Advanced Vision-Language Model Optimization: Group Relative Policy Optimization for Visual Question Answering

Mehul Bafna

March 24, 2025

Abstract

This report presents a comprehensive implementation of Group Relative Policy Optimization (GRPO) for enhancing the performance of vision-language models on Visual Question Answering (VQA) tasks. We develop a sophisticated reinforcement learning framework that optimizes the Microsoft Phi-3-vision-128k-instruct model without requiring human feedback, instead relying on a carefully designed multi-component reward function that evaluates the quality of generated answers relative to ground truth. Our approach addresses key challenges in multimodal learning by: (1) implementing a group-based normalization scheme that stabilizes the training signal, (2) developing a composite reward function that captures semantic and syntactic aspects of answer quality, (3) employing varied sampling strategies to effectively explore the model's output space, and (4) implementing robust tensor handling to accommodate the complex requirements of vision-language architectures. The proposed methodology demonstrates promising potential for improving the precision and relevance of responses in VQA systems without the scalability limitations of human feedback collection.

Contents

1 Introduction

Visual Question Answering (VQA) represents one of the most challenging frontiers in artificial intelligence, requiring models to simultaneously process visual and linguistic information to generate relevant, accurate responses. Recent advances in multimodal learning have led to vision-language models with remarkable capabilities, yet these models often struggle with generating concise, precise answers—particularly for factual or reasoning-based questions about images.

Traditional supervised fine-tuning approaches for VQA typically employ maximum likelihood estimation (MLE) via cross-entropy loss, which minimizes the divergence between predicted and ground truth answer distributions. However, this approach has limitations:

- It optimizes for token prediction rather than holistic answer quality
- It can be overly sensitive to surface-level linguistic patterns
- It may not align with evaluation metrics used to assess model performance
- It struggles with the one-to-many nature of valid answers in VQA contexts

Reinforcement Learning from Human Feedback (RLHF) has emerged as an alternative paradigm that directly optimizes for human preferences, but it faces significant scalability challenges due to the requirement for extensive human annotation. In this report, we explore Group Relative Policy Optimization (GRPO), a reinforcement learning approach that addresses these challenges by using automatically computed rewards based on answer quality metrics.

1.1 Contributions

This project makes several key contributions to the field of vision-language model optimization:

1. A complete implementation of GRPO for VQA tasks, demonstrating its applicability to multimodal learning contexts
2. A novel multi-component reward function that combines lexical, semantic, and structural evaluation of answer quality
3. A robust dataset processing pipeline for handling the complexities of image-text paired data
4. A theoretical framework for optimization dynamics, including the effects of sampling strategies, reward normalization, and tensor handling for vision-language models

1.2 Problem Formulation

Formally, we define the VQA task as follows: Given an image I and a question Q, the model must generate an answer A that maximizes some quality metric $R(A, A^*)$ where A^* is the ground truth answer. In conventional supervised learning, we optimize the likelihood of the ground truth answer:

$$\mathcal{L}_{\text{MLE}} = -\log P(A^*|I, Q; \boldsymbol{\theta}) \tag{1}$$

where $\boldsymbol{\theta}$ represents the model parameters. In contrast, our GRPO approach directly optimizes for the expected reward:

$$\mathcal{L}_{\text{GRPO}} = -\mathbb{E}_{A\sim P(\cdot|I,Q;\boldsymbol{\theta})}\left[\frac{R(A, A^*) - \mu_R}{\sigma_R}\right] \cdot \mathcal{L}_{\text{MLE}} \tag{2}$$

where μ_R and σ_R are the mean and standard deviation of rewards within a group of samples. This formulation introduces a relative reward structure that stabilizes training by comparing samples within a group rather than optimizing against absolute reward values.

2 Background

2.1 Visual Question Answering

Visual Question Answering emerged as a distinct research problem in the mid-2010s, aiming to evaluate machine understanding of both visual and textual content through question-answering tasks. Unlike image captioning, VQA requires targeted information extraction based on specific questions, often necessitating spatial reasoning, attribute recognition, counting, and commonsense knowledge.

VQA datasets typically contain triplets of (I, Q, A^*) where I is an image, Q is a question about the image, and A^* is the ground truth answer. Questions span various categories:

- **Object recognition**: "What animal is in the image?"
- **Attribute identification**: "What color is the car?"
- **Counting**: "How many people are in the room?"
- **Spatial reasoning**: "What is to the left of the table?"
- **Action recognition**: "What is the person doing?"
- **Abstract reasoning**: "Why is the person smiling?"

