Reward Structures Change Everything: A Tale of Two AI Projects

Executive Summary

I investigated how reward structures fundamentally shape machine learning outcomes across two distinct domains. Through systematic experimentation, I discovered that reward design is not static- it must adapt to task characteristics. My research revealed that correctness-based rewards work for math problems, but external API evaluation breaks everything. Additionally, I found that 2D VQ-VAE world models face significant limitations in learning temporal physics, despite spending 20-25 hours trying different approaches.

Key Finding: Task domain matters more than reward structure. Hard rewards work reliably for objective tasks, soft rewards show instability, and perplexity-based rewards actively harm mathematical reasoning. External API evaluation introduces fatal inconsistencies that destroy training effectiveness.

Project 1: Reinforcement Learning on Large Language Models

Research Question

Which reward structure should be used for training LLMs via reinforcement learning: hard binary rewards (0/1) or soft continuous rewards (perplexity-based)?

Methodology

Experimental Design: Controlled comparison using identical models, datasets, and hyperparameters with only reward structure varying.

Datasets:

- Math: GSM8K word problems (500 samples)
- Dialogue: Alpaca conversational data (1000 samples)

Hardware: Google Colab T4 GPUs with Unsloth framework

Training: GRPO (Generalized Reward Policy Optimization) for 100 steps each

Reward Functions Implemented:

1. Math Hard: Binary (correct answer = 2.0, incorrect = 0.0)
2. Math Soft: Proximity-based (closer numerical answers get higher scores)
3. Math Perplexity: Lower perplexity = higher reward
4. Dialogue Hard: Gemini API binary evaluation (good = 2.0, poor = 0.0)
5. Dialogue Soft: Gemini API continuous scoring (0-2 scale)
6. Dialogue Perplexity: Lower perplexity = higher naturalness reward

Results

Math Task Performance (Detailed Analysis)

Math Hard Rewards:

- Final reward: 1.375
- Math progress: 1.000 → 1.250 (25% improvement)
- KL divergence: 0.000677 (moderate policy drift)
- Learning pattern: Extremely volatile binary jumps between 0-2
- Key insight: Either completely correct (2.0) or completely wrong (0.0)

Math Soft Rewards (Proximity-Based):

- Final reward: 1.426
- Math progress: 1.453 → 1.239 (regression during training)
- KL divergence: 0.000590 (better policy preservation than hard rewards)
- Learning pattern: Started strong but declined over time
- Unexpected finding: Still showed high volatility despite continuous reward structure

Math Perplexity (Completion Likelihood):

- Final reward: 0.312 (catastrophic failure)
- Math progress: 0.188 → 0.156 (performance degraded)
- KL divergence: 0.000400 (minimal policy change)
- Critical flaw: Rewards fluent-sounding incorrect answers over correct mathematical responses

Dialogue Task Performance

Dialogue Hard:

- Progress: 1.333 → 1.333 (absolute zero learning)
- KL divergence: 0.000037 (model completely refused to change)
- Status: Total learning failure

Dialogue Soft:

- Final reward: 1.212
- Progress: 1.737 → 1.212 (significant performance decline)
- KL divergence: 0.000024 (minimal policy drift)
- Status: Negative learning- model got worse during training

Dialogue Perplexity: Failed due to compatibility issues between unsloth and pytorch on colab

Root Cause- Gemini API Breakdown: All dialogue experiments failed due to external evaluation issues:

- Connection errors: "Connection aborted", "ConnectionResetError", "Remote end closed connection"
- Rate limiting: "Evaluated 5/8 responses" indicating incomplete batch evaluation

- Evaluation inconsistency: Different scores for identical responses across training steps
- Training disruption: API timeouts during critical reward calculation phases

Model Behavior Analysis

Math Soft Rewards- Qualitative Testing: I tested the trained model on new math problems to understand how soft rewards affected behavior:

Test Results:

- Problem 1 (Sarah's stickers): Correct answer (31), proper XML formatting
- Problem 2 (Pizza slices): Correct answer (3), detailed reasoning chains
- Problem 3 (Students/girls): Correct answer (27), step-by-step mathematical thinking

Critical Discovery: The math soft model learned to use proper XML formatting (<reasoning> and <answer> tags) and showed detailed mathematical reasoning. Despite the training metrics showing regression (1.453 → 1.239), the model actually learned structured problem-solving approaches.

