TTS-Optimized Language Generation via Reinforcement Learning from Synthetic Preference Signals

Jake Bodea
CS329H: Machine Learning from Human Preferences
Stanford University
TaxRise Inc.
jpbodea@stanford.edu, jake.bodea@taxrise.com

Abstract

Large language models generate responses with bulleted lists and structured formatting optimized for visual consumption, creating unnatural experiences when converted to speech for voice interfaces. We demonstrate that explicit prompting fails to reliably prevent list generation, motivating an alignment-based approach. Using customer service transcripts from TaxRise, a tax resolution company, we construct synthetic preference pairs by having an LLM rewrite list-heavy outputs into TTS-friendly prose with natural discourse markers. Interpreting these rewrites as implicit preference signals inspired by coactive learning, we train Llama 3.2 1B-Instruct using Group Relative Policy Optimization (GRPO) against a reward function that penalizes list structures while encouraging conversational flow. Our fine-tuned model achieves a 66.7% reduction in list usage rate (from 30.6% to 10.2%) while increasing discourse marker usage, demonstrating that preference learning can effectively address modality-specific stylistic constraints where prompting alone fails.

1 Introduction

Voice interfaces are becoming increasingly prevalent across customer service, healthcare, education, and accessibility applications. As organizations deploy large language models (LLMs) to power these interfaces, a critical usability problem emerges: LLMs are trained primarily on written text and generate outputs optimized for visual consumption. When these outputs are converted to speech via text-to-speech (TTS) systems, the result is often unnatural and difficult to follow.

A particularly problematic pattern is the overuse of bulleted and numbered lists. Consider a customer service agent explaining a multi-step process: a written response might naturally present this as a bulleted list, which is easy to scan visually. However, when spoken aloud, phrases like "bullet point one," "dash," or awkward pauses between items create a robotic, disjointed experience that undermines user trust and comprehension.

Despite explicit prompting to avoid lists and use conversational language, LLMs persistently generate structured formatting. Our preliminary experiments showed that even with detailed instructions and few-shot examples demonstrating conversational style, models still frequently produce lists. This suggests that stylistic preferences for modality-appropriate generation require deeper alignment than prompt engineering alone can provide.

We frame TTS-optimized generation as a preference learning problem. Our approach consists of three stages: (1) collecting customer service transcripts where the LLM generated list-heavy responses, (2)

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

using a separate LLM to rewrite these responses into natural prose as synthetic preference signals, and (3) fine-tuning the base model using reinforcement learning to prefer the conversational style.

This work connects to the broader literature on learning from implicit feedback. Tucker et al. (12) demonstrate that user edits to model outputs can serve as preference signals in a coactive learning framework. While we use synthetic LLM edits rather than human edits for scalability, the underlying principle remains: learning from improved versions of outputs rather than explicit ratings. Our contribution demonstrates that this approach can effectively address modality-specific stylistic constraints, specifically TTS-friendliness, rather than general alignment objectives like helpfulness or harmlessness.

A brief note on data availability: the customer service transcripts and fine-tuned weights are proprietary to TaxRise and hosted within their Databricks environment; all examples here are anonymized or synthetic, and code/reward scripts are in the public repository (see Data Collection and Section A.2).

1.1 Contributions

- We identify and characterize the list-generation problem for TTS applications, showing that prompting-based approaches are insufficient.
- We construct a dataset of synthetic preference pairs from real customer service transcripts, where LLM rewrites serve as implicit preference signals for TTS-friendly style.
- We demonstrate that reinforcement learning with a reward function penalizing list structures can effectively align a language model for TTS-appropriate generation while preserving semantic content.
- We document our training pipeline and evaluation framework to support future work on modality-specific alignment.[1]

2 Related Work

2.1 Preference Learning and RLHF

Reinforcement Learning from Human Feedback (RLHF) has become the dominant paradigm for aligning language models with human preferences (2; 5). The standard pipeline involves collecting human preference data, training a reward model, and fine-tuning the policy using Proximal Policy Optimization (PPO) (8).

Direct Preference Optimization (DPO) (6) reformulated this objective to enable direct policy optimization without explicit reward model training. DPO assumes preferences follow the Bradley-Terry model (1), where the probability of preferring response y_1 over y_2 is determined by the difference in their latent utilities. Our work builds on this preference modeling perspective but uses reinforcement learning with an explicit reward signal rather than direct optimization.

