Cognitive Modeling with Scaffolded LLMs: A Case Study of Referential Expression Generation

Polina Tsvilodub [1] **Michael Franke** [1] **Fausto Carcassi** [2]

Abstract

To what extent can LLMs be used as part of a cognitive model of language generation? In this paper, we approach this question by exploring a neuro-symbolic implementation of an algorithmic cognitive model of referential expression generation by Dale & Reiter (1995). The symbolic task analysis implements the generation as an iterative procedure that scaffolds symbolic and `gpt-3.5-turbo`-based modules. We compare this implementation to an ablated model and a one-shot LLM-only baseline on the A3DS dataset (Tsvilodub & Franke, 2023). We find that our hybrid approach is cognitively plausible and performs well in complex contexts, while allowing for more open-ended modeling of language generation in a larger domain.

1. Introduction

Large language models (LLMs) have shown impressive performance on different benchmarks on a variety of tasks (e.g., Brown et al., 2020; Bommasani et al., 2021; Chowdhery et al., 2022; Touvron et al., 2023). Recently, LLMs have been embedded within larger systems, also called LLM *agents*, which can for instance retrieve relevant information (Lewis et al., 2020; Liu et al., 2022), make use of additional computational components for math problem solving (He-Yueya et al., 2023), more complex reasoning tasks (Creswell et al., 2022; He-Yueya et al., 2023; Paranjape et al., 2023; Poesia et al., 2023), programming tasks (Gao et al., 2022), or generating better texts through additional computational steps (Piriyakulkij et al., 2023). Other recent work has taken a more cognitively inspired perspective using LLMs in hybrid, neuro-symbolic models for extending explanatory cognitive models (e.g., Lew et al., 2020) or as part of cognitive architectures (e.g., Sumers et al., 2023; Wong et al., 2023). Here, rather than focusing on LLM performance we take the perspective of cognitive scientists, focusing on task analysis with the goal of building computational models of cognitive processes. In this perspective, we pose the question: To what extent can LLMs be used as components in implementation of general algorithmic models of cognitive processes?

We focus on linguistic cognition and pick out a minimal non-trivial case study of *referential expression* generation in a *contrastive reference game task*. This task requires the production of a description to identify a *target* referent among a set of possible alternative *distractor* referents. Several prominent approaches in computational linguistics and cognitive science treat reference games, essentially, as a mapping problem. Given a set of distractors, the target, and a fixed list of expressions (with a known semantics), the task is construed as selecting the best, or a good enough, expression to describe the target (Kramer & van Deemter, 2012; Frank & Goodman, 2012). While these approaches have successfully captured that cooperative speakers exploit pragmatic reasoning rooted in Gricean Maxims to convey the intended content while keeping utterances brief (Grice, 1975), they require a pre-specified list of, or a construction procedure for, the descriptions to select from, which essentially reduces the task to the problem of mapping utterances onto states. This "closed world" problem is shared by many instances of whole classes of approaches, including many types of Bayesian models or model-based reinforcement learning.

Several neuro-symbolic models for reference games have been proposed to overcome this limitation. Some use pragmatic agent models to learn suitable semantic meaning representations (Monroe & Potts, 2015; Ohmer et al., 2021). Others train neural networks to perform well on the problem of in-context discrimination (Mao et al., 2016; Hendricks et al., 2016). More cognitively inspired models build on extant probabilistic models (Frank & Goodman, 2012), adding modules for neural language generation (Andreas & Klein, 2016; Vedantam et al., 2017; Cohn-Gordon et al., 2018; Nie et al., 2020; Zarrieß & Schlangen, 2019). However, all these

[1] Department of Linguistics, University of Tübingen, Tübingen, Germany [2] ILLC, University of Amsterdam, Amsterdam, Netherlands. Correspondence to: Polina Tsvilodub <polina.tsvilodub@uni-tuebingen.de>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

require task-specific training or appropriately pre-trained base models. Moreover, even if inspired by extant cognitive models, they often do not aspire to remain faithful to the original in favor of achieving higher task accuracy.

In this work, we explore how successfully modern LLMs can be used to overcome this "closed-world" problem for known algorithmic solutions to the reference game task. Specifically, we consider a simple algorithmic idea from Dale & Reiter (1995) and extend it with LLM modules to make it more widely applicable without requiring hand-specified, closed sets of utterance alternatives. We critically assess this neuro-symbolic model against an ablation and a baseline.

2. Model & Experiments

2.1. Iterative Model

We build on the Incremental Algorithm (IA) by Dale & Reiter (1995), in which a referential expression is constructed by iterating through a pre-specified list of attributes and adding each attribute that applies to the target and rules out at least one distractor. While the IA is relatively simple and more performant solutions exist, ordering the attributes according to human preferences is psycholinguistically motivated. Moreover, the general idea of incremental search algorithms has a long history (Newell & Simon, 1972), demonstrable advantages also for LLM-based architectures (Yao et al., 2023), and is supported in the particular domain of language generation (Ferreira, 2019).

