CSC490 Assignment 4 - LLM Judge

Team:
Son Nguyen (ID 1009656560), Kyle (ID 1007785229),
Daniel (ID 1008035378), Jinbo (ID 1004821419)

Project: ArXplorer - Semantic Search for arXiv

Part One: Background research on GRPO

Paper	Commentary
Training-Free Group Relative Policy Optimization (Cai et al.) - 2025 -ArXiv Link	Proposes a low-cost, training-free variant that exploits group-relative semantics, enabling large-scale model adaptation without heavy RL training. Extends RLVR by reducing training overhead, suitable for resource-constrained settings.
Rejection Sampling Fine-Tuning (Yuan et al.) - 2023 - ArXiv Link	Uses preference-based rejection sampling to improve model alignment, emphasizing verifiable, safe outcomes with minimal additional training. Alternative approach to RLVR focusing on data quality and safety guarantees.
Direct Preference Optimization (DPO) (Rafailov et al.) - 2023 - ArXiv Link	Performs direct optimization of human preferences, bypassing traditional RL frameworks and value functions while maintaining safety constraints. Offers a simplified, verifiable alternative for training aligned models, aligned with RLVR goals.
Hybrid Group Relative Policy Optimization (Sane) - 2025 - ArXiv Link	Combines features of PPO and GRPO to balance stability and efficiency, using multiple responses per prompt for advantage estimation. Advances RLVR by synthesizing methods for improved training stability of large models.
Meta Reinforcement Learning with Verifiable Rewards (Zhang et al.) - 2024 - ArXiv Link	Introduces a meta-learning framework for RLVR that adapts quickly to new verification criteria with minimal data, improving generalization and reward verifiability. Highly relevant for RLVR, focusing on rapid adaptation and verifiable reward signals in evolving environments.

Part Two: Metrics and Evaluations

Metric	Why	How to Measure	Challenges at Scale
Precision@k	Measures whether the top-k search results are relevant to the user query; important for user trust and satisfaction.	**Quantitative:** Calculate the fraction of relevant papers among top k retrieved results using a ground-truth dataset. **Qualitative:** Manually inspect whether retrieved papers align with query intent.	Requires ground-truth labels or human judgment for "relevance," which is subjective and costly to gather for thousands of papers.
Recall@k	Measures the ability to retrieve all relevant papers for a given query; important for comprehensive literature discovery.	**Quantitative:** Count how many of the total relevant papers are retrieved in top k using a ground-truth dataset. **Qualitative:** Manually inspect whether key papers show up in a related query.	The total set of "relevant" documents is often unknown; estimating recall for open-domain queries is difficult.
Mean Reciprocal Rank (MRR)	Evaluates ranking quality i.e. whether the most relevant paper appears near the top. Important for user search experience such that they see most relevant results first.	**Quantitative:** Compute the average of reciprocal ranks of the first relevant paper for each query using a ground-truth dataset. **Qualitative:** Examine user interaction logs for whether users click early results.	Needs high-quality annotations or user click data; biased if click data reflects popularity rather than relevance.
Embedding Quality	Measures how well the embeddings capture semantic similarity between papers (e.g., papers on the topic "transformers" cluster together).	**Quantitative:** Use clustering metrics such as pairwise cosine similarity among known related papers. **Qualitative:** Visualize embedding space using t-SNE or UMAP for interpretability.	Visualization and clustering become computationally expensive with millions of embeddings; hard to interpret dense embedding spaces.
Retrieval Latency	Directly affects user experience and scalability; a slow semantic search would limit usability.	**Quantitative:** Measure average query time across multiple runs and load conditions. **Qualitative:** User testing to see if latency feels acceptable	Balancing low latency with high accuracy. Larger models and indexes improve relevance but increase compute costs.

Part Three: Establishing a baseline with a LLM

Metric Chosen: Precision@k

To evaluate our semantic retrieval system for ArXiv papers, we focused on precision@k as the primary metric, since user satisfaction depends heavily on receiving relevant results among the top-k retrieved papers. High precision@k indicates that users receive useful papers, while low precision@k would suggest that most results are irrelevant.

Although our evaluation metric is precision@k, each model in our setup functions as a binary classifier that predicts whether a specific query-paper pair is relevant or irrelevant. Thus, these classifiers operate on individual pairs rather than ranked lists. The models can be used to estimate precision@k for our retrieval system by applying the classifier to the top-k candidate papers for a given query and computing precision using the classifiers predictions.

