Preserving Step Order in Recipe Instruction Generations: Methods and Semantic Evaluation

Daniel Costa
University of California, Berkeley
daniel_costa@berkeley.edu

Abstract

LLM-generated recipe instructions have improved in quality as larger models continue to excel at reasoning tasks. However, even SoTA models struggle to generate instructions in a logical order that reflects the state of ingredients at each step. This is a barrier to their widespread adoption in the culinary space. My study compares a novel reference-based metric with an LLM-as-a-judge reference-free metric to explore the impact of Chain-of-Thought (CoT) prompting, retrieval-augmented in-context learning, and fine-tuning on a 7B parameter decoder only model's ability to preserve sequence order in generated recipe instructions. I find that CoT prompting provides measurable improvements over baseline for both metrics, with fine-tuning and few-shot prompting improving primarily with respect to reference-based metrics given diverse in-context examples.

1 Introduction

As large language models (LLMs) continue to advance, their ability to generate natural and fluent text has transformed a wide range of applications, including recipe generation. Automatically producing coherent cooking instructions from a list of ingredients offers value across domains, from consumer-facing apps to food industry automation. However, one persistent shortcoming remains: even state-of-the-art models frequently fail to preserve the correct sequential order of recipe steps, resulting in outputs where ingredients are used before they are prepared, and actions are logically misplaced. This is more than a cosmetic flaw - misordered instructions can make recipes confusing, inefficient, or even unusable.

The inspiration for this work comes from a combination of informal critiques, such as those seen in the New York Times Cooking evaluation of the practicality of GPT-3 generated recipes, and literature published by the creator of CulinAI, a pioneer in the consumer-facing recipe generation application space (H. Lee et al., 2020). As LLMs are increasingly used by cooks of all levels, coherent instruction generation will be essential to sustain trust in their use in dedicated applications.

While frontier models like GPT-4 have made impressive progress, deploying such models in practical, resource-constrained environments (e.g., mobile apps or edge devices) remains challenging. This motivates a closer examination of smaller, more efficient models - on the order of 10 billion parameters - that are better suited for low-latency, cost-effective inference.

To this end, my work explores whether reasoning-augmented prompting strategies and fine-tuning can help smaller models generate more logically ordered instructions. Specifically, I use both reference-based and reference-free metrics to study the impact of Chain-of-Thought (CoT) prompting, retrieval-augmented few-shot learning, and lightweight supervised QLoRA fine-tuning on a 7B parameter instruction-tuned model's ability to preserve instruction sequence and coherence.

2 Background

Prior work has largely focused on improving output quality via lexical metrics (e.g., BLEU (Mohbat and Zaki, 2025), ROUGE (Taneja et al., 2024)) without explicitly targeting the preservation of procedural structure. While some studies like those by H. Lee et al., 2020 used normalized tree edit distance (NTED) to evaluate the structure of the generated recipe against the reference text, this approach heavily penalizes semantically equivalent yet lexically divergent test examples. Given the variety of recipe sources used in standard datasets such as Recipe1M+ , consistently evaluating the procedural integrity of generated recipe text across a wide array of recipe styles requires a semantic approach (Marin et al., 2019).

With regards to fine-tuning vs. CoT prompt-

Table 1: *Bounds on dataset attributes after cleaning*

Attribute	Ingredients	Steps	Mean Instruction Tokens	Total Instruction Tokens
Lower Limit (5th percentile)	4	3	5.2	15
Upper Limit (95th percentile)	16	17	44	237

ing, some studies have found that fine-tuning smaller T5 models on reasoning tasks outperforms GPT-3 with few-shot CoT prompting (Min et al., 2022). However, others have found that few-shot CoT prompting outperforms fine-tuned models in generalization-heavy tasks (Yao et al., 2023) (Huang et al., 2022). Within the context of entity-tracking tasks such as recipe generation, smaller language models like Flan-T5-XL (3B params) have been shown to implicitly track object states only after being fine-tuned (Kim and Schuster, 2023). The best method for improving reasoning in long-form generation tasks is domain-dependent, and evaluating recipe instruction generation's position in this space is a central task of this study.

Other works have seen success using semantic similarity to retrieve few-shot examples similar to the test prompt at inference time (Gedeon, 2025), while others consider syntactic structure to improve the generalization of models prompted in this way (Wang et al., 2024). Progress in this area indicates that retrieval may be a valuable tool for improving the structure of recipe instructions.

3 Methods

3.1 Data/Cleaning

Recipe data comes from RecipeNLG (Bień et al., 2020), which is a cleaned and expanded version of the standard Recipe1M+ dataset Salvador et al., 2017 used extensively in recipe NLP literature (Marin et al., 2019) (Liu et al., 2022)(Sakib et al., 2022). This dataset shows improved BLEU, GLEU, and WER scores on recipe generation tasks as compared to the base Recipe1M+ dataset. Due to the aims and capacity constraints of this study, I performed a number of cleaning operations on the original dataset.

