Does discourse structure help action prediction? A look at Correction Triangles

Kate Thompson[†‡], **Akshay Chaturvedi**[†‡], **Nicholas Asher***[‡]
[†]LINAGORA Labs, [‡]IRIT, *CNRS
Toulouse, France

Abstract

An understanding of natural language corrections is essential for artificial agents that are meant to collaborate and converse with humans. We present some preliminary experiments using language-to-action models investigating whether discourse structure, in particular CORRECTION relations, improves the action prediction capabilities of language-to-action models for simple block world tasks. We focus on scenarios in which a model must correct a previous action, and present a corpus of synthetic dialogues to help explain model performance.

1 Introduction

In order to successfully complete a shared task, such as building a block structure in a shared environment, participants must accumulate a body of shared information about the goal of the task, the changing state of play, and their beliefs and intentions, collectively referred to as *common ground* (Clark, 1996). Errors are integral to the process of building common ground, as participants naturally explore and test their strategies through trial and error (Thomaz et al., 2019). When natural language is among the modes of communication available to participants, corrective speech acts provide an efficient and information-rich mechanism with which they can identify and quickly resolve errors (Benotti and Blackburn, 2021). For artificial agents that can collaborate with humans using natural language, the ability to understand and use corrections is essential.

While most speech acts entail a monotonic update of the common ground (elaborations or acknowledgments, for example), a correction entails a revision to the common ground, and is an example of a *divergent* speech act (Asher and Lascarides, 2003). An agent that understands corrections must:

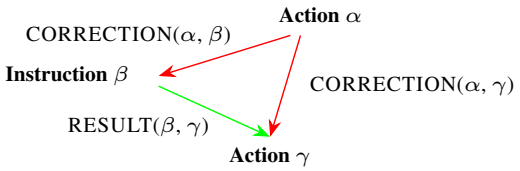

Figure 1: A **Correction Triangle** is formed by two CORRECTION relations and a RESULT relation, and appears in dialogues where an initial correction to an action results in a new action.

1. Recognize an utterance as an instance of a divergent speech act.

2. Determine the content of the correction by identifying the parts of the previous dialogue and/or shared environment it refers to.

3. Revise the common ground according to 1 and 2, and make any required changes to the shared environment.

The Minecraft Dialogue Corpus (MDC) (Narayan-Chen et al., 2019) features interactions in which two humans, playing the roles of *Builder* and *Architect*, collaborate to construct block structures in a simulated 3D environment. Architect and Builder communicate via chat window, where Architect describes the structure to Builder, who may then place and remove blocks on the grid. The MDC provides paradigmatic examples of collaborative conversation situated in a shared environment, wherein the players' linguistic contributions and the non-linguistic Builder actions are highly interdependent, particularly when Builder performs an erroneous action which the Architect must then verbally correct. The Minecraft Structured Dialogue Corpus (MSDC) (Thompson et al., 2024b) adds an annotation layer to the MDC, drawing semantically-typed relations (Asher and Lascarides, 2003) between player utter-

Proceedings of the 16th International Conference on Computational Semantics, pages 166–174

ances and actions, which elucidate the overarching semantic structure—or *discourse structure*—of the interaction. In the MSDC, correction scenarios are captured in substructures called *Correction Triangles*: an Architect instruction β stands in the CORRECTION relation to the previous erroneous action α, eliciting a new action γ, which is the RESULT of the Architect correction and also a CORRECTION of the erroneous action (Figure 1).

The MSDC discourse structures, including Correction Triangles, were shown to be automatically retrievable with state-of-the-art accuracy using the Llamipa discourse parser (Thompson et al., 2024a). Presumably, an agent utilizing the output of this parser would have at least a partial understanding of correction scenarios, since predicting the relation CORRECTION(α, β), amounts to identifying utterance α as a correction (*"no, I said add a blue block"* (1), and connecting it to the previous context β (*Builder places red block*) (2). However, the question we want to address in this paper is whether an agent can leverage the discourse structure of an interaction to inform its subsequent manipulation of the environment. This goes beyond the parsing task: given a dialogue context and its semantic structure, including CORRECTION(α, β), we want to know whether the presence of the structure improves agent predictions of the action sequence γ that would appropriately complete the Triangle.