2.2 Implicit Feedback and Coactive Learning

Tucker et al. (12) study coactive learning for large language models, where users iteratively edit model outputs and these edits serve as preference signals rather than explicit ratings. They show that such implicit feedback can efficiently improve model quality and that modeling the structure of edits is crucial. Our approach adapts this insight: we use LLM-generated rewrites as synthetic coactive feedback, treating the transformation from list-heavy to prose-based output as an implicit preference signal. While we sacrifice the richness of human edits for scalability, the principle of learning from improved versions remains.

[1] Code and methodology available at: `https://github.com/jakebodea/cs329h-final.git`. Note: Due to proprietary data restrictions, the training dataset and model weights cannot be released. The repository contains the training scripts and reward function implementation, but full reproduction requires access to TaxRise's Databricks environment where data and compute are hosted.

2.3 TTS Evaluation and Naturalness

Prior work on TTS evaluation has established that discourse structure significantly impacts listener perception (11). Howcroft et al. (3) emphasized the importance of metrics capturing discourse coherence for generated text evaluation. Our evaluation metrics, including list usage rate, discourse marker frequency, and semantic preservation, draw on this literature to assess TTS-appropriateness.

3 Problem Setting

3.1 The List Generation Problem

Modern LLMs are trained primarily on written corpora where structured formatting (bullet points, numbered lists, headers) is common and often preferred for conveying information clearly. This training bias manifests as a strong tendency to generate lists even when instructed otherwise.

We illustrate this with an example from our customer service domain. Given a prompt asking about company services, a typical LLM response might be:

```
We help clients by:
1. Reviewing your tax situation
2. Preparing and filing missing returns
3. Negotiating with the IRS or state tax agencies
4. Setting up payment plans or settlements
```

When converted to speech, this becomes awkward: "We help clients by: one, reviewing your tax situation, two, preparing and filing missing returns..." The TTS system may read the numbers literally, insert unnatural pauses, or fail to convey the logical structure that the visual formatting provides.

A TTS-friendly alternative conveys the same information using natural discourse:

```
Our process typically starts with reviewing your tax
situation, then we prepare and file any missing returns,
and negotiate with the IRS or state tax agencies to set
up payment plans or settlements.
```

This version uses transition words ("starts with," "then," "and") to convey sequence, producing natural speech flow.

3.2 Why Prompting Fails

We conducted preliminary experiments adding explicit instructions to system prompts:

```
IMPORTANT: Never use bullet points, numbered lists, or
dashes. Always write in natural flowing sentences using
transition words like "first", "then", "next", "finally".
```

Even with such system instructions and few-shot examples, models continued generating lists in approximately 30% of the responses we sampled. This aligns with findings that instruction-following for stylistic constraints is unreliable (5), motivating our alignment-based approach.

4 Methods

4.1 Data Collection

We use customer service call transcripts from TaxRise, where users ask questions to a voice agentic system requiring multi-part explanations. From approximately 500 transcripts, we identify conversations where the LLM agent generated responses containing bulleted or numbered lists.

Data Availability: The source data contains proprietary TaxRise customer information and cannot be publicly released. All examples in this paper are anonymized or synthetically generated to preserve the structure without revealing private details. The preference pairs used for training contain only

the conversational context and responses, with personally identifiable information removed. Due to these data restrictions, the training dataset and fine-tuned model weights are not available for public distribution.

Each transcript contains a multi-turn conversation with roles (user, agent) and the agent's system prompt. We extract prompts where the final agent response contains at least two consecutive list items, as detected by regex patterns matching common list formats (bullets, dashes, numbered items).

4.2 Synthetic Preference Pair Generation

For each list-heavy response, we generate a TTS-friendly alternative using a separate LLM (Llama 3.2 3B-Instruct). Crucially, the rewriting model receives *only* the list-heavy response as context, not the full conversation, ensuring the rewrite focuses purely on stylistic transformation rather than content generation.

The rewrite prompt instructs the model to:

- Remove all bullet points, numbered lists, and dashes
- Use natural transition words ("first," "then," "next," "finally," "also")
- Preserve the same information and meaning
- Make it sound like natural spoken conversation

This produces preference pairs (x, y_w, y_l) where x is the conversational context, y_w is the preferred prose-based rewrite, and y_l is the dispreferred list-heavy original. We validate each pair by verifying that y_w contains no detected lists and that y_l contains at least two list items.

From 500 source transcripts, this process yielded 450 valid preference pairs for training and 50 for evaluation.

4.3 Reward Function Design

Rather than training a neural reward model on the preference pairs, we implement a rule-based reward function that directly captures our optimization objective. This design choice reflects the observation that list detection is straightforward via regex, avoiding the overhead of training and potential overfitting of a learned reward model.

