emissions, and k_{word} is number of parses to consider at word boundaries. During decoding we only consider the valid set of actions for a given parse state given the rules of ArcStandard.

The results of these experiments are shown in Figure 4, where we set $k_{\text{action}} = k_{\text{word}} = 10$. MAP outperforms GAP as expected (since it makes weaker assumptions). Interestingly, while NAP outperforms the other probes in terms of PPL, when evaluated as an incremental parser, it has worse UAS than MAP. This is significant because while MAP makes explicit use of a stack (i.e., the embeddings it uses to make decisions are entirely determined by the stack), NAP does not and instead uses the attention distribution over the prefix at each time step to implicitly encode the stack. We take these results as an important data point in adjudicating between representational hypotheses of incremental syntax in ALMs. Under the assumption that high-performance ALMs are adept incremental parsers—as has been previously demonstrated (Marvin and Linzen, 2018; Hu et al., 2020)—the high performance of MAP suggests our best current model of syntactic parsing in ALMs may be stack-based but not necessarily geometry-based.

Probe performance to an extent implies the *existence* of implicit representaitons of syntax with ALMs. However, it does not necessarily imply that these structures mediate model behavior. In the

[5]That is, even though our incremental-parse probes only predict GEN, action likelihoods in our probes are conditioned on $\texttt{GEN}(w)$ for words in the prefix: $w \in w_{\leq t}$.

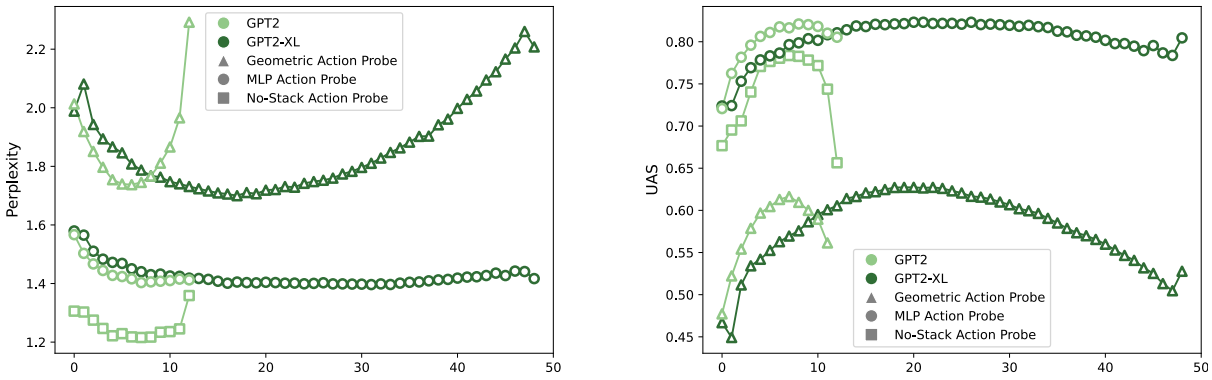

Figure 4: Action perplexity (left) and UAS (right) for each of the incremental-parse probes.

following section, we conduct experiments to see whether we can predict and control model behavior with our probes.

4.2 Probing Incremental Disambiguation

We evaluate our probes to predict model behavior. For this purpose, we extend the dataset of NP/Z ambiguous sentences of Futrell et al. (2019) by augmenting it to include continuations that disambiguate toward the Zero- and NP-complement interpretations (Figure 2 shows an example data point). We also include continuations that are consistent with both parses ("Even though the band left the party *that was raging* [...]") or neither parse (we use the period token, which is ungrammatical in either case, i.e., "Even though the band left the party .") for a more complete comparison of the effects of disambiguation. Concretely, the likelihoods of 'Both' and 'Neither' continuations are expected to increase and stay the same, respectively, regardless of the direction of disambiguation.

