Evaluating LLMs' Reasoning Over Ordered Procedural Steps

Anonymous ACL submission

Abstract

Reasoning over procedural sequences, where the order of steps directly impacts outcomes, is a critical capability for large language models (LLMs). In this work, we study the task of reconstructing globally ordered sequences from shuffled procedural steps, using a curated dataset of food recipes, a domain where correct sequencing is essential for task success. We evaluate several LLMs under zero-shot and few-shot settings and present a comprehensive evaluation framework that adapts established metrics from ranking and sequence alignment. These include Kendall's Tau, Normalized Longest Common Subsequence (NLCS), and Normalized Edit Distance (NED), which capture complementary aspects of ordering quality. Our analysis shows that model performance declines with increasing sequence length, reflecting the added complexity of longer procedures. We also find that greater step displacement in the input, corresponding to more severe shuffling, leads to further degradation. These findings highlight the limitations of current LLMs in procedural reasoning, especially with longer and more disordered inputs.

Recipe: Apple cheese casserole

Shuffled Steps:
1. bake 325: for about 30-45 minutes
2. serves 4-6
3. add flour and mix well-batter will be stiff
4. place apples in a buttered baking dish about 1.5 qt size
5. cream butter and sugar in a mixing bowl , add cheese & combine well
6. spread the cheese / flour mixture over the apples covering the apples well

Correct Order: [5, 3, 4, 6, 1, 2]

Correctly Ordered Steps:
1. cream butter and sugar in a mixing bowl , add cheese & combine well
2. add flour and mix well-batter will be stiff
3. place apples in a buttered baking dish about 1.5 qt size
4. spread the cheese / flour mixture over the apples covering the apples well
5. bake 325: for about 30-45 minutes
6. serves 4-6

Figure 1: Example of the step ordering task. Given a shuffled list of recipe instructions (top), the goal is to recover the correct sequence (bottom) required to successfully complete the recipe. The middle row shows the gold permutation that reorders the input into the correct order.

1 Introduction

Understanding and generating correctly ordered action sequences is a key aspect of reasoning. Many real world tasks, such as cooking recipes or carrying out technical procedures, require steps be completed in a precise order to achieve the intended outcome. LLMs have demonstrated strong performance on various reasoning tasks including arithmetic computation (Imani et al., 2023; Ahn et al., 2024), commonsense inference (Rajani et al., 2019), and question answering (Robinson et al., 2022). While much prior work has evaluated LLMs on step-by-step reasoning, their ability to reason over and reconstruct ordered procedural steps remains relatively underexplored.

Step ordering tasks, where the correctness of the output depends on recovering a globally coherent sequence, pose a unique challenge. Most existing research focuses on predicting the immediate next step (Yong et al., 2025; Wang et al., 2023), rather than reconstructing the full sequence from a shuffled set. Moreover, prior evaluations rely only on accuracy (Quan and Liu, 2024), measuring exact matches between predicted and reference positions. This limits our ability to fully understand LLMs procedural reasoning. In this work, we evaluate LLMs' step ordering capabilities using a curated dataset of food recipes due to their clearly defined structure and strong ordering constraints. As illustrated in Figure 1, the model receives a shuffled list of recipe instructions and must recover the correct sequence that reflects the intended preparation process. We incorporate complementary metrics—Kendall's Tau for rank correlation, Normalized Longest Common Subsequence (NLCS) for subsequence preservation, and Normalized Edit Distance (NED) for reordering cost to provide a deeper analysis of model performance. We conduct a systematic evaluation across multiple LLMs under 0-shot and few-shot settings. We further analyze performance as a function of sequence complexity, examining how models respond to longer recipes and greater amounts of step shuffling. Our main contributions are:

- We evaluate step-order reasoning in LLMs using a structured cooking recipe dataset for full sequence reconstruction.
- We propose a multi-metric evaluation framework capturing partial correctness, subsequence alignment, and reordering cost.
- We analyze performance variations with step count and shuffling difficulty, revealing gaps and challenges in procedural reasoning.