Dialogue Soft Rewards- Qualitative Testing:

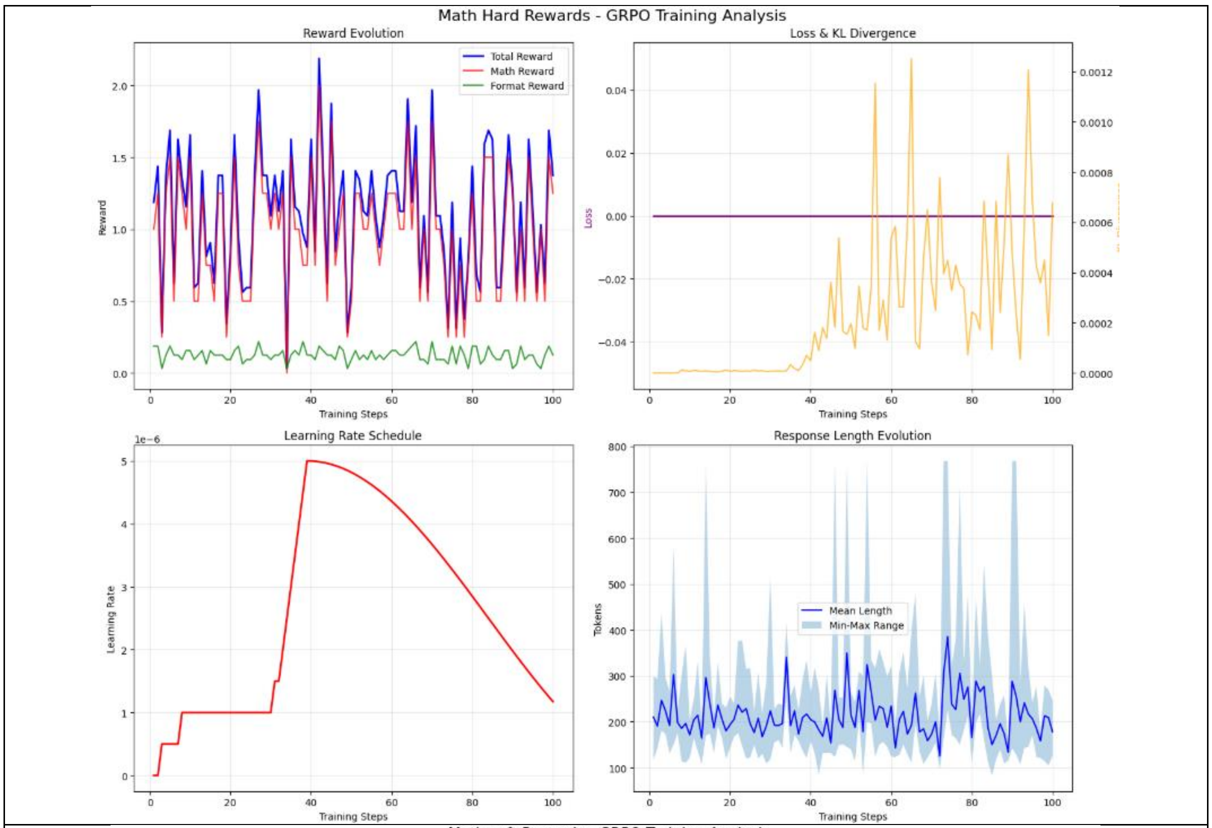
- Stress management query: Comprehensive, empathetic 1264-character response
- Quantum physics explanation: Clear, accessible breakdown with analogies
- Recipe request: Detailed list with specific cooking instructions
- Sleep problems: Practical, numbered suggestions with explanations