Predicting Behavior. First, we replicate Futrell et al. (2019)'s result showing that ALMs are sensitive to verb transitivity and use it to guide next-word predictions, resulting in incremental parser-like behavior. Specifically, we show that changing the verb in NP/Z-ambiguous prefixes from intransitive (unambiguously Z-complement favoring) to transitive (ambiguous), e.g.,:

(1) Even though the band **left** [ambiguous], Even though the band **performed** [unambiguous],

causes ALMs to prefer the continuation consistent with the zero-complement parse. Following Futrell et al. (2019) we use the ALM's surprisal over the words in the continuation[6] to quantify the ALM's expectations, i.e., $S(\text{"went on."}) = -\log P_{\text{model}}(\text{"went on."}|\textit{prefix})$. In Figure 5 we show the change (difference) in surprisal from the unambiguous to the ambiguous case and observe that GPT2 and GPT2-XL both find the Z continuation more surprising in the presence of a transitive verb (Figure 5, right). Somewhat surprisingly, the surprisal of the NP continuation is unchanged by verb transitivity. As expected, the surprisal of the 'Both' and 'Neither' conditions are largely unchanged between conditions.

We find that our probes can predict model behavior in this setting. We target the first action that differentiates the NP and Z parses, which corresponds to the decision to place and arc from the verb to the head of the noun phrase (Figure 2) and plot the difference in surprisal of this decision from the intransitive condition to the transitive condition (Figure 5, left). This probe illustrates a preference for the Z-complement parse in the presence of an intransitive verb which provides representational evidence for incremental parse disambiguation on the part of the model.

In a further study, we present each of the probes with the two NP/Z parses in two conditions: 1) where the sentence suffix was consistent with the parse being probed, and 2) where the sentence suffix was consistent with the other parse. This can be seen as a probabilistic version of Figure 1 (right). We plot the difference in negative log-likelihoods from condition 2 to condition 1 in Figure 6. As expected, we find each of the probes assign higher negative log likelihood (i.e., lower likelihood) to parses when matched with incongruent suffixes. This effect is especially pronounced for the MAP probe.

Editing Parse States. Finally, we use counterfactual analysis (Tucker et al., 2021, 2022) at the level of ALM hidden states to probe the extent to which our probes capture essential information

[6]Note that as the sentence is not fully disambiguated until the period is reached. We include the period token in our surprisal estimates.

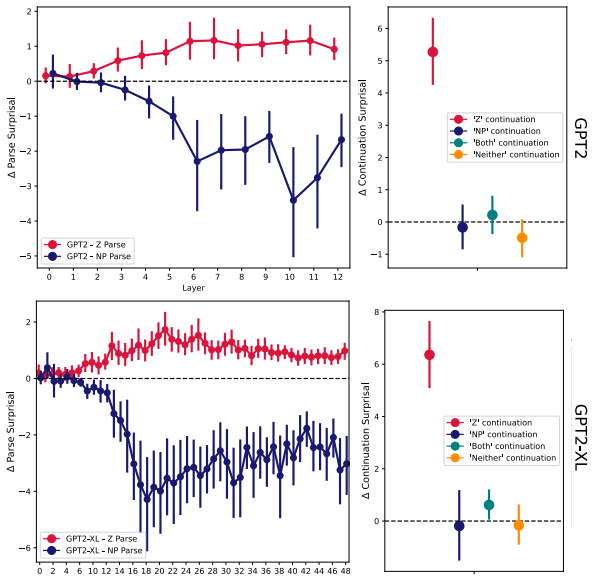

Figure 5: Disambiguation as evaluated by MAP and model behavior. (Right) Difference in *GPT2* surprisal over the continuations in our corpus from the unambiguous condition to the ambiguous condition (higher means the model is more surprised in the unambiguous condition). (Left) Difference in (MAP) *probe* surprisal over the disambiguating parse action. Error bars show 95% confidence intervals across our corpus.

about incremental parses that mediate model behavior. Our approach generates counterfactual ALM hidden states by propagating the loss of a particular parse state ($\mathbf{a}$) from our probe output to model embeddings. Specifically, we iteratively update GPT2 hidden states with the following: $\widehat{\mathbf{h}} = \mathbf{h} + \epsilon \nabla_{\mathbf{h}} P_{\text{probe}}(\mathbf{a} \mid \mathbf{h})$, for a small step size ϵ (see Appendix 7 for further details). Crucially, we perturb towards incremental, incomplete structures before a word is generated, rather than complete structures.