The IA of Dale & Reiter (1995) requires *a priori* specification of (i) the order in which the attributes are considered, (ii) all possible states and their attribute values, (iii) a procedure for utterance construction, as well as (iv) semantics of utterances applied to any state, all of which may be domain-specific. To address these limitations, we propose a variation of the IA which combines the IA's symbolic computations with LLM generation.

Our *iterative model* (IM) is illustrated in Figure 1 (IM; see Algorithm 1 in the appendix for full details). The model takes as input a context which consists of a target state s^* and one or more distractors D. It outputs a contrastive description for s^*. It uses an LLM for (i) proposing descriptions (module: *UtterancesProposer*) and (ii) evaluating whether a proposed description is semantically compatible with any state (target or distractor, module: *SemanticEvaluator*).[1] Symbolic components subsequently (i) check for contrastivity of each proposed description, and (ii) potentially iterate the procedure if no fully contrastive description has been found.

Contrastivity is defined as the proportion of distractors for which the utterance is false. For instance, if there are two or four distractors and the utterance is false of one, the contrastivity is 0.5 or 0.75, respectively (see Fig. 1). At each iteration, if none of the utterances is fully contrastive (i.e., uniquely identifies the target), the symbolic component selects the most informative utterances available (i.e., the most contrastive ones), and passes them to the UtterancesProposer again, which is prompted to add some detail about the target to the utterance based on the previous utterance, and starts a new iteration.[2] This is repeated until an utterance is found that solves the task or the maximal number of iterations (in our simulations, five) is achieved. The most contrastive utterance is greedily selected and returned.

An example of a pass through the IM can be found in Figure 1 (to be read counter-clockwise, starting at the top and ending at the bottom right). Example inputs for one target and two distractors are shown in the top box. The first iteration starts with the target description (marked with "T") being passed to the UtterancesProposer (A.1.1A/B); example outputs are shown in the next box (e.g., "The floor is purple"). The sampled utterances are then passed to the SemanticEvaluator (A.1.3, shown as columns) evaluated for all states (shown as rows). Results of evaluating each utterance for each state are shown in the cells, with column C showing the contrastivity values. The more contrastive utterances (i.e., those setting the target apart from one distractor) are not fully contrastive, so they are passed to the UtterancesProposer (A.1.1A/B) again, and the model is prompted to generate more detailed samples (e.g., "The floor is purple and the wall is green"). The second iteration through the model is computed and since a fully contrastive extended utterance is found (i.e., with $C = 1$, two utterances in this example), it is returned as the final expression (A.1.4).

The IM implements a search over a tree of possible referential utterances (cf., Yao et al., 2023), which has been a long-standing approach in AI (van Rooij et al., 2023). Since each iteration adds a single detail about the target, in an order proposed by the LLM itself rather than a manually-specified order like in the original IA, the tree depth roughly corresponds to the number of details included in the generated utterance, while the width corresponds to the number of sampled utterance proposals on each iteration. The IM implements a partial search, by passing only the currently most contrastive utterances to the subsequent iteration.

[1] For all LLM components and the baseline, GPT-3.5 (`gpt-3.5-turbo`, checkpoints of summer 2023) with temperature $\tau = 0.1$ was used. All LLM prompts and details on functionality and evaluation of the single modules can be found in the Appendix A.

[2] Note that distractors are not taken into account when extending the description of the target. Instead, we consider only the utterances produced so far along with the target. This potentially makes the module more task-agnostic and reusable for future applications. Note that the UtterancesProposer in Alg. 1 uses a different initial prompt for the first iteration; see App. A.1.1.