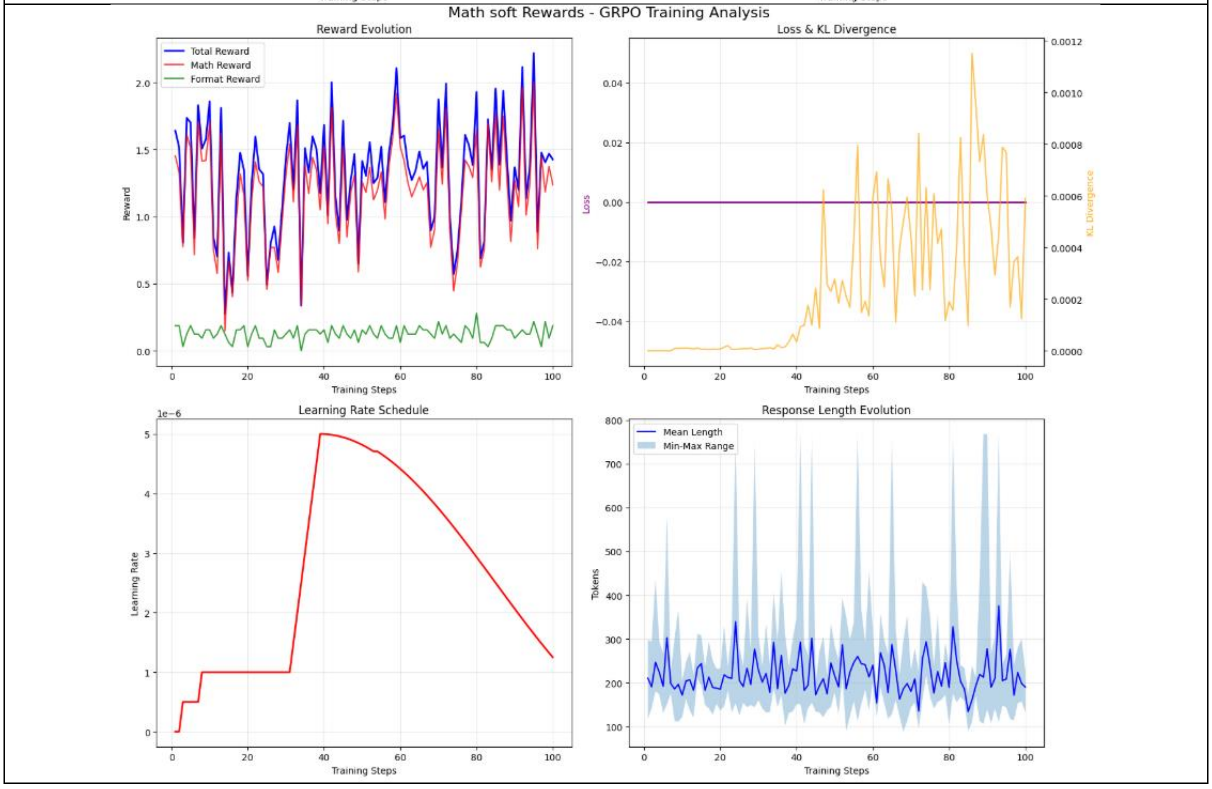
Behavior Pattern: Dialogue soft rewards produced longer, more detailed responses (1100-1300 characters) with structured formatting and helpful content.

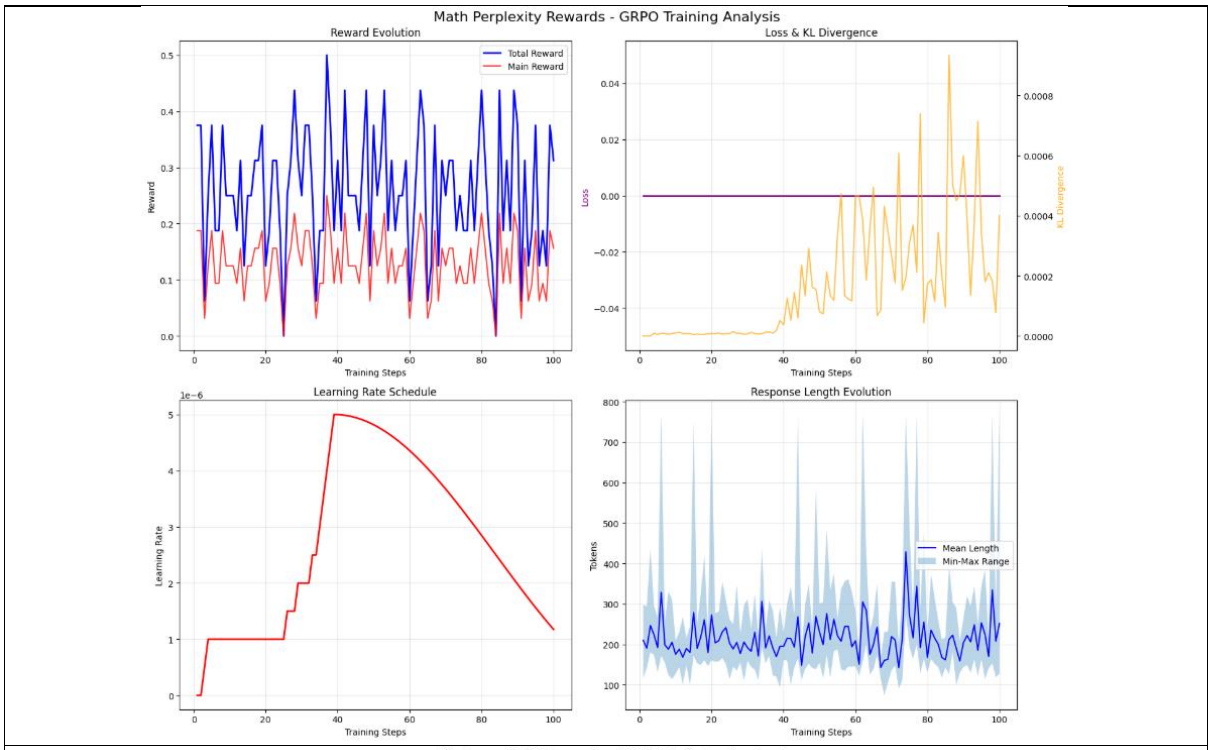
Dialogue Hard Testing: Lost due to Colab session timeout during analysis