The full RecipeNLG dataset contains 2.23M deduplicated records, each with a recipe title, ingredient/portion list, instructions, pre-extracted NER values, and source URL. Since this study aims to evaluate the sequence order preservation of generated instructions, only recipes with well-formed lists of at least 3 instructions were kept. Additionally, to constrain the form of the reference text, records were removed if they were in the bottom or top 5th percentiles with respect to number of ingredients, instruction steps, average instruction step tokens, and total instruction tokens. Recipe sources that showed consistently incoherent, grammatically incorrect, or underdeveloped recipe instructions such as food.com or cookbooks.com were omitted to preserve the quality of the reference dataset.

At the end of the cleaning process, the dataset consisted of 380K high-quality, well-formed recipes that were randomly split into validation and test sets each of size 1K and a retrieval set of 10K examples for in-context learning (ICL), with the rest available for training.

3.2 Model, Fine-Tuning, and Inference

Model selection considered the combined aims of evaluating fine-tuning, CoT prompting, and few-shot learning, as well as the constraints of my computing environment. With respect to model size, other studies have found reliable results from CoT prompting models with as few as 3B parameters, and even found improved results after fine-tuning and few-shot prompting (Kim et al., 2023). As far as the type of model, (Kojima et al., 2023) show that instruction-tuned models are best in a CoT paradigm. As structure and sequence is particularly important for coding tasks, I also searched for models trained to generate code. Among HuggingFace models fitting these criteria, the OPENHERMES-2.5-MISTRAL-7B model fine-tuned on code instruction data showed outstanding performance on both coding and non-coding benchmarks as compared to its base model MISTRAL-7B-V0.1.

I fine-tuned this model using 4-bit quantization and a LoRA adapter with dimension 8. The LoRA adapter was applied to the query and value modules of the top layers (29-32) of the model. The model was trained using the HuggingFace library on a random sample of 1K records from the training set for 2 epochs, with batch size 32 and final validation perplexity of 205.88 (see results for discussion of perplexity significance). Details of my hyperparameter search appear in Appendix Appendix B.

For CoT fine-tuning, Kim et al., 2023 found success using a 12B parameter teacher model to aug-

ment 1.84M reasoning examples for fine-tuning student models. Again, given the constraints of this computing environment, I use a QWEN2.5-14B-INSTRUCT-1M teacher model trained on code reasoning tasks to create a training set of 1000 reasoning examples on which to fine-tune OPENHERMES-2.5-MISTRAL-7B for reasoning generation. Training inputs included title and ingredient list, with recipe instruction reasoning as outputs. The second call for this experiment class was done with a model fine-tuned to accept title, ingredients, and reasoning and produce instructions.

For all experimental inference, the maximum number of words in recipe instructions in the training set was 237, so max_new_tokens was set to 256. I used a temperature of 0.7, top_p of 0.9, and repetition penalty of 1.1 for all experiments. See appendix Appendix C for examples of prompt formatting.

4 Experiments

I analyzed 4 treament variables: fine-tuned (FT) vs. non-FT, normal prompting vs. CoT prompting, 0, 1, 2, and 3-shot prompting, as well as 1, 2, and 3-shot prompting with retrieved high-similarity reference examples.

Non-retrieval non-COT few-shot experiments included fixed, randomly selected examples from the retrieval set. For retrieval experiments, I gathered examples for each test recipe in three steps. First, I formed a sentence representation of each test and retrieval recipe using only the prompt input elements: title and ingredient list. Second, I embedded these sentences using a lightweight MULTI-QA-MINILM-L6-COS-V1 sentence transformer model. Finally, I computed each test example's nearest neighbors in the retrieval set by performing a cosine similarity search on a HuggingFace dataset with an FAISS index on the embedding column.

I implemented CoT inference in two calls: one to generate reasoning, and one to generate recipe instructions from the first call's generated reasoning. This ensured a consistent number of recipe instruction tokens across experiments. Wei et al. 2022 suggest that prompting with both CoT and few-shot examples performs worse than either alone. To test this, I ran experiments on the non-fine-tuned CoT-prompted model with few-shot output examples. The predicted degradation in performance materialized as shown in Tables 2 and 3, and I did not conduct any further few-shot CoT experiments.

5 Evaluation

Choosing an evaluation metric to adequately represent the sequential order of recipe instructions proved challenging, as there are many ways to represent a given set of actions. Reference and generated text can have dissimilar word choices and significantly different numbers of steps, and yet two cooks following either set of instructions could produce the exact same result.

Informed by this observation and similar ones throughout the literature, I did not use word-based methods such as ROUGE-L, ROUGE-W, and NTED to evaluate experiment results. A study by Bhandari et al., 2020 demonstrated that word-based metrics correlate poorly with human judgments on abstractive tasks such as recipe generation, while another by Gao et al., 2022 concluded the same for procedural text. Surveys of evaluation metrics have suggested a semantic approach within this study's domain (Liu et al., 2017)(Bhandari et al., 2020).

I propose a metric inspired by the Regressor Using Sentence Embeddings (RUSE) metric for evaluating sentence-level semantic order preservation (Shimanaka et al., 2018). For each reference-generated recipe pair in the test set, I construct an injective mapping $f : \{1, 2, \ldots, |G|\} \rightarrow \{1, 2, \ldots, |R|\}$ from the indices of the generated instructions G to those of the reference instructions R. This mapping is done by mapping the element in G to its nearest neighbor in R using the cosine similarity of their sentence embeddings. I then compute the spearman rank correlation between $[1, 2, \ldots, |G|]$ and $[f(G)]$ as a measure of how well instruction order and cardinality was preserved. I refer to this metric as Mean Instruction Spearman Correlation (MISC).