In what follows, we briefly discuss current work relevant to our question. We then describe the language-to-action model that will serve as our agent for action prediction experiments using the MSDC. We explain how our preliminary results incentivize the creation of synthetic correction dialogues, which allow for more tightly controlled experiments. We then discuss model performance on synthetic correction dialogues and directions for future work.

2 Related Work

Previous approaches to building agents that understand corrections in collaborative tasks differ with respect to model architectures. Rubavicius et al. (2024) and Appelgren and Lascarides (2020) build agent models based on pared down cognitive architectures for interactive task learning (Laird et al., 2017), where a corrective utterance generates symbolically encoded, probabilistically weighted hypotheses, and is used to update the agent's belief state and inform an action plan. Alternatively, Chiyah-Garcia et al. (2024) create a language-to-action model by fine-tuning a large vision language model (VLM) on instruction-action pairs from a block world dataset (Bisk et al., 2016). They augment the instructions with third position repairs (Schegloff, 1992), and use masking techniques during finetuning to encourage the model to recognize the repair. In our experimental setup, we also use an LLM-based language-to-action model: our agent model is based on Nebula (Chaturvedi et al., 2024), a Llama3 architecture fine-tuned on the MDC dialogue-to-action corpus. Furthermore, we use the discourse structure annotations from the MSDC, which make corrections explicit, whereas the repairs in Chiyah-Garcia et al. (2024) were unmarked.

Discourse parsing predicts the semantic relations that hold between the elementary units of a dialogue, and produces a structural representation of the interaction. The most recent approaches to parsing based on LLMs formulate the structure prediction task as a sequence-to-sequence generation task (Li et al., 2024), where the parser takes the dialogue units as input, and outputs the discourse structure as a sequence of typed tuples. The Llamipa parser (Thompson et al., 2024b) provides state-of-the-art results on the MSDC using this approach. Our agent model uses the Llamipa structure representation, in which the discourse graph is flattened into a string of typed tuples, where each tuple represents a single relation (see Figure 2).

Discourse structure has been used to improve performance on various downstream tasks. Devatine et al. (2023) leverages discourse information to predict political orientation of news articles, while Rennard et al. (2024) uses it to improve extractive meeting summarization. Sharma et al. (2025) demonstrate that discourse structure can improve a model's performance on mathematical reasoning tasks. The experiments described in this paper are the first to use discourse structure to improve action prediction in situated collaborative tasks.

Finally, data synthesis has been increasingly used to provide high-quality training data for LLMs and LLM-based agents (Liu et al., 2024; Shichman et al., 2024), as well as to perform targeted tests of LLM knowledge (Wu et al., 2024). Synthetic data has been shown to be especially helpful in determining which concepts an agent trained on the MDC's ambiguous natural language instructions actually learned (Chaturvedi et al., 2024; Jayan-

	MDC (F1)		SynthCorr300 (Accuracy) [Action error/ Site error]			
	All relations	Correction	Overall	D1	D3	D5
Nebula	**0.39**	**0.57**	0.79 [0.18/0.03]	**1.00** [0/0]	0.73 [0.07/0.02]	0.64 [0.1/0.02]
Nebulipa	0.37	0.50	**0.80** [0.17/0.02]	**1.00** [0/0]	**0.75** [0.06/0.02]	**0.66** [0.1/0.01]
Nebulipa-E	0.37	0.52	0.67 [0.29/0.04]	0.98 [0.01/0]	0.57 [0.12/0.02]	0.46 [0.17/0.01]

Table 1: Performance of the context-aware (Nebula) and structure-aware (Nebulipa) models. Nebulipa-E(mpty) shows the results of an ablation in which structure is removed from the test samples. Column 1 shows the net action F1 scores on the MDC **test** set using all relation types; Column 2 shows F1 calculated on just those MDC **validation** samples whose predicted action sequence is mediated by a CORRECTION relation (see Figure 3). For a full breakdown of MDC splits see Appendix C. Columns 3-6 give the accuracy scores on the 300 synthetic correction dialogues broken down by the CORRECTION distance.

navar et al., 2025). This work is the first to create synthetic dialogues with discourse annotations to test the efficacy of discourse structure in a downstream task.