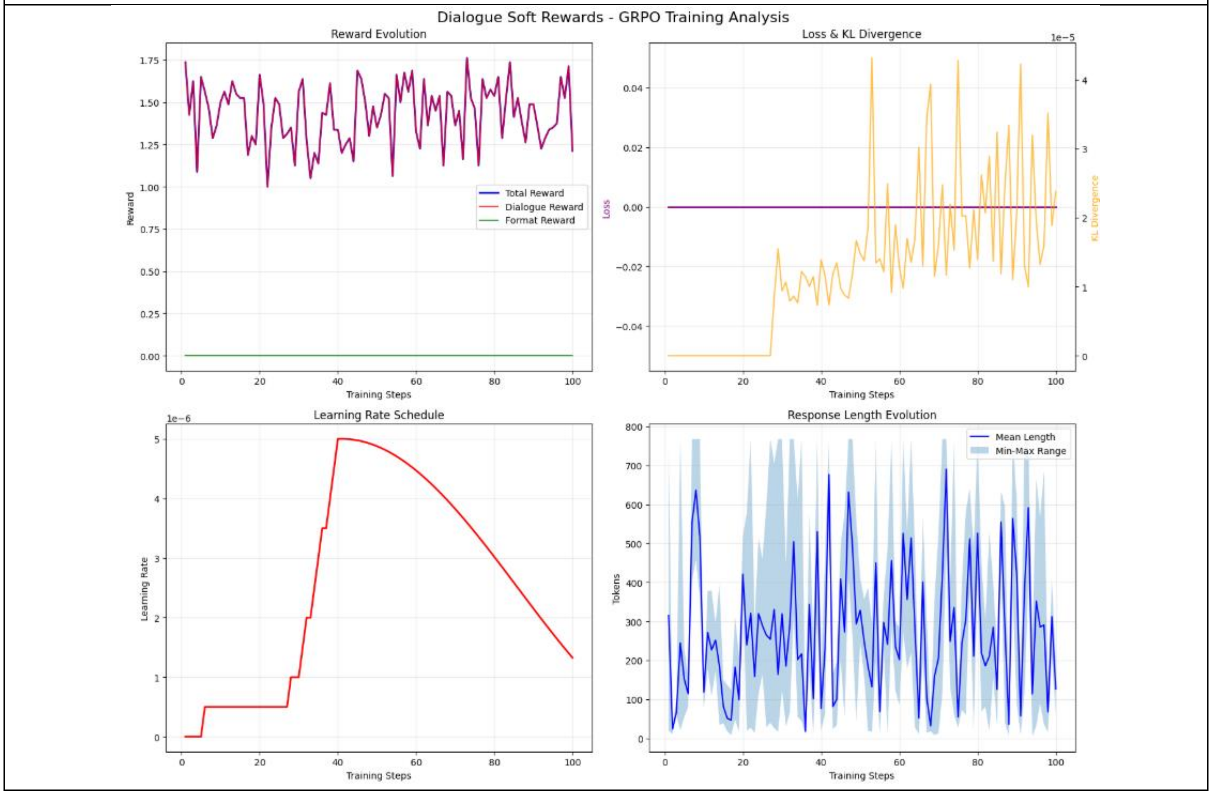
Math Perplexity Testing: Not performed due to model failure during training

