LLM PROMPTING – FINE TUNING

© DAN MITCHELL

Outline

- Introduction to fine tuning
  - What?
  - Why?
  - Examples

- Methodologies for fine tuning
  - Where?
  - How much?
  - Impact?

What is Fine Tuning?

- An LLM is a neural network made up of many weights and biases
  - Also, a dictionary of tokens
- Pre-trained on a huge corpus of text over many tasks
- Fine Tuning changes something about the model
  - Smaller training set of a few tasks
  - Change the model using SGD over this training set

Why Fine Tune?

- A pre-trained LLM is trained on a broad set of tasks
- Sometimes we want to do a better job at accomplishing specific tasks
  - RAG inserts context to help answer questions
- Fine tuning changes the style of the output
  - Encourages specific output formats

Customer Support

- Reflect brand’s tone
- Comply with internal policies

- Airline
  - Respond to delays with empathy
  - Offer standard compensation

Regulated Industries

- Enforce constraints and compliance
- Prevent topic specific hallucination

- Banking
  - Don't provide financial advice
  - Provide requisite legal caveats

Sales Outreach

- Outbound messages should comply with brand's tone
- Use industry specific language
- Startup Marketing
  - Cold emails
  - Emails can mimic top salespeople

Legal Documents

- Contract drafting matches required format
- Avoid hallucination

- Real estate firm
  - Consistent format of lease agreements
  - Conform to multiple jurisdictions

Language Dialects

- Many countries speak Spanish
- Each country has their own dialect
- Spanish companies should fine tune to get the model to use the Spanish dialect – not the Mexican dialect!
  - Ustedes / Vosotros

Full Fine Tuning

- An LLM is a neural network made up of many weights and biases
- These weights and biases are trained using SGD on a HUGE training set
- We have a smaller dataset of text that we want the LLM to mimic
  - Take a few more steps of SGD using this training set

Full Fine Tuning

- Advantages
  - Can dramatically impact model output
- Drawbacks
  - Efficiency
  - Catastrophic Forgetting
  - Scalability

Soft Prompting

- Add a consistent set of words to the beginning of all prompts
- Encourage consistent format
- Soft Prompting
  - Prepend all queries with consistent text
  - Specific instructions
  - Few-shot prompting
  - Also known as prompt engineering

Soft Prompting

- My plane was delayed by 4 hours because of a thunderstorm; I would like a refund please.

- Reply to the following request using empathy and compassion. It is our policy to only provide refunds if a delay was our fault, not because of weather.

  Request: My plane was delayed by 4 hours because of a thunderstorm; I would like a refund please.

Soft Prompt

- Benefits
  - Easy to implement
  - Intuitive
- Drawbacks
  - Ad hoc: no training
  - Prompt injection

Parameter-Efficient Fine Tuning

- Full fine tuning changes the entire model
- Soft prompting doesn't change the model at all
- Parameter-Efficient Fine Tuning changes just part of the model
  - Which part?

Prompt Tuning

- Instead of prepending words to the beginning of a prompt, prepend trainable tokens to the prompt
- A prompt is first transformed to a sequence of tokens
- Add some new tokens to the dictionary, prepend the sequence

Prompt Tuning

- I like ice cream.
- [BOS] [T2313] [T4652] [T21] [T912] [T7743] [EOS]