The advantage of MISC is that its semantic approach avoids the issues that plague word-based metrics, and prevents different styles/levels of expressivity from penalizing otherwise equivalent instructions. The disadvantage of this metric is that the grouping (by numbered instruction groups) it imposes on the text penalizes syntactically equivalent instructions that have different structures. However, the magnitude of this penalty in the worst case is preferable to the converse, where ROUGE-L heavily penalizes semantically equivalent examples. See appendix Appendix A.1 for explicit examples and analysis.

Liu et al., 2017 also suggested using discriminative reference-free metrics when evaluating long-

			Non-Retrieval			Retrieval		
		0-shot	1-shot	2-shot	3-shot	1-shot	2-shot	3-shot
Non-CoT								
	Non-FT	.4311	.4873	.5253	**.5592**	.4992	.5535	.5346
	FT	.5491	.5740	**.5756**	.5646	.5664	.5700	.5502
CoT								
	Non-FT	**.4468**	.3710	.4239	.3270	.3738	.3734	.3683
	FT	**.4107**	-	-	-	-	-	-

Table 2: MISC scores on test set for all experiments on a OPENHERMES-2.5-MISTRAL-7B. Best scores for each experiment class are bolded.

			Non-Retrieval			Retrieval		
		0-shot	1-shot	2-shot	3-shot	1-shot	2-shot	3-shot
Non-CoT								
	Non-FT	7.414	7.032	7.404	**7.532**	6.897	7.077	7.16
	FT	6.930	**7.258**	7.220	7.228	6.949	6.840	6.838
CoT								
	Non-FT	**7.708**	6.561	6.343	6.347	6.643	6.158	6.240
	FT	**6.990**	-	-	-	-	-	-

Table 3: LLM-as-a-judge scores on test set for all experiments on a OPENHERMES-2.5-MISTRAL-7B, as judged by QWEN2.5-CODER-32B-INSTRUCT on a 1-10 scale. Best scores for each experiment class are bolded.

form generations. Upon analyzing outputs like that of Example 1 in Appendix Appendix A.1, it became apparent that the strong performance of the fine-tuned models could be partially explained by the use of a reference-based metric. To gain a more complete understanding of the implications of this metric for evaluating instruction sequence, I also used LLM-as-a-judge to serve as a reference-free evaluator of the stand-alone logical coherence of recipe instructions. Hu et al., 2025 provide a comprehensive overview of best practices associated with employing LLM judges, such as fine-tuning smaller judges across an array of evaluation tasks and prompting for both a numeric score and explanation. While typical judges are trained using data from teacher models on the order of 1.7T parameters, this is not feasible on a single 40GB RAM A100 GPU. I searched for models small enough to load in memory but large enough to exceed the capabilities of the 7B parameter experimental model, while also prioritizing instruction models with concise, focused output. Within the HuggingFace platform, QWEN2.5-CODER-32B-INSTRUCT fit these criteria best.

6 Results and Discussion

Below are some key findings from my experiments as well as analyses of the results.

More shots are not always better for smaller models

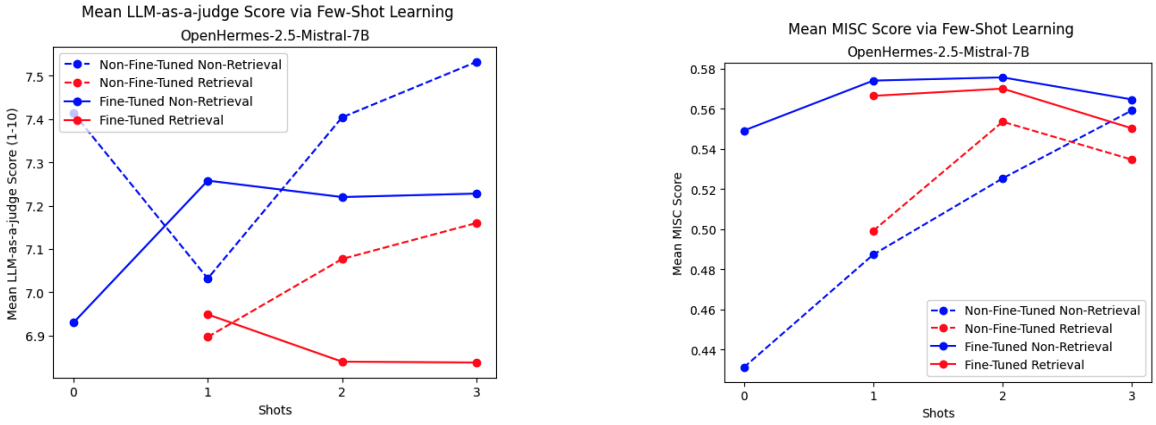
In line with findings from Brown et al., 2020, smaller language models like the Mistral 7B model can benefit from from few-shot prompting, but this benefit is neither monotonic nor unbounded with respect to the number of examples.