We conjecture that if our probes pick up on essential syntactic information in model embeddings, they should be able to generate model embeddings that produce the expected effect on model behaviour (e.g., perturbing toward the Z-complement parse should make the Z-complement continuation more likely). We generate two sets of counterfactual embeddings (one for each syntactic interpretation; Figure 2) for each sentence in our modified NP/Z data set. We do this for each layer of the models considered for each of our probes and evaluate the effect of these treatments by generating surprisal estimates from each of the counterfactual embeddings.

The results of our analysis are shown in Figure 7. We find that counterfactuals generated with MAP

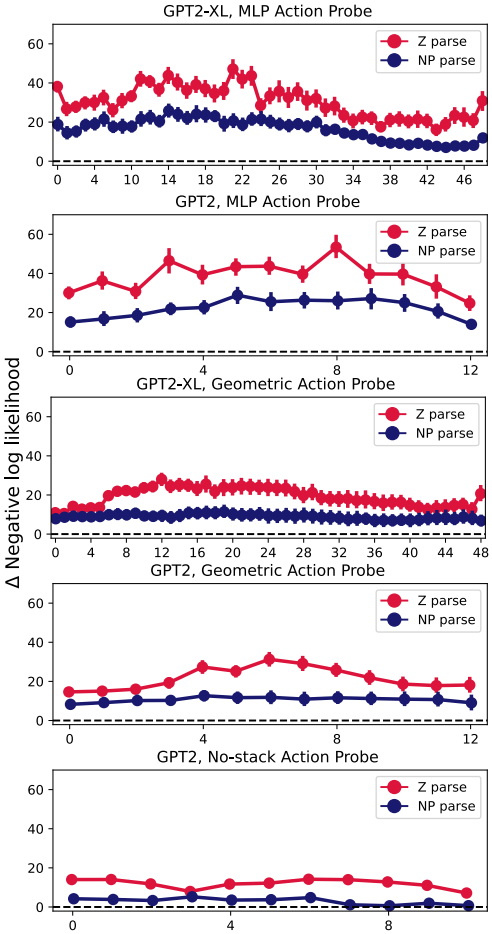

Figure 6: Effect of sentence continuations on parse likelihood. For each of the incremental parse probes, we produce the negative log likelihood for each parse given 1) congruent suffix words and 2) incongruent suffix words (larger is better) and show each probe assigns higher negative log-likelihood to parses when matched with an incongruent suffix (lines above origin in all cases).

produce the predicted effect on both GPT2 and GPT2-XL for the Z- and NP-congruent continuations (but not for the 'Both' or 'Neither' conditions; Appendix A.2). This effect is significant across several layers of both models and is most significant at layer 2 of GPT2 small, where Z-complement-congruent continuations increase in likelihood by ∼8 fold (3 nats) on average after model intervention. We find that counterfactuals generated with the other probes do not mediate next-word predictions in the expected way (Appendix A.2).

5 Related Work

Our work is related to the growing literature quantifying the syntactic sensitivity of neural models of language at behavioral (Linzen and Baroni, 2021; Aina and Linzen, 2021; Lakretz et al., 2021) and representational (Pimentel et al., 2020; Hewitt and