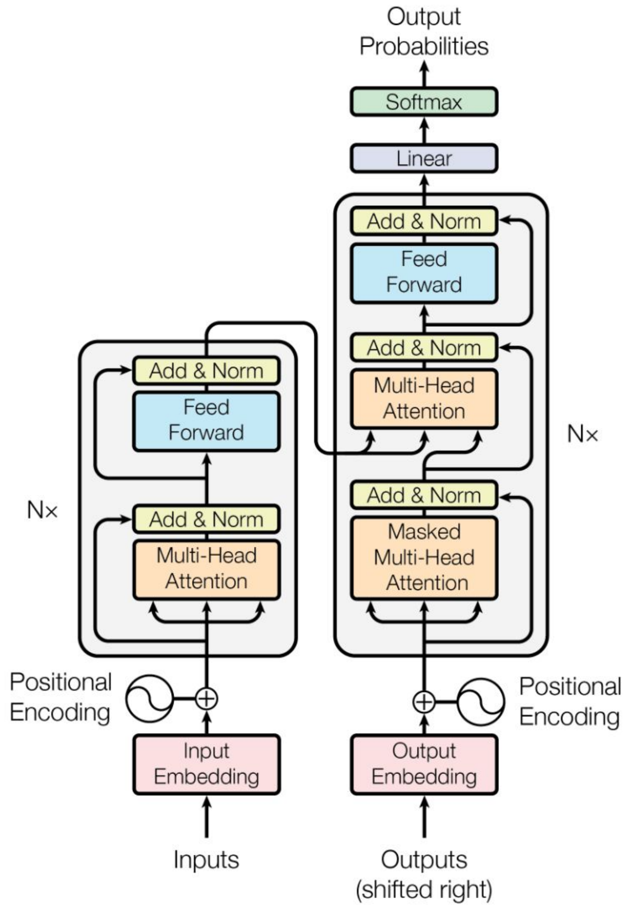
- [V1] [V2] [V3] … [VK] [BOS] [T2313] [T4652] [T21] [T912] [T7743] [EOS]
- [V1] – [VK] are vectors whose values are modified using SGD on the smaller training set

Prompt Tuning

- Advantages
  - Same general idea of prompt engineering, but more tunable
  - Can prevent prompt injection
  - Only a few parameters to fit
  - Can change appended tokens depending on task
- Drawbacks
  - Doesn’t actually change the model
  - Not as powerful as newer tools

Prefix Tuning

- Prompt tuning prepends trainable vectors to the very beginning of the transformer process
- Prefix tuning prepends trainable vectors to the beginning of the sequence of vectors before every attention layer