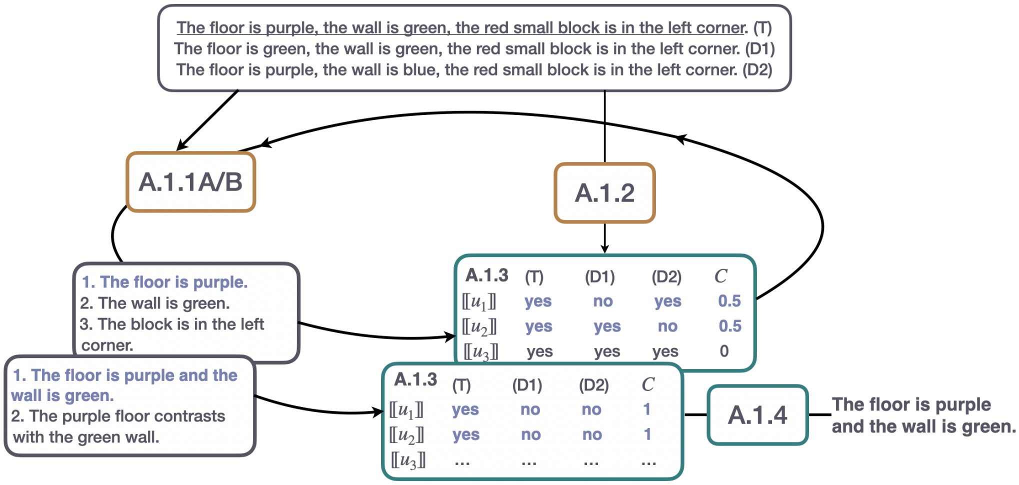
The floor is purple, the wall is green, the red small block is in the left corner. (T)
The floor is green, the wall is green, the red small block is in the left corner. (D1)
The floor is purple, the wall is blue, the red small block is in the left corner. (D2)
A.1.1A/B
A.1.2
1. The floor is purple.
2. The wall is green.
3. The block is in the left corner.
A.1.3 (T) (D1) (D2) C
$[\![u_1]\!]$ yes no yes 0.5
$[\![u_2]\!]$ yes yes no 0.5
$[\![u_3]\!]$ yes yes yes 0
1. The floor is purple and the wall is green.
2. The purple floor contrasts with the green wall.
A.1.3 (T) (D1) (D2) C
$[\![u_1]\!]$ yes no no 1
$[\![u_2]\!]$ yes no no 1
$[\![u_3]\!]$ ... ...
A.1.4
The floor is purple and the wall is green.

Figure 1. The figure shows two iterations of the model, ending with the production of a contrastive utterance. T, D1, and D2 denote the target and the two distractor states respectively. C indicates the contrastivity values. Only the target is passed to the utterance proposer. Boxes with a brown border indicate LLM-based modules, components with green symbolic modules. The labels of the modules indicate section numbers in the appendix containing the full details.

The modules used for such computational models might generally fall into *functionally different types* (cf., Sumers et al., 2023). *Evaluators* provide context-dependent assessment of alternatives, *proposers* supply these possible alternatives or contingencies (e.g., plausible utterances for a given context), and symbolic or simply rule-based (as in this case) modules which can generally be called *selectors* combine and process information supplied by the other two types of modules. Given their context-dependence, the first two types will often be neural.

2.2. Ablated Model

To test the impact of iteration in the IM, we compare it to an ablated *single-pass* model (SP; Algorithm 2), which samples utterances, evaluates them and selects the best alternative in a single pass, without iteration. We expect the performance of the SP model to scale with the number of sampled utterances. Maintaining a very large number of alternatives may be cognitively implausible due to resource constraints, lending credence to more iterative approaches (Ferreira, 2019).

2.3. Baseline

We compare the results of the two neuro-symbolic models (IM and SP) with a baseline model. The baseline consists of a single call to the LLM asking for an utterance that solves the task. We use a one-shot chain-of-thought prompt, shown in Appendix A.3, (Wei et al., 2022, among others).

2.4. Simulations

We test the models for reference games based on a derivative of the 3Dshapes dataset (Burgess & Kim, 2018), called A3DS (Tsvilodub & Franke, 2023). A3DS contains textual descriptions of scenes consisting of a 3D geometric object in an otherwise empty room. All scenes in the used subset are unique and consist of a combination of the following attributes: shape of the object (four values); size of the object (three values); color of the wall, the floor, the object itself (independent of each other, seven possible values); object position relative to the background (three values). The scene descriptions used as input to the models were of the form “The floor is {floor color}, the wall is {wall color}, the {object color} {size} {object} is in the {position}.” Example inputs are shown in Figure 1 (top box). This state-space is highly structured and would, in principle, allow the specification of a large description set with a grammar and compositional semantic rules. We use this data set not because it provides an insurmountable obstacle to the original IA, but rather because it simplifies the evaluation of the models’ outputs, allowing us to automatically verify whether a generated utterance mentions contrastive features and calculate the overall contrastivity for a given context, without human annotation.

To compare the performance of the IM, SP and baseline models, we construct reference games by first sampling a target state, and then adding one, four or eight distinct distractors. Each distractor differs from the target by maximally two features. In the IM, the UtterancesProposer sampled either four or eight utterances. For the SP, the UtterancesProposer always sampled ten utterances. We tested all models on 100 reference games for each of the parameter configurations.

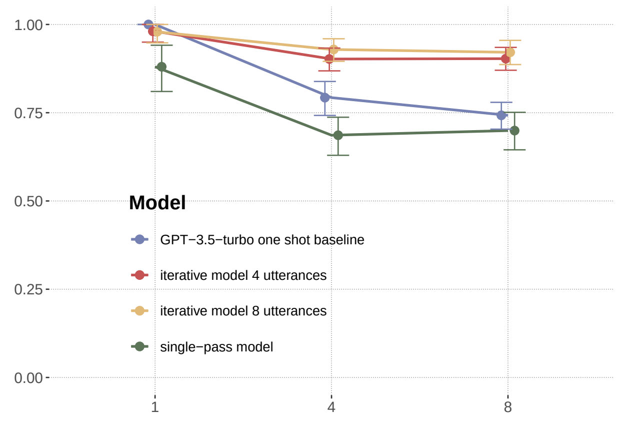

Figure 2. Distribution over contrastivity values (y-axis) by number of distractors (x-axis) and number of utterances proposed (color). Error bars show bootstrapped 95%-CIs.