The reward function $r(y)$ combines multiple components:

List Penalty: The primary signal penalizes list structures:

$$r_{\text{list}}(y) = \begin{cases} -0.4 \cdot \min(n_{\text{items}}, 5) & \text{if } n_{\text{items}} \geq 2 \\ +0.5 & \text{otherwise} \end{cases} \tag{1}$$

where n_{items} is the count of detected list items (bullets, numbered items, lettered items).

Semantic Similarity: To prevent degenerate solutions (e.g., generating empty or off-topic responses), we include a semantic similarity term when a reference response is available:

$$r_{\text{sim}}(y, y_{\text{ref}}) = 0.8 \cdot \text{cos\_sim}(\text{enc}(y), \text{enc}(y_{\text{ref}})) \tag{2}$$

using sentence embeddings from a lightweight model (all-MiniLM-L6-v2).

Discourse Markers: We provide a small bonus for using natural transition words:

$$r_{\text{discourse}}(y) = 0.1 \cdot \min(n_{\text{markers}}, 3) \tag{3}$$

where n_{markers} counts occurrences of words like "first," "then," "next," "finally," "also," "additionally."

Length Ratio: To discourage overly short or verbose responses:

$$r_{\text{length}}(y, y_{\text{ref}}) = \begin{cases} -0.4 & \text{if } |y|/|y_{\text{ref}}| < 0.5 \\ -0.2 & \text{if } |y|/|y_{\text{ref}}| > 2.0 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

The total reward is: $r(y) = r_{\text{list}}(y) + r_{\text{sim}}(y, y_{\text{ref}}) + r_{\text{discourse}}(y) + r_{\text{length}}(y, y_{\text{ref}})$.

4.4 Reinforcement Learning Training

We fine-tune Llama 3.2 1B-Instruct using Group Relative Policy Optimization (GRPO) (10) via the `trl` library. GRPO is a recent alternative to PPO that eliminates the need for a separate value network by computing advantages relative to other samples in a group, reducing memory requirements and simplifying training.

The objective maximizes expected reward while regularizing against the reference policy:

$$\max_{\theta} \mathbb{E}_{x,y\sim\pi_\theta} \left[r(y) - \beta \cdot \mathrm{KL}(\pi_\theta(\cdot|x) \| \pi_{\text{ref}}(\cdot|x)) \right] \tag{5}$$

where $\beta = 0.2$ controls the strength of the KL penalty.

Training Configuration:

- **Model:** Llama 3.2 1B-Instruct with LoRA adapters (rank 16, $\alpha = 32$)
- **Optimizer:** AdamW with learning rate 10^{-5}
- **Batch size:** 4 per device with gradient accumulation (effective batch 8)
- **Epochs:** 3 over the 450 training examples
- **Generation:** 4 completions per prompt, max 256 tokens, temperature 0.7
- **Hardware:** Single GPU with 16GB VRAM (Databricks cluster)

Parameter-efficient fine-tuning via LoRA (4) targets attention projection layers (q_proj, k_proj, v_proj, o_proj) and MLP layers, updating approximately 0.5% of total parameters while keeping the base model frozen.

5 Experiments

5.1 Evaluation Metrics

We evaluate on a held-out set of 50 prompts, comparing the base Llama 3.2 1B-Instruct model against our GRPO-tuned model.

List Usage Rate (LUR): Percentage of generated responses containing at least two consecutive list items, detected via regex patterns for bullets, dashes, and numbered/lettered items.

Discourse Marker Frequency (DMF): Average count of transition words per response from a curated lexicon of 20+ markers including "first," "then," "next," "finally," "also," "additionally," "moreover."

Average Reward: Mean reward score under our reward function, providing an aggregate measure of TTS-friendliness.

Semantic Preservation (BERTScore): BERTScore F1 (13) between generated responses and reference responses, measuring whether the fine-tuned model preserves the semantic content of its outputs.

5.2 Results

Table 1: Comparison of base vs. GRPO-tuned Llama 3.2 1B on TTS-friendliness metrics. Evaluated on 49 held-out prompts.

