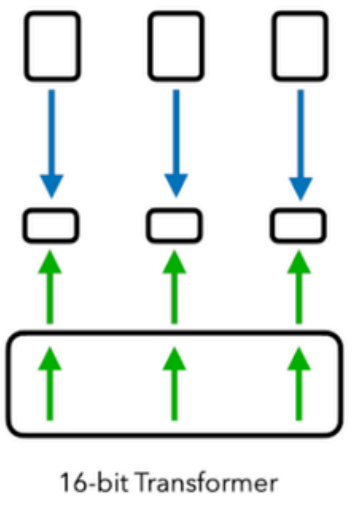
Mastering LLMs

Analytics Vidhya

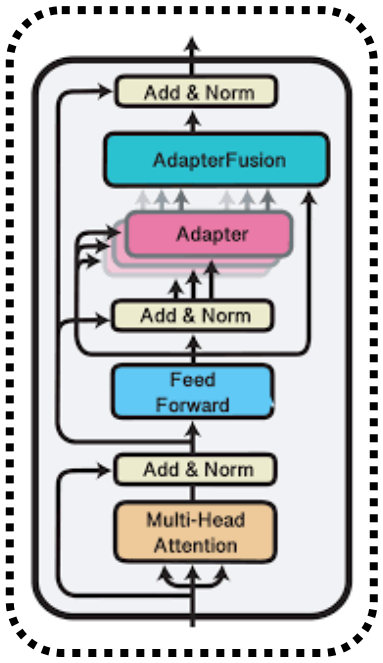
Day 28: PEFT Beyond LoRA

LoRA



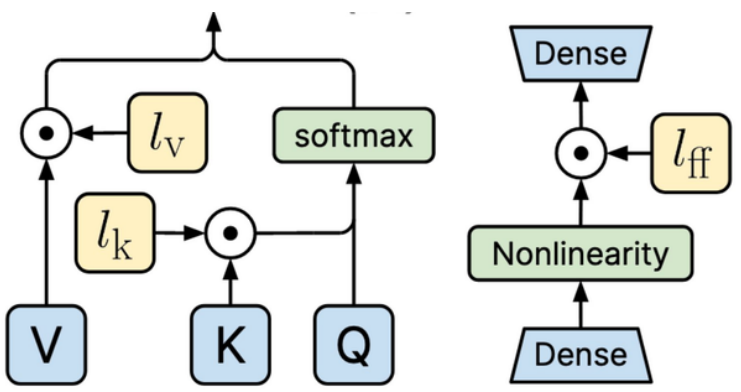
Use LoRA if you want a general-purpose, low-rank adaptation with strong efficiency

AdapterFusion



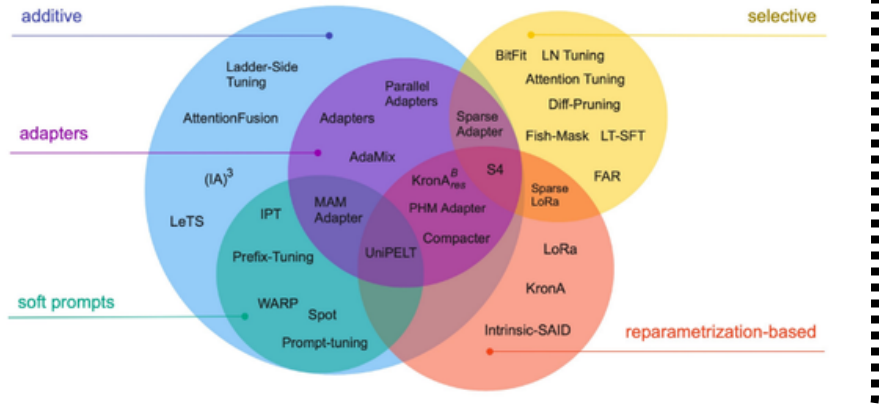
Use AdapterFusion if you work on multi-task learning and need a composable adapter-based approach.