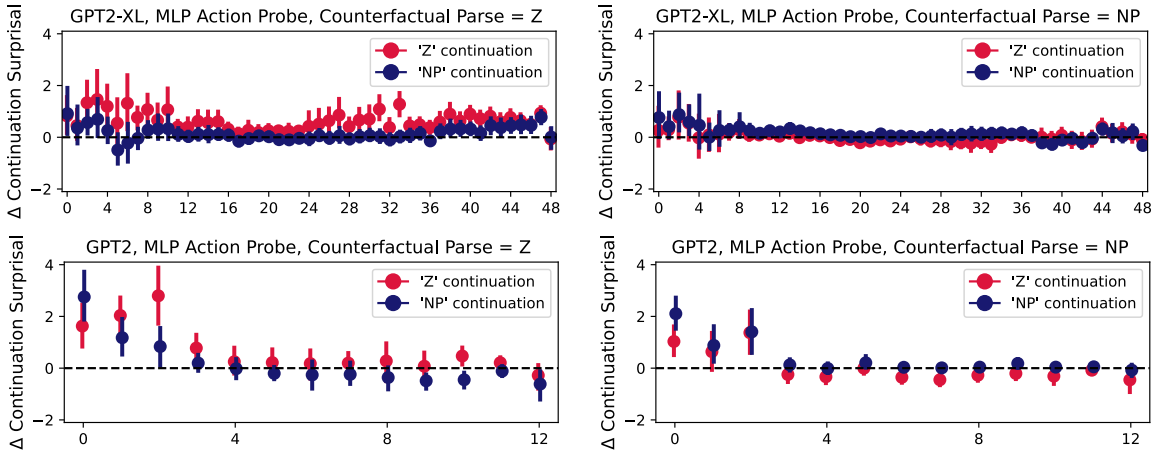

Figure 7: Results of counterfactual syntactic perturbations. The y-axis shows the surprisal difference over the sentence continuations from unperturbed GPT2 to GPT2 perturbed with counterfactual embeddings. We see differences in the expected directions for the Z-complement and NP-complement (i.e., red above blue in the plots on the left and blue above red in the rightward plots) completions for MAP across model. Error bars show 95% confidence intervals across our corpus.

Manning, 2019; Manning et al., 2020; Müller-Eberstein et al., 2022) levels. Of particular note among these studies is the dependency-arc labeling task introduced in Tenney et al. (2019) which bears resemblance to our MAP architecture. Our work extends this literature by developing several new types of syntactic probes in the incremental setting.

Similar to our work but in semantics, Li et al. (2021) use probes to interpret text-encoders as maintaining an evolving semantic information state while interacting with a text-based game. Relatedly, our work is among others that attempt to control neural models of language for counterfactual analysis or controllable AI more generally (De Cao et al., 2021; Ravfogel et al., 2021; Elazar et al., 2020; Meng et al., 2022; Dathathri et al., 2019).

Lastly, we expect our work to be of consequence to computational psycholinguistics, where neural language models and incremental parsers have been historically prominent as candidate cognitive models (Hale et al., 2018; Hale, 2001; Wilcox et al., 2020; Levy, 2008; Jurafsky, 1996; Eisape et al., 2020). This work draws representational parallels between these two models.

6 Conclusion

Motivated by recent work showing human-like incremental parsing behaviour in ALMs, this work extends structural probing to the incremental setting. We find that autoregressive language models can perform on par with bidirectional models in terms of global parsing metrics despite the fact that an ALM's representation cannot condition on future words. We present several hypotheses of the representation of incremental structure in ALMs, instantiate them in probe architectures, and evaluate their explanatory power across a wide range of experiment types: parsing performance, predicting model behavior, and counterfactual analyses. Cumulatively, MAP performs the best, which is significant because this architecture incorporates strong constraints based on linguistic theory (i.e., stack-based representations) while still being relative agnostic (compared to the geometric probe) to the details of how parse probabilities might be encoded. This suggests that despite a lack of explicit feedback, language models not only rediscover linguistic structure through pretraining but learn to make 'inferences' over this structure in real-time comprehension.

