On Grounded Planning for Embodied Tasks with Language Models: A Study Using Synthetic Environments

Research Seminar Number Theoretic and Algebraic Methods in Data Analysis Project Report

Faculty of Computer Science
National Research University Higher School of Economics

Muhammad Sufyan
Zahid Hussain

December 29, 2025

Contents

1 Introduction

This project explores grounded plan generation, where a model receives a short goal description and must produce a sequence of high-level actions that complete the task. The idea is based on the work presented in the paper *"On Grounded Planning for Embodied Tasks with Language Models (G-PlanET)"*. The original study shows that language models can benefit from additional structured information about the environment when generating multi-step plans.

To examine these ideas in a controlled way, We created a small synthetic dataset consisting of simple kitchen-style environments, goal sentences, and reference plans. We trained several versions of BART on this dataset and compared them with TAPEX and GPT-4o. Our aim was to understand how single-pass generation, table-augmented generation, iterative decoding, and model size influence plan accuracy.

1.1 Team Contribution

This project is completed jointly by both team members Muhammad Sufyan and Zahid Hussain. We worked together on all major parts of the project, including understanding the original G-PlanET paper, designing the synthetic dataset, implementing the models, running experiments, and analyzing the results. The report, code and presentation were also prepared collaboratively, with both of us contributing to writing, discussion, and final revisions.

2 Dataset and Experimental Setup

Each example in the dataset contains:

- an environment table describing objects, their positions, and their parent locations,
- a goal sentence such as "move the apple from the sink to the counter",
- a four-step ground truth plan.

A total of 300 examples were generated, with 240 used for training and 60 for validation.

Models evaluated

The following systems were tested:

- **BART-base (goal only):** receives only the goal sentence and generates the plan in one pass.

- **BART-base with table:** the environment table is added to the input to see whether the model can use structured information.
- **Iterative BART model:** predicts one step at a time, using previously generated steps as context.
- **BART-large (goal only):** a larger version of BART with better capacity for long context.
- **BART-large with table:** tests whether model size improves the use of structured input.
- **TAPEX:** a table-focused QA model evaluated without fine-tuning to see how it behaves on sequential action generation.
- **GPT-4o:** evaluated on a small sample of examples to compare its reasoning style with the fine-tuned models.

All fine-tuned models were trained for two epochs with the same tokenizer, batch size, and beam search settings.

3 Methods

Single-pass generation

The model receives the goal (or goal + table) and produces the entire plan in one sequence. This serves as the simplest baseline.

Table-based generation

The environment table is flattened into text and appended to the input. This tests whether the model can benefit from structured information.

Iterative decoding

The model predicts the plan step by step. The input contains the goal, the environment table, and all previously generated steps. This reduces long-sequence drift and often stabilizes weaker models.

Evaluation metrics

To evaluate performance, four common metrics were used:

- **KAS:** action-level overlap between predicted and reference actions.

- **BLEU:** precision-based score comparing predicted tokens to reference tokens.
- **ROUGE-1:** recall of single tokens, measuring how much of the reference text appears in the prediction.
- **ROUGE-L:** based on the longest common subsequence, rewarding predictions that follow similar structure and order.

These metrics provide a balanced view of both action correctness and textual similarity.

4 Results

Evaluation Table

Model	KAS	BLEU	ROUGE-1	ROUGE-L
BASE_SMALL	0.690	0.7276	0.8900	0.8240
TABLE_SMALL	0.200	0.0342	0.3636	0.3636
ITERATIVE	0.650	0.6210	0.7580	0.7530
BASE_LARGE	0.800	0.9015	0.9700	0.9600
TABLE_LARGE	0.545	0.6842	0.8470	0.8180
TAPEX	0.200	0.000002	0.1990	0.1990
GPT-4o	0.695	0.3158	0.8483	0.8039