Early approaches to VQA combined convolutional neural networks (CNNs) for image processing with recurrent neural networks (RNNs) for text processing. Modern approaches leverage transformer-based architectures that handle both modalities, either through separate encoders with cross-attention mechanisms or through unified encoders that process interlaced image and text tokens.

2.2 Group Relative Policy Optimization

GRPO belongs to a family of policy gradient methods in reinforcement learning, specifically designed to improve training stability through relative reward normalization. Unlike standard policy gradient methods, which optimize for absolute reward values, GRPO normalizes rewards within groups of samples, reducing sensitivity to reward scaling and outliers.

The core insight of GRPO is to use the statistics of a batch of samples to normalize the reward signal, leading to more stable gradient updates. This approach shares conceptual similarities with advantage estimation in actor-critic methods but operates at the batch level rather than using a learned value function.

Mathematically, standard policy gradient methods optimize the objective:

$$\mathcal{L}_{\text{PG}} = -\,\mathbb{E}_{a\sim\pi_{\boldsymbol{\theta}}(a|s)}[R(a)] \tag{3}$$

GRPO modifies this objective to use relative rewards:

$$\mathcal{L}_{\text{GRPO}} = -\,\mathbb{E}_{a\sim\pi_{\boldsymbol{\theta}}(a|s)}\left[\frac{R(a)-\mu_R}{\sigma_R}\right] \tag{4}$$

where μ_R and σ_R are the mean and standard deviation of rewards within a batch. This normalization ensures that the training signal has zero mean and unit variance, making optimization less sensitive to the absolute scale of rewards.

In the context of language model fine-tuning, GRPO can be combined with the standard language modeling loss to create a hybrid objective that balances likelihood maximization with reward optimization.

2.3 Multimodal Foundation Models

Recent advances in foundation models have led to the development of powerful multimodal architectures that seamlessly integrate visual and textual information. Models like CLIP, DALL-E, and Flamingo demonstrate remarkable zero-shot and few-shot capabilities across various multimodal tasks.

The Phi-3-vision model used in this study represents a new generation of vision-language models that can process high-resolution images and generate coherent, contextually relevant text responses. These models typically employ:

- Vision transformers (ViT) or similar architectures for image encoding
- Large language models for text processing and generation
- Cross-modal attention mechanisms or projection layers for modality fusion

Phi-3-vision specifically uses a vision encoder that divides input images into patches, processes them with a transformer architecture, and projects the resulting embeddings into the language model's representation space. The language model then generates text conditioned on both the visual embeddings and input text tokens.

Fine-tuning these models presents unique challenges due to their scale, the heterogeneity of visual and textual data, and the complexity of optimization objectives. Our implementation addresses these challenges through careful dataset design, tensor management, and a reinforcement learning approach that directly optimizes for task-specific metrics.

3 Methodology

3.1 Model Architecture

Our implementation uses the Microsoft Phi-3-vision-128k-instruct model, a state-of-the-art vision-language model that combines a vision encoder with an autoregressive language model. The architecture follows the general pattern of recent multimodal models:

1. **Vision Encoder**: Processes the input image into a set of visual embeddings. In the case of Phi-3-vision, the image is divided into 17 patches which are individually encoded.
2. **Language Model**: A transformer-based language model with 128k context length that processes text tokens and generates responses.
3. **Multimodal Integration**: A mechanism to project visual embeddings into the language model's embedding space, allowing the model to condition text generation on visual features.

The model is designed to work with interleaved text and image inputs, using special tokens to denote image positions within the prompt. We structure prompts as follows:

```
<|user|>
<|image_1|>
{question} Answer with a single word or short phrase.
<|end|>
<|assistant|>
```

This prompt structure guides the model to generate concise answers appropriate for VQA tasks, while the special tokens ensure proper handling of the multimodal input.

3.2 Dataset Processing

Our implementation features a sophisticated dataset processing pipeline designed to handle the complexities of multimodal data. The dataset class handles:

- Loading and preprocessing images with robust error handling for missing files
- Customizable column mapping to adapt to different dataset schemas
- Integrated prompt formatting according to the model's expected input structure
- Processing through the model's tokenizer and image processor for consistent inputs

The dataset implementation includes several robust features for handling image-question-answer triplets, including support for various image file formats and paths, graceful error recovery for corrupted images, and appropriate preprocessing of both visual and textual inputs.