For MISC scores, besides the non-fine-tuned non-retrieval experiments, 2-shot learning seems to offer the best balance between prompt complexity, information gain, and generalizability. This difference is especially evident in the retrieval case. The semantically similar examples provide the simultaneous forces of more examples giving more information while also allowing more potential to overfit to the similar examples.

With regards to MISC, retrieval hurts performance shot-for-shot on the fine-tuned experiments, and mostly improves on non-fine-tuned experiments (see Table 2). In the retrieval case for the fine-tuned experiments, there may be a mismatch between the input similarity on which the examples were retrieved and the output similarity on which the model was scored. Fine-tuning has already taught the model to mimic the training distribution better, but these few-shots may cause the fine-tuned model to adjust its output too similarly to an output sequence it encountered during training.

Teacher-tuned CoT prompting increased expressiveness and degraded performance

Fine-Tuning CoT reasoning on decoder-only models like those from Qwen and Mistral helps models to imitate the teacher's style more than develop reasoning skills. Thus, the student models'



(a) (b)

Figure 1: **a)** Mean LLM Judge Score across experiment types and number of shots. While non-FT performance scales with the number of prompt examples, this does not result in a significant improvement over 0-shot baselines. **b)** Mean MISC scores on the same dimensions. Fine-tuning and few-shot learning confer a much clearer benefit on this reference-based objective.

ability to generate helpful CoT reasoning text is inherited from the teacher LLM. Upon closer inspection, this finding seems to be more a fault of the metrics used than the quality of the reasoning or instruction generation.

LLM-as-a-judge had subjective and even incorrect explanations for why it penalizes generations. There were many instances in which the larger teacher models taught student models to diverge from a more conservative, minimalist interpretation of the recipe instructions to add dimensions like texture or depth of flavor to a dish. The LLM judge describes these additions as filler that detract from the logical coherence of the instructions, even if the extra steps enhance the dish as in the examples in Appendix Appendix A.2.

Added expressiveness also lowered MISC scores, as extra details may not correlate well with the less verbose reference text. For the fine-tuned CoT examples, the mean number of steps for the reference and generated instructions were 6.926 and 8.435 respectively (+21.8%), and the mean word counts were 91.44 and 118.59 respectively (+29.7%). Beyond just additional details, though, the MISC evaluation will penalize the model even for *correcting* a step that was sub-optimally sequenced in the reference, as in Example 2 in Appendix Appendix A.1.

Non-CoT examples did worse on LLM-as-a-judge

CoT-Prompting had a similar improvement over baseline according to both MISC and the LLM judge. MISC score was 3.64% better (0.4468 vs. 0.4311) , and LLM score was 3.96% better (7.7075 vs. 7.414). However, the non-CoT experiments were uniformly rated lower by the LLM judge in comparison with the baseline CoT experiment.

Non-fine-tuned CoT-prompted output is rated higher by the LLM judge because instruction-tuned models are biased towards structure, which CoT-prompted answers naturally exhibit (Kim et al., 2023). Goel et al., 2025 finds that LLM judges significantly favor responses from models whose training data most closely resembles their own. Fine-tuning or seeing few-shot examples from the RecipeNLG dataset distribution causes the non-CoT data exposure to deviate from that of the LLM judge. As evidence of this, the LLM judge cited the clear "step-by-step" structure of generated instructions for 27.3% of CoT-prompted outputs, compared to just 14.5% for non-CoT-prompted outputs.

LLM judges lack internal consistency

The overall structure of the generated instructions, regardless of content, likely accounts for some of the improvement over the baseline from CoT prompting. This improvement could also contain some genuine signal according to the LLM judge, but the lack of internal consistency found in LLM-as-a-judge provides insufficient evidence for such a signal.

After conducting 30 separate ratings of the same 1000 reference examples in the test set, 30 mean scores were recorded, with a mean of 7.983 and between-experiment standard deviation 0.17. Com-

		0-shot Baseline	Retrieval		Non-Retrieval	
MISC	FT	.5491	.5700	(+3.81%)	.5756	(+4.83%)
	Non-FT	.4311	.5535	(**+28.39%**)	.5592	(**+29.71%**)
LLM Judge	FT	6.93	6.949	(+0.27%)	7.258	(+4.73%)
	Non-FT	7.414	7.16	(-3.43%)	7.532	(+1.59%)

Table 4: *Maximum ICL improvement over 0-shot baseline by retrieval and fine-tuning, summarized from Tables 2 and 3. Non-FT experiments saw largest improvements on reference-based MISC scores.*

paring the 7.477 baseline 0-shot LLM score and 7.707 CoT score by treating these means as an estimate of the distribution of the LLM judge scoring mean, a two-tailed z-test yields $p = 0.347$. Thus, the improvements over baseline from applying CoT prompting should be viewed with caution.

Few-shot metric improvements

Few-shot examples are more crucial for non-fine-tuned models only for MISC scores, as MISC is reference-based and these models benefit greatly from exposure to the output distribution. The LLM judge was not as impressed by few-shot examples, again perhaps because the structure that the human-generated examples impose on the output differ from what it expects based off its training data. Looking at the LLM judge scores in Table 3 and keeping in mind the between-experiment standard deviation of 0.17, these improvements over baseline are not significant enough to draw a conclusion as to the performance of few-shot prompting on reference-free evaluation metrics.

