Analysis of Reinforcement Learning with Verifiable Rewards for Language Model Reasoning

Luka Meladze
National University of Singapore
Project: **LLM Post-training with Reinforcement Learning**

Abstract

This report provides an analysis of two papers that explore enhancing Large Language Models' (LLMs) reasoning capabilities through "pure reinforcement learning". The first paper, "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning," introduces a method to cultivate advanced reasoning in LLMs from a base model using RL (Guo et al., 2025). The second, "Understanding R1-Zero-Like Training: A Critical Perspective," critically explains the underlying assumptions and methodologies of the first paper, offering a more nuanced view and suggesting refinements (Liu et al., 2025). The paper will highlight details about "pure reinforcement learning," the differences between between Reinforcement Learning from Human Feedback (RLHF) and the technique used here, which we can term Reinforcement Learning with Verifiable Rewards (RLVR), the algorithms that drive it and the importance of algorithmic design and base model selection.

1 From Subjective Feedback to Verifiable Outcomes: RLHF vs. RLVR

1.1 Shift in Reinforcement Learning for LLMs

The common application of Reinforcement Learning (RL) to LLMs is through Reinforcement Learning from Human Feedback (RLHF). The primary objective of RLHF is to align a model with human preferences, which are often subjective and difficult to quantify. In this paradigm, a separate reward model is trained on a dataset of human-provided comparisons (e.g., selecting which of two responses is better). The LLM is then fine-tuned using this reward model as the source of its learning signal.

The papers by Guo et al. (2025) and Liu et al. (2025) focus on a different more objective **Reinforcement Learning with Verifiable Rewards (RLVR)**. This approach is used for domains where correctness can be programmatically verified.

- **RLHF**: Optimizes for subjective, human-aligned behavior using a learned reward model. The goal is often helpfulness, harmlessness, and adherence to a specific experts response style.
- **RLVR**: Optimizes for verifiable correctness using a rule-based or verifier-based reward function. The goal is to improve performance on specific reasoning tasks like mathematics, coding, and logic puzzles.

RLVR sidesteps the costly human data collection and potential reward-hacking pitfalls of a learned reward model, providing a direct and unambiguous signal for improving a model's reasoning faculties.

1.2 Language Model Reasoning as a Markov Decision Process

The process of generating a reasoned response can be framed as a Markov Decision Process (MDP), a foundational concept in RL.

- **State (s_t)**: The current state is the concatenation of the input question (q) and the sequence of tokens generated so far ($o_{<t}$).
- **Action (a_t)**: The action is the selection of the next token (o_t) from the model's vocabulary.
- **Policy (π_θ)**: The LLM itself is the policy. Parameterized by θ, it maps a state s_t to a probability distribution over all possible actions (tokens).
- **Reward (r_t)**: In the RLVR setting, the reward is sparse. It is zero for all intermediate tokens and a terminal reward is given only at the end of the sequence. For example, $R(q, o) = 1.0$ if the final answer in the response o is correct, and 0.0 otherwise.

The objective of reinforcement learning is to update the policy's parameters θ to maximize the expected cumulative reward, effectively teaching the model to generate token sequences that lead to correct final answers.

2 The DeepSeek-R1 Methodology: A Blueprint for Reasoning

Guo et al. (2025) introduces a novel pipeline for developing reasoning capabilities, centered around two key models: **DeepSeek-R1-Zero** and the more refined **DeepSeek-R1**.

2.1 The "Pure RL" Paradigm: DeepSeek-R1-Zero

The most significant contribution is the concept of training a reasoning model via "pure reinforcement learning," which means starting directly from a base model without an initial Supervised Fine-Tuning (SFT) phase on reasoning data.

Base Model: The process begins with DeepSeek-V3-Base, a foundational model not specifically tuned for complex reasoning.

Training Template: To guide the model's output structure, a specific template is enforced. This is critical for the verifier to parse the output.

```
User: {question}
Assistant: <think> reasoning process here </think> <answer> answer here </answer>
```

Reward Model: A simple, rule-based reward function provides the learning signal.

- **Accuracy Reward**: A primary reward is given if the content within the <answer> tag is correct. This is verified by checking against the ground-truth solution for math problems or running code against unit tests.
- **Format Reward**: A small auxiliary reward is given for correctly using the <think> and </think> tags, encouraging the model to adhere to the desired structure.

This pure RL process was shown to be remarkably effective, increasing the AIME 2024 pass@1 score from 15.6% to 71.0%.

2.2 The GRPO Algorithm: An Efficient Policy Gradient Method

The RL updates are driven by **Group Relative Policy Optimization (GRPO)**, a policy gradient algorithm designed for efficiency at scale.

Training Steps:

1. **Group Sampling:** For each question q in a batch, the policy $\pi_{\theta_{old}}$ generates a group of G different responses $\{o_1, o_2, \ldots, o_G\}$.
2. **Reward Calculation:** Each response o_i is evaluated by the verifier, yielding a set of rewards $\{r_1, r_2, \ldots, r_G\}$.

3. **Advantage Estimation:** The advantage A_i for each response is calculated relative to the other responses in its group. This is the core of GRPO's efficiency.

$$A_i = \frac{r_i - \text{mean}(\{r_1, \ldots, r_G\})}{\text{std}(\{r_1, \ldots, r_G\})} \tag{1}$$

4. **Policy Update:** The policy parameters θ are updated using a clipped surrogate objective, similar to PPO, to maximize the likelihood of responses with high advantage.