3.3 Reward Function Design

A critical component of our GRPO implementation is the reward function, which evaluates the quality of generated answers relative to ground truth. Our approach uses a composite reward function that combines multiple metrics to capture different aspects of answer quality:

Definition 3.1 *Composite VQA Reward Function: For a generated answer a and ground truth answer a^*, we define the reward $R(a, a^*)$ as:*

$$\begin{aligned} R(a, a^*) = w_1 \cdot \textit{BLEU-1}(a, a^*) + w_2 \cdot \textit{ExactMatch}(a, a^*) \\ + w_3 \cdot \textit{SubstringMatch}(a, a^*) + w_4 \cdot \textit{Jaccard}(a, a^*) \\ + w_5 \cdot \textit{Extractable}(a, a^*) \end{aligned} \tag{5}$$

where w_i are weight coefficients summing to 1, and each component evaluates a specific aspect of answer similarity.

The individual components are defined as follows:

- **BLEU-1**: Unigram precision, measuring token-level overlap

$$\text{BLEU-1}(a, a^*) = \frac{\sum_{w \in a} \text{Count}_{\text{clip}}(w)}{\sum_{w \in a} \text{Count}(w)} \tag{6}$$

- **ExactMatch**: Binary indicator of exact string matching

$$\text{ExactMatch}(a, a^*) = 1\left[a = a^*\right] \tag{7}$$

- **SubstringMatch**: Binary indicator of substring inclusion

$$\text{SubstringMatch}(a, a^*) = 1\left[a^* \text{ is a substring of } a\right] \tag{8}$$

- **Jaccard**: Set similarity of tokens

$$\text{Jaccard}(a, a^*) = \frac{|T(a) \cap T(a^*)|}{|T(a) \cup T(a^*)|} \tag{9}$$

  where $T(\cdot)$ extracts the set of tokens.

- **Extractable**: Binary indicator of regex-based extraction

$$\text{Extractable}(a, a^*) = 1\,[\text{regex pattern match of } a^* \text{ in } a] \tag{10}$$

The implementation includes preprocessing steps such as lowercasing, tokenization, and whitespace normalization to ensure robust evaluation. The weights assigned to each component (0.2, 0.3, 0.2, 0.1, 0.2) reflect our assessment of their relative importance, with exact matching receiving the highest weight due to its strong correlation with human judgment of answer quality.

3.4 GRPO Algorithm Implementation

The core of our approach is the GRPO algorithm, which generates multiple samples, computes their rewards, normalizes the rewards within the group, and scales the language modeling loss accordingly. The algorithm proceeds as follows:

Algorithm 1 Group Relative Policy Optimization for VQA

Procedure ComputeGRPOLoss(model, processor, batch, num_samples)
  temperatures ← [0.7, 0.8, 0.9, 1.0][: *num_samples*]
  top_p_values ← [0.9, 0.85, 0.95, 0.8][: *num_samples*]
  samples ← [ ]
  for $i \leftarrow 0$ **to** num_samples − 1 **do**
    temperature ← temperatures[i]
    top_p ← top_p_values[i]
    model_inputs ← PrepareBatchForModel(batch)
    generate_ids ← model.generate(model_inputs, temperature, top_p)
    response ← DecodeAndClean(generate_ids)
    samples.append(response)
  end for
  rewards ← ComputeRewards(samples, batch["ground_truth"])
  $\mu_R \leftarrow$ mean(rewards)
  $\sigma_R \leftarrow$ std(rewards) + ϵ *ϵ prevents division by zero*
  normalized_rewards ← (rewards − μ_R)/σ_R
  model_inputs ← PrepareBatchForModel(batch)
  outputs ← model(model_inputs, labels=model_inputs["input_ids"])
  $\mathcal{L}_{\text{base}} \leftarrow$ outputs.loss
  $\mathcal{L}_{\text{GRPO}} \leftarrow -\text{mean}(\text{normalized\_rewards}) \cdot \mathcal{L}_{\text{base}}$
  return $\mathcal{L}_{\text{GRPO}}$
end Procedure

The implementation includes several sophisticated features:

- **Varied sampling strategy**: Different temperature and top-p values encourage diverse outputs
- **Reward normalization**: Zero-mean, unit-variance normalization within sample groups
- **Hybrid loss function**: Combines reinforcement learning with language modeling objectives
- **Numerical stability**: Handles edge cases like zero rewards and zero standard deviations

This approach avoids known pitfalls in reinforcement learning for language models, such as mode collapse and reward hacking, through careful normalization and integration with the base language modeling objective.