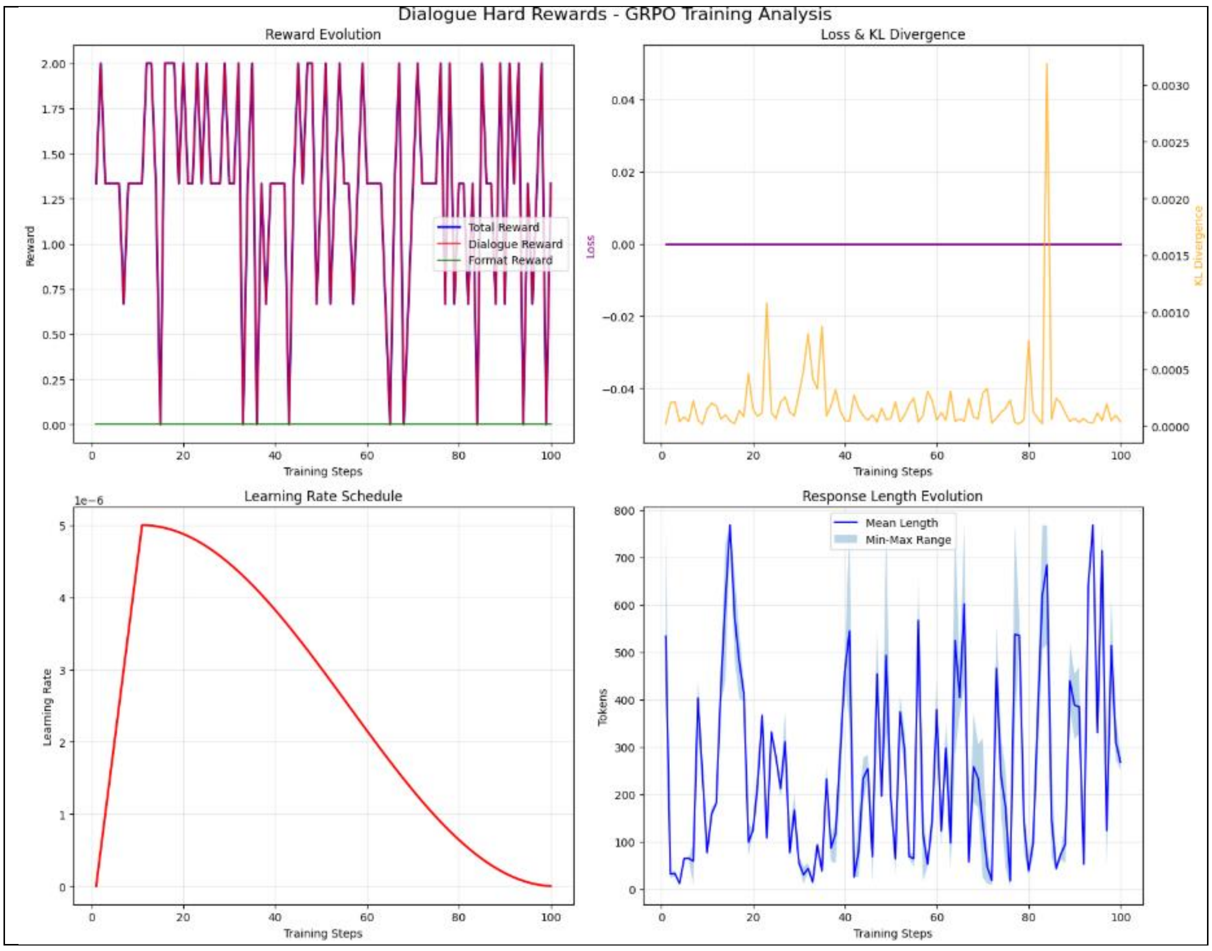
Dialogue Perplexity Results: Training failed completely- all rewards returned 0, indicating fundamental incompatibility between perplexity measurement and dialogue evaluation