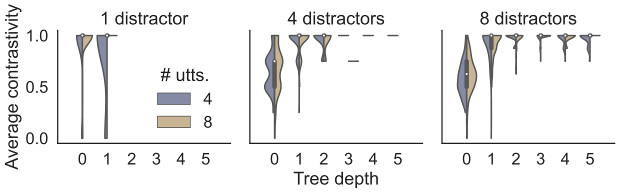

Figure 3. Development of task success over increasing tree depth in the IM: distribution over contrastivity values (y-axis) over increasing tree depth (extended utterance proposal and evaluation iterations; x-axis), by number of distractors (facets) and tree width (number of proposed utterances; color). Dots indicate means, thick bars indicate quartiles, thinner lines indicate minimal values.

2.5. Results

Model performance is measured by average contrastivity across references games (Figure 2). The IM generated highly contrastive utterances which successfully set apart the target from the distractors, even with more distractors, outperforming the baseline and the SP. The average contrastivity of the IM for four and eight distractors was above the baseline (bootstrapped $P = 1$), while the difference was not significant for one distractor due to a ceiling effect. The performance of the ablated SP model decreased as the number of distractors in the context increased. The performance of the ablated SP model was credibly worse than all other models, across the number of distractors (bootstrapped $P = 1$). The LLM baseline outperformed the SP.

The number of iterations required for the IM until fully contrastive utterances were produced increased with the number of distractors and thus with the difficulty of identifying contrastive features (Figure 3). In particular, this shows that the IM increased the complexity of the computation and of the generated utterances in a context-dependent way. Furthermore, Figure 3 suggests a slight trade-off between the tree width and tree depth required for producing contrastive utterances (x-axis vs. color): when more utterances were proposed at each step, it was more likely that at least one of them mentioned contrastive features, so that fewer iterations were required overall.

3. Discussion

In this paper, we focused on contrastive utterance generation in a reference game setting. We implemented an iterative neuro-symbolic model based on the IA (Dale & Reiter, 1995). The model adapts the generated utterances to the complexity of the task at hand, while producing diverse language that is natural in context, which is enabled by the performance of LLMs. We found that the IM outperforms both an ablated single-pass model and an LLM baseline in complex reference games.

The proposed approach has certain advantages over approaches requiring fine-tuning. For instance, Tsvilodub & Franke (2023) used the A3DS data set to fine-tune a pre-trained image captioner to produce contrastive captions via reinforcement learning. In a comparable test setting (one distractor), the pragmatic model only achieved an accuracy of 0.54. Although directly conditioned on visual input rather than "oracle" text input, it required costly fine-tuning and did not match the performance shown here. In contrast, the considered neuro-symbolic approach didn't require fine-tuning, and showed better performance, while being agnostic to the neural model. Additionally, the IM model offers an algorithmic cognitively motivated (symbolic) task decomposition which requires spelling out assumptions about the modeled process, which scaffolds the black-box LLM modules. Such a task analysis allows for easy explorations of a given model via plug-and-play with the modules. For instance, other search algorithms could be considered, which would also allow to explore more computationally efficient solutions (e.g., via cognitively plausible search heuristics, or implementing amortization of the reasoning, cf. Gershman & Goodman, 2014). Further, likelihood estimates supplied by the LLMs could be used for evaluations instead of the sampling based approximation here (cf., Hu & Levy, 2023). This would allow for quantitative model comparison (Lee & Wagenmakers, 2015; Franke et al., 2024). Finally, neural modules implementing specific subtasks can be reused across different cognitive models. For instance, the UtterancesProposer module could be reused for sampling plausible utterances in other tasks or contexts. Repeatedly deployed and evaluated modules could over time accumulate into a toolbox of well-tested modules for cognitive modeling. Modularity is cognitive motivated (e.g., Sperber, 2001), and additionally modules for common sub-tasks could be fine-tuned, which might become increasingly important as more agent models and cognitive architectures are deployed (cf. Sumers et al., 2023).

In sum, we take this case study to be an informative starting point outlining some methodological reference points for further investigating the potential of explanatory cognitive models augmented with neural modules within the toolbox of cognitive scientists.

Acknowledgements

We gratefully acknowledge support by the state of Baden-Württemberg, Germany, through the computing resources provided by bwHPC and the German Research Foundation (DFG) through grant INST 35/1597-1 FUGG. MF is a member of the Machine Learning Cluster of Excellence, EXC number 2064/1 – Project number 39072764.