Perplexity does not correlate with top-line metrics in long-form generation tasks

Despite a perplexity of 124.29 for the 1-layer fine-tune at 2 epochs and 205.88 for the 4-layer fine-tune at 2 epochs, the 4-layer fine-tuned model had a higher MISC score (0.4923 vs. 0.5028). Perplexity - both in an absolute and relative sense - is not always predictive of top-line metric performance for long-form generation. This can clearly be seen in Figure 1b of (Fang et al., 2025), which describes how per-token perplexity cannot capture the uncertainty present in tasks that are evaluated over longer contexts.

7 Conclusion

This study investigated strategies to improve the logical sequencing of LLM-generated recipe instructions—a critical but often overlooked aspect of culinary text generation. Results suggest that CoT prompting provides measurable improvements over baseline, though its effectiveness is heavily dependent on the quality and alignment of the teacher model's reasoning style. Few-shot prompting benefits non-fine-tuned models more consistently, while fine-tuned models may suffer from exposure bias when shown overly similar examples.

While semantic evaluation metrics like the ones explored in this study offer better alignment with human judgment than traditional word-based methods, my results reveal limitations in the ability of these metrics to capture sequential coherence using tools at this scale. The amorphous nature of recipe instructions and the fact that even the most prominent datasets in the culinary text generation domain contain ambiguities and sub-optimally sequenced examples presents a challenge for evaluating model output with reference-based metrics.

Ultimately, the success of many methods employed in this study hinges on using larger teacher models, training on a larger volume of data, and fine-tuning with reasoning generated for a wide array of tasks. Promise may lie in adapting larger scale LLM judges fine-tuned on tasks external to recipe generation for reference-free evaluation, however even this comes with known limitations. Still, efforts such as FoodKG by Haussmann et al., 2019 to impose a standardized evaluation topology in the form of a ground-truth knowledge graph could augment the LLM evaluation process and reduce variance in measurement. Taken together, these insights underscore the need for more scalable, task-agnostic evaluation frameworks and richer reasoning-aware training pipelines to advance the reliability of structured generation across domains.

References

Manik Bhandari, Pranav Narayan Gour, Atabak Ashfaq, Pengfei Liu, and Graham Neubig. 2020. Re-evaluating evaluation in text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9347–9359, Online. Association for Computational Linguistics.

Michał Bień, Michał Gilski, Martyna Maciejewska, Wojciech Taisner, Dawid Wisniewski, and Agnieszka Lawrynowicz. 2020. RecipeNLG: A cooking recipes dataset for semi-structured text generation. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 22–28, Dublin, Ireland. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Lizhe Fang, Yifei Wang, Zhaoyang Liu, Chenheng Zhang, Stefanie Jegelka, Jinyang Gao, Bolin Ding, and Yisen Wang. 2025. What is wrong with perplexity for long-context language modeling?

Shen Gao, Haotong Zhang, Xiuying Chen, Rui Yan, and Dongyan Zhao. 2022. Summarizing procedural text: Data and approach. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2216–2225, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Máté Gedeon. 2025. Retrieval-enhanced few-shot prompting for speech event extraction.

Shashwat Goel, Joschka Struber, Ilze Amanda Auzina, Karuna K Chandra, Ponnurangam Kumaraguru, Douwe Kiela, Ameya Prabhu, Matthias Bethge, and Jonas Geiping. 2025. Great models think alike and this undermines ai oversight.

Helena H. Lee, Ke Shu, Palakorn Achananuparp, Philips Kokoh Prasetyo, Yue Liu, Ee-Peng Lim, and Lav R. Varshney. 2020. Recipegpt: Generative pre-training based cooking recipe generation and evaluation system. In *Companion Proceedings of the Web Conference 2020*, WWW '20, page 181–184. ACM.

Steven Haussmann, Oshani Seneviratne, Yu Chen, Yarden Ne'eman, James Codella, Ching-Hua Chen, Deborah Mcguinness, and Mohammed Zaki. 2019. *FoodKG: A Semantics-Driven Knowledge Graph for Food Recommendation*, pages 146–162.

Renjun Hu, Yi Cheng, Libin Meng, Jiaxin Xia, Yi Zong, Xing Shi, and Wei Lin. 2025. Training an llm-as-a-judge model: Pipeline, insights, and practical lessons. In *Companion Proceedings of the ACM on Web Conference 2025*, WWW '25, page 228–237. ACM.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models.

Seungone Kim, Se June Joo, Doyoung Kim, Joel Jang, Seonghyeon Ye, Jamin Shin, and Minjoon Seo. 2023. The cot collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large language models are zero-shot reasoners.

Chia-Wei Liu, Ryan Lowe, Iulian V. Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2017. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation.