2 Related Work

Previous studies have explored LLMs reasoning on procedural tasks. STEPS (Wang et al., 2023) proposes a benchmark to assess models' procedural reasoning through two subtasks: next-step prediction and multiple-choice selection of the correct next step. While valuable, these tasks focus only on local coherence by predicting or identifying a single correct step rather than requiring the model to recover an entire global sequence. ProcBench (Fujisawa et al., 2024) focuses on multi-step reasoning over structured tasks like string manipulation and arithmetic operations. It evaluates whether LLMs can follow explicit instructions step-by-step, minimizing the need for external knowledge or path exploration. AttackSeqBench (Yong et al., 2025) evaluates LLMs' understanding of sequential patterns in cybersecurity reports through a suite of question-answering tasks. These are designed to probe models' ability to reason about adversarial behavior over time. However, the setting remains extractive QA, and models are not required to reconstruct full procedural chains. EconLogicQA (Quan and Liu, 2024) introduces a benchmark targeting sequential reasoning over interdependent events drawn from economic articles, emphasizing complex temporal and logical relationships. However, like other QA-style evaluations, it relies mainly on accuracy or exact match at each step, missing partial correctness or structural misalignment. In contrast, our study focuses on full-sequence reconstruction and introduces additional metrics for a more comprehensive assessment of procedural reasoning.

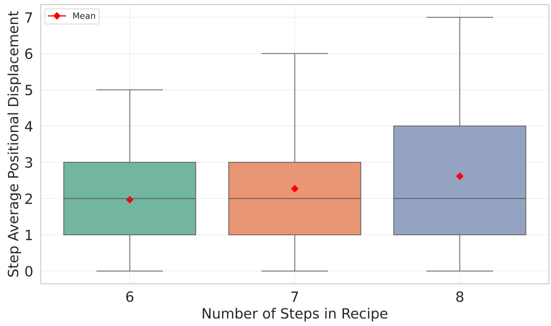

Figure 2: Distribution of step movement distances across recipes of different sequence lengths.

3 Problem Definition

Given a shuffled set of procedural steps $S = \{s_1, s_2, \ldots, s_n\}$, the goal is to find a permutation $\hat{S} = \{\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_n\}$ that best approximates the ground truth ordered sequence $S^* = \{s^*_1, s^*_2, \ldots, s^*_n\}$. The predicted sequence $\hat{S}$ is aligned with S^* to assess ordering quality.

4 Dataset

Majumder et al. (2019) has introduced a dataset containing 230K recipes from Food.com[1]. From this corpus, we select 5,000 samples with 6 to 8 steps and 5 to 6 ingredients, ensuring moderate sequence length and complexity. Food recipes are inherently sequential, and prior work has treated step ordering as critical to successful execution (Wang et al., 2023). However, some recipes may have some steps that may be interchangeable without affecting the outcome (e.g., cutting onions and cutting potatoes). To focus on sequences where step order is necessary, we apply an additional curation step using a LLM to filter recipes requiring strict ordering (see Appendix A) which yields to 1,740 recipes. Each recipe provides a coherent step sequence $S = \{s_1, \ldots, s_n\}$, which we shuffle randomly (with fixed seed) to produce $\hat{S}$. The task is to recover the original order from $\hat{S}$. We generate a permutation label $\pi \in \{1, \ldots, n\}^n$, where

[1] https://www.food.com

π_i denotes the original position of the i-th step in the shuffled sequence. The dataset is balanced with 29.6% (515 samples) having 6 steps, 36.7% (638 samples) with 7 steps, and 33.7% (587 samples) with 8 steps. We quantify the extent of step permutation by measuring the *average positional displacement*, defined as the mean absolute difference between the original position p_i and the shuffled position s_i of each step i in a sequence of length n, i.e., $\frac{1}{n}\sum_{i=1}^{n}|p_i - s_i|$. This metric captures the average magnitude of step movement caused by shuffling. As shown in Figure 2, longer sequences show higher average displacement, increasing from 1.97 for 6-step to 2.62 for 8-step recipes, indicating greater complexity in step rearrangements. The median displacement remains at 2 across all lengths.

5 Experimental Setup

5.1 Inference Settings

We evaluate four instruction-tuned LLMs: `Llama-3.1-8B-Instruct`(Grattafiori et al., 2024), `Mistral-7B-Instruct`(Jiang et al., 2024), `Gemma-2-9b-it`(Team et al., 2024), and `GPT-4o-mini`(OpenAI et al., 2024) under three settings: zero-shot, 3-shot, and 5-shot. Model receives task instructions, the recipe name, and a list of shuffled steps (see Appendix B, C). It is expected to output the reordered steps and their corresponding indices. From 1,740 samples, we use 1,700 for testing and the rest for few-shot demonstrations.