Impact Statement / Limitations

Our approach also has various limitations. The model crucially depends on the quality of the LLM-based modules, especially since small errors in early iterations might be accumulated and propagated. For example, we observed that evaluation of literal semantics should be critically assessed, and SOTA LLMs tend towards verbose generations without explicit instructions (see Appendix A.1.1, A.1.2 for details). Manual inspection of the model outputs revealed that, when approaching the maximal number of iterations, utterances sometimes became more repetitive, but remained true. LLM modules also inherit issues of LLMs, such as excessive sensitivity to apparently minor changes to the prompt, uninterpretability, hallucinations and biases (Bender et al., 2021; Ji et al., 2023; Liu et al., 2023; Shi et al., 2023; Zhao et al., 2023). Further, they do not perform equally well across different domains (e.g., Ahn et al., 2024); modeling language cognition with such hybrid models where LLMs are employed "in-distribution" might therefore be a particularly natural starting point. An important goal for future work is also to analyze the performance of the modules backed by different (open-source) LLMs; this case study can be seen as a reference point of performance with one of the best available closed-source models.

References

Ahn, J., Verma, R., Lou, R., Liu, D., Zhang, R., and Yin, W. Large language models for mathematical reasoning: Progresses and challenges. In Falk, N., Papi, S., and Zhang, M. (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 225–237, St. Julian's, Malta, March 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.eacl-srw.17.

Andreas, J. and Klein, D. Reasoning about pragmatics with neural listeners and speakers. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1173–1182, Austin, TX, 2016. Association for Computational Linguistics.

Bender, E. M., Gebru, T., McMillan-Major, A., and Mitchell, M. On the dangers of stochastic parrots:

Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, pp. 610–623, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383097. doi: 10.1145/3442188.3445922. URL https://doi.org/10.1145/3442188.3445922.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. A large annotated corpus for learning natural language inference. In Màrquez, L., Callison-Burch, C., and Su, J. (eds.), *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1075. URL https://aclanthology.org/D15-1075.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020.

Burgess, C. and Kim, H. 3d shapes dataset, 2018.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Cohn-Gordon, R., Goodman, N. D., and Potts, C. Pragmatically informative image captioning with character-level inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 439–443, Stroudsburg, PA, June 2018. Association for Computational Linguistics.

Creswell, A., Shanahan, M., and Higgins, I. Selection-inference: Exploiting large language models for interpretable logical reasoning. *arXiv preprint arXiv:2205.09712*, 2022.

Dale, R. and Reiter, E. Computational interpretations of the gricean maxims in the generation of referring expressions. *Cognitive science*, 19(2):233–263, 1995.

Ferreira, V. S. A mechanistic framework for explaining audience design in language production. *Annual Review of Psychology*, 70(1):29–51, 2019.

Frank, M. C. and Goodman, N. D. Predicting pragmatic reasoning in language games. *Science*, 336(6084):998–998, 2012.

Franke, M., Tsvilodub, P., and Carcassi, F. Bayesian Statistical Modeling with Predictors from LLMs, 2024.

Gao, L., Madaan, A., Zhou, S., Alon, U., Liu, P., Yang, Y., Callan, J., and Neubig, G. Pal: Program-aided language models. *arXiv preprint arXiv:2211.10435*, 2022.

Gershman, S. and Goodman, N. Amortized inference in probabilistic reasoning. In *Proceedings of the annual meeting of the cognitive science society*, volume 36, 2014.

Grice, H. P. Logic and conversation. In *Speech acts*, pp. 41–58. Brill, 1975.

He-Yueya, J., Poesia, G., Wang, R. E., and Goodman, N. D. Solving math word problems by combining language models with symbolic solvers. *arXiv preprint arXiv:2304.09102*, 2023.

Hendricks, L. A., Akata, Z., Rohrbach, M., Donahue, J., Schiele, B., and Darrell, T. Generating visual explanations. In Leibe, B., Matas, J., Sebe, N., and Welling, M. (eds.), *Computer Vision – ECCV 2016*, pp. 3–19, Cham, 2016. Springer International Publishing.

Hu, J. and Levy, R. Prompt-based methods may underestimate large language models' linguistic generalizations. *arXiv preprint arXiv:2305.13264*, 2023.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12), mar 2023. ISSN 0360-0300. doi: 10.1145/3571730. URL https://doi.org/10.1145/3571730.

Kramer, E. and van Deemter, K. Computational generation of referring expressions: A survey. *Computational Linguistics*, 38(1):173–218, 2012.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lee, M. D. and Wagenmakers, E.-J. *Bayesian Cognitive Modeling: A Practical Course*. Cambridge University Press, Cambridge, MA, 2015.