Baseline Classifiers

- Embedding-based model (all-MiniLM-L6-v2): Both queries and paper abstracts are embedded into dense vectors. Cosine similarity is computed, and a relevance label is assigned based on a threshold optimized for accuracy on the validation set.
- LLM-based model (llama3-8B via Ollama): The model is prompted with a query-abstract pair and asked to output "yes" or "no" for relevance.

Dataset

- Dataset of 1617 query-abstract pairs, labeled as relevant or irrelevant.
- 883 Training Examples, 386 Validation Examples, 347 Evaluation Examples (60/30/30 Split)
- Dataset contains 100 distinct ML-related queries.
- Each query is paired with several ML-related abstracts, some relevant and others irrelevant.
- Each query-abstract pair is labeled as relevant/irrelevant accordingly.

Results on Evaluation Set

Model	Accuracy	Precision	Recall	F1	False Pos	False Neg
Embedding Baseline	0.8069	0.9620	0.5429	0.6941	3	64
LLM Baseline	0.8011	0.7052	0.8714	0.7796	51	18

Note: Precision in this table refers to precision in binary classification not precision@k

Comparison

- Both models have comparable accuracy; they differ in their precision/recall trade-off.
- Embedding baseline achieves higher precision but lower recall; the model is more conservative.
- Embedding baseline is highly accurate when it identifies abstract as relevant but will frequently miss borderline cases resulting in a high amount of false negatives.
- LLM baseline achieves better recall but lower precision; the model overestimates relevance.
- LLM baseline is better at identifying non-relevant pairs avoiding false negatives but frequently overestimates relevance resulting in high amount of false positives.

Common Error Patterns

- **Specificity Mismatch**: Both models can produce false positives when the query expresses a specific subtopic or application but the abstract only discusses the broader category.
- **Keyword Bias**: Both models produce false positives when they have high keyword match in the query and abstracts but the underlying meaning differs.
- An example from the test set both models failed on illustrates the above error patterns:
  - Query: “Graph Neural Networks for drug discovery and molecular property prediction”
  - Abstract: discussed feature extraction from graphs (not relevant to drug application)
- **Overgeneralization (LLM)**: The llama3 model tends to predict ”yes” for loosely related abstracts due to its broader semantic interpretation, resulting in low precision.

Limitations

- **Dataset quality and size**; creating high-quality, labeled query-abstract pairs is time-consuming if done manually and difficult to scale synthetically. Synthetic labels risk biasing results since they would likely rely on similar models used in our baselines, potentially inflating performance metrics.
- **Baseline model simplicity**; both baseline models are relatively simple: all-MiniLM-L6-v2 is less expressive than larger or domain-specific models, and llama3-8B was used only with zero-shot prompting without fine-tuning.

Next Steps

- **Fine-tune embedding baseline**; perform contrastive fine-tuning on the embedding model using data derived from the ArXiv citation graph to encourage co-cited papers to be closer in the vector space.
- **Improve LLM baseline**; improve the LLM classifier via reinforcement learning such as the GRPO implemented in part 4.
- **Improve the dataset**; increase the diversity and number of labeled pairs to improve generalization.