5.2 Evaluation Metrics

We use four complementary metrics:

Step Accuracy (Acc). We report accuracy at the step level: $Acc = \frac{1}{n}\sum_{i=1}^{n}(\hat{\pi}_i = \pi_i)$. This metric measures the fraction of steps placed at the correct positions provides an measure of how often the model recovers the exact step location.

Kendall's Tau (τ) (KTau). Kendall's tau is a rank correlation metric (Lapata, 2006) that evaluates the relative order of all possible step pairs between the predicted permutation $\hat{\pi}$ and the ground truth π. It is computed as $\tau = \frac{C-D}{\frac{1}{2}n(n-1)}$, where C is the number of concordant pairs and D is the number of discordant pairs. It is suitable for assessing whether the predicted step sequence agrees with the ground truth in terms of relative step precedence, regardless of their absolute positions.

Normalized Edit Distance (NED). Edit distance counts the number of insertions, deletions, or swaps required to convert the predicted order into the gold sequence. We use its normalized form (Marzal and Vidal, 2002), $NED = \frac{EditDistance(\hat{\pi},\pi)}{n}$. This metric measures the total transformation cost and is particularly sensitive to local misplacements.

Normalized Longest Common Subsequence (NLCS). We compute the length of the longest common subsequence (LCS) between $\hat{\pi}$ and π, normalized by the length of the reference: $NLCS = \frac{LCS(\hat{\pi},\pi)}{n}$. This metric rewards the preservation of correct subsequences and reflects the extent to which a model recovers partial ordering structure. It is robust to small local reorderings and has been widely used in structured sequence evaluation.

Model	Shots	Acc	NLCS	KTau	NED
Llama-3.1	0-shot	0.33	0.62	0.70	0.56
	3-shot	0.45	0.73	0.83	0.42
	5-shot	0.44	0.73	0.83	0.43
Mistral	0-shot	0.29	0.61	0.73	0.55
	3-shot	0.32	0.66	0.79	0.51
	5-shot	0.31	0.66	0.79	0.51
Gemma-2	0-shot	0.59	0.81	0.87	0.32
	3-shot	0.62	0.84	**0.90**	0.28
	5-shot	0.61	0.84	**0.90**	0.28
GPT-4o	0-shot	0.63	0.83	0.89	0.29
	3-shot	**0.64**	**0.85**	**0.90**	**0.27**
	5-shot	**0.64**	0.84	**0.90**	**0.27**

Table 1: Performance of different models across few-shot settings (0, 3, 5) using Accuracy (Acc), Normalized Longest Common Subsequence (NLCS), Kendall Tau (KTau), and Normalized Edit Distance (NED). Best values are bolded (lowest for NED).

6 Results and Analysis

6.1 Performance in Zero-Shot and Few-Shot Settings

Table 1 reports LLMs performance in 0-shot and few settings. All models show notable improvements from 0-shot to 3-shot prompting, with no gains beyond 3-shot. This suggests that a small number of demonstrations helps models learn structural reordering patterns, but further they do not add additional value. GPT-4o consistently achieves the best performance across all metrics. In the 3-shot setting, it reaches the highest accuracy (0.64), NLCS (0.85), and KTau (0.90), and the lowest NED (0.27), indicating better absolute positioning, strong preservation of subsequences, and minimal local reordering. Gemma-2 performs competitively whereas Mistral and Llama-3.1 fall behind across

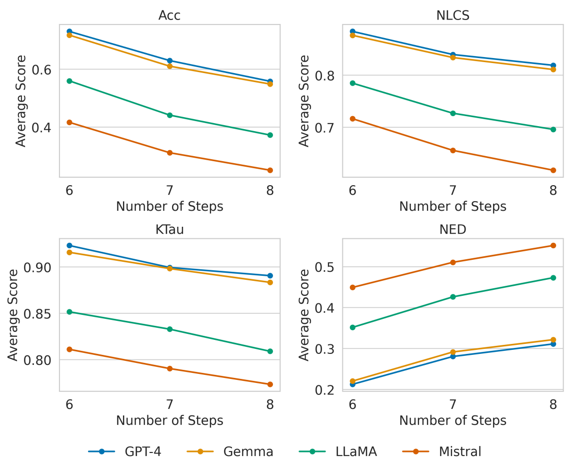

Figure 3: 3-shot performance of models (Acc, NLCS, KTau, NED) across varying numbers of steps.