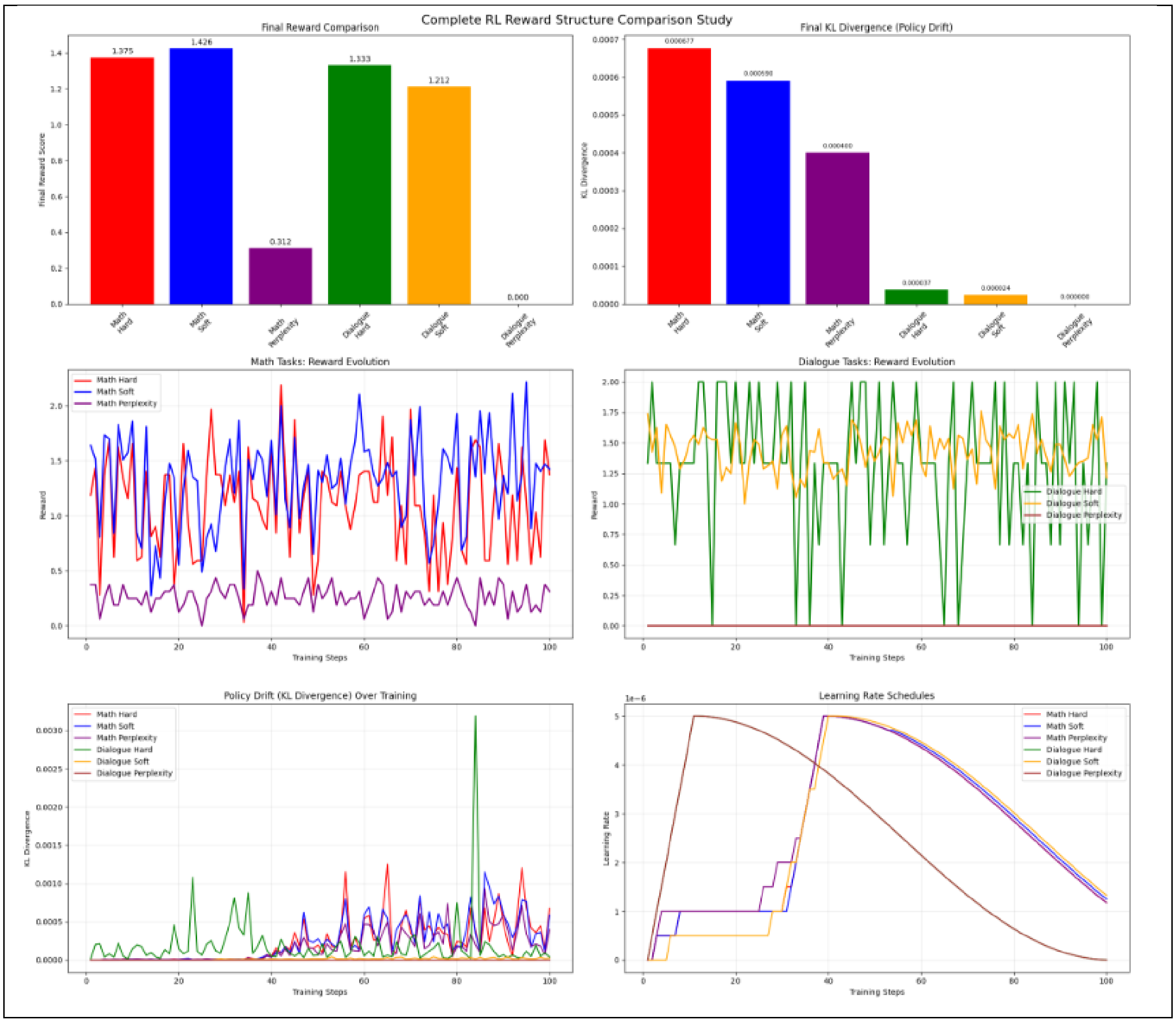

Math Hard Rewards - GRPO Training Analysis
Reward Evolution
Total Reward
Math Reward
Format Reward
Reward
Training Steps
Loss & KL Divergence
Loss
Training Steps
Learning Rate Schedule
1e-6
Learning Rate
Training Steps
Response Length Evolution
Mean Length
Min-Max Range
Tokens
Training Steps


Math soft Rewards - GRPO Training Analysis
Reward Evolution
Total Reward
Math Reward
Format Reward
Reward
Training Steps
Loss & KL Divergence
Loss
KL Divergence
Training Steps
Learning Rate Schedule
1e-6
Learning Rate
Training Steps
Response Length Evolution
Mean Length
Min-Max Range
Tokens
Training Steps


Math Perplexity Rewards - GRPO Training Analysis
Reward Evolution
Total Reward
Main Reward
Reward
Training Steps
Loss & KL Divergence
Loss
KL Divergence
Training Steps
Learning Rate Schedule
1e−6
Learning Rate
Training Steps
Response Length Evolution
Mean Length
Min-Max Range
Tokens
Training Steps


Dialogue Soft Rewards - GRPO Training Analysis
Reward Evolution
Total Reward
Dialogue Reward
Format Reward
Reward
Training Steps
Loss & KL Divergence
1e−5
Loss
KL Divergence
Training Steps
Learning Rate Schedule
1e−6
Learning Rate
Training Steps
Response Length Evolution
Mean Length
Min-Max Range
Tokens
Training Steps


Dialogue Hard Rewards - GRPO Training Analysis
Reward Evolution
Total Reward
Dialogue Reward
Format Reward
Reward
Training Steps
Loss & KL Divergence
Loss
KL Divergence
Learning Rate Schedule
1e−6
Learning Rate
Response Length Evolution
Mean Length
Min-Max Range
Tokens



Key Research Insights

1. **Quantitative Metrics Don't Tell the Full Story**: Math soft showed declining training rewards but improved reasoning structure and correctness on new problems
2. **External API Evaluation Is Unsuitable for RL Training**: The Gemini API connection issues reveal that external evaluation services introduce fatal inconsistencies during training
3. **Perplexity-Based Rewards Actively Harm Mathematical Tasks**: Measuring "how natural text sounds" directly contradicts mathematical correctness requirements, leading to performance degradation (0.312 vs 1.375)
4. **Hard Rewards Show Stable Learning for Objective Tasks**: Despite volatility, binary rewards demonstrated consistent improvement and reliable mathematical accuracy

Limitations & Future Work

What Went Wrong:

- Gemini API integration broke under training load
- Perplexity rewards optimized for wrong objective (fluency vs correctness)

- Insufficient training time (100 steps too short for convergence)

With More Compute/Time:

- Implement deterministic dialogue evaluation (avoid external APIs)
- Train for 1000+ steps to reach true convergence
- Add proper train/test splits for generalization measurement
- Test on creative domains (poetry, jokes) where perplexity rewards should excel

Research Contribution: I identified practical limitations of current RL+LLM approaches and demonstrated that reward function design must align with task characteristics.