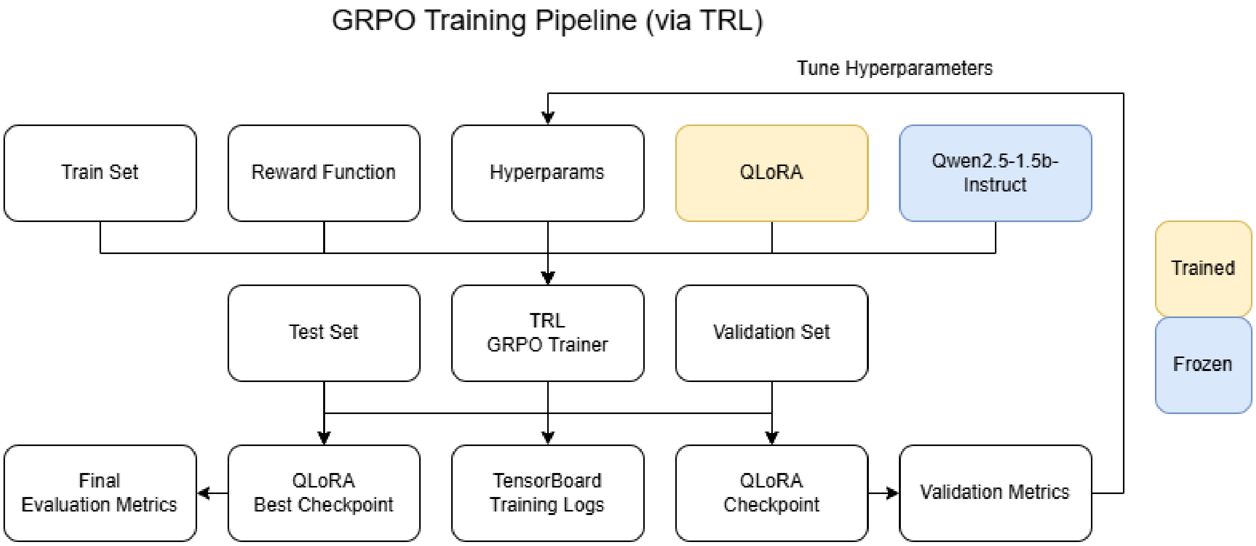
Part 4: Training your Judge

Metric and Hypothesis

We chose precision@k as our metric. Users want the retrieved papers to be relevant to their query, and precision@k measures how well the model delivers on retrieving relevant papers. However, our model remains a binary relevance classifier (predicting "yes" or "no" for each query–abstract pair). This classifier can then be used to estimate precision@k as explained in part 3.

Given that our baseline models already achieved relatively strong accuracy (≈ 0.80) and that most errors involved complex semantic distinctions, such as recognizing when a query refers to a specific subtopic rather than a general area, we did not expect reinforcement learning to yield dramatic overall performance gains. However, we hypothesized that fine-tuning through RLVR could modestly improve the model's ability to interpret user intent and mitigate some of the failure modes from the baseline, particularly the keyword bias demonstrated by the baseline.

Training Pipeline



Base Model: Qwen2.5-1.5b-Instruct with QLoRA adapter
RL algorithm: GRPO via TRL

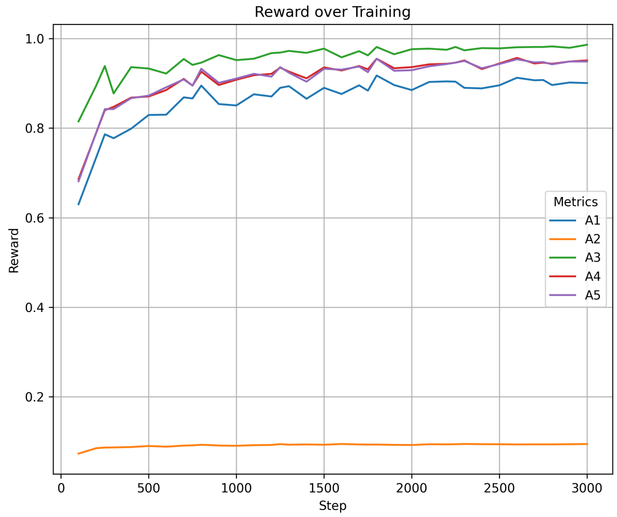
Our pipeline uses the GRPO trainer from the TRL library. We use the Qwen model with a QLoRA adapter, and the model is prompted to answer "yes" or "no" to whether a query-abstract pair is relevant. We use a simple reward function with +1 for correct predictions and +0 for incorrect with a tunable penalty for breaking formatting (single token yes/no). Metrics were logged with TensorBoard. We used GRPO because it effectively handles RL for language models that produce short and discrete text outputs. In our case, the model must choose between "yes" or "no" for each query–abstract pair, and GRPO stabilizes learning by comparing multiple sampled responses and rewarding those that are more accurate than others in the group. This relative feedback helps the model refine its predictions without overfitting to sparse binary rewards.