2.3 The Full Pipeline: The Creation of DeepSeek-R1

While DeepSeek-R1-Zero demonstrated powerful reasoning, its outputs were often unpolished and sometimes mixed different languages. To create a more robust and user-friendly model, **DeepSeek-R1**, a multi-stage pipeline was developed.

1. **Cold Start (SFT):** The `DeepSeek-V3-Base` model is first fine-tuned on a small (thousands) set of high-quality, human-readable reasoning examples. This provides a better starting point for RL.
2. **Reasoning-Oriented RL:** The GRPO algorithm is applied to this "cold-started" model to aggressively enhance its core reasoning abilities on math and code.
3. **Rejection Sampling & SFT:** The powerful RL-trained model is used to generate a large dataset of correct solutions ($\sim$600k). This high-quality synthetic data, combined with data for general tasks (writing, Q&A), is used for another round of SFT to broaden the model's capabilities.
4. **RL for All Scenarios:** A final RL stage is performed to align the model with human preferences for helpfulness and harmlessness, using a hybrid of RLVR for reasoning and traditional RLHF for general dialogue.

3 A Critical Perspective: Deconstructing the R1-Zero Paradigm

The second paper by Liu et al. (2025) provides a critical re-evaluation of the "R1-Zero-like" training paradigm, revealing that the narrative of "pure emergence" is more complex.

3.1 The Illusion of Emergence: Pre-existing Capabilities

The analysis reveals that the base models are not the blank slates they might appear to be.

- **Latent Abilities:** Base models like `DeepSeek-V3-Base` and especially the `Qwen2.5` family already possess significant reasoning capabilities before any RL is applied.
- **"Aha Moment" is Not Emergent:** The phenomenon of the model appearing to self-correct mid-thought (the "aha moment") was found to exist in the base models themselves. RL amplifies this behavior rather than creating it from nothing.
- **Pre-training Contamination:** Some models, like `Qwen2.5`, perform best without any template, suggesting they were likely pre-trained on concatenated question-answer text, making them functionally similar to SFT models from the outset.

3.2 Algorithmic Flaws and Unintended Biases in GRPO

Most critically, the paper identifies two significant biases in the GRPO objective function that can lead to misleading interpretations of model behavior.

1. **Response-Level Length Bias:** The GRPO loss is normalized by the length of the response. For an incorrect response (negative advantage), a longer response receives a smaller penalty. This incentivizes the model to generate increasingly long, meandering chains of thought for its *incorrect* answers, which can be mistaken for deeper reasoning.

2. **Question-Level Difficulty Bias:** Normalizing the advantage by the standard deviation of rewards within a group gives disproportionately high weight to questions that are either very easy or very hard (where reward variance is low). This biases the learning process.

3.3 The Fix: GRPO Done Right (Dr. GRPO)

The authors propose a simple and effective solution: **Dr. GRPO**, which removes the biasing normalization terms from the objective. This seemingly small change has a profound impact. The following code snippet highlights the difference between a typical, biased PPO loss implementation (as found in many libraries) and the unbiased approach.

```
# A common, but biased, way to calculate loss in many RL libraries
# This normalizes loss by the length of each individual response, introducing bias.
def biased_loss_calculation(ppo_loss, response_mask):
    # masked_mean normalizes by the number of non-pad tokens in each response
    per_response_loss = masked_mean(ppo_loss, response_mask, dim=-1)
    return per_response_loss.mean()

# The unbiased approach proposed by Dr. GRPO
# This normalizes by a fixed constant, removing the length bias.
MAX_TOKENS = 4096 # A fixed budget
def unbiased_loss_calculation(ppo_loss, response_mask):
    # Here, the loss for each token is summed and normalized by a constant
    total_loss = (ppo_loss * response_mask).sum()
    return total_loss / (BATCH_SIZE * MAX_TOKENS)
```

Dr. GRPO achieves comparable or better performance while being significantly more token-efficient, as it no longer rewards the model for generating lengthy incorrect responses.

4 Conclusion and Key Takeaways

The insights from DeepSeek-R1's initial breakthrough paper to the critical analysis in the second paper provides important considerations for the successful application of RL to LLMs.

1. **RLVR is a Powerful Tool for Objective Tasks:** For domains with verifiable correctness, RLVR provides a direct, scalable, and powerful method for improving model capabilities without the complexities and extra costs of RLHF.

2. **Base Models are Not Blank Slates:** The reasoning success of RL is heavily dependent on the pre-existing, latent capabilities of the base model. The notion of "pure emergence" should be treated with skepticism; RL is more accurately described as a process of amplifying and refining these latent skills.

3. **Algorithmic Design is Critical:** Subtle choices in the RL algorithm's objective function can introduce significant, unintended biases. An unbiased optimizer like Dr. GRPO is more robust, efficient, and leads to more interpretable model behavior.

4. **Iterative Refinement and Distillation are State-of-the-Art:** The most effective reasoning models are built through a multi-stage, iterative process that combines the strengths of SFT (for learning patterns) and RL (for exploration and generalization). The reasoning abilities of these large models can then be effectively distilled into smaller, more accessible models.

References

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Liu, Z., Chen, C., Li, W., Qi, P., Pang, T., Du, C., Lee, W.S., & Lin, M. (2025). Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.