Ammielle Wambo Becker – 24/06/2025

AI Alignment via RLHF using PPO

Aligning a code-generation model to PEP8 compliance via RLHF & PPO

Goal: Align the model's behaviour with a **desired behaviour**

Technique to align an intelligent agent with human preferences

AI Alignment via Reinforcement Learning from Human Feedback (RLHF) using Proximal Policy Optimization (PPO)

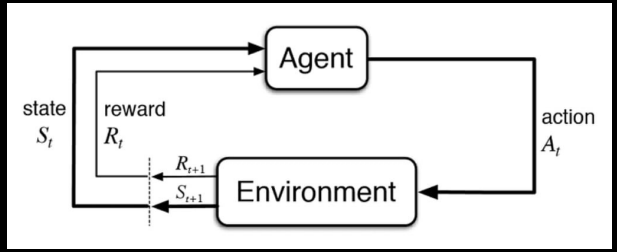
Policy gradient method, often used for deep RL when the policy network is very large

Aligning a code-generation model to PEP8 compliance via RLHF & PPO

Guide a pretrained code language model (LM) to produce PEP8-compliant Python snippets

Goal

RL Setup: Language Modeling as a MDP



Agent: CodeParrot-small
State (S_t): Prompt (input tokens)
Action (A_t): Next selected token
Environment: Transformer Reinforcement Learning wrapper:

- Presents each prompt as an episode
- Appends generated tokens to the state
- Ends the episode at an EOS or max length
- Returns the full token trajectory for scoring

Policy: LM's conditional token probabilities → modelingthe % of A_t space given the current S_t

- Wrapped in AutoModelForCausalLMWithValueHead for value estimation

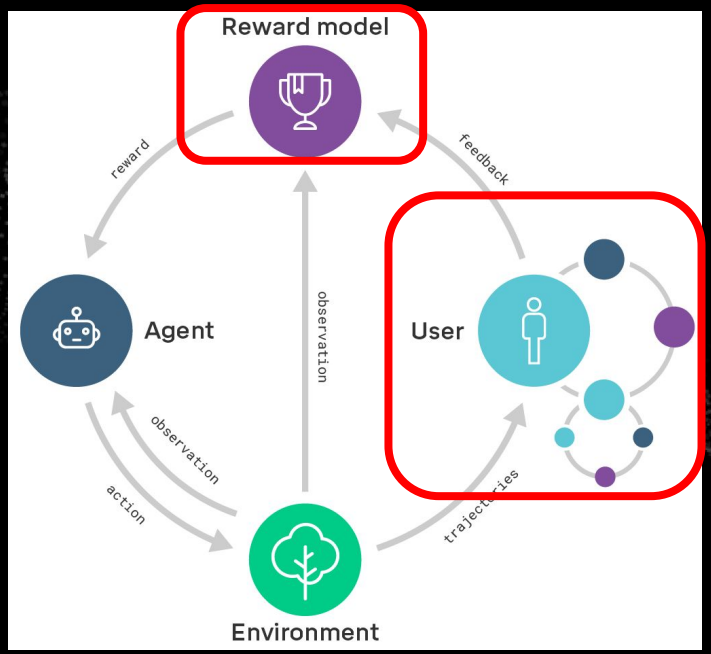
Reward model: DistilBERT binary classifier predicting PEP8 compliance (trained on flake8-labeled snippets)

WAT.ai

Reward Model and Trajectories for LMs in RLHF

Why RLHF? Human-in-the-loop feedback (here: style checker + reward model) drives nuanced preferences that supervised fine-tuning alone can't capture.

Trajectory: sequence of (S_t, A_t) over 1 prompt → 1 snippet



Reward signal: scalar style score (0–1) from the classifier

Frozen reference model applies a KL divergence penalty → keeps model fluent

PPO

Clips policy updates to prevent over-stepping (cliprange=0.2)

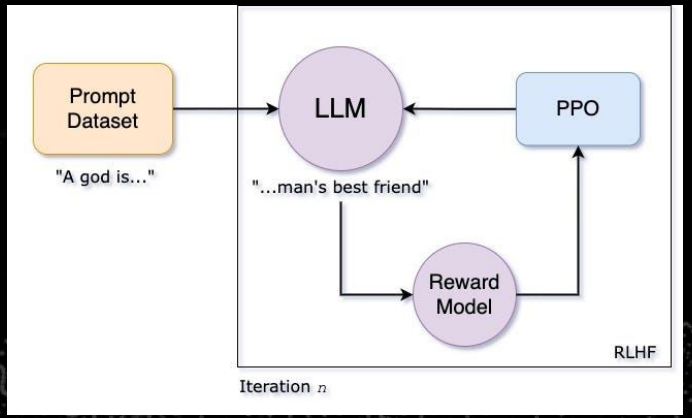
Optimizes a surrogate objective balancing:

$$L_t^{CLIP+VF+S}(\theta) = \hat{\mathbb{E}}_t\left[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi_\theta](s_t)\right]$$

c1 and c2 are coefficients.

Add an entropy bonus **to ensure sufficient exploitation.**

Squared-error value loss: $(V_\theta(s_t) - V_t^{targ})^2$



- RLHF reward (style compliance)
- KL penalty (fluency preservation)

Configured for 2 episodes, 1 PPO epoch, batch_size=4

WAT.ai

Code Style Enforcer

https://github.com/ammiellewb/wataionboarding

WAT.ai

Project Overview: Code Style Enforcer

Goal: Align an open-source code-generation model to follow a specific style guide (PEP8 standard)

Steps:

1. Supervised fine-tuning: CodeParrot-small (110 M GPT-2 backbone) on 200 perfectly PEP8-compliant Python snippets from CodeParrot-clean dataset
2. Reward model: DistilBERT classifier trained on 400 labeled examples (flake8 pass/fail)
3. RLHF via PPO: 2 gradient updates to refine style compliance
4. Evaluation: measure average style score before vs. after PPO

System Design Choices

Dataset: **CodeParrot-clean** (deduped, denoised Python) to avoid inconsistencies

Model: **CodeParrot-small (110 M)** for fast iteration and low memory footprint

Code style guide: **PEP8** enforced via **flake8** wrapper for transparent scoring

Reward model: **DistilBERT** for quick training **(34 M params)** and strong classification accuracy

Episodes: **2-shot demo** balances proof-of-concept vs. Google Colab compute budget

Lessons Learned

Introspect modules to understand sub-modules, expected argument names and results **(import inspect)**

Few-shot PPO works: small iteration counts suffice for demos, but real tasks need hundreds of episodes

Watch out for **GPU and System RAM usage limits** in Colab :(

Reward latency: **calling flake8 per snippet is slow** → train a faster classifier for more labels

WAT.ai

Sources

- Reinforcement Learning from Human Feedback explained with math derivations and the PyTorch code.
- TRL - Transformer Reinforcement Learning
- PPO Trainer
- [2403.17031] The N+ Implementation Details of RLHF with PPO: A Case Study on TL;DR Summarization
- An introduction to Policy Gradient methods - Deep Reinforcement Learning

Model: codeparrot/codeparrot-small · Hugging Face
Dataset: codeparrot/codeparrot-clean · Hugging Face
flake8 docs: Flake8


Thanks for listening
WAT.ai