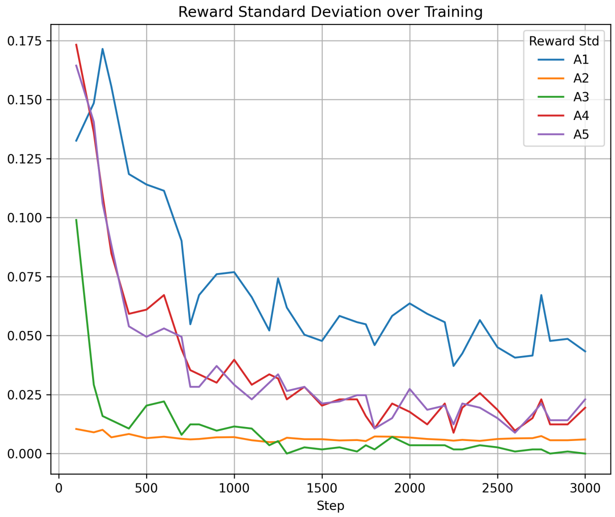
Hyperparamters and Ablations

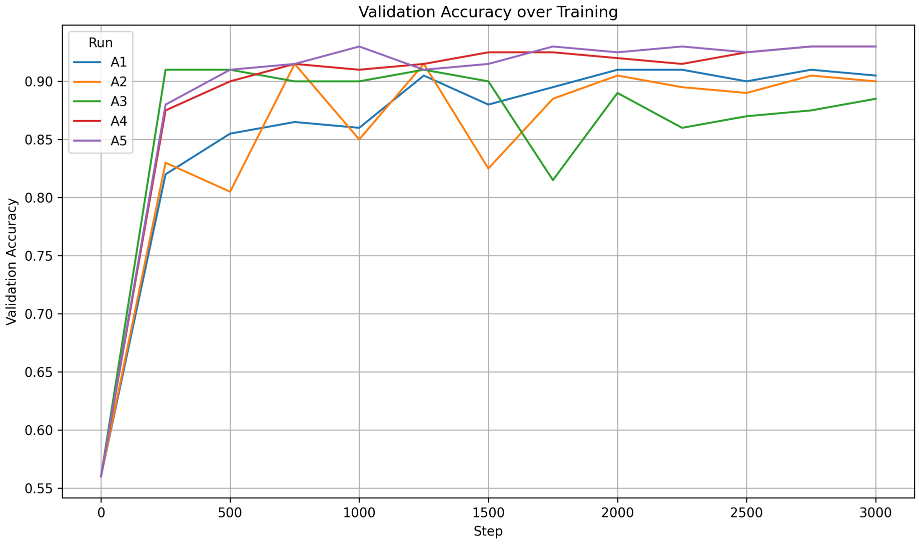
The following table shows the hyperparameter for training runs tested.

Setting	LR	KL Coeff.	Batch Size	Format Penalty	Max Prompt Tokens	Max New Tokens	Num Generations
A1	1e-4	0.05	64	0.1	1024	6	4
A2	1e-4	0.05	32	0.9	1024	6	4
A3	2e-5	0.01	64	1	1024	1	3
A4	5e-6	0.01	64	1	1800	1	4
A5	1e-5	0.001	16	1	1800	1	2

The following plots are from the above training runs extracted from TensorBoard.







Challenges and Observations

- We encountered problems with the format penalty during runs A1 and A2.
  - When format penalty was low (A1) the reward signal from correct answers dominated the penalty term causing the model to fail to learn the format (single token yes/no).
  - When format penalty was high (A2) the penalty term dominated the reward signal from correct answers and the model received very little reward causing training to stall.
  - This was addressed in the later runs by changing max new tokens to 1.
- Max Prompt Tokens made a significant difference in validation performance.
  - Lower Max Prompt Tokens results in some of the longer abstracts getting cut off when added to the prompt during training to save memory.
  - In A1-A3, training metrics were inflated compared to validation metrics.
  - This was addressed by increasing the max prompt tokens in the later training runs.
- OOM errors on higher memory training runs such as A4.

Final Performance on Evaluation Set

The following table is the best checkpoint (determined by validation set accuracy) evaluated on the evaluation set (A5 setting at 3000 steps).

Model	**Accuracy**	**Precision**	**Recall**	**F1**	**False Pos**	**False Neg**
Best Checkpoint	0.8991	0.8431	0.9214	0.8805	24	11

- The model achieves higher accuracy (≈ 0.9) than the baseline models (≈ 0.8).
- The model had better precision/recall tradeoff than the LLM baseline indicating that GRPO training mitigated some of the error patterns observed in part 3.
- Compared to the LLM baseline the model showed less keyword bias correctly identifying negatives with keyword matches where the LLM baseline identified false positives.
- The model still showed the specificity mismatch error pattern (although less severe than in the LLM baseline) sometimes failing to identify a query is referring to a subtopic rather than general topic. For example, it failed on the Graph Neural Net example shown in part 3.

Next Steps

- **Larger Base Model**: Experiment with a larger base model such as Qwen3-8B.
- **Constrained Decoding**: Implement constrained decoding to avoid the problems we encountered with the format and format penalty.
- **Reward Function**: Experiment with different reward function setups such as reward that incorporates logits/probabilities of yes/no.
- **Improve the dataset**; increase the diversity and number of labeled pairs to improve generalization.