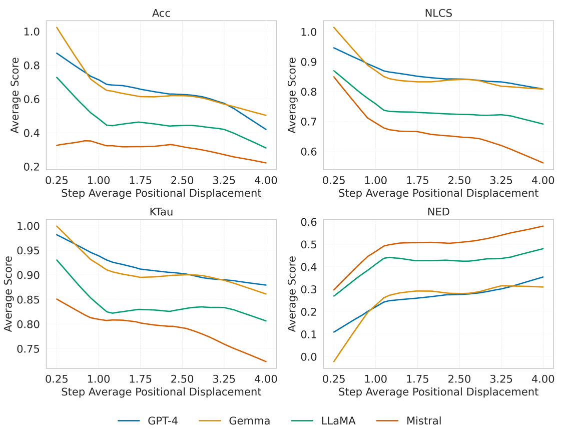

Figure 4: Smoothed 3-shot performance of models (Acc, NLCS, KTau, NED) across average positional displacement

all metrics, often producing more fragmented sequences (lower NLCS) and higher reordering costs (higher NED), despite moderate KTau scores.

KTau values show that even when models make positional errors, they may still preserve correct relative ordering. For example, Llama-3.1 in 3-shot achieves 0.83 KTau despite only 0.45 accuracy, indicating good understanding of step precedence even with absolute misplacements. NED values further expose models' tendency to make local misorderings— GPT-4o and Gemma-2B consistently yield the lowest values. NLCS emphasizes preservation of long subsequences; GPT-4o and Gemma-2B again score highest, indicating better retention of step continuity. Despite overall improvements in few-shot settings, all models exhibit gaps in fine-grained step-level reasoning.

6.2 Impact of Number of Steps on Ordering Performance

We analyze model performance by the number of steps in the sequence (n), where longer sequences indicate increased complexity. As shown in Figure 3, with n increasing from 6 to 8, a general performance decline is observed across all models, reflecting the added difficulty in recovering longer step sequences. GPT-4o consistently performs best, maintaining high accuracy (0.73 $\rightarrow$ 0.56), strong subsequence alignment (NLCS: 0.88 $\rightarrow$ 0.82), and low edit cost (NED: 0.21 $\rightarrow$ 0.31) as complexity increases. Gemma-2B shows similar robustness, with slightly lower performance. Llama-3.1 and Mistral performance drops more significantly indicating their tendency to produce fragmented and disordered outputs under increased complexity.

6.3 Impact of Step Average Positional Displacement on Model Performance

We further assess robustness to reordering by analyzing model performance with respect to *average positional displacement*. As shown in Figure 4 with displacement increasing, indicating more severe shuffling, all models show performance drops. For Accuracy, Gemma starts highest but drops sharply from near 1.0 to 0.4, while GPT-4 declines more gradually, indicating greater stability. Mistral performs lowest overall but declines steadily, suggesting consistent underperformance rather than heightened sensitivity. In NLCS, all models degrade with displacement, but GPT-4 and Gemma maintain relatively stable and higher scores. NED increases with displacement, reflecting greater divergence from the reference; here, GPT-4 shows a smaller increase compared to the other models.

7 Conclusion

We evaluated four LLMs on step ordering tasks using four complementary metrics. All models improved from 0-shot to 3-shot prompting, with no gains beyond. GPT-4o consistently achieved the best performance, followed by Gemma-2, while Llama and Mistral performed less reliably. As sequence length and reordering complexity increased, performance declined across the board. While models often preserved relative ordering (high KTau) and subsequences (high NLCS), they still struggled with precise step level reasoning highlighting limitations in LLMs' procedural understanding.