$(IA)^3$



Use IA3 if you need ultra-low memory fine-tuning and prefer simple activation rescaling.

Compacter



Use Compacter when memory constraints are extreme, but task performance remains critical.

- Parameter-Efficient Fine-Tuning (PEFT) has revolutionized the field of deep learning, particularly in adapting large-scale models without the need for full fine-tuning.

- While LoRA (Low-Rank Adaptation) has gained significant popularity, other methods like **IA3** (Infused Adapter-Attention), **AdapterFusion**, and **Compacter** offer alternative approaches, each with unique strengths. This post explores these techniques beyond LoRA, discussing their mechanisms, advantages, and ideal use cases.

Why PEFT?

- Fine-tuning massive models like GPT and BERT can be computationally expensive and storage-intensive. PEFT methods tackle this challenge by introducing small, trainable parameters that modify specific layers of the model while keeping the majority of weights frozen. This dramatically reduces computational overhead and enables efficient adaptation to new tasks.

Beyond LoRA

IA3 (Infused Adapter-Attention)

- **Key Idea**: Instead of modifying the model's weights directly, IA3 introduces multiplicative learnable vectors that rescale activations within the Transformer layers.

How it Works

- It introduces learnable scalars per attention head and MLP layer.

- These scalars adjust activations before applying the feedforward transformations.

- Unlike LoRA, which introduces additional low-rank matrices, IA3 modifies existing activations without adding extra trainable parameters in weight matrices.

Advantages

- Reduces memory and computation even further than LoRA.
- Less intrusive modification to pre-trained weights.
- Suitable for large-scale distributed settings where communication overhead matters.

Ideal Use Cases

- When fine-tuning with ultra-low memory constraints.
- Applications where maintaining pre-trained weights' integrity is crucial.

AdapterFusion

Key Idea: AdapterFusion combines multiple fine-tuned adapters from different tasks into a single model, leveraging their shared knowledge.

How it Works:

- Fine-tune separate adapters on different tasks.
- Introduce a fusion mechanism that combines learned representations from multiple adapters.
- This allows transfer learning across domains without retraining the whole model.

Advantages:

- Facilitates knowledge transfer across multiple tasks.
- Reduces the need for training new adapters from scratch.
- Can improve generalization on downstream tasks.

Ideal Use Cases:

- Multi-task learning and knowledge transfer.
- Scenarios where pre-trained adapters exist for related tasks.

Compacter

Key Idea: Compacter reduces adapter parameters by using low-rank reparameterization and hypercomplex multiplications to compress adapter layers.

How it Works:

- Utilizes Kronecker product-based parameterization to reduce redundancy.
- Instead of training full adapter layers, it learns a compact set of task-specific parameters.

Advantages:

- Further parameter reduction compared to standard adapters.
- Ideal for extreme low-resource environments.
- Retains effectiveness while using significantly fewer parameters.

Ideal Use Cases:

- Devices with extreme memory limitations (e.g., edge AI, IoT).
- Resource-efficient multi-task adaptation.

Analysis of PEFT Methods

Method	Trainable Params	Memory Efficiency	Modularity	Multi-task Learning	Performance Trade-offs
LoRA	Low	High	Moderate	Limited	May underperform in task composition
IA3	Very Low	Very High	Low	No	Can be less expressive than LoRA
AdapterFusion	Medium	Moderate	High	Yes	Requires multiple adapters
Compacter	Very Low	Very High	Moderate	Yes	More complex implementation

Choosing the Right PEFT Method

Choosing the right PEFT approach depends on compute resources, modularity requirements, and task diversity:

- **Use LoRA** if you want a general-purpose, low-rank adaptation with strong efficiency.
- **Use IA3** if you need ultra-low memory fine-tuning and prefer simple activation rescaling.
- **Use AdapterFusion** if you work on multi-task learning and need a composable adapter-based approach.
- **Use Compacter** when memory constraints are extreme, but task performance remains critical

Analytics Vidhya

Stay Tuned for **Day 29** of

Mastering LLMs