Graphs

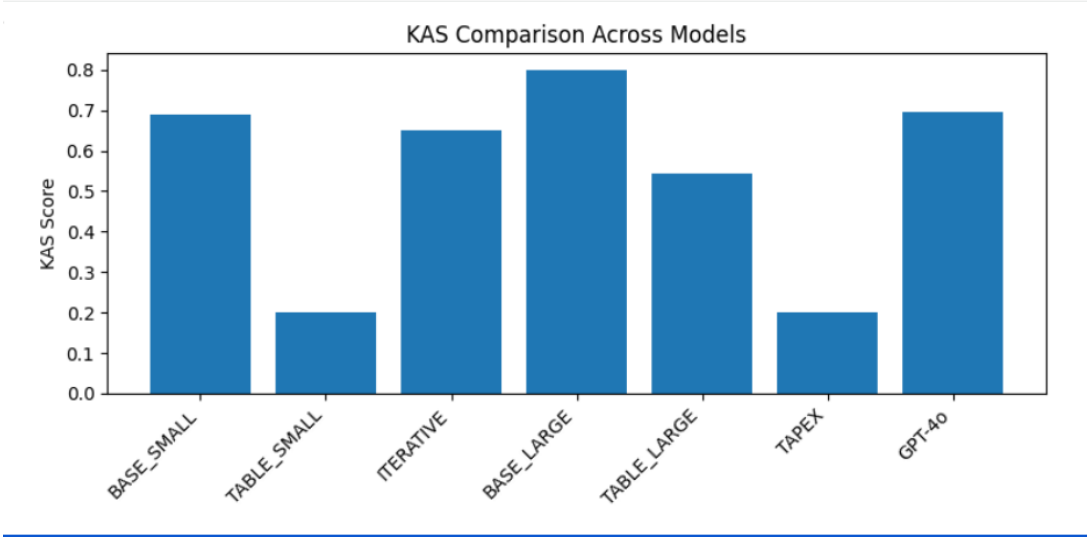

Figure 1: KAS comparison across models.

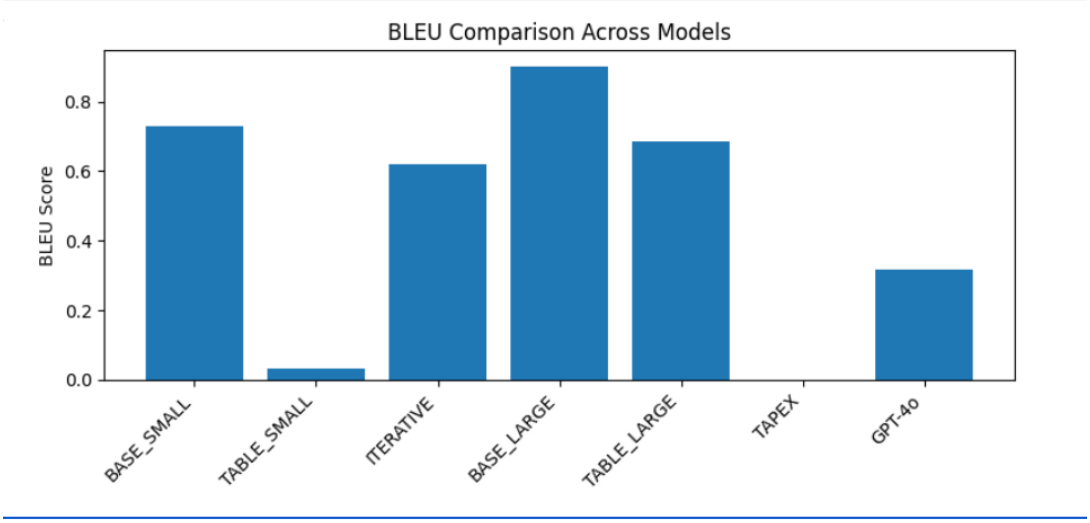

Figure 2: BLEU score comparison across models.

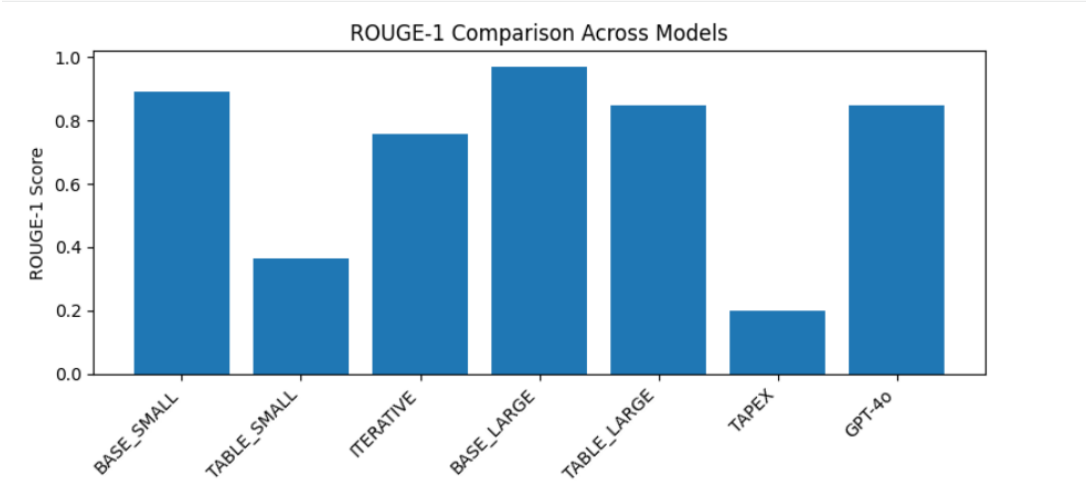

Figure 3: ROUGE-1 score comparison.

