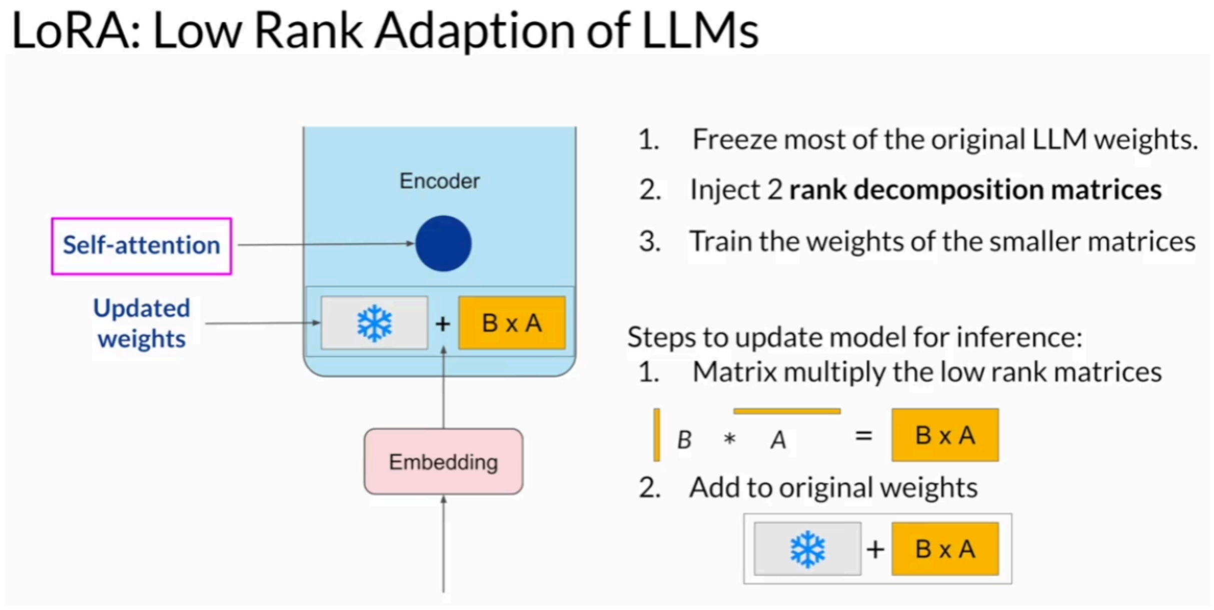
PEFT techniques 1: LoRA

Training large Language Models (LMs) is computationally demanding, necessitating innovative approaches to reduce memory requirements while maintaining performance. Low-rank Adaptation (LoRA) emerges as a promising parameter-efficient fine-tuning technique within the re-parameterization category. In this article, we delve into the mechanics of LoRA, its benefits, and its practical implications for LM training.

LoRA aims to streamline fine-tuning by minimizing the number of parameters updated while preserving LM performance. It achieves this by freezing original model parameters and introducing low-rank matrices alongside them. These matrices, with dimensions tailored to match the original weights, undergo supervised learning, significantly reducing trainable parameters.

Implementation of LoRA:

During fine-tuning, LoRA freezes the original LM weights and trains the low-rank matrices. These matrices are multiplied to create a matrix with dimensions matching the frozen weights, which is then added to the original weights. This process results in a fine-tuned model with performance gains, achieved with a fraction of the parameters.



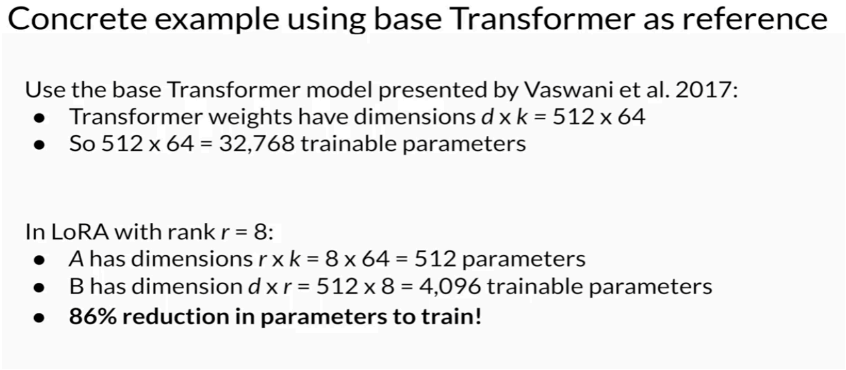
LoRA: Low Rank Adaption of LLMs

1. Freeze most of the original LLM weights.
2. Inject 2 **rank decomposition matrices**
3. Train the weights of the smaller matrices

Steps to update model for inference:
1. Matrix multiply the low rank matrices
2. Add to original weights

Advantages of LoRA:

One significant advantage of LoRA is its ability to drastically reduce the number of trainable parameters, often by over 80%. This reduction enables fine-tuning on a single GPU, eliminating the need for distributed computing clusters. Additionally, LoRA facilitates efficient model adaptation for multiple tasks, as low-memory requirements allow storing LoRA matrices for various tasks.



Concrete example using base Transformer as reference

Use the base Transformer model presented by Vaswani et al. 2017:

- Transformer weights have dimensions d x k = 512 x 64
- So 512 x 64 = 32,768 trainable parameters

In LoRA with rank r = 8:

- A has dimensions r x k = 8 x 64 = 512 parameters
- B has dimension d x r = 512 x 8 = 4,096 trainable parameters
- **86% reduction in parameters to train!**

Consider fine-tuning a transformer architecture with LoRA. For instance, if the original transformer weights are 512x64, each matrix contains 32,768 parameters. With LoRA using a rank of eight, the total trainable parameters reduce to just 4,608, representing an 86% reduction. This efficiency enables seamless fine-tuning for diverse tasks without significant computational overhead.

Performance Comparison:
Comparing LoRA fine-tuning with full fine-tuning reveals promising results. While full fine-tuning yields slightly higher performance gains, LoRA achieves comparable improvements with significantly fewer resources. The choice of LoRA rank impacts model performance, with ranks between 4-32 offering a balanced trade-off between parameter reduction and performance preservation.

Future Directions:
Optimizing the choice of LoRA rank remains an active area of research, with practitioners exploring various ranks and evaluating their impact on model performance. As LoRA gains traction, its principles extend beyond LM training, offering insights into efficient fine-tuning methods across diverse domains.