3 A structure-aware language-to-action model

The Nebula language-to-action model (Chaturvedi et al., 2024), was trained using the MDC to predict a Builder action sequence given the previous dialogue, and was evaluated using the same net action F1 metric as the baseline MDC model (Jayannavar et al., 2020). Given a completed action sequence, net action F1 is computed on newly placed blocks that exactly match the color and position of those in the corresponding gold action sequence. Nebula leveraged the large context window of the Llama3-8b architecture (Dubey et al., 2024), which allowed it to predict an action sequence using the entire previous dialogue context, resulting in a *context-aware* model that doubled the baseline F1 on the MDC.

In order to see whether the addition of discourse structure might further improve performance on the action prediction task, we augmented each MDC training sample with the gold[1] discourse structure from the MSDC. We formatted the structure as a sequence of typed tuples, and appended it to the dialogue context (see Figure 2). Following the Nebula training regime, we finetuned Llama3-8b using QLoRA (Dettmers et al., 2023) for 3 epochs on the augmented data (training parameters shown in Appendix A). The result was *Nebulipa*, a *structure-aware* language-to-action model.

The leftmost column of Table 1 compares the net action F1 scores of Nebula and Nebulipa on the MDC augmented with the full MSDC structures; i.e., all 17 relation types (Thompson et al., 2024b). Nebulipa-E ("Nebulipa-Empty") shows the result of an ablation in which the structure was removed from the test samples, in order to provide some further indication of whether Nebulipa, trained with structure, learned to use it. We see that this brute inclusion of structure hinders rather than improves model performance, as F1 drops two points. Also, Nebulipa-E shows no change with respect to Nebulipa. If Nebulipa were using discourse information for action prediction, we would expect its removal to result in a drop in F1; yet this result indicates that, overall, training with structure did not lead to the model to exploit it.

4 Focusing on Correction Triangles

The preliminary result above shows that including full discourse structures, containing many different relation types, does not improve language-to-action model performance on the MDC action prediction task. We note that the relational structure presents each relation uniformly, even though some types are more informative than others, given the discourse context. As mentioned in Section 1, CORRECTIONS describe revisions to the common ground, and so are often more informative than other relation types holding between less salient parts of the context: COMMENT, ACKNOWLEDGEMENT, etc.

To test this, we took a subset of MDC samples[2] in which the final action sequence to be predicted by the model is the result of an Architect correction, i.e. where a CORRECTION relation is critical to the final prediction (Figure 3). The second column of

[1] Since the purpose of these experiments is to see whether the inclusion of structure makes any difference at all, we use the gold annotations. Of course, a fully autonomous agent would predict actions as well as structure.

[2] For this test we looked at a subset of the MDC validation set, 149 of 1051 samples. See Appendix C for a description of MDC splits.

Table 1 shows that F1 improves on the subset, but is still higher for Nebula, thus corroborating the first result. Further, F1 *improves* when structure is removed (Nebulipa-E), suggesting that the addition of structure hinders performance.

Nevertheless, we maintain that this result must not be taken as decisive for three reasons. The first is that the longer dialogue contexts feature a dense relational structure (with uniform relations, as just mentioned), presenting the possibility that the signal provided by more informative relation types, such as CORRECTION, is greatly diminished; this is illustrated in Appendix B. Second, the MDC instructions contain highly context-dependent language, rich in anaphora and ellipsis, which often does not indicate a single correct action sequence. For example, we see in Figure 3 that Nebulipa performs the correct net action given the previous dialogue, but then places additional blocks—yet there is nothing in the dialogue that prohibits this. The second reason, already mentioned above, is that the contexts are long and can be very dense. Lastly, since the net action F1 metric requires action sequences match the exact positions of the gold sequences, it unjustly discounts actions that are the result of instructions that are naturally ambiguous, e.g. *put a block in a corner*, and thus obscures the model's true performance.