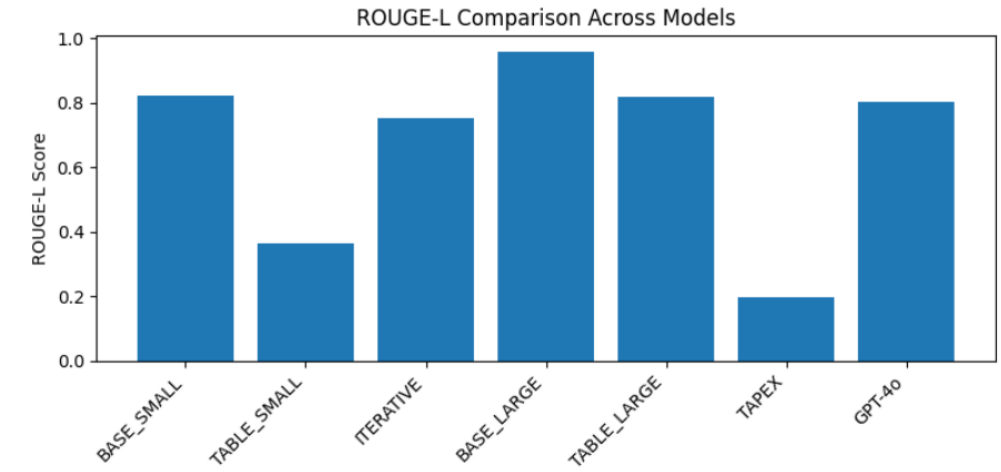

Figure 4: ROUGE-L score comparison.

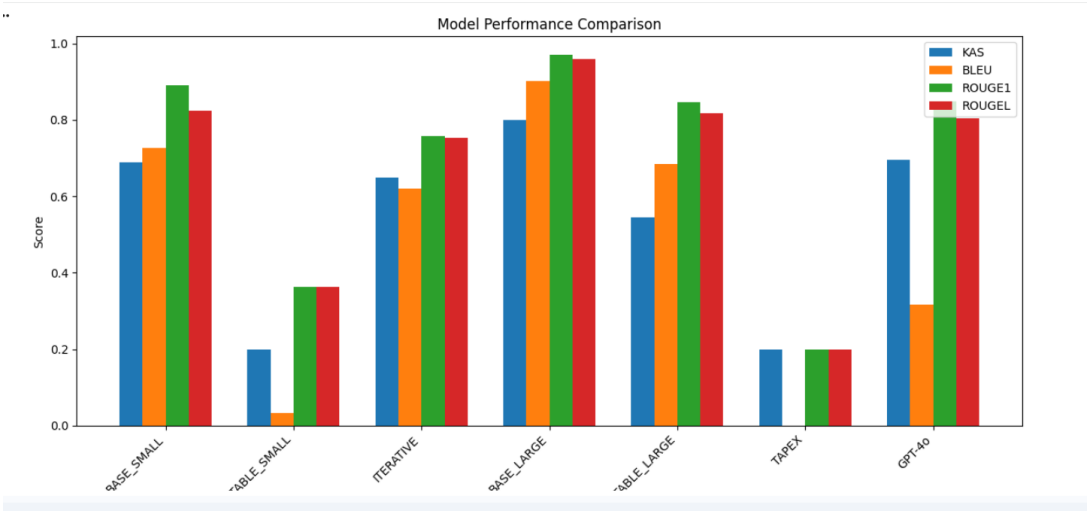

Figure 5: Grouped comparison of all metrics.