3.5 Tensor Shape Management

A significant technical challenge in working with vision-language models is handling the complex tensor structures required for multimodal inputs. Our implementation includes dedicated functions for managing tensor shapes to ensure compatibility with the model architecture.

The tensor management system handles various tensor shape configurations that might arise from different dataset processing pipelines or model processors. It ensures that tensors are properly formatted before being passed to the model, preventing shape mismatch errors that could otherwise disrupt training.

This component addresses a key challenge in multimodal learning: different models expect different tensor layouts, and ensuring compatibility requires careful handling of dimensions, particularly for batched inputs and specialized formats like Phi-3-vision's image patch representation.

3.6 Training Procedure

Our training procedure integrates GRPO with standard training practices such as gradient accumulation, learning rate scheduling, and periodic evaluation. The main training loop includes:

- Mixed precision training (bfloat16) for memory efficiency
- Gradient checkpointing to trade computation for reduced memory usage
- Gradient accumulation for effective larger batch sizes
- Gradient clipping to prevent exploding gradients
- Periodic evaluation and best model checkpointing
- Exception handling for robust recovery from errors

This comprehensive approach ensures stable and efficient training even with large models and complex multimodal inputs.

4 Experimental Setup

4.1 Model Configuration

Our experiments use the Microsoft Phi-3-vision-128k-instruct model, which offers several advantages for VQA tasks:

- Long context window (128k tokens) supporting complex image-text interactions
- Advanced vision encoder capable of processing high-resolution images
- Instruction-tuning that aligns with our task formulation
- Flash Attention 2 implementation for efficient training

We configure the model with the following optimization settings:

- Gradient checkpointing enabled for memory efficiency
- Mixed precision training with bfloat16 data type
- Batch size of 1 with gradient accumulation of 2 steps
- Learning rate of 2e-7 with AdamW optimizer
- Weight decay of 0.01 for regularization

4.2 Dataset Preparation

Our implementation is designed to work with standard VQA datasets containing image-question-answer triplets. The dataset is loaded into a pandas DataFrame with columns for:

- Image identifiers (linking to image files)
- Questions about the images
- Ground truth answers

We split the dataset into training and evaluation sets, with approximately 90% of data used for training and 10% for evaluation. For initial testing and hyperparameter tuning, we use smaller subsets of 20 training samples and 10 evaluation samples to enable rapid iteration.

4.3 Evaluation Metrics

We evaluate model performance using the same composite reward function used during training, which combines:

- BLEU-1 score for token overlap
- Exact match rate for perfect answers
- Substring match rate for partial correctness
- Jaccard similarity for semantic overlap
- Extractable match rate for answer extractability

During evaluation, we generate a single response with temperature 0.7 and top-p 0.9, balancing diversity and coherence. The evaluation procedure processes a fixed number of samples (configurable, typically 5-10) to provide a reliable estimate of model performance.

5 Advanced Implementation Considerations

5.1 Memory Optimization

Working with large vision-language models presents significant memory challenges, particularly during reinforcement learning where multiple forward and backward passes are required. Our implementation addresses these challenges through several techniques:

- **Gradient checkpointing**: Trades increased computation for reduced memory by recomputing intermediate activations during backpropagation rather than storing them.
- **Mixed precision training**: Uses bfloat16 data type to reduce memory footprint while maintaining numerical stability.
- **Explicit memory management**: Calls to `torch.cuda.empty_cache()` after processing each batch to prevent memory fragmentation.
- **Gradient accumulation**: Updates model weights after accumulating gradients over multiple batches, enabling effective larger batch sizes.
- **Selective tensor handling**: Carefully manages which tensors are moved to GPU and when they are released.

These optimizations enable training on consumer-grade GPUs with 16-24GB of memory, making the approach more accessible for research and development.

5.2 Potential Pathologies

Reinforcement learning for language models can exhibit several pathological behaviors that our implementation takes steps to mitigate:

- **Mode collapse**: The model converges to a narrow set of "safe" responses that receive moderate rewards. We address this through temperature variation during sample generation and by maintaining the base language modeling loss.
- **Reward hacking**: The model exploits patterns in the reward function that don't align with true answer quality. Our multi-component reward function makes this more difficult by requiring the model to optimize across different metrics simultaneously.
- **Partial optimization**: The model optimizes for a subset of reward components while ignoring others. The balanced weighting of components and group normalization help prevent this behavior.
- **Training instability**: Policy gradient methods can exhibit high variance in gradient estimates. The group normalization in GRPO provides a more stable training signal.

