Week 3

RLHF (Reinforcement learning from Human feedback)

↑ Helpful ↓ Harm X Dangerous Topics & Personalized LLM (learn preferences) / learning plans / AI assistants

Rollout = Sequence of actions & states

Human feedback on whether the generated text is toxic

on a scale [0 1] → some prompt to multiple labelers → consensus

↓

expensive → use Reward Model to classify the outputs, no more humans in the loop after training

Reward Model: binary classifier (can be LM)

classify: favour human prefered outputs

GOAL ↓

reduce the probability of toxicity (AVG across completions)

PPO RL algorithm

you can use PEFT to only update limited # weights ↓ use same LLM

Proximal Policy optimization → makes updates to the LLM for it to be more aligned to human preferences

keep changes in small region

BUT Reward hacking could happen → prevent that the RL-updated model shifts too far from OG Reward Model

- Add penalization term (KL Dwergence between 2 completions)
- IF using PEFT the same underlying LLM is used, just update PPO model with trained parameters

Optimization Techniques

1 Distillation

→ larger teacher model (already fine tuned on training data) trains a student model to minimize

→ student model used for inference — quicker loading, ↓ latency!

→ work best for encoder only models

Distillation Loss

Compare probability distributions of 2 models

Teacher one matches Ground truth so

we add Temperature > 1 (↑ creativity)

distribution is broader, ↓ peaked

set of tokens (soft labels) similar to GT

(student model generates soft prediction)

Student Loss

Here Temperature = 1

generate predictions (Hard predictions) correct also based on GT data! (HARD Labels)

2 Post Training Quantization

→ Here we reduce dimension of LLM to optimize it for deployment
- ↓ model size
- ↓ memory footprint
- ↓ compute needed to serve the model

→ Transform its weights to lower precision representation
- 16 bit floating point
- 8 bit integer

→ can be applied to model weights (+ Activation layers for higher impact)

→ Requires Calibration to capture range of original parameters

3 Pruning

→ Reduce model size by eliminating that are not contributing to overall model performance

$w \sim \phi$

→ either full retraining or PEFT / LORA

→ could have not much impact if low % of $w = 0$

	Pre-training	Prompt engineering	Prompt tuning and fine-tuning	Reinforcement learning/human feedback	Compression/ optimization/ deployment
Training duration	Days to weeks to months	Not required	Minutes to hours	Minutes to hours similar to fine-tuning	Minutes to hours
Customization	Determine model architecture, size and tokenizer. Choose vocabulary size and # of tokens for input/context Large amount of domain training data	No model weights Only prompt customization	Tune for specific tasks Add domain-specific data Update LLM model or adapter weights	Need separate reward model to align with human goals (helpful, honest, harmless) Update LLM model or adapter weights	Reduce model size through model pruning, weight quantization, distillation Smaller size, faster inference
Objective	Next-token prediction	Increase task performance	Increase task performance	Increase alignment with human preferences	Increase inference performance
Expertise	High	Low	Medium	Medium-High	Medium

Using LLMs in Applications

RETRIEVAL AUGMENTED GENERATION [RAG]

→ framework to make use of External Data Sources at inference time

→ solves CUTOFF ISSUE (outdated info) w/o retraining

→ different implementations → Facebook 2020

↓

Retriever — Query encoder* / External info (CSV, DB, web pages)

→ expanded prompt!

* encodes users' input so that it can be used to query the data source

→ Issues

1 context window is limited so external Data sources are 'chopped'

2 for efficient identification of semantically related text we need embedding vectors of external data stored in vector stores

vector DB each vector has a key → useful for source citation

Chain of Thoughts Prompting

→ breaking problems down in steps to 'think like a human'

Program Aided Language Models

→ interact w/ external code interpreter → carries calculations in the generated code

→ may need orchestrator → ReACT (2022) { Chain of thought + Action Planning

→ Lang Chain Framework : provides modular pieces that contain the components needed by LLM

e.g prompt templates

memory to store interaction

prebuilt tools to call external DB or API

both pre-determined chains

Agent interprets input & chooses which tools to use

(PAL & ReAct)