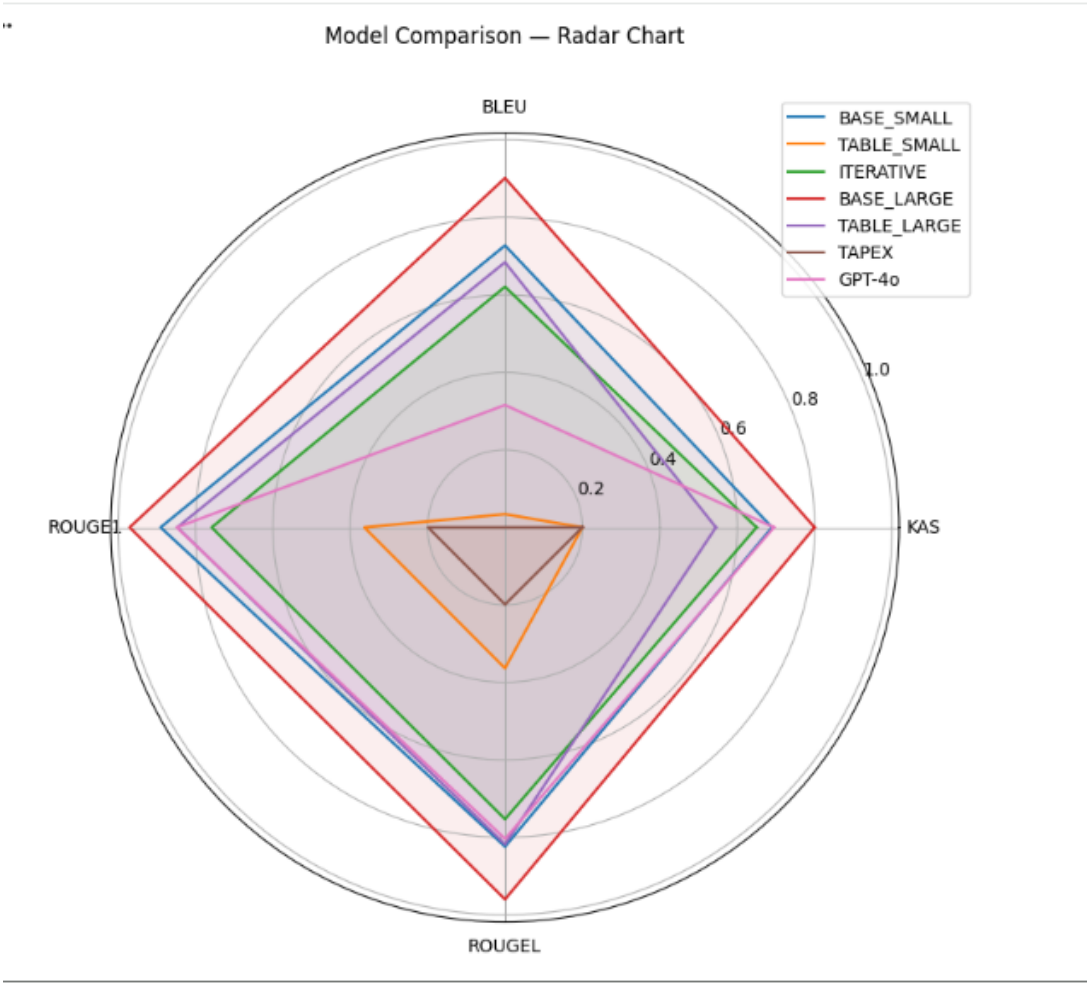

Figure 6: Radar chart showing overall performance patterns.

General Observation

The plots show clear patterns across all metrics. The large BART models achieve the strongest and most consistent results. The small baseline performs well, while the small table-based model performs poorly, suggesting difficulty in using structured input. The iterative model remains stable across metrics. TAPEX still performs extremely poorly because it is not designed for action generation. Even after adding the fallback rule, its predictions do not match the reference structure, which results in very low metric scores.. GPT-4o shows high ROUGE scores but lower BLEU because it sometimes adds extra steps or wording not present in the reference plan.

5 Discussion

Several points stand out clearly from the figures and results:

1. **Model size has a strong effect on accuracy.** The large BART model gave the

best results across all metrics. It handled both the goal text and the environment table without losing consistency.

2. **Adding the table does not always help.** For the small BART model, including the table caused a noticeable drop in performance, suggesting that the model did not have enough capacity to use the structured information correctly.

3. **Step-by-step decoding stabilizes weaker models.** The iterative version of BART-base produced more reliable actions compared to the full-sequence version, even though it still lagged behind the large model.

4. **TAPEX is not suited for action generation.** Even after adjusting outputs to avoid empty strings, TAPEX still had low overlap with the reference plans. This aligns with its training objective, which focuses on answering table queries, not generating sequences of actions.

5. **GPT-4o produced valid actions but did not follow the strict four-step structure.** Because the evaluation depends on matching the reference format, any extra movements or combined steps resulted in lower BLEU and ROUGE scores.

6 Conclusion

This project reproduced the main ideas from the G-PlanET paper using a synthetic dataset. The experiments showed that large models deliver the most stable results and benefit from additional structured inputs. Small models do not gain from environment tables, and iterative decoding improves their stability. TAPEX performs poorly in this task because it is not designed for step-based plan generation, and GPT-4o provides additional insights but does not follow the strict format used for evaluation.

Even with a small dataset, the overall trends match the findings of the original work, showing that the main ideas generalize well in a controlled environment.

7 Future Work

There are several directions in which this work can be extended in the future. First, the size and diversity of the dataset can be increased to better reflect real-world environments and reduce overfitting to simple synthetic layouts. Second, instead of fixing the plan length to four steps, future experiments could allow variable-length plans to better match natural task complexity. Finally, it would be interesting to test other planning-oriented language models or incorporate explicit reasoning or constraint-checking modules to improve consistency and reduce invalid action sequences.