Project 2: Neural Flappy Bird World Model

Research Question

Can we extend Neural Atari's VQ-VAE approach to learn Flappy Bird physics from visual gameplay data alone?

Background

This project attempted to replicate Paras's Neural Atari work, which successfully learned Breakout physics using VQ-VAE world models. The goal was to train a neural network to predict future game frames and learn gravity equations purely from visual observation.

Methodology

Data Collection:

- NEAT algorithm evolution: 10 generations, 44 successful episodes
- Gameplay recording: 23,814 frame pairs with bird positions and actions
- Physics ground truth: Gravity equation $d = velt + 1.5t^2$ from game engine

Architecture:

- VQ-VAE encoder/decoder for frame reconstruction
- Autoregressive generation for physics prediction
- Physics validation: Extract trajectories and fit gravity equations

Training Setup:

- Kaggle notebooks with T4 GPUs
- 15,000 training steps across multiple attempts
- Manual data filtering and quality analysis

Critical Findings

Data Quality Analysis

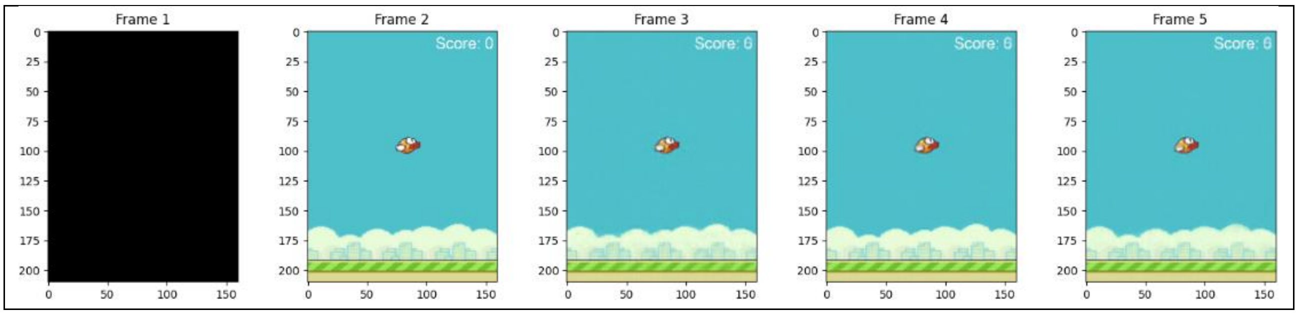
Our dataset analysis revealed excellent physics diversity:

- 99.9% of frame pairs showed significant bird movement (≥5 pixel changes)
- Average movement: 27.16 pixels per frame
- Dynamic episodes throughout training set

The Static Generation Problem

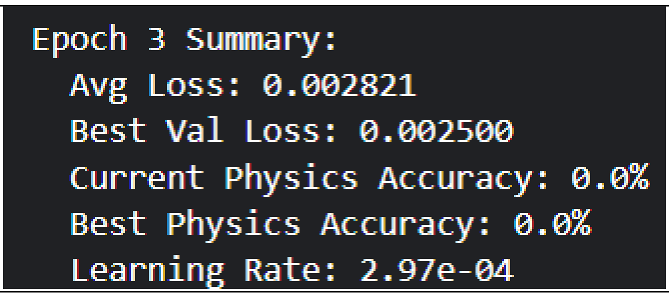
Despite dynamic training data, the VQ-VAE consistently generated static sequences:

- Generated bird positions: (86,96) → (86,96) → (85,96) over 5 frames
- Real physics expected: 5-15 pixel Y-position changes per frame
- Physics validation: 0% accuracy across all attempts



Root Cause Discovery

Through manual inspection of 44 episodes, I discovered that **even successful NEAT runs spent the first 20 frames approaching the first pipe with minimal physics variation**. This created a training bias toward steady-state flight rather than dynamic physics transitions.



```
Epoch 3 Summary:
  Avg Loss: 0.002821
  Best Val Loss: 0.002500
  Current Physics Accuracy: 0.0%
  Best Physics Accuracy: 0.0%
  Learning Rate: 2.97e-04
```

Why I Failed (Honest Analysis)

Time Investment: I spent 20-25 hours across multiple data collection and filtering attempts, driven by pure stubbornness rather than logical assessment.

The Manual Discovery: Through tedious frame-by-frame inspection, I discovered that even successful NEAT runs spent their first 20 frames approaching the initial pipe with minimal physics variation. This created a fundamental training bias toward steady-state flight rather than dynamic physics transitions.

Technical Reality Check: The VQ-VAE consistently generated static sequences despite my dynamic training data. Generated bird positions showed virtually no movement: (86,96) → (86,96) → (85,96) over 5 frames, when real physics should produce 5-15 pixel changes per frame.