Taking these factors into consideration, we determined that a set of short dialogues in simple correction scenarios, in which we could zero in on Correction Triangles, would help us more clearly assess whether structure can be leveraged for action prediction. To this end, we synthetically generated *SynthCorr300*, a set of 300 short dialogues[3]. In each dialogue, Architect gives three instructions for simple shapes, one of which Builder botches, eliciting a correction in Architect's final turn (Figure 2). The shapes are towers and rows, which Nebula was shown to build with high accuracy (Chaturvedi et al., 2024), as well as single blocks, which Nebula was able to place and remove on rows and towers already present on the grid. For each instruction, a shape and its parameters were chosen randomly: one of six possible colors and a size (for towers and rows) of 3, 4, or 5 blocks. Repeated colors, shapes, or sizes occurs in a majority of the dialogues (Table 2), however, each shape is disambiguated by its location descriptor (*centre*, *corner edge*). To botch an instruction for a tower or row, we randomly chose whether to remove or add (+1 or -1) a single block from the number of blocks given by Architect. For single block placement instructions, we changed the color of the block given by Architect by randomly choosing from the five remaining colors.

0 <Build> Mission has started.
1 <Arch> Let's start with some basic shapes.
2 <Arch> Build a row of 3 red blocks at the centre.
3 <Build> `place red 010, place red 110`
4 <Arch> And place a red block at an edge.
5 <Build> `place red -215`
6 <Arch> Also build a yellow tower of size 3 at a corner.
7 <Build> `place yellow -515, place yellow -525, place yellow -535`
8 <Arch> The red row at the centre should be 3 blocks.

STRUCTURE:
Continuation(0,1), Continuation(1,2), Result(2,3), Continuation(2,4), Result(4,5), Continuation(4,6), Result(6,7), Correction(3,8)

BUILDER PREDICTION:
`place red -110 [OR place red 210]`

Figure 2: A SynthCorr300 dialogue example in which the given CORRECTION connects the Architect at turn 8 with the Builder error at turn 3, and so is of distance 5 (D5). NB: the SynthCorr300 dialogues use the Llamipa structure representation (Thompson et al., 2024a), where the relations are typed tuples appended after the dialogue in a "Structure" field.

We generated 100 dialogues for each of the three possibilities for Builder error: after the first, second, or third instruction. We also generated the discourse structure, which was identical for each dialogue except for the first relation of the Correction Triangle CORRECTION(α, β). This latter varied with the position of the error, e.g. if it was after the first instruction, the CORRECTION would reach farther back into the dialogue context (distance 5) than it would if it was after the second (distance 3) or third (distance 1) instruction.

SynthCorr300 tests whether a model can accurately produce the action sequence γ which would effectively complete the Correction Triangle, and whether the addition of discourse structure improves its performance. The correct action sequence for each dialogue is clearly defined and can be checked automatically.

[3]The SynthCorr300 data, and the code used to generate it, are available at https://huggingface.co/datasets/linagora/synthetic_corrections

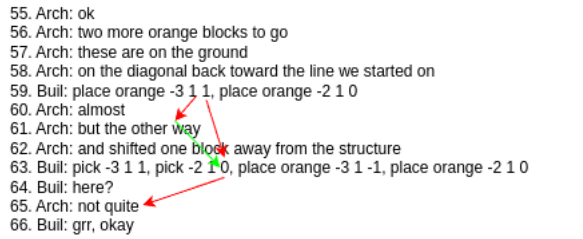

Figure 3: MDC sample where the predicted sequence is the RESULT of Architect CORRECTION in 65 and a CORRECTION of the action sequence in 63.
NB: to conserve space we truncated the dialogue context of this sample, shown in full in Appendix B.

	D1	D3	D5	All
No ambiguity	17	12	13	42
Color only	7	11	9	27
Shape only	41	39	43	123
Color and shape	35	38	35	108

Table 2: Sample counts by relation distance D between the CORRECTION source and target, and ambiguity type: *Color only*: shapes are all different, but colors are repeated; *Shape only*: colors are different but shapes are repeated.