Yinhong Liu, Yixuan Su, Ehsan Shareghi, and Nigel Collier. 2022. Plug-and-play recipe generation with content planning. In *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 223–234, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Javier Marin, Aritro Biswas, Ferda Ofli, Nicholas Hynes, Amaia Salvador, Yusuf Aytar, Ingmar Weber, and Antonio Torralba. 2019. Recipe1m+: A dataset for learning cross-modal embeddings for cooking recipes and food images.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Fnu Mohbat and Mohammed J Zaki. 2025. KERL: Knowledge-enhanced personalized recipe recommendation using large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19125–19141, Vienna, Austria. Association for Computational Linguistics.

Nazmus Sakib, G M Shahariar, Md.Mohsinul Kabir, Md Kamrul Hasan, and Hasan Mahmud. 2022. Assorted, archetypal and annotated two million (3a2m) cooking recipes dataset based on active learning.

Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marín, Ferda Ofli, Ingmar Weber, and Antonio Torralba. 2017. Learning cross-modal embeddings for cooking recipes and food images. pages 3068–3076.

Hiroki Shimanaka, Tomoyuki Kajiwara, and Mamoru Komachi. 2018. RUSE: Regressor using sentence embeddings for automatic machine translation evaluation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 751–758, Belgium, Brussels. Association for Computational Linguistics.

Karan Taneja, Richard Segal, and Richard Goodwin. 2024. Monte carlo tree search for recipe generation using gpt-2.

Qianlong Wang, Hongling Xu, Keyang Ding, Bin Liang, and Ruifeng Xu. 2024. In-context example retrieval from multi-perspectives for few-shot aspect-based sentiment analysis. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8975–8985, Torino, Italia. ELRA and ICCL.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models.

Appendix A Metric Analysis

Appendix A.1 MISC

Consider the following edge-case which questions my metric's ability to reflect proper instruction sequencing. Assume the reference text has two steps with two sentences each, while the generated text is the same word for word, but has four steps with one sentence each. The mapping f would be $f(1) = f(2) = 1$, $f(3) = f(4) = 2$. Then, the Spearman correlation between $[1, 2, 3, 4]$ and $[1, 1, 2, 2]$ would be 0.894, demonstrating that the instructions' differing partition schemes did somewhat affect their evaluation scores. On the one hand, this is inconvenient for my metric's ability to capture the true essence of the instructional text. On the other, this feature of my metric is useful in comparing not just the order of steps, but the overall structure of the model output compared to human-written recipes.

However, Figure 2 illustrates that the potential harms of this effect are constrained. Furthermore, consider the example illustrated in Figure **??**. The ROUGE-L F1 score for this pair is 0.23, which would place it in the bottom 20% of evaluated examples across experiments. MISC, however, gives this pair a perfect score, and reflects the practical equivalence of this instruction pair.

While MISC has its limitations, the magnitude of its errors are preferable to that of word-based metrics like ROUGE.

Generated	Reference
1: Set your oven to 375 degrees Fahrenheit.	1: Preheat the oven to 375 F.
2: Combine eggs, sweetener, and flour inside mixer.	2: Mix flour, sugar, and eggs in a bowl.
3: Transfer mixture to buttered baking dish.	3: Pour the batter into a greased pan.
4: Cook twenty-five minutes in the oven.	4: Bake for 25 minutes.

Example 1: Semantically similar recipes perfectly scored by MISC and heavily penalized by ROUGE.

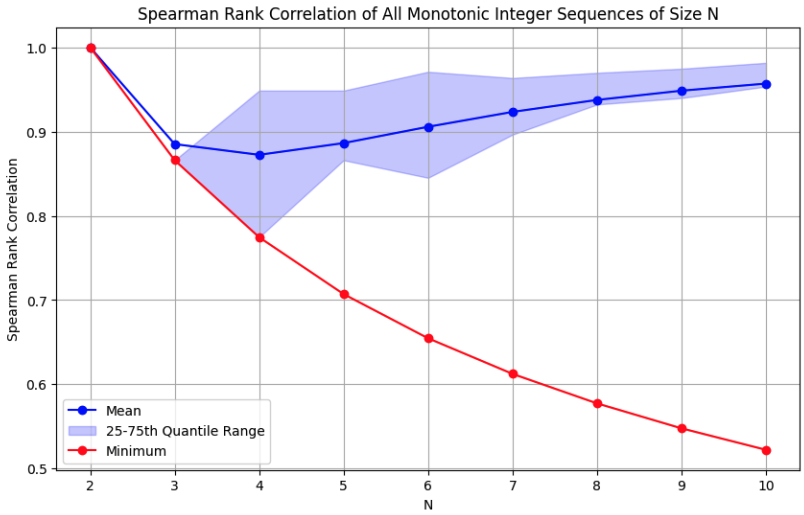
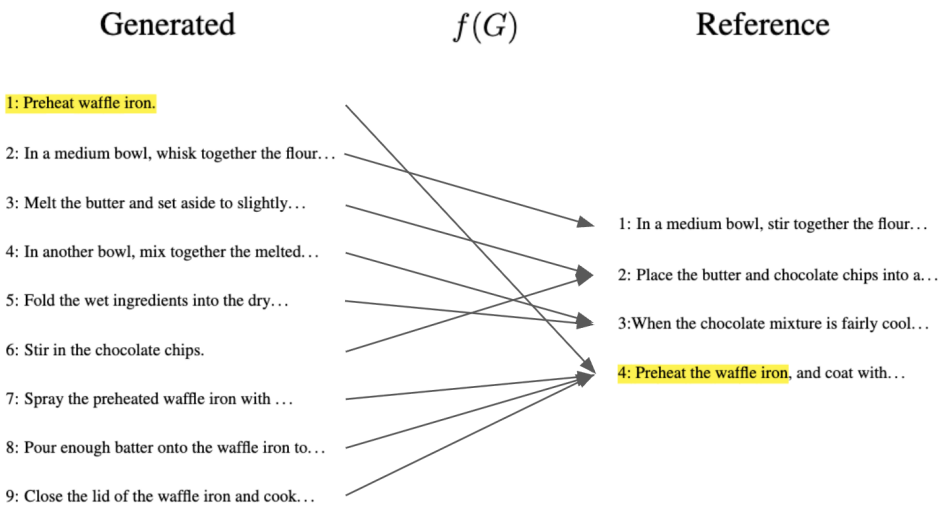