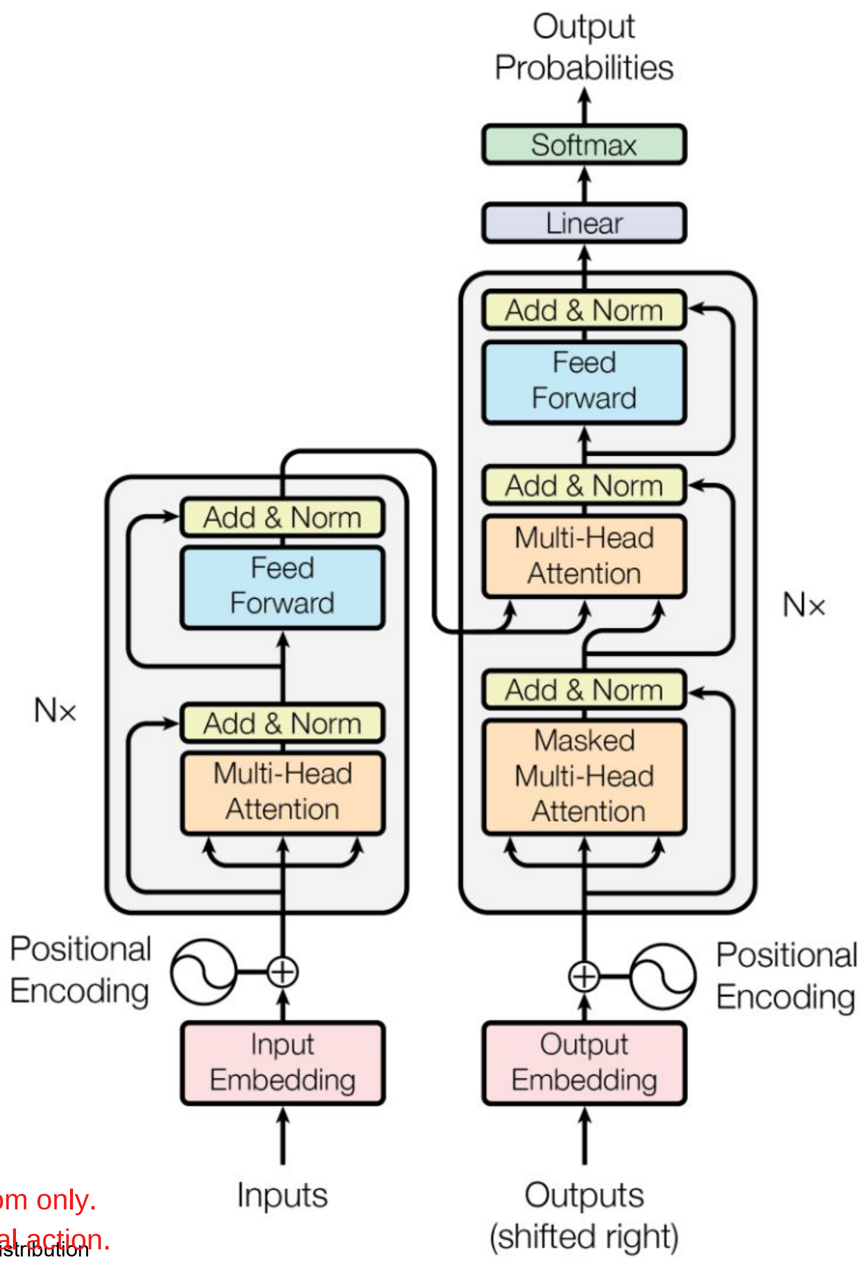


Prefix Tuning

- Advantages
  - Same idea as prompt tuning
  - Still not many parameters to fit
  - More powerful than prompt tuning
- Drawbacks
  - Does not change the transformer
  - Just changes what is sent through it

Adapter Layers

- Now let’s change the actual transformer!
- Insert more FF layers



Adapter Layers

- Advantages
  - Can change layers for specific tasks
  - Modifies how tokens are processed
  - Not sequence based
  - Not many parameters to fit
- Drawbacks
  - Requires structural change to model

LoRA

- Self Attention
  - Start with a sequence of vectors
  - Multiply each vector by a query, key, and value matrix
  - Take outputs of query and key matrix multiplications and calculate dot-product similarity
  - Calculate weighted average of output from value matrix multiplication

LoRA

- This requires 3 sets of matrices – Q, K, V
- The values in these matrices are pre-trained using the huge dataset
- Low-Rank Adaptation seeks to change the values of Q and V using the smaller training set
  - $Q^* = Q + \Delta Q$
  - $V^* = V + \Delta V$

LoRA

- Q and V are typically large matrices
  - M x N
- This means ΔQ and ΔV need to be the same size
- Instead of fitting all these parameters, rephrase as:
  - $\Delta Q = A_Q \cdot B_Q$
  - A_Q is M x R
  - B_Q is R x N
  - R is a small number – typically 4
- Imagine M and N are 1000
  - Q and V each have 1M parameters
  - ΔQ and ΔV each have 8000 parameters

LoRA

- Advantages
  - Structure of LLM stays the same
  - Very few parameters to fit
  - Most powerful fine-tuning strategy so far
- Drawbacks
  - Struggles with very complex tasks, like writing code
  - Still needs to use huge model

QLoRA

- Although LoRA is lightweight, you still need to load the entire LLM into memory to fit weights of A and B
- An LLM like LLaMA or DeepSeek has billions of parameters
- Still may not be possible to fine tune these on consumer grade hardware

QLoRA

- All the billions of parameters in an LLM are simply numbers stored in binary
  - Typically, 16 or 32 bit
- Quantization rounds those numbers to fewer bits
  - ~5 bit is common
  - This requires MUCH less memory to store the entire LLM
- Once the LLM has been quantized, then use LoRA

QLoRA

- Advantages
  - More memory efficient
  - All the benefits of LoRA on consumer grade hardware
- Drawbacks
  - Some accuracy loss from quantization
- QLoRA is the most popular method for fine tuning

Scalability

- We may want to do some fine tuning for several different tasks
- Many methods for PEFT are scalable
  - In LoRA, train ΔQ and ΔV for each task individually
  - Store the base model and then store each ΔQ and ΔV
  - We don’t have to store the whole new model for each task
- This is NOT possible for full fine tuning!

Fine Tuning

Model	Alteration	Efficiency	Effectiveness
Full Fine Tuning	All model parameters	Very low	Very high (risks)
Soft Prompting	Input words	Extremely high	Very low
Prompt Tuning	Input tokens	Very high	Low
Prefix Tuning	All token sequences	High	Moderate
Adapter Layers	Adds FF layers	Moderate	High
LoRA	Change attention layers	High	Very high
QLoRA	LoRA + Quantization	Very high	High