Model	LUR ↓	DMF ↑	Avg Reward ↑	BERTScore F1
Base Llama 3.2 1B	30.61%	0.51	−0.165	–
GRPO-tuned	10.20%	0.63	0.304	–
Δ (absolute)	−20.41	+0.12	+0.469	–
Δ (relative)	−66.7%	+23.5%	–	–

GRPO fine-tuning achieved a 66.7% relative reduction in list usage rate, dropping from 30.61% to 10.20%. The model also increased discourse marker frequency by 23.5%, indicating more frequent

use of natural transition words like "first," "then," and "finally." The average reward improved substantially from -0.165 to 0.304, reflecting the combined effect of reduced list penalties and increased discourse marker bonuses. BERTScore was not computed due to compute constraints and time limitations; semantic preservation is instead demonstrated qualitatively in Section 2.

5.3 Training Dynamics

Training completed in 440 minutes (7.3 hours) over 675 optimization steps (3 epochs on 450 samples). The average reward improved steadily from 0.12 at step 10 to 0.64 at convergence, representing a $5\times$ improvement. KL divergence from the reference policy stabilized around 0.02 throughout training, indicating the $\beta = 0.2$ regularization effectively prevented the policy from diverging too far from the base model.

One notable pattern was increasing completion length: average generations grew from 28 tokens early in training to 174 tokens by the end, with 30% of completions hitting the 256-token maximum. This verbosity increase occasionally caused response truncation during evaluation, suggesting future work could incorporate a length penalty in the reward function to encourage more concise outputs.

5.4 Qualitative Examples

Table 2 shows representative examples comparing base model and GRPO-tuned model outputs.

Table 2: Example outputs from base vs. GRPO-tuned model on the same prompt. Responses truncated for space.

Base Model (List-heavy)	**GRPO-tuned (TTS-friendly)**
Here's a high-level overview of how we work: 1. **Get in touch with us**: You can reach us through phone, email, or our online chat... 2. **Get your tax information**: Once you've reached out to us, we'll ask you for some basic information... 3. **Get a personalized consultation**: Our team will review your tax situation... 4. **Get updates and guidance**: Throughout the tax season, we'll keep you informed...	I'd be happy to explain how TaxRise works. Our goal is to help you navigate the complex world of taxes... TaxRise is a team of experts who are here to guide you through the tax preparation process. When you contact us, we'll ask you to provide some information about your tax situation. We'll then use this information to understand your situation and provide you with a customized plan to get you back on track. Our team will work with you to resolve any tax issues you may be facing...
Here's what I recommend: 1. **Contact your case manager**: Reach out to your case manager as soon as possible and explain the situation... 2. **Propose a payment plan**: Ask your case manager to help you create a payment plan... 3. **Review your tax obligations**: Your case manager will likely need to review your tax obligations...	I can see that the letter is clear on the payment amount, but the penalty is still on the table. Let me clarify with you. The IRS typically wants to see a payment to resolve the case... Regarding the case management, I'd be happy to connect you with your case manager. They'll be able to provide you with more information on the current status of your case and help you understand what's required to move forward. In the meantime, I can also help you explore options to resolve the pending penalty and interest...

6 Discussion

6.1 Effectiveness of Rule-Based Rewards

Our results demonstrate that a rule-based reward function can effectively guide RL fine-tuning for well-defined stylistic constraints. List detection is amenable to regex-based rules, avoiding the complexity and potential overfitting of neural reward models. This approach may generalize to other stylistic objectives with clear formal definitions (e.g., sentence length constraints, vocabulary restrictions, formatting rules).

However, rule-based rewards have limitations. They cannot capture nuanced preferences that resist formalization, and the hand-crafted components (penalty magnitudes, similarity weights) require tuning. For more subtle stylistic goals, learned reward models remain necessary.

6.2 Synthetic Preferences as Coactive Feedback

Interpreting LLM rewrites as synthetic coactive feedback enabled scalable preference data collection. Unlike human annotation, which is costly and slow, LLM rewrites can be generated in bulk while maintaining reasonable quality. The key insight from Tucker et al. (12), that edits implicitly encode preferences, transfers to the synthetic setting: the transformation from list-heavy to prose-based output encodes a preference for TTS-friendly style.

A limitation is that synthetic preferences may not perfectly align with human preferences. The rewriting model might introduce artifacts or fail to capture all aspects of natural spoken language. Future work could validate synthetic preferences against human judgments.

6.3 Limitations

Domain Specificity: Our experiments focus on customer service conversations in the tax resolution domain at TaxRise. The effectiveness of our approach on other domains (e.g., technical explanations, creative writing) remains to be validated.

Evaluation: We rely on automated metrics rather than human listening studies. While our metrics correlate with TTS-friendliness, direct human evaluation of spoken output quality would strengthen our conclusions.

