Fine tuning LLMs

Arun Kumar Rajasekaran

What and Why?

- General-Purpose vs. Specialized: Pre-trained LLMs are excellent for general tasks, but they might not be ideal for specific applications.
- Fine-Tuning Tailors the Model: Fine-tuning adjusts the pre-trained LLM's parameters using your dataset to improve its performance on a particular task.

Fine-tuning is the process of taking a pre-trained model and further training it on a domain-specific dataset. The fine-tuning process offers considerable advantages, including lowered computation expenses and the ability to leverage cutting-edge models without the necessity of building one from the ground up.

Some scenarios

- Question Answering: Train the LLM to answer questions in your specific domain (e.g., legal, medical).
- Machine Translation: Fine-tune for a specific language pair or domain (e.g., scientific articles).
- Text Summarization: Tailor the LLM to summarize documents in a particular style or length.
- Creative Text Generation: Fine-tune for a specific genre (e.g., writing poems, scripts).
- Code Generation: Train the LLM to generate code in a particular programming language.
- Chatbots: Train the LLM on conversational data to create chatbots that can engage in more natural and informative dialogues.

Benefits ..

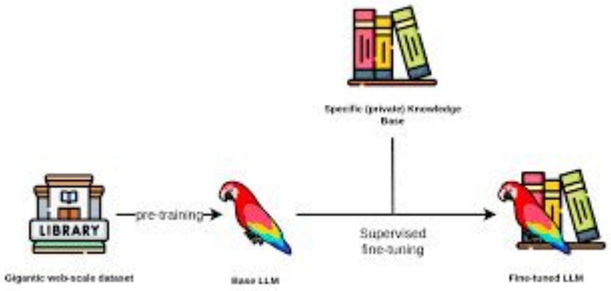
- Improved Performance: Fine-tuning can significantly enhance the LLM's accuracy and effectiveness on your specific task.
- Domain-Specific Knowledge: The LLM learns the nuances and terminology of your domain, leading to more relevant and accurate outputs.
- Reduced Training Time: Compared to training an LLM from scratch, fine-tuning leverages the pre-trained model's knowledge, saving time and resources.

The Different Types of Fine-tuning

1. Supervised fine-tuning

   The model is further trained on a labeled dataset specific to the target task to perform, such as text classification or named entity recognition.

   In this method, a dataset comprising labeled examples is utilized to adjust the model's weights, enhancing its proficiency in specific tasks.



Few types within…

- Full Fine Tuning (Instruction fine-tuning): Instruction fine-tuning is a strategy to enhance a model’s performance across various tasks by training it on examples that guide its responses to queries. However, it demands sufficient memory and computational resources, similar to pre-training, to handle the storage and processing of gradients, optimizers, and other components during training.
- Parameter Efficient Fine-Tuning (PEFT) is a form of instruction fine-tuning that is much more efficient than full fine-tuning.PEFT addresses this by updating only a subset of parameters, effectively “freezing” the rest.

PEFT …

There are various ways of achieving Parameter efficient fine-tuning. Low-Rank Adaptation LoRA & QLoRA are the most widely used and effective.

- LoRA is an improved finetuning method where instead of finetuning all the weights that constitute the weight matrix of the pre-trained large language model, two smaller matrices that approximate this larger matrix are fine-tuned. These matrices constitute the LoRA adapter. This fine-tuned adapter is then loaded into the pre-trained model and used for inference.

PEFT …

There are various ways of achieving Parameter efficient fine-tuning. Low-Rank Adaptation LoRA & QLoRA are the most widely used and effective.

- QLoRA represents a more memory-efficient iteration of LoRA. QLoRA takes LoRA a step further by also quantizing the weights of the LoRA adapters (smaller matrices) to lower precision (e.g., 4-bit instead of 8-bit). This further reduces the memory footprint and storage requirements. In QLoRA, the pre-trained model is loaded into GPU memory with quantized 4-bit weights, in contrast to the 8-bit used in LoRA. Despite this reduction in bit precision, QLoRA maintains a comparable level of effectiveness to LoRA.

The Different Types of Fine-tuning

2. Few-shot learning

There are some cases where collecting a large labeled dataset is impractical. Few-shot learning tries to address this by providing a few examples (or shots) of the required task at the beginning of the input prompts. This helps the model have a better context of the task without an extensive fine-tuning process.

The Different Types of Fine-tuning

3. Transfer learning

Even though all fine-tuning techniques are a form of transfer learning, this category is specifically aimed to allow a model to perform a task different from the task it was initially trained on. The main idea is to leverage the knowledge the model has gained from a large, general dataset and apply it to a more specific or related task.

The Different Types of Fine-tuning

4. Domain-specific fine-tuning

This type of fine-tuning tries to adapt the model to understand and generate text that is specific to a particular domain or industry. The model is fine-tuned on a dataset composed of text from the target domain to improve its context and knowledge of domain-specific tasks.

For instance, to generate a chatbot for a medical app, the model would be trained with medical records, to adapt its language understanding capabilities to the health field.

Fine tuning best practices

1. Data Quality and Quantity
2. Hyperparameter tuning
3. Regular evaluation

Things to note …

1. Overfitting

   Using a small dataset for training or extending the number of epochs excessively can produce overfitting. This is usually characterized by the model showing high accuracy on our training dataset but failing to generalize to new data.

2. Underfitting

   Conversely, insufficient training or a low learning rate can result in underfitting, where the model fails to learn the task adequately.

Things to note …

3. Catastrophic forgetting

In the process of fine-tuning for a particular task, there's a risk that the model might lose the broad knowledge it initially acquired. This issue, referred as catastrophic forgetting, can diminish the model's ability to perform well across a variety of tasks using natural language processing.

4. Data leakage

Always make sure that training and validation datasets are separate and that there's no overlap, as this can give misleading high-performance metrics.

Useful links …

https://www.datacamp.com/tutorial/fine-tuning-large-language-models

https://medium.com/@mohitdulani/fine-tune-any-llm-using-your-custom-dataset-f5e712eb6836

https://medium.com/@mohitdulani/fine-tune-any-llm-using-your-custom-dataset-f5e712eb6836

Next steps …

-> Fine tuning vs Prompt engineering vs RAG (Retrieval-Augmented Generation)

-> How to quantitatively evaluate base vs fine tuned model?