8 Limitations

While our study offers a comprehensive evaluation of LLMs on step ordering tasks, it leaves room for further exploration. First, we restrict our analysis to relatively short sequences (6–8 steps), extending the evaluation to longer instructions could uncover new insights. Second, we evaluate only instruction-tuned models without task-specific fine-tuning. Targeted fine-tuning on step ordering or procedural datasets may yield improved performance. Finally, although our dataset is carefully curated to ensure strong ordering constraints, it is focused solely on the cooking domain; evaluating cross-domain generalization would offer a broader view of LLM procedural reasoning capabilities.

9 Ethics Statement

The research conducted for this paper adheres to ethical principles and guidelines. The study utilizes publicly available datasets from reputable sources, ensuring compliance with data usage policies and respecting the privacy and confidentiality of individuals involved. All methodologies follow established scientific practices, emphasizing transparency, validity, and reliability. As the study does not involve human subjects or sensitive information, no ethics approval was sought.

References

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*.

Ippei Fujisawa, Sensho Nobe, Hiroki Seto, Rina Onda, Yoshiaki Uchida, Hiroki Ikoma, Pei-Chun Chien, and Ryota Kanai. 2024. Procbench: Benchmark for multi-step reasoning and following procedure. *arXiv preprint arXiv:2410.03117*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models.

AQ Jiang, A Sablayrolles, A Mensch, C Bamford, DS Chaplot, Ddl Casas, F Bressand, G Lengyel, G Lample, L Saulnier, and 1 others. 2024. Mistral 7b. arxiv 2023. *arXiv preprint arXiv:2310.06825*.

Mirella Lapata. 2006. Automatic evaluation of information ordering: Kendall's tau. *Computational Linguistics*, 32(4):471–484.

Bodhisattwa Prasad Majumder, Shuyang Li, Jianmo Ni, and Julian McAuley. 2019. Generating personalized recipes from historical user preferences. *arXiv preprint arXiv:1909.00105*.

Andres Marzal and Enrique Vidal. 2002. Computation of normalized edit distance and applications. *IEEE transactions on pattern analysis and machine intelligence*, 15(9):926–932.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, and 401 others. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Yinzhu Quan and Zefang Liu. 2024. Econlogicqa: A question-answering benchmark for evaluating large language models in economic sequential reasoning. *arXiv preprint arXiv:2405.07938*.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. *arXiv preprint arXiv:1906.02361*.

Joshua Robinson, Christopher Michael Rytting, and David Wingate. 2022. Leveraging large language models for multiple choice question answering. *arXiv preprint arXiv:2210.12353*.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, and 179 others. 2024. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Weizhi Wang, Hong Wang, and Xifeng Yan. 2023. Steps: A benchmark for order reasoning in sequential tasks. *arXiv preprint arXiv:2306.04441*.

Javier Yong, Haokai Ma, Yunshan Ma, Anis Yusof, Zhenkai Liang, and Ee-Chien Chang. 2025. Attackseqbench: Benchmarking large language models' understanding of sequential patterns in cyber attacks. *arXiv preprint arXiv:2503.03170*.

A Dataset Curation

A.1 Dataset Curation Prompt

We used LLaMA-3 models with the prompt shown in Figure 5 to curate a dataset of 5,000 samples. Each sample was processed in two independent runs, where the model was asked to determine whether the order of steps matters. We retained only those samples for which both runs returned a positive response ("yes"), indicating that step order is important.

A.2 Dataset Curation Examples

Some examples of data curation is provided in Figure 6.

B Task Details

B.1 Prompt

The prompt used for evaluation of the LLMs for step ordering is given in Figure 7.

B.2 Few Shot Examples Selections

The few shot examples in this experiment were chosen randomly out of the samples considered for few shot demponstartions. For 3 shot setting- 1 example with 6 steps, 1 example with 7 steps, 1 example with 8 steps were chosen. For 5 shot setting- 2 examples with 6 steps, 1 example with 7 steps, 2 examples with 8 steps were chosen. These fixed set of chosen examples were used for all evaluation.

C Experimental Details

The table shows the hyperparameters of the LLM model used for experimentation and their respective values. We have used 1 A100 GPUs.

Hyperparameter	Value
temperature	0.9
max_new_tokens	512
top_p	0.9

Table 2: Hyperparameter Values

D AI Assistance

We have used ChatGPT for writing assistance in the paper writing

Prompt For Data Curation:

You will be given a list of steps from a food recipe. Your task is to determine whether the steps must be followed in the given order, or if the steps can be rearranged without affecting the outcome.