Model Scale: We fine-tune a 1B parameter model due to computational constraints. Larger models may exhibit different list-generation tendencies and respond differently to RL fine-tuning.

Semantic Preservation: While BERTScore suggests semantic content is preserved, we do not evaluate factual accuracy or completeness of the generated responses.

Reproducibility: The training data is proprietary and the pipeline is tightly coupled with TaxRise's Databricks infrastructure where data, compute, and model weights are hosted. While we release our training scripts and reward function implementation, full reproduction of our results requires access to the private dataset or construction of an analogous corpus.

7 Conclusion

We addressed the problem of TTS-inappropriate list generation in language models through preference learning. By constructing synthetic preference pairs from LLM rewrites and fine-tuning with GRPO against a rule-based reward function, we achieved significant reductions in list usage while maintaining semantic content. Our approach demonstrates that modality-specific stylistic alignment can be achieved through preference learning, with implications for voice interfaces and audio applications.

Future work could extend this approach to other modalities (e.g., generation for screen readers, chat interfaces with character limits) and explore the combination of synthetic and human preferences for more nuanced stylistic objectives.

References

[1] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[2] Paul F Christiano, Jan Leike, Tom Brown, Miljan Marber, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems*, 30, 2017.

[3] David M Howcroft, Anya Belz, Miruna-Adriana Clinciu, Dimitra Gkatzia, Sadid A Hasan, Saad Mahamood, Simon Mille, Emiel van Miltenburg, Sashank Santhanam, and Verena Rieser.

Twenty years of confusion in human evaluation: Nlg needs evaluation sheets and standardised definitions. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 169–182, 2020.

[4] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[5] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[6] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.

[7] Dorsa Sadigh, Anca D Dragan, Shankar Sastry, and Sanjit A Seshia. Active preference-based learning of reward functions. In *Robotics: Science and Systems*, 2017.

[8] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[9] Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

[10] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Y Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[11] Catherine J Stevens, Nico Lees, Julie Vonwiller, and Denis Burnham. On-line experimental methods to evaluate text-to-speech (tts) synthesis: Effects of voice gender and signal quality on intelligibility, naturalness and preference. *Computer Speech & Language*, 19(2):129–146, 2005.

[12] Aaron Tucker, Markus Anderljung, and Allan Dafoe. Coactive learning for large language models using implicit user feedback. *arXiv preprint*, 2024.

[13] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

A Pre-Analysis Plan

The original pre-analysis plan submitted on November 5, 2025 is included below for reference. Section A.2 documents deviations from the original plan.

A.1 Original Abstract

Large language models generate responses with bulleted lists and structured formatting that create unnatural text-to-speech (TTS) experiences, and explicit prompting fails to prevent this. We frame TTS-optimized generation as a preference learning problem, constructing a dataset of preference pairs from customer service transcripts where list-heavy LLM outputs are contrasted with LLM-rewritten prose using natural discourse markers. We first train a reward model that assigns higher scores to conversational prose and explicitly penalizes bulleted lists, and then use the `trl` library to fine-tune gpt-oss-20b with reinforcement learning against this reward. We compare the RL-optimized model to the provider's vanilla instruction-tuned baseline. Evaluation through automated metrics and human listening tests will demonstrate that preference learning reduces list-based formatting while maintaining content quality. Interpreting the LLM rewrites as synthetic preference signals inspired by coactive learning, our work shows that stylistic alignment for modality-appropriate generation can be achieved through preference learning, with implications for voice interfaces and audio applications.

A.2 Deviations from Pre-Analysis Plan

Model Selection: The original plan proposed using "gpt-oss-20b." Due to GPU memory constraints (16GB VRAM), we switched to Llama 3.2 1B-Instruct. This smaller model fits comfortably in memory without quantization while still demonstrating the effectiveness of our approach.

Training Algorithm: The original plan specified PPO via the `trl` library. We used GRPO (Group Relative Policy Optimization) instead, as PPOTrainer was deprecated in `trl` version 0.25+ and GRPO is the recommended replacement. GRPO simplifies training by eliminating the need for a separate value network while achieving comparable results.

Reward Model: The original plan proposed training a neural reward model on Bradley-Terry preference pairs. We implemented a rule-based reward function instead. Since list detection is straightforward via regex, a neural reward model was unnecessary and would have added training complexity and memory overhead.