7 Limitations

The methods presented here have several limitations. Principally, the question of what constitutes a probe remains a potential confound (Hewitt and Liang, 2019; Belinkov, 2021). While we show several effects that suggest our probes identify meaningful syntactic information, including counterfactual perturbations, our counterfactual effects, though significant for many of the layers of the models considered, are small in absolute terms. Our approach to mitigating these has been to apply a varied set of analyses that, in aggregate, adjudicate between the hypotheses we present here. Finally, our probes rely on a particular oracle—it is possible that action sets from other oracles (e.g.

ArcEager) are better-suited towards representing incremental syntactic structures than ArcStandard.

8 Ethical and Broader Impacts

Language models are increasingly used for tasks beyond language that assume the ability of the language model to structure their input and make 'decisions' in real-time. Our study targets this ability in a basic linguistic phenomenon and thus has the potential to be useful for interpretability and alignment. While the model control aspects of our study pose some risks in that it may enable malicious parties to generate harmful content with language models, the authors believe the benefits in terms of interpretability reasonably balance these.

Acknowledgments

We would like to thank Peng Qian, Mycal Tucker, Jacob Andreas, Josh Tenenbaum, and Ted Gibson for helpful discussions. TE acknowledges support from the GEM consortium and the National Science Foundation Graduate Research Fellowship under Grant No. 1745302. RPL acknowledges support from NSF award BCS-2121074, NIH award U01-NS121471, and a Newton Brain Science Award. YK acknowledges support from MIT-IBM Watson AI lab. Lastly, our codebase is built, in part, on code from open source projects for prepossessing and parsing PTB (Qi and Manning, 2017; Hewitt and Manning, 2019; Noji and Oseki, 2021) as well deploying transformer language models (Paszke et al.; Wolf et al., 2019).

References

Laura Aina and Tal Linzen. 2021. The language model understood the prompt was ambiguous: Probing syntactic uncertainty through generation.

Suhas Arehalli and Tal Linzen. 2020. Neural language models capture some, but not all, agreement attraction effects.

Yonatan Belinkov. 2021. Probing classifiers: Promises, shortcomings, and advances.

Thomas G Bever. 1970. The cognitive basis for linguistic structures. In *Cognition and the development of language*. Wiley.

Charlotte Caucheteux and Jean-Rémi King. 2022. Brains and algorithms partially converge in natural language processing. *Communications Biology*, 5(1):1–10.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation.

Nicola De Cao, Leon Schmid, Dieuwke Hupkes, and Ivan Titov. 2021. Sparse interventions in language models with differentiable masking.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A Smith. 2016. Recurrent neural network grammars.

Tiwalayo Eisape, Noga Zaslavsky, and Roger Levy. 2020. Cloze distillation: Improving neural language models with human Next-Word prediction. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 609–619, Online. Association for Computational Linguistics.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2020. Amnesic probing: Behavioral explanation with amnesic counterfactuals.

William Falcon et al. 2019. Pytorch lightning. *GitHub. Note: https://github. com/PyTorchLightning/pytorch-lightning*, 3(6).

Richard Futrell, Ethan Wilcox, Takashi Morita, Peng Qian, Miguel Ballesteros, and Roger Levy. 2019. Neural language models as psycholinguistic subjects: Representations of syntactic state. 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 32–42, Minneapolis, Minnesota. Association for Computational Linguistics.

John Hale. 2001. A probabilistic Earley parser as a psycholinguistic model. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

John Hale, Chris Dyer, Adhiguna Kuncoro, and Jonathan R Brennan. 2018. Finding syntax in human encephalography with beam search.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks.

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.

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

Daniel Jurafsky. 1996. A probabilistic model of lexical and syntactic access and disambiguation. *Cogn. Sci.*, 20(2):137–194.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Yair Lakretz, Théo Desbordes, Dieuwke Hupkes, and Stanislas Dehaene. 2021. Causal transformers perform below chance on recursive nested constructions, unlike humans.

Roger Levy. 2008. Expectation-based syntactic comprehension. *Cognition*, 106(3):1126–1177.