Our theoretical analysis suggests that these mitigations should be effective, though careful monitoring during training is necessary to detect and address potential pathologies as they emerge.

5.3 Scaling Considerations

The current implementation focuses on efficient training with limited resources, but several modifications could enable scaling to larger datasets and models:

- **Distributed training**: Implementing data parallelism across multiple GPUs using frameworks like DeepSpeed or PyTorch Distributed.
- **Optimized reward computation**: Moving reward calculation to GPU and batching computation across multiple samples.
- **Parameter-efficient fine-tuning**: Implementing techniques like LoRA or QLoRA to reduce the number of trainable parameters.
- **Selective sampling**: Prioritizing training examples that show larger gradients or lower rewards to focus computational resources.

These enhancements would enable application of the GRPO approach to larger vision-language models and more extensive VQA datasets.

6 Future Directions

6.1 Reward Function Enhancements

Several promising directions for enhancing the reward function include:

- **Semantic similarity**: Incorporating embeddings-based similarity metrics that capture semantic equivalence beyond lexical overlap.
- **Answer type awareness**: Adapting reward components based on the question type (e.g., different metrics for yes/no questions versus counting questions).

- **Visual grounding**: Adding components that measure the alignment between answers and visual features, potentially using attention maps or object detections.
- **Calibrated confidence**: Rewarding appropriate expression of uncertainty when questions are ambiguous or unanswerable from the image.

These enhancements could address some potential weaknesses in the current formulation and lead to more robust performance across different question types.

6.2 Architectural Extensions

The GRPO approach could be extended to alternative architectures and training paradigms:

- **Critic networks**: Incorporating learned value functions to provide more sophisticated reward estimation.
- **Multi-task optimization**: Extending the approach to jointly optimize for VQA and related tasks like image captioning or visual reasoning.
- **Few-shot adaptation**: Exploring GRPO as a method for rapidly adapting pre-trained models to new domains with limited data.
- **Human feedback integration**: Combining automated metrics with sparse human feedback for hybrid optimization.

These extensions could broaden the applicability of GRPO beyond the specific VQA task addressed in this implementation.

6.3 Evaluation Framework

A comprehensive evaluation framework could provide deeper insights into model performance:

- **Question type stratification**: Evaluating performance separately across different question categories.
- **Human evaluation**: Comparing model outputs with human judgments to validate the alignment between automated metrics and perceived quality.
- **Robustness testing**: Assessing performance on adversarially designed questions or perturbed images.
- **Benchmarking**: Systematic comparison against state-of-the-art VQA models using standardized datasets and metrics.

Such a framework would provide a more nuanced understanding of model capabilities and limitations.

7 Conclusion

This report has presented a comprehensive implementation of Group Relative Policy Optimization for Visual Question Answering, demonstrating the applicability of reinforcement learning approaches to multimodal tasks without requiring human feedback. Our approach addresses key challenges in vision-language model fine-tuning through:

1. A sophisticated reward function that captures multiple aspects of answer quality

2. A group-based normalization scheme that stabilizes training
3. Careful tensor management to accommodate the complexities of vision-language models
4. Memory-efficient training techniques that enable work with large models on limited hardware

The proposed methodology offers a promising middle ground between MLE-based supervised learning and RLHF, potentially enabling improved performance without the scalability limitations of human feedback collection.

Our implementation provides a foundation for further research in reinforcement learning for multimodal tasks, with potential applications beyond VQA to other domains requiring precise alignment between visual content and textual outputs. By making the code accessible and documenting implementation details, we aim to contribute to the broader research community's efforts to develop more capable and aligned vision-language models.

References

[1] Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Lawrence Zitnick, C., & Parikh, D. (2015). VQA: Visual question answering. In *Proceedings of the IEEE international conference on computer vision* (pp. 2425-2433).

[2] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning* (pp. 8748-8763).

[3] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., ... & Sutskever, I. (2021). Zero-shot text-to-image generation. In *International Conference on Machine Learning* (pp. 8821-8831).

[4] Alayrac, J. B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., ... & Zisserman, A. (2022). Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems* (pp. 18431-18441).

[5] Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., & Amodei, D. (2017). Deep reinforcement learning from human preferences. *Advances in neural information processing systems, 30*.

[6] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., ... & Schulman, J. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems, 35*, 27730-27744.