LLM fine tuning

GPT-3.5 model & fine tuning it with your own data for your specific application.

* fine tuning lets you get more out of the models available through the API by

  (i) Higher quality results than prompting.
  (ii) Ability to train on more example than can fit in a prompt
  (iii) Token saving due to shorter prompts
  (iv) Lower latency requests.

* Fine-tuning is not available for all models.

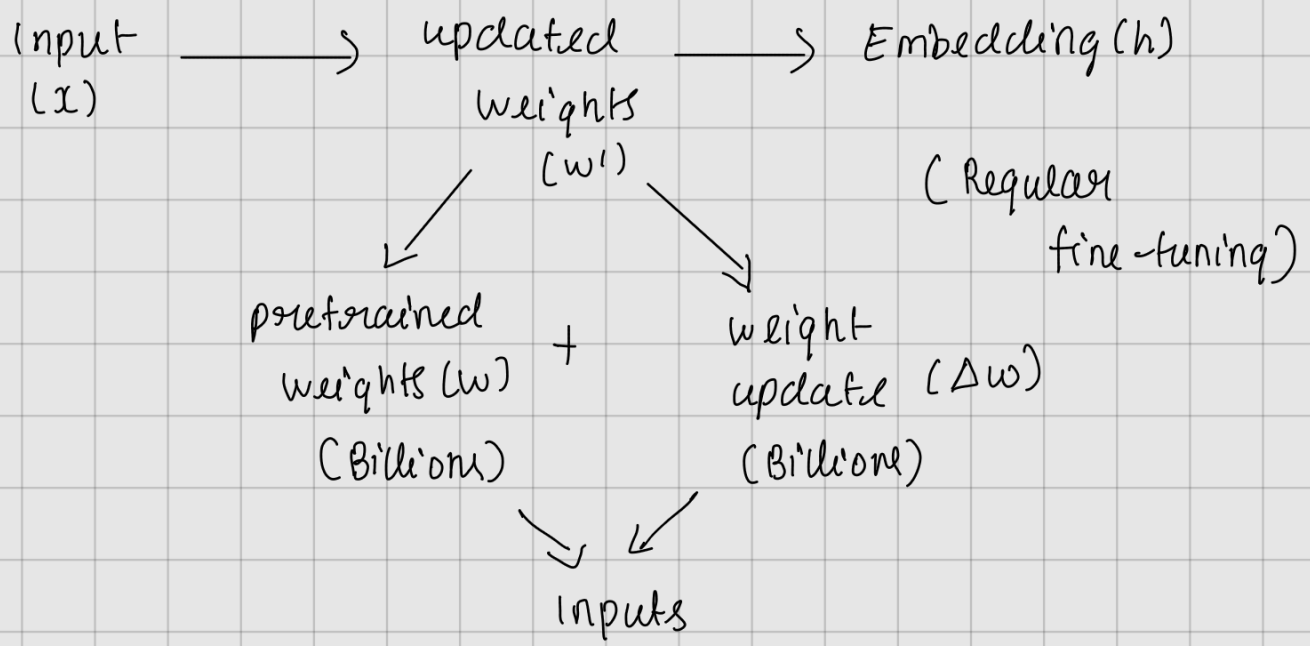
* Improve results?

  (i) Setting the style, tone, format or other qualitative aspects
  (ii) Improving reliability at producing a desired output
  (iii) correcting failures to follow complex prompts
  (iv) Handling many edge cases in specific ways
  (v) performing a new skill/task that's hard to articulate in a prompt.

How fine-tuning works?

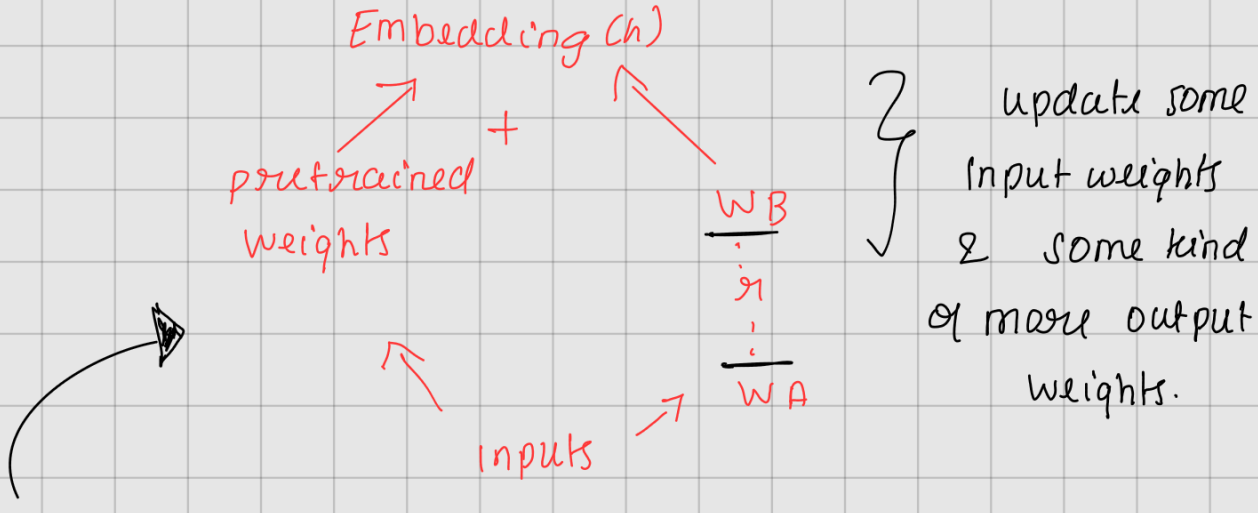
↳ LORA (Low Rank Adaptive) fine tuning.

Regular fine-tuning, is really just continuing the entire training process on a particular set of data.



fine-tuning process

LORA weights W_A & W_B represent Δw



Instead of having to re-train the billions of parameters in the large GPT models, we can instead only need to update and train 1.1. - 2.1. of the parameters weights (OpenAI save those weights) fine tuned Model.

Fine tuning (requirement)

- base model
- training dataset
  - known inputs
  - Desired known outputs.

Fine-tuning Application & use cases:

(i) changing style & tone to your own custom style.

(ii) Fine-tuning for very specific structured output or custom code function output.

(iii) classifications based on proprietary data sources.

(iv) Remember to always use the base-model first and see if prompt engineerings or extra context can achieve your desired result

(v) Fine-tuning the model is more expensive process that require datasets.

(vi) it will also be more costly to query your own fine-tuned model verus the base model.

(Storing LORA weights)