Belinda Z Li, Maxwell Nye, and Jacob Andreas. 2021. Implicit representations of meaning in neural language models.

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

Christopher D Manning, Kevin Clark, John Hewitt, Urvashi Khandelwal, and Omer Levy. 2020. Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proc. Natl. Acad. Sci. U. S. A.*

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

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual knowledge in GPT.

Max Müller-Eberstein, Rob van der Goot, and Barbara Plank. 2022. Probing for labeled dependency trees.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. pages 50–57.

Hiroshi Noji and Yohei Oseki. 2021. Effective batching for recurrent neural network grammars.

Adam Paszke, Sam Gross, and Soumith Chintala. Automatic differentiation in PyTorch.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-Theoretic probing for linguistic structure.

Peng Qi and Christopher D Manning. 2017. Arc-swift: A novel transition system for dependency parsing.

Peng Qian, Tahira Naseem, Roger Levy, and Ramón Fernandez Astudillo. 2021. Structural guidance for transformer language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3735–3745, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Shauli Ravfogel, Grusha Prasad, Tal Linzen, and Yoav Goldberg. 2021. Counterfactual interventions reveal the causal effect of relative clause representations on agreement prediction.

Martin Schrimpf, Idan Asher Blank, Greta Tuckute, Carina Kauf, Eghbal A Hosseini, Nancy Kanwisher, Joshua B Tenenbaum, and Evelina Fedorenko. 2021. The neural architecture of language: Integrative modeling converges on predictive processing. *Proc. Natl. Acad. Sci. U. S. A.*, 118(45).

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.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations.

Mycal Tucker, Tiwalayo Eisape, Peng Qian, Roger Levy, and Julie Shah. 2022. When does syntax mediate neural language model performance? evidence from dropout probes.

Mycal Tucker, Peng Qian, and Roger Levy. 2021. What if this modified that? syntactic interventions with counterfactual embeddings. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alex Warstadt and Samuel R Bowman. 2020. Can neural networks acquire a structural bias from raw linguistic data?

Ethan Wilcox, Peng Qian, Richard Futrell, Miguel Ballesteros, and Roger Levy. 2019. Structural supervision improves learning of Non-Local grammatical dependencies.

Ethan Wilcox, Pranali Vani, and Roger P Levy. 2021. A targeted assessment of incremental processing in neural LanguageModels and humans. *arXiv e-prints*, page arXiv:2106.03232.

Ethan Gotlieb Wilcox, Jon Gauthier, Jennifer Hu, Peng Qian, and Roger Levy. 2020. On the predictive power of neural language models for human Real-Time comprehension behavior.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M Rush. 2019. HuggingFace's transformers: State-of-the-art natural language processing.

A Appendix

A.1 Training

Our probes are built in PyTorch (Paszke et al.). All probes were trained with Adam (Kingma and Ba, 2014) with a learning rate of 10^{-3} except for the GAP, which used a learning rate of 10^{-5} after pretraining on the regression task of Hewitt and Manning (2019). For multilayer architectures, a dropout rate of 0.2 was used between layers (3 layers), and hidden state size was fixed to be the same size as the model emission. The attentive probe was implemented using a GRU of 200 hidden units, biaffine attention, and a 3 layer MLP read-out network. All architectures were optimized in PyTorch lightning (Falcon et al., 2019). Each model trained for at most 40 each but used early stopping with a plateau tolerance of 3.

A.2 Counterfactual details

We find training our probes with a high dropout rate (Tucker et al., 2022) improves counterfactual results. Thus, we train separate checkpoints for evaluation (0 dropout on model embeddings) and counterfactual generation (0.4 dropout). We found MAP achieved the largest counterfactual effect size in the predicted direction when evaluating NP- and Z-parse consistent completions. It had less interpretable effects on the 'Both' and 'Neither' conditions. We include the full range of experiments in Figures 8 and 9.