Figure 2: Although the MISC metric can penalize syntactically equivalent text for differences in structure, in practice this effect is relatively minor. If a generated text is syntactically equivalent to its reference, its step order under its semantic mapping will be an increasing monotonic positive integer sequence of size $|G|$ consisting of elements from $\{1, 2, \ldots, |R|\}$. This figure visualizes the distribution of Spearman rank correlation between all such sequences of size N and the sequence $[1, 2, \ldots, N]$for $N \in [2, 10]$. While worst case performance decays exponentially, average case typically exceeds 0.9 and actually increases with N.



Example 2: This example from the baseline model serves both to illustrate how MISC maps between generated and reference instructions, and to reveal a central challenge of using reference-based metrics to evaluate model outputs. The generated instructions G contain 9 steps, compared to 4 steps in the reference R. Arrows indicate the mapping $f : G \rightarrow R$ for this example. Despite mostly adhering to the reference's step sequence, MISC penalizes the generated text for suggesting to preheat the waffle iron at the beginning as opposed to the end. However, it may be preferable to pre-heat a waffle iron before mixing ingredients to account for heating time (an evaluation by GPT-4o concurs). While the MISC score for this example is 0.4831, removing just the first generated step from the mapping yields 0.8648 - a drastic penalty given that GPT-4o actually favored the generated example.

Appendix A.2 LLM-as-a-Judge

In the LLM judge's explanation for why it gave one fine-tuned CoT-prompted output a low score, it cited that the generated instructions included "a step to toast the rice, which could alter the texture and flavor of the patties, making the recipe less effective.", giving the generated instructions a 6/10, compared to 9/10 it gave the non-fine-tuned model's generations on the same example. After prompting GPT-4o with both the generated example and the LLM judge's evaluation, GPT-4o approved the toasting of rice to achieve "enhanced flavor through Maillard browning" and gave the generated instructions a 9/10.

Similarly, another fine-tuned CoT-prompted output asked the cook to form a loaf of bread both in a bread machine as well as a loaf pan, whereas a non-FT CoT-prompted model only mentioned a bread machine. The LLM judge docked the fine-tuned output 6 points compared to the non-fine-tuned output simply for using an "unnecessary" loaf pan. Once again, GPT-4o disagreed, noting that use of the loaf pan was "crucial for achieving a fluffy texture and well-formed crust."

These examples illustrate how verbosity can often hurt fine-tuned CoT outputs' LLM judge scores.

Appendix B Fine-Tuning Hyperparameter Search

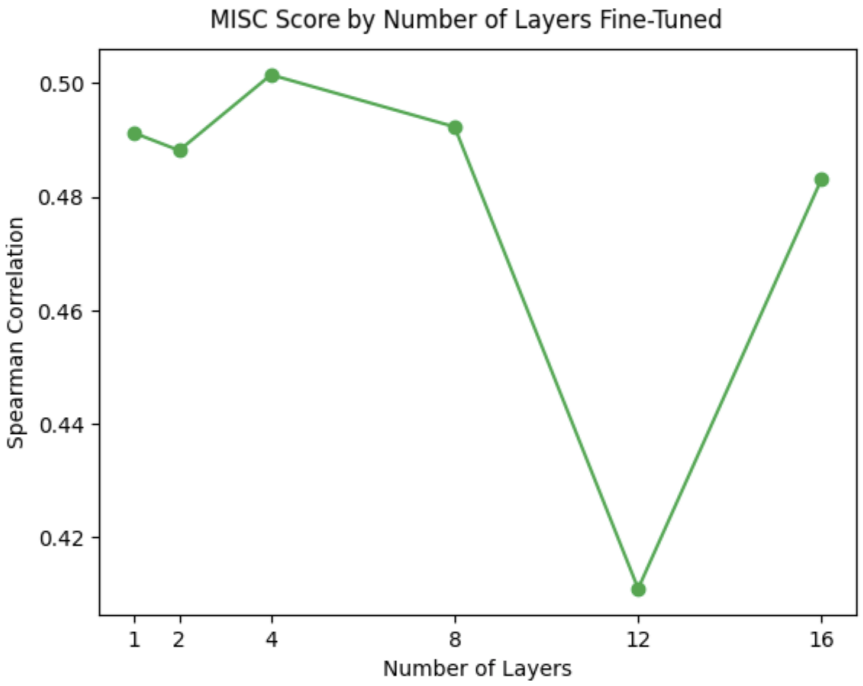


Figure 3: MISC scores for test outputs of OPENHERMES-2.5-MISTRAL-7B trained on various numbers of transformer layers (out of 32 total layers). Each model was trained on 1K examples for 2 epochs.