Dataset Size and Evaluation: The original plan mentioned 100 initial preference pairs with active learning expansion, plus a human listening study with 25 evaluators. We opted for a fully synthetic approach instead: collecting 500 source transcripts and generating 450 training pairs via LLM rewrites, with automated metrics (LUR, DMF, BERTScore) for evaluation. This decision prioritized scalability; synthetic preference generation and automated evaluation can scale to arbitrary dataset sizes without per-sample human cost, better reflecting how such a system would be deployed and iterated on in practice.

Baseline Comparison: The original plan compared against "the provider's vanilla instruction-tuned baseline" (GPT-4.1). We observed the list-formatting limitations with GPT-4.1 and reproduced the same pattern with the base Llama 3.2 1B; for consistency we report only the Llama baseline and omit GPT-4.1 outputs while focusing on the effect of RL fine-tuning.

B Data Format Example

Table 3 shows an anonymized example of the preference pair format used for training.

C Reward Function Implementation

The complete reward function implementation is available in our code repository. The core list detection function uses the following patterns:

Table 3: Example preference pair format (anonymized).

Field	Content
`prompt`	`[{"role": "system", "content": "You are a helpful customer service agent..."}, {"role": "user", "content": "Can you tell me about your organization?"}]`
`rejected`	I'm happy to give you a quick overview! [Company] helps people resolve tax issues. We help clients by: - Reviewing your tax situation - Preparing and filing missing returns - Negotiating with the IRS - Setting up payment plans
`chosen`	I'd be happy to give you a quick overview. TaxRise is a tax resolution company that helps people resolve IRS and state tax issues. Our process typically starts with reviewing your tax situation, then we prepare and file any missing returns, and negotiate with the IRS or state tax agencies to set up payment plans or settlements.

```
list_patterns = [
    r"^\s*[-*+]\s+.+$",       # Bullet points
    r"^\s*\d+[.)]\s+.+$",      # Numbered lists
    r"^\s*[a-zA-Z][.)]\s+.+$" # Lettered lists
]
```

A response is classified as containing a list if at least 2 consecutive lines match any pattern.

D Plagiarism, Bias, and Inaccuracies Statement

This work safeguards against plagiarism, bias, and inaccuracies. For plagiarism, all external ideas and text fragments are paraphrased and cited in the bibliography; when using exact phrasing (e.g., algorithm names or equations), quotation-like formatting and citations are provided, and drafts were manually checked to avoid unattributed reuse of wording or ideas. Idea-level attribution is explicit for coactive learning, RL from human feedback, LoRA, and GRPO. To limit bias, proprietary transcripts were anonymized, and synthetic rewrites were constrained to preserve meaning rather than inject opinions; results were reported with domain limitations to avoid overstating generality. Accuracy was addressed by cross-checking metric definitions, data splits, and reported hyperparameters against the training scripts; unsupported metrics (e.g., human listening tests) were removed rather than speculated. Any claims about proprietary data availability clarify that datasets and weights remain private. We monitored for domain skew and acknowledged the customer service context to avoid unfair generalization.

E AI Tool Usage Reflection

We used AI tools in two roles. First, we relied on ChatGPT and Claude in deep research modes to survey preference-learning literature, scope RL design choices, and sanity-check reward and LoRA settings under a 16GB GPU budget. Second, we used Cursor as the bridge between local development and Databricks: wiring the uv/dbconnect bootstrap, resolving environment and path issues, running jobs from the editor, and accelerating debugging. These tools shortened scoping and implementation cycles, but we treated their outputs as hypotheses and verified them against code, metrics, and logs; this discipline clarified GRPO vs. PPO trade-offs, exposed regex reward edge cases, and improved our judgment about synthetic preference quality. Looking ahead, we will continue to use LLM research modes for early scoping and citation trails, supplementing them with manual validation, versioned notes, and occasional grammar/LaTeX polishing reviewed by hand.

F Broader Impact Statement

This project aims to make voice interfaces easier to follow by discouraging list-like speech, improving accessibility for visually impaired users and reducing cognitive load for anyone relying on audio. By nudging models toward conversational flow, we hope to strengthen trust and comprehension in customer support and other speech-first settings. Voice AI in general can be misused by bad actors for persuasive or deceptive purposes; while our work focuses narrowly on naturalness rather than persuasion, deployment should pair stylistic tuning with governance, abuse monitoring, and human review in sensitive contexts. Training data consists of proprietary TaxRise transcripts that are access controlled within Databricks, and outputs reported only in aggregate. Attribution follows standard citations for external methods and models, and AI-assisted contributions were recorded. We adhered to Stanford's academic integrity and privacy standards by citing sources, avoiding unverifiable claims, and withholding data and weights that contain customer information.