Definition:
- steps_matter: "yes" → All steps depend on the previous ones and must be performed in order (e.g., you cannot bake before marinating).
- steps_matter: "no" → The steps are not fully order-dependent and can be rearranged without significantly affecting the outcome (e.g., you can prepare the sauce before or after cooking the meat).

Instructions:
- Analyze the sequence of steps.
- For each adjacent step pair, determine if the later step depends on the previous one.
- For example, if there are 4 steps, evaluate: (step 1 → step 2), (step 2 → step 3), (step 3 → step 4)
- If all transitions require the given order, return steps_matter = "yes".
- If even one pair can be reordered without affecting the result, return steps_matter = "no".
- Provide a detailed explanation justifying your conclusion, explaining whether each step depends on its predecessor.
- Return your output in the following JSON format **only**:

```
{
    steps_matter: <yes|no>,
    explanation: <elaborate explanation justifying the decision for each step transition>
}
```

- Do NOT include any other text in your output before or after the JSON, ONLY return the JSON Object.

Recipe Name: <recipe_name>
Recipe steps: <ordered_recipe_steps>

Figure 5: LLM prompt used for data curation

Data Curation Example 1:

1. 'rinse the quinoa in water using a strainer , as you would was rice',
2. 'add the quinoa to the water in a medium sized saucepan',
3. 'bring to the boil , reduce heat and gentley simmer for 5 minutes',
4. 'add the apples , raisins and cinnamon',
5. 'simmer until the water has absorbed',
6. 'serve with milk of your choice and sweeten with honey / rice syrup if desired']

LLM Decision:
Step Matter: Yes
LLM Reasoning: The quinoa must be cooked before adding the liquid, and the liquid must be added before the quinoa is tender. The quinoa must be drained before adding the soymilk, apples, sugar, and cinnamon. The mixture must be simmered before being transferred to a blender. Each step depends on the previous one, so the steps must be followed in the given order.

Data Curation Example 2

Steps:

1. 'in a bowl combine eggs , sugar , flour and milk',
2. 'grease a stoneware dish with all the butter',
3. 'pour in the dough and add fruit , raisins or almonds to taste', '
4. bake in the cold oven at 180c for about 60 minutes or until custard has set',
5. 'you might need to cover the dish after 40 minutes or so',
6. 'allow to stand and set for at least 10 minutes before serving']

LLM Decision:
Step Matter: No
LLM Reasoning: The steps can be rearranged without affecting the outcome. For example, step 2 (greasing the dish) does not depend on step 1 (combining the ingredients).

Figure 6: Examples of data curation

Model	Details	License
LLaMA-3.1	meta-llama/Llama-3.1-8B (Hugging Face)	llama 3.1
Mistral-7	mistralai/Mistral-7B-Instruct-v0.2 (Hugging Face)	apache-2.0
Gemma-2	google/gemma-2-9b-it (Hugging Face)	gemma
GPT-4o	gpt-4o-mini (OpenAI)	proprietary

Table 3: List of models used in our experiments.

Prompt Used For LLM Evaluation on Step Ordering Task of Food Recipes :

````
You are given a list of randomly shuffled cooking recipe steps. These steps are out of their intended logical order. Your task is to reorder them based on your understanding of typical recipe structure.
Cooking recipes usually follow a logical progression: starting with preparation, then actual cooking, and ending with final touches (e.g., serving, garnishing).

### Instructions:
- Analyze the shuffled steps and infer the most logical correct order.
- Return ONLY a JSON object in the exact format described below — no extra text or explanation.
- Do not renumber or reword the steps; return them as-is from the input, just reordered.
- You must respond with ONLY the JSON object — no explanations, comments, or markdown formatting. Your output will be parsed automatically, so format strictly.
### Output (JSON format only):
```json
{
"reordered_steps": [<step_1>, <step_2>, ..., <step_n>],
"order": [<index in the original shuffled list of step_1>, <index of step_2>, ..., <index of step_n>]
}
```

### Examples: Here are some examples:

Here's the input
### Input
Recipe Name: <recipe_name>
Recipe steps: <shuffled_recipe_steps>
````

Figure 7: LLM prompt used for step ordering