5 Results

There was only a one-point difference between Nebula and Nebulipa on SynthCorr300 (Table 1), but unlike on MDC tests, Nebulipa was in the superior position. Furthermore, Nebulipa-E ablation conformed to previous expectations: accuracy dropped substantially when the structure was removed, indicating that a model trained with structure does learn to leverage it. When we look at the performance by CORRECTION distance, we see that the pattern holds at longer distances, although performance degraded for all models as distance increased, which is unsurprising given that the majority of CORRECTIONS in the MSDC training data ($\sim$ 70%) are of distance 3 or less.

To consider another angle of comparison, we looked at two ways in which models failed to generate the correct action sequences. *Action errors* occurred when the model correctly identified the shape to be changed (after the first, second, or third instruction), but did not perform the correct actions to do so. *Site errors* occurred when the model changed a shape that was not indicated by the CORRECTION, misidentifying the botched sequence. Returning to the discussion in Section 1 of what is in involved in understanding corrections, we can roughly align Site errors with the failure to identify what portion the previous dialogue the CORRECTION refers to (2), e.g. which instruction. We can align Action errors with the failure to properly revise the common ground (3), e.g. to perform the correct block placements and removals.

Table 1 gives action errors and site errors as a proportion of *total* samples. There was little difference in error rates between Nebula and Nebulipa, although with Nebulipa-E we saw an increase in Action errors. Since a CORRECTION representation *Corr(x,y)* effectively acts as a pointer to the botched sequence x, we would expect an increase in Site errors once the pointer was taken away. Instead, there was a greater incidence of Action errors. While the SynthCorr300 data is too small to support conclusions on the relationship between error types, the preliminary indication here is that the CORRECTION relations are not used to pick out the error site (perhaps the model can already do this using linguistic context) but rather to provide important semantic information about what the model is supposed to do at the site. For instance, it is possible the Architect utterance *"The tower ... should be 3 blocks."* might only lead to model to correctly infer the correct actions (i.e., *remove one block from the tower*) when combined with a semantic marker for CORRECTION. Possible future work might thus involve varying the synthetic dialogues by replacing the final CORRECTIONS with a different but coherent relation type such as ELABORATION, and testing for changes in Action errors.

6 Conclusion

In this paper we broach a question in discourse that has gotten relatively little attention—*can discourse structure guide action predictions?*—and explore a particular LLM-based approach for an initial investigation. The results of our synthetic dialogue experiments showed that, overall, access to large contexts overrides the effects of adding explicit discourse representations. However, there was some indication that models trained with structure did learn to exploit CORRECTIONS, using them to correct relevant parts of the discourse context with higher accuracy. In future work, we will enlarge the synthetic data in order to further investigate the action error results, as well as consider more varied correction scenarios.

7 Limitations

This work explores the role of discourse structure in conversational instruction following scenarios where the agent's goal is to perform the correct action sequences in a shared environment. It only considers one instantiation of such a scenario, in which the agent builds block structures on a 3D grid. The experimental results are obtained using agent models based on generative LLMs, where the discourse structure is represented as a string of typed tuples, and appended after the dialogue text in the model inputs. Certainly there are other ways to represent and feed structure into the agent model—or perhaps to integrate structural information into the model architecture rather than the data inputs—as well as other model architectures which would be worth exploring, such as graph neural networks. The synthetic data generated is small, and covers only a portion of the variation possible in correction dialogues with respect to the source and complexity of the Builder action errors, and to the referential ambiguity of correction language.

Acknowledgments

We thank our reviewers for their observations and constructive criticism. For financial support, we thank the National Interdisciplinary Artificial Intelligence Institute ANITI (Artificial and Natural Intelligence Toulouse Institute), funded by the French 'Investing for the Future–PIA3' program under the Grant agreement ANR-19-PI3A-000. This project has also been funded by the France 2030 program and is funded by the European Union - Next Generation EU as part of the France Relance. We also thank the projects COCOBOTS (ANR-21-FAI2-0005) and DISCUTER (ANR-21-ASIA-0005), and the COCOPIL "Graine" project of the Région Occitanie of France. This work was granted access to the HPC resources of CALMIP supercomputing center under the allocation 2016-P23060.