Lew, A. K., Tessler, M. H., Mansinghka, V. K., and Tenenbaum, J. B. Leveraging unstructured statistical knowledge in a probabilistic language of thought. In *Proceedings of the annual conference of the cognitive science society*, 2020.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

Liu, J., Liu, A., Lu, X., Welleck, S., West, P., Le Bras, R., Choi, Y., and Hajishirzi, H. Generated knowledge prompting for commonsense reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3154–3169, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.225. URL https://aclanthology.org/2022.acl-long.225.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. Lost in the middle: How language models use long contexts, 2023.

Mao, J., Huang, J., Toshev, A., Camburu, O., Yuille, A., and Murphy, K. Generation and comprehension of unambiguous object descriptions. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11–20, 2016.

Monroe, W. and Potts, C. Learning in the Rational Speech Acts model. In *Proceedings of 20th Amsterdam Colloquium*, Amsterdam, December 2015. ILLC.

Moskal, M., Musuvathi, M., and Kıcıman, E. AI Controller Interface. https://github.com/microsoft/aici/, 2024.

Newell, A. and Simon, H. A. *Human problem solving*. Prentice-Hall, Englewood Cliffs, NJ, 1972.

Nie, A., Cohn-Gordon, R., and Potts, C. Pragmatic issue-sensitive image captioning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1924–1938, Online, November 2020. Association for Computational Linguistics.

Ohmer, X., Franke, M., and König, P. Mutual exclusivity in pragmatic agents. *Cognitive Science*, 46(1):e13069, 2021.

Paranjape, B., Lundberg, S., Singh, S., Hajishirzi, H., Zettlemoyer, L., and Ribeiro, M. T. Art: Automatic multi-step reasoning and tool-use for large language models. *arXiv preprint arXiv:2303.09014*, 2023.

Piriyakulkij, T., Kuleshov, V., and Ellis, K. Asking clarifying questions using language models and probabilistic reasoning. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023. URL https://openreview.net/forum?id=2SjoG6lVz3.

Poesia, G., Gandhi, K., Zelikman, E., and Goodman, N. D. Certified deductive reasoning with language models, 2023.

Shi, F., Chen, X., Misra, K., Scales, N., Dohan, D., Chi, E., Schärli, N., and Zhou, D. Large language models can be easily distracted by irrelevant context, 2023.

Sperber, D. In defense of massive modularity. In Dupoux, E. (ed.), *Language, Brain and Cognitive Development: Essays in Honor of Jacques Mehler*. MIT Press, 2001.

Sumers, T., Yao, S., Narasimhan, K., and Griffiths, T. L. Cognitive architectures for language agents. *arXiv preprint arXiv:2309.02427*, 2023.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Tsvilodub, P. and Franke, M. Evaluating pragmatic abilities of image captioners on A3DS. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1277–1285, Toronto, Canada, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-short.110. URL https://aclanthology.org/2023.acl-short.110.

van Rooij, I., Guest, O., Adolfi, F. G., de Haan, R., Kolokolova, A., and Rich, P. Reclaiming AI as a theoretical tool for cognitive science. 2023.

Vedantam, R., Bengio, S., Murphy, K., Parikh, D., and Chechik, G. Context-aware captions from context-agnostic supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 251–260, 2017.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., and Zhou, D. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

Wong, L., Grand, G., Lew, A. K., Goodman, N. D., Mansinghka, V. K., Andreas, J., and Tenenbaum, J. B. From word models to world models: Translating from natural language to the probabilistic language of thought, 2023.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.

Zarrieß, S. and Schlangen, D. Know what you don't know: Modeling a pragmatic speaker that refers to objects of unknown categories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 654–659. Association for Computational Linguistics, 2019.

Zhao, H., Chen, H., Yang, F., Liu, N., Deng, H., Cai, H., Wang, S., Yin, D., and Du, M. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 2023.

Algorithm 1 Iterative model. Components in brown are implemented as LLM modules in this case study, components in green are symbolic.

```
Generate($s^*$, $D$, $n$):
partialUtt $\leftarrow$ []
while True do
    for $u' \in$ partialUtt do
        $U \leftarrow$ UtterancesProposer$(s^*, n, u')$
    end for
    $C_{\text{new}} \leftarrow$ []
    $T \leftarrow$ []
    for $u \in U$ do
        append $T_u$ = SemanticEvaluator$(\{s^*\} \cup D, u)$ to $T$
        append ConstrastivitySelector$(T_u)$ to $C_{\text{new}}$
    end for
    indices $\leftarrow i \mid c_i \in \max(C_{\text{new}})$
    $U^* \leftarrow U$[indices]
    $T^* \leftarrow T$[indices]
    if $\max(C_{\text{new}}) = 1$ then
        $u^* \leftarrow$ InfoMaxSelector$(T^*, U^*)$
        return $u^*$
    end if
    for $u \in U^*$ do
        partialUtt $\leftarrow U^*$
        $U \leftarrow$ []
    end for
end while
```