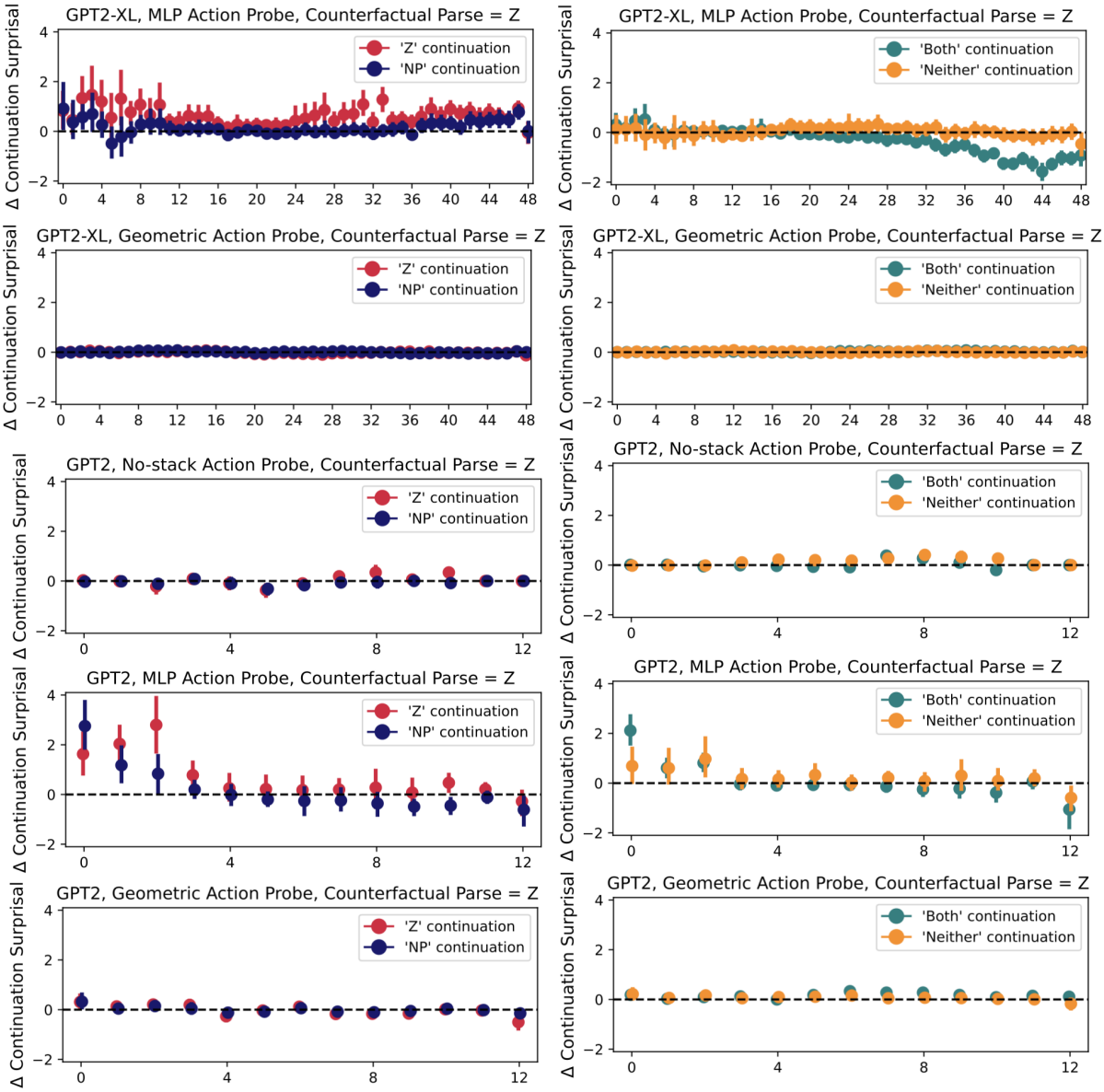

Figure 8: Extended results of counterfactual analyses.