References

Mattias Appelgren and Alex Lascarides. 2020. Interactive task learning via embodied corrective feedback. *Autonomous Agents and Multi-Agent Systems*, 34:1–45.

Nicholas Asher and Alex Lascarides. 2003. *Logics of Conversation*. Cambridge University Press.

Luciana Benotti and Patrick Blackburn. 2021. Grounding as a collaborative process. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 515–531, Online. Association for Computational Linguistics.

Yonatan Bisk, Deniz Yuret, and Daniel Marcu. 2016. Natural language communication with robots. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 751–761.

Akshay Chaturvedi, Kate Thompson, and Nicholas Asher. 2024. Nebula: A discourse aware Minecraft builder. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6431–6443, Miami, Florida, USA. Association for Computational Linguistics.

Javier Chiyah-Garcia, Alessandro Suglia, and Arash Eshghi. 2024. Repairs in a block world: A new benchmark for handling user corrections with multi-modal language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11523–11542, Miami, Florida, USA. Association for Computational Linguistics.

Herbert H Clark. 1996. *Using language*. Cambridge university press.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115. Curran Associates, Inc.

Nicolas Devatine, Philippe Muller, and Chloé Braud. 2023. An integrated approach for political bias prediction and explanation based on discursive structure. In *Findings of the Association for Computational Linguistics (EACL 2023)*, pages 11196–11211. ACL: Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Prashant Jayannavar, Anjali Narayan-Chen, and Julia Hockenmaier. 2020. Learning to execute instructions in a Minecraft dialogue. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2589–2602, Online. Association for Computational Linguistics.

Prashant Jayannavar, Liliang Ren, Marisa Hudspeth, Charlotte Lambert, Ariel Cordes, Elizabeth Kaplan, Anjali Narayan-Chen, and Julia Hockenmaier. 2025. Bap v2: An enhanced task framework for instruction following in minecraft dialogues. *arXiv preprint arXiv:2501.10836*.

John E Laird, Kevin Gluck, John Anderson, Kenneth D Forbus, Odest Chadwicke Jenkins, Christian Lebiere, Dario Salvucci, Matthias Scheutz, Andrea Thomaz,

Greg Trafton, et al. 2017. Interactive task learning. *IEEE Intelligent Systems*, 32(4):6–21.

Chuyuan Li, Yuwei Yin, and Giuseppe Carenini. 2024. Dialogue discourse parsing as generation: A sequence-to-sequence LLM-based approach. In *Proceedings of the 25th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 1–14, Kyoto, Japan. Association for Computational Linguistics.

Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi Yang, Denny Zhou, et al. 2024. Best practices and lessons learned on synthetic data. *arXiv preprint arXiv:2404.07503*.

Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. 2019. Collaborative dialogue in Minecraft. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5405–5415, Florence, Italy. Association for Computational Linguistics.

Virgile Rennard, Guokan Shang, Michalis Vazirgiannis, and Julie Hunter. 2024. Leveraging discourse structure for extractive meeting summarization. *arXiv preprint arXiv:2405.11055*.

Rimvydas Rubavicius, Peter David Fagan, Alex Lascarides, and Subramanian Ramamoorthy. 2024. Secure: Semantics-aware embodied conversation under unawareness for lifelong robot learning. *arXiv preprint arXiv:2409.17755*.

Emanuel A Schegloff. 1992. Repair after next turn: The last structurally provided defense of intersubjectivity in conversation. *American journal of sociology*, 97(5):1295–1345.

Krish Sharma, Niyar R Barman, Akshay Chaturvedi, and Nicholas Asher. 2025. Dimsum: Discourse in mathematical reasoning as a supervision module.

Mollie Frances Shichman, Claire Bonial, Taylor A. Hudson, Austin Blodgett, Francis Ferraro, and Rachel Rudinger. 2024. PropBank-powered data creation: Utilizing sense-role labelling to generate disaster scenario data. In *Proceedings of the Fifth International Workshop on Designing Meaning Representations @ LREC-COLING 2024*, pages 1–10, Torino, Italia. ELRA and ICCL.