A. Implementation and Evaluation Details

A.1. Iterative Model Details

More technically, IM (Algorithm 1) takes as input a list of full state descriptions, one of which is the target state and the remaining ones are distractors. The target is always passed as the first state. First, the (LLM-based) UtterancesProposer generates candidate utterances that describe a single detail of the target state based on the target state description. Second, the SemanticEvaluator determines the (literal) truth value of all candidate utterances for each state (target and distractors). Third, based on the semantic evaluation in the previous step, the ContrastivitySelector evaluates the contrastivity of the generated utterances, and determines whether any utterance is fully contrastive (i.e., only true of the target). For each utterance u_i, this is computed as $C_i = 1 - \frac{\#\text{distractors for which } [\![u_i]\!](d)=1}{\#\text{distractors}}$. A fully contrastive utterance has $C_i = 1$. If utterances with $C_i = 1$ are found, one utterance is greedily chosen among the contrastive utterances by the InfoMaxSelector and returned.

Otherwise, a set of the so far most contrastive utterances is constructed, and the passed to the UtterancesProposer module again. This module produces new alternatives for each utterance, each of which includes one more detail from the full description. The loop repeats from the semantic evaluation on until a fully contrastive utterance is produced or the maximal iteration steps have been reached, in which case the InfoMaxSelector greedily selects the most contrastive among the utterances produced in the most recent iteration.

A.1.1. UTTERANCESPROPOSER

On the first iteration of the model, there are no previous utterances yet. We prompt the LLM to generate initial utterances which should only mention a single feature of the target. This prompt can be considered as simulating a production cost pressure, which is present for humans and usually included in cognitive models, but absent in LLMs. Based on our explorations, we observe that for `gpt-3.5-turbo` the prompt needs to explicitly instruct the LLM to NOT generate exhaustive utterances. We used the following prompt (A) for the first iteration:

```
You will be given a target
description.  Please produce
{num_samples} sentence(s) that
only mention one detail from
the target description.  The
produced sentences should
include exclusively content
mentioned in the target.  Please
provide the sentences in a
bullet list format.

Target:  {target state}
Sentences:
```

`num_samples` was set to either four or eight in our simulation.

For further iterations of the model, the prompt (B) was adapted to also include previously generated utterances:

```
Your task is to produce some
sentences.  Each sentence
should repeat the information
in "{partial_description}", but
add one more detail taken from
"{full_description}" Do not make
up any new detail.

Please produce {num_samples}
sentence(s) in a bullet list
format.  Be very concise!

Sentences:
```

Evaluation Samples from all of the reference game were manually inspected by the authors and no grammatical errors were observed. It was observed that in some of the samples more than one feature of the target was added in one iteration. For instance, the object type (i.e,. the shape) was mentioned in addition to some other new feature, akin to a basic-level bias.

A.1.2. SEMANTICEVALUATOR

The purpose of the SemanticEvaluator is to check whether the candidate utterances can be used to refer to each of the states (target and distractors). The task is cast as checking the utterance semantics and checking if the utterance is literally true of the state. Semantic evaluation is applied to each utterance-state combination.

Based on a state description, an utterance and a prompt, the LLM is instructed to answer the question contained in the prompt with ‘yes’/‘no’ and output a chain of thought. The generated response is processed with a regular expression to extract ‘yes’/‘no’ and convert these to 1, 0, respectively. We note that since the time of implementation, new methods for retrieving LLM generations that fit a specific format and contain specific values have been developed (e.g., Moskal et al., 2024); such tools could be of great use for future implementations of such modules.

To exploit the model’s knowledge of intuitive language use, the prompt chosen to determine semantic compatibility can be seen as implementing an *intuitive entailment* task. The prompt was:

```
Consider the following sentence:
{state}

Does the following statement
provide exclusively information
also contained in the sentence
above:  {utterance}

Explain your answer step by
step.

Importantly, the last line of
your answer should exclusively
contain "yes" or "no", and
nothing else.

Here the the structure of the
answer:

"""

[step-by-step explanation,

possibly over multiple lines]

[empty line]

[yes/no]
```

```
"""
Your answer:
```

During development, variations of this prompt were tested. For instance, we added a one-shot chain of thought prompt exemplifying the reasoning (no effect observed); various terms used to refer to the state and the previous utterance were used ("statement", "sentence", "fact"). Alternative more natural-sounding prompt was tested:

```
Suppose you already know the
following facts: {state}
Do you learn anything new
from the following statement:
{utterance}?
Explain your answer step by
step.
```

The final prompt presented above led to best results during testing and was, therefore, used throughout simulations for both the IM and the SP model.