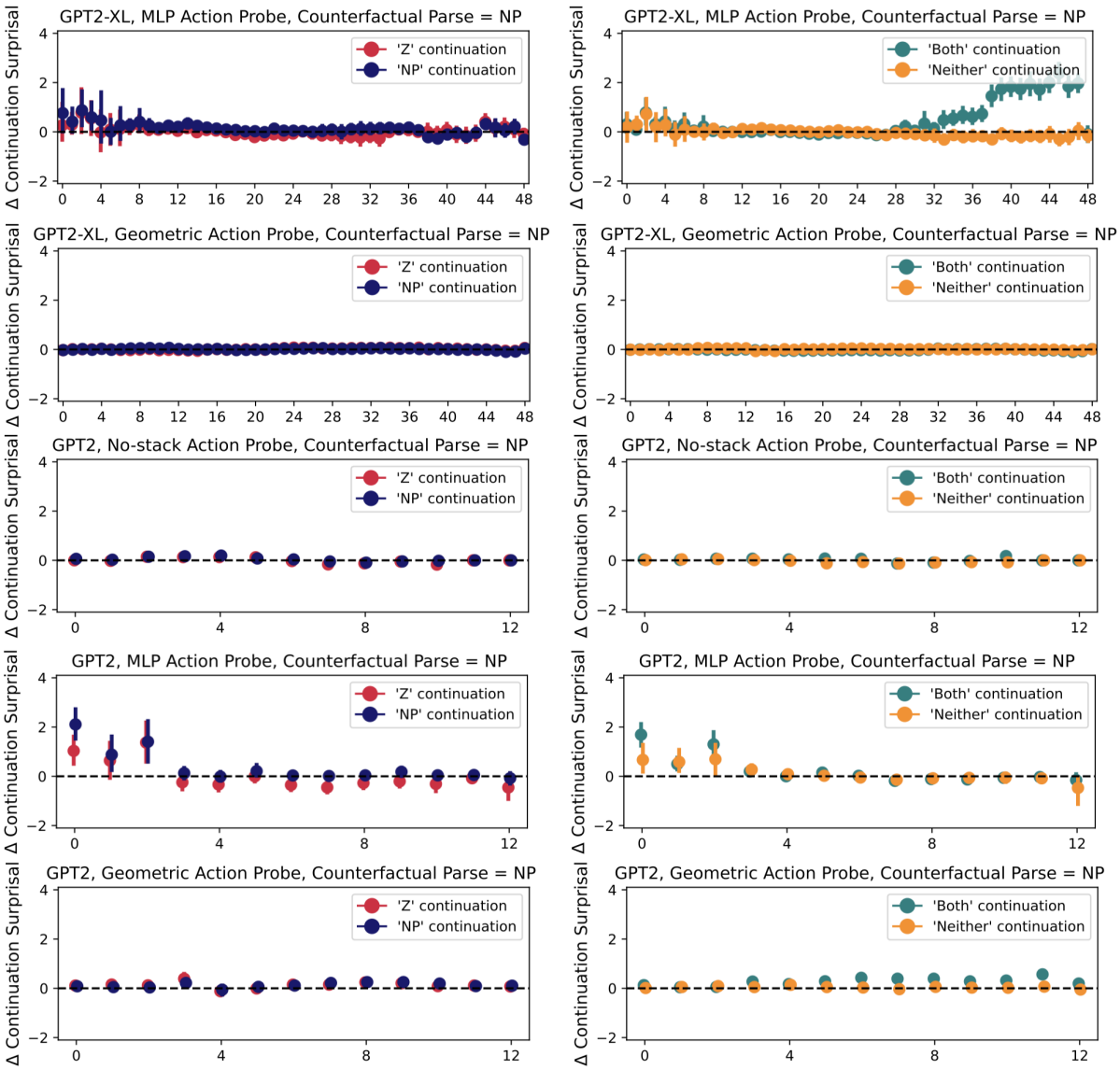

Figure 9: Extended results of counterfactual analyses