Andrea L Thomaz, Elena Lieven, Maya Cakmak, Joyce Y Chai, Simon Garrod, Wayne D Gray, Stephen C Levinson, Ana Paiva, and Nele Russwinkel. 2019. Interaction for task instruction and learning. In *Interactive task learning: Humans, robots, and agents acquiring new tasks through natural interactions*, pages 91–110. MIT Press.

Kate Thompson, Akshay Chaturvedi, Julie Hunter, and Nicholas Asher. 2024a. Llamipa: An incremental discourse parser. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6418–6430, Miami, Florida, USA. Association for Computational Linguistics.

Kate Thompson, Julie Hunter, and Nicholas Asher. 2024b. Discourse structure for the Minecraft corpus. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4957–4967, Torino, Italia. ELRA and ICCL.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. 2024. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1819–1862, Mexico City, Mexico. Association for Computational Linguistics.

A Model Parameters

GPUs	
4 NVIDIA Volta V100	
Hyperparameters	
Training epochs	3
batch size	4
optimizer	Adam
learning rate	2e-4
learning rate scheduler	linear warm-up and cosine annealing
warm-up ratio	0.03
gradient clipping	0.3
lora r	64
lora (alpha)	16
lora dropout ratio	0.1
lora target modules	Only Attention Blocks (q_proj, v_proj)
quantization for Llama3	4-bit NormalFloat

Table 3: Details on computing resources and hyperparameters for finetuning Llamipa.

Table 3 gives the hyperparameters used for finetuning Nebula and Nebulipa, along with the computing resources. We adapt the finetuning code from the following repository[4].

[4] https://github.com/mlabonne/llm-course/blob/main/Fine_tune_Llama_2_in_Google_Colab.ipynb

B MDC sample

Figure 4: MDC sample where the predicted sequence is the RESULT of Architect CORRECTION in 65 and a CORRECTION of the action sequence in 63. The full discourse structure is given with CORRECTIONS highlighted. The Correction Triangles are superimposed on the context for reference.

C MDC data

	MDC test		MDC validation			
	# samples	F1	# samples	F1	# Correction samples	F1
Nebula	1615	0.39	1335	0.39	149	0.57
Nebulipa	1471	0.37	1194	0.355	149	0.50
Nebulipa-E	1471	0.37	1194	0.363	149	0.52

Table 4: Number of samples and model F1 for the MDC test and validation splits. The Correction set discussed in Section 4 is a subset of the validation set.

The Nebula language-to-action model predicts Builder actions given the entire previous dialogue context. We prepared the MDC dialogues for training and testing Nebula by dividing it into dialogue-action pairs. Thus the number of data samples in a split is equal to the total number of Builder actions across all dialogues in that split. The 100 validation dialogues contain 1335 action sequences, and the 101 test dialogues contain 1615 action sequences—the dialogues in test were on average longer (had greater number of utterances) than those in validation (Thompson et al., 2024b).

The MSDC [5] provides complete dialogue structure annotations for all MDC dialogues. Adding structure from the MSDC to the MDC samples for Nebulipa training was not straightforward. During the MSDC annotation campaign, some of the Builder action sequences were fused together into *Complex Discourse Units* (see Figure 1 in Thompson et al. (2024b)), which lead to a reduction in the overall number of separate action sequences, as can be seen in Table 4. When the action sequences were combined, the dialogue moves between them also shifted. As a result, the MDC data used for Nebulipa is not *identical* to the data used for Nebula (ignoring the addition of structure to the Nebulipa data). However, the overlap between them is large enough to warrant their comparison: 116 of the 1194 ($\sim$ 9%) samples in the Nebulipa validation set were not present in the Nebula set, and 99 out of 1471 ($\sim$ 7%) of samples in the test set.

In Section 4 we isolated the samples in the validation set where the action sequence to be predicted was the target of a CORRECTION relation. There were 182 such samples in the Nebulipa data, but only 149 of these were also present in the Nebula data.

[5] https://huggingface.co/datasets/linagora/MinecraftStructuredDialogueCorpus