Evaluation Since the semantic evaluation crucially carries the performance of the entire model, we analysed the performance of this module in isolation. Specifically, we evaluated the SemanticEvaluator on two groups of tests. First, we used tasks from the SuperGLUE and SNLI benchmarks (Bowman et al., 2015; Wang et al., 2019). One set of tests is based on five samples from each of the "axb", "axg", "copa", "rte" tasks within the SuperGLUE benchmark, and five entailment and contradiction datapoints each from the SNLI benchmark. Pairs of states and utterances where constructed, and the ground truth semantic value was derived from the dataset values. For SNLI, the sentence 1 was used as the state and sentence 2 as the utterance. These tests mostly contained naturalistic sentences and strongly focused on testing NLI. There were 39 test pairs. Second, we used tests containing example sentences closely matching A3DS in phrasing and content, checking for synonym and modification understanding, as well as some further semantic tests with examples of quantifiers. These are intended to broaden the set of tests and include tests matching the reference game setting more closely. There were 12 test sentences.

All evaluations were conducted with manual evaluation. The prompt was optimized based on performance on these tests. The accuracy of the final SemanticEvaluator with the prompt reported above was 0.82. Inspecting cases where the module failed throughout development, we identified some systematic ways in which this prompt fails. For instance, if the chain of thought of the model seems to indicate that the LLM answers the question "Is there any new information *present* in the utterance?", it leads the model to incorrectly answer with "no", although the reasoning suggests "yes".

A.1.3. ContrastivitySelector

This module is a rule-based module which takes a matrix of truth values T computed by the semantic evaluator (i.e., a matrix of shape state×utterance), and checks the proportion of distractors of which each available utterance is *false*. That is, it computes the sum S of the truth values over the states and computes $C = 1 - softmax(S)$, resulting in a list of contrastivity values for the utterances. If there is at least one utterance which is fully contrastive (i.e., $max(C) = 1$), the loop is terminated and this utterance is returned. If no utterance is fully contrastive, the utterances with the highest contrastivity are selected and passed to the extended UtterancesProposer.

A.1.4. InfoMaxSelector

We implement a simple informativity maximization (InfoMax) utterance selector which returns the utterance with the highest contrastivity value, which is derived from the semantic truth values.

Specifically, the selector takes as input the result of the ContrastivityEvaluator and selects the optimal contrastive utterance u: $u^* = argmax(C)$

We note that this module could be extended to return a distribution over utterances, akin to the pragmatic speaker S_1 in RSA models (Frank & Goodman, 2012):

$$P_{S_1}(u \mid s) \propto \exp(\alpha\ (\log\ L_0(s \mid u) - \text{cost})) \qquad (1)$$

$$P_{L_0}(s \mid u) \propto [\![u]\!](s) \qquad (2)$$

A.2. Ablated Single-Pass Model Details

As for the IM, the input to the model is a list of state descriptions, including the target state and one or more distractors. The model proceeds in three steps (Algorithm 2). First, an UtterancesProposer module generates ten candidate utterances for the target state based on the target state description. Second, the SemanticEvaluator module determines the truth value of each candidate utterance for each state (target and distractors). Lastly, the InfoMaxSelector module selects the most contrastive utterance. The functionality of the SemanticEvalutor and the InfoMaxSelector are identical to the IM.

A.2.1. UtterancesProposer

The prompt used for generating the utterance proposals did not include the restriction to mentioning a single feature, i.e., it did not include the production cost approximation.

```
You will be given a target
description.  Please produce
{num_samples} sentence(s) based
on the target that leave out
some part of the description
but are still well-formed.  The
reduced sentences should include
exclusively content already
mentioned in Target.  Please
provide the sentences in a
bullet list format.
```

{num_samples} was set to ten across simulations.

A.3. LLM Baseline

We conducted the same simulations with a one-shot CoT LLM baseline, where the LLM was prompted with all context information and was directly prompted to solve the task of generating a contrastive utterance. The following prompt was used:

```
You will be given a target state
and one or more distractors.
Your task is to describe the
target state in natural language
in a way that distinguishes it
from the distractors.
Try to be as concise as possible.
You do not need to list all the
features of the target state.
Please think step by step,
motivating why you decide to
mention some features.

Here is an example of a good
answer.

Target state:
- The floor is purple, the wall
is green, the red small block is
in the left corner.

Distractors:
- The floor is red, the wall is
green, the red small block is in
the middle.

Your answer:
One difference between the
target and the distractor is
the color.  This difference is
enough to distinguish between
them.  Utterance:  "The target
state has a purple floor".
Now the real input.
```

Algorithm 2 Ablated (i.e., single-pass) model.

```
Generate($s^*$, $D$, $n$):
$U_* \leftarrow \text{UtterancesProposer}(s^*, n)$
$T \leftarrow \text{SemanticEvaluator}(\{s^*\} \cup D, U_*)$
$u^* \leftarrow \text{InfoMaxSelector}(T, U_*)$
return $u^*$
```