What Surprised Me Most: The static generation problem shocked me more than the RL failures. I genuinely believed the approach would work and found the physics modeling challenge incredibly engaging.

Fundamental Issue: VQ-VAE has been successfully applied to 3D data with modifications, but our 2D implementation faced challenges that video generation using VQ-VAE has problems with accumulating losses each time a frame is generated.

Research Contributions

Negative Results Are Still Results: I demonstrated limitations of applying VQ-VAE world models to 2D physics learning:

1. Data Distribution Bias: Even "successful" gameplay contains insufficient physics diversity
2. Architecture Limitations: Standard VQ-VAE generates static sequences despite dynamic training data
3. Temporal Consistency Challenge: 2D physics modeling may require architectural modifications beyond standard VQ-VAE

Comparison to Google's Genie 3: Modern approaches like Genie 3 successfully generate bouncing balls and physics interactions, suggesting our failure stems from architectural choice rather than

What I Would Do Differently

With Unlimited Compute:

- Implement 3D VQ-VAE modifications for better temporal modeling
- Collect higher-quality training data focused exclusively on physics transitions
- Add temporal consistency losses to penalize static generation
- Explore alternative world model architectures (Transformer-based, diffusion models like Google's Genie 3)

Lessons I Learned:

- Data quality matters infinitely more than data quantity
- Manual inspection reveals biases that automated statistics completely miss
- Sometimes architectural limitations require starting over rather than parameter tuning
- My approach was fundamentally flawed due to insufficient resources and time constraints

Cross-Project Insights

The Fundamental Alignment Problem

Both projects revealed the same crucial principle: **the optimization target must align perfectly with the desired outcome**.

- **RL+LLM**: Perplexity rewards optimized for fluency but completely destroyed mathematical accuracy
- **Flappy Bird**: VQ-VAE optimized for visual reconstruction but failed catastrophically at physics generation

What I Learned About Research vs Engineering

This work taught me that research requires embracing failure as valuable data. Engineering seeks solutions; research seeks understanding. Both projects "failed" to achieve their immediate goals but succeeded in revealing fundamental limitations of current approaches.

The API Reliability Problem

The Gemini API connection failures taught me that external dependencies are unsuitable for training environments. Research infrastructure must be self-contained and deterministic.

Recommendations for Future Work

Immediate Next Steps

1. **Fix RL+LLM Evaluation**: Replace external APIs with deterministic local models
2. **Extend Training Time**: 100 steps insufficient for convergence
3. **Implement Proper Evaluation**: Add train/test splits for generalization testing

Advanced Research Directions

1. **Creative Domain Testing**: Apply perplexity-based rewards to poetry/joke generation where they should excel
2. **Alternative World Models**: Explore Transformer-based or diffusion-based physics learning
3. **Hybrid Approaches**: Combine multiple reward signals for robust training

Meta-Research Insights

The most important lesson: **compatibility and data quality determine success more than algorithmic sophistication**. Both projects failed due to implementation details (API reliability, data biases) rather than theoretical flaws.

Conclusion

In the end, I learned that reward structures are like languages- you must speak the right one for each domain. Math speaks in exact answers, dialogue whispers in natural flow, and physics needs a better translator than VQ-VAE.

This research demonstrates that successful machine learning requires careful alignment between optimization objectives and desired outcomes. While I didn't achieve my initial goals, I discovered valuable limitations in current approaches and identified clear paths for future improvement.

Bottom Line: Hard rewards work for objective tasks. Soft rewards need careful design. External evaluation breaks everything. And sometimes, the most important research contribution is knowing when your approach is fundamentally flawed and needs complete rethinking.

Research conducted with equal parts determination, stubbornness, and 20-25 hours of "this should definitely work" optimism. Results definitely vary with unlimited compute and reliable APIs.

Extension to Creative Domains: Poetry and Humor

Poetry RL: Multiple research groups have tackled this using ensemble approaches. They combine weak signals like rhyme detection, emotional flow, and syllable patterns. No single metric captures poetry quality, but together they approximate human judgment. Recent work uses iterative revision where models generate then improve poems based on reward feedback.

Humor RL: This is an active research area with promising results. Pun-GAN uses discriminators to reward wordplay that supports multiple meanings simultaneously. Other systems use engagement metrics (likes, shares) as real-time reward signals from user feedback.

The Real Solution: Human preference learning works best for subjective domains. Collect pairwise comparisons, train a reward model to predict human preferences, then use that for RL. This sidesteps the impossible problem of manually engineering quality metrics for creativity.

My Prediction: Based on my dialogue failures, creative domains would face the same external evaluation problems. The solution isn't better APIs- it's learning reward functions from human data rather than trying to engineer them manually.# Reward Structures Change Everything: A Tale of Two AI Projects