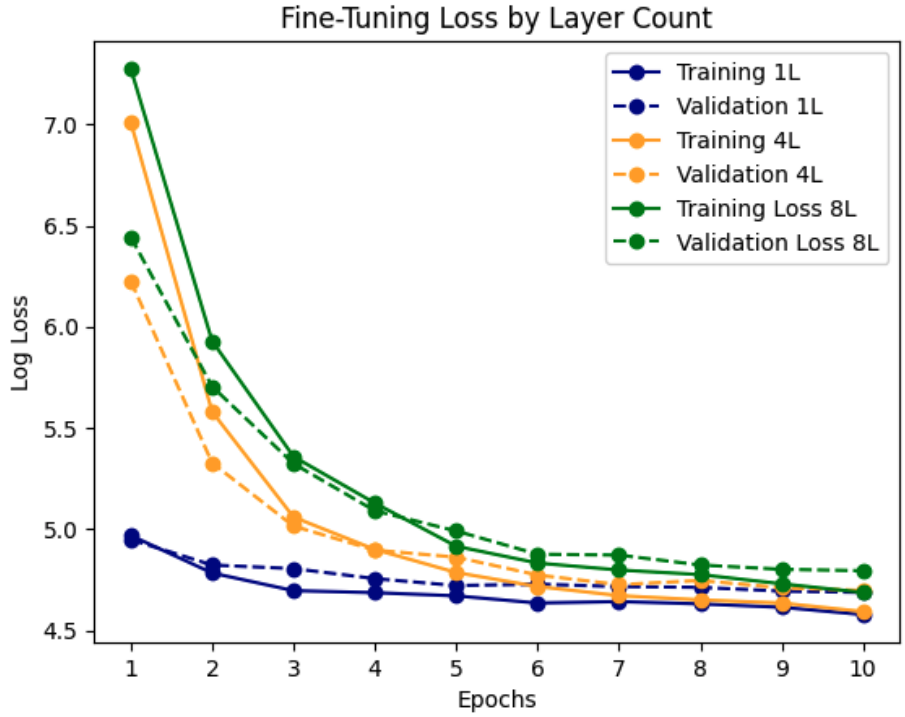


Figure 4: Training loss (on 1K examples) for the 3 QLoRA configurations with highest MISC-objective scores. After further hyperparameter tuning, the final experimental model used the optimal 4-layer 2 epoch configuration, highlighting that models with high perplexity can still perform well in long-form generation tasks.

Appendix C Experiment Prompts

LLM Evaluation Prompt

```
<|im_start|>system
You are a professional chef and expert in culinary instruction writing.<|im_end|>
<|im_start|>user
Your task is to evaluate a set of recipe instructions generated by an AI model.

You should assess the instructions based on the following criteria:
1. Whether the steps are presented in a logical and executable order
2. Whether all ingredients in the recipe were used at least once
3. Whether ingredients are prepared before they are used
4. Whether all key steps are included and nothing important is missing
5. Whether the instructions are free of contradictions or redundancy

After reading the instructions, provide:

- A coherence rating from 1 (poor) to 10 (excellent)
- A brief explanation justifying your score, citing specific strengths or issues
 ---
 Recipe Prompt:
 {original_recipe_prompt}
 ---
 Model's' Instructions:
 {generated}
 ---
 Your Evaluation:
[1] Coherence Rating (1-10):
[2] Explanation (2-5 sentences):

Prepend your answer to the items in these last 2 lines with [1] and [2] respectively.
<|im_end|>
```

Few-shot Prompt

```
Fill in the missing Recipe Instructions. Be sure to number each step. After the following examples,
provide your response.
{examples}
Recipe Title: {title}
Ingredients: {ingredients}
Recipe Instructions:
```

Reasoning Transfer Prompt

```
Write recipe instructions for the following recipe title and ingredient list.
Use the provided reasoning to inform your response.
Enclose your recipe instructions in <instructions> tags and number each step.
Recipe Title: {title}
Ingredients: {ingredients}
Reasoning: {reasoning}
Your response:
```

CoT Prompt

```
Before writing the recipe instructions, first think step by step through the logic for all
ingredients: what ingredients are needed, in what order tasks must be done, and how long each
step takes.
Are there any dependencies like marinating, chopping, or cooking that must be done before certain
steps?
Only generate the reasoning in this response, the recipe instructions will be generated in a
later response.
Recipe Title: {title}
Ingredients: {ingredients}
Reasoning:
```

System Prompt

```
You are an expert in generating recipe instructions from the recipe title and ingredient list.
```

Base Prompt

```
Fill in the missing Recipe Instructions. Be sure to number each step.
Recipe Title: {title}
Ingredients: {ingredients}
Recipe Instructions:
```