Capek: A Competitive Distributed Artificial Intelligence

capekai@proton.me

Abstract

A distributed system that competitively tunes models in response to human feedback.

Introduction

Large language models allow the condensing of the vast corpus of human knowledge into a trained dataset. This initial training is best done in a highly interconnected datacenter to process many petabytes of data. However, the models that are generated from this training process are suitable for distribution over the internet for further tuning. Efficient fine tuning (see Llama-Adapter R Zhang et al 2013) can take a trained model and vastly increase its usefulness by applying fine tuning and Reinforcement Learning from Human Feedback.

Currently this tuning is done on models by discrete teams using external funding. We propose a system where an unbounded number of models compete for fees that are paid by end users. Fees are distributed to miners and models, thus incentivising models to carefully select further tuning functions and thus become more attractive to consumers of the network.

Nodes

Nodes load publicly shared models from the wider internet. These nodes then accept jobs from the network which may be inference tasks or tuning functions. A distributed network of mining nodes and an efficient task routing system allows trust-less edge compute of large scale models.

Because inference tasks will not always be present, mining nodes may elect instead to process tuning functions, which are of lower precedence (and thus less reward) than inference tasks. Heterogenous tuning functions are distributed across many nodes for an individual model, and merged and snapshotted at the end of an epoch.

Nodes receive fees from consumers for inference against the models, and from models for further tuning of models.

Inference Tasks

A node receives input and runs inference against the chosen model, returning the result to the network. The network may choose to compare the results of multiple nodes against known model states to ensure the nodes are being truthful in their responses.

Tuning Functions

Tuning functions are run by nodes in a sandboxed runtime. They are turing complete and are used to apply reinforcement learning to tune the models. Tuning functions will commonly incorporate RLHF by requesting preferences from consumers.

Models

Models are large neural networks that support inference and can be trained. Models receive a fee paid by end users for inference.

Models are uploaded to the network by trainers, who are issued a fixed number of tokens in return.

Token holders vote for tuning functions. These votes are processed by the network and fees earned by the model are distributed to further refine the model in the next epoch.

Consumers

Consumers submit inference tasks to the network, specifying an input, a model and fee. Consumers may tend to favour particular models as they become adapted to the responses of that model and its tuning over time.

Consumers may earn fees by indicating their preferences to tuning functions. These responses are used by nodes to apply reinforcement learning to the issuing model.

Orchestration

A distributed ledger is used to receive fees from consumers and distribute them to nodes and models. This ledger records votes from token holders and uses it to create tuning jobs for models using inference fees at the end of each epoch.

Incentive

Consumers pay fees to have rapid inference of high quality models. These models may be too large or complex to efficiently compute on their own hardware, or they may be mutating sufficiently fast that is infeasible for consumers to keep a local copy of the model. Consumers can introduce funds to the system from an external source, or they may earn fees by participating in RLHF by revealing their preferences to tuning functions.

Nodes receive fees by providing permission-less compute to models, whether inference or applying tuning functions.

Models receive fees from consumers and use these fees to pay nodes to further tune the model to make the model more attractive to consumers and thus receive future fees.

Holders own tokens in models. Markets may form that value these models relative to each other and allows holders to sell their interest in a model.

Nodes and models have a symbiotic relationship whereupon they depend on each other to generate more attractive models for consumers to interact with.