

Adam (Adaptive Moment Estimation) 和 AdamW 是深度学习中两种非常流行的优化器，它们都基于自适应学习率的概念。它们的主要区别在于如何处理 权重衰减 (Weight Decay)，这是一种常用的正则化技术。

### Adam 优化器

Adam 是一种将 AdaGrad 和 RMSProp 优点结合起来的优化算法。它能够为每个模型参数计算独立的自适应学习率。

Adam 的主要特点：

自适应学习率：Adam 为每个参数维护两个指数移动平均值：

一阶矩 (m)：梯度的指数移动平均值（类似于动量）。

二阶矩 (v)：梯度平方的指数移动平均值。

这些矩用于调整每个参数的学习率，使得对于稀疏梯度（不经常出现的特征），学习率更大；对于常见特征，学习率更小。

动量：结合了动量的思想，利用过去梯度的信息来加速收敛并减少震荡。

偏差修正：在初始迭代中，一阶矩和二阶矩的估计值会偏向于零。Adam 包含偏差修正步骤来抵消这种早期偏差，使其在训练初期更稳定。

默认选择：由于其出色的性能和较少的超参数调整需求，Adam 经常被推荐为深度学习模型的默认优化器。

Adam 中权重衰减的实现方式（经典方式）：

在标准的 Adam 实现中，L2 正则化（即权重衰减）通常是通过向梯度中添加一个与权重成比例的项来实现的。其数学形式类似于：

$$g_t = \nabla L(\theta_t) + \lambda \theta_t$$

其中：

$g_t$  是调整后的梯度。

$\nabla L(\theta_t)$  是原始损失函数对参数  $\theta_t$  的梯度。

$\lambda$  是权重衰减系数。

这意味着，权重衰减的影响被耦合到了梯度更新中，并且会受到 Adam 自身自适应学习率的影响。

### AdamW 优化器

AdamW 是 Adam 优化器的一个变体，由 Ilya Loshchilov 和 Frank Hutter 在 2017 年的论文 “Decoupled Weight Decay Regularization” 中提出。它解决了 Adam 在处理权重衰减时的一个问题。

### AdamW 的核心改进：解耦的权重衰减

AdamW 的主要区别在于它解耦（decouples）了权重衰减和梯度更新。这意味着权重衰减不再通过修改梯度来实现，而是作为一个独立的步骤直接应用于权重。

AdamW 的更新规则大致如下：

计算自适应学习率更新：像 Adam 一样，根据一阶和二阶矩计算参数的自适应学习率更新。

应用权重衰减：在应用上述更新之后，独立地将权重衰减项从权重中减去：

$$\theta(t+1) = \theta_t - \text{learning\_rate} \cdot \text{Adam\_update\_term} - \text{weight\_decay} \cdot \theta_t$$

### 为什么解耦很重要？


Adam 的问题：在 Adam 中，由于权重衰减被添加到梯度中，它会与 Adam 的自适应学习率机制相互作用。对于那些具有大梯度移动平均值（大  $m$ ）或小梯度平方移动平均值（小  $v$ ）的参数，自适应学习率可能会显著降低权重衰减的效果，或者导致权重衰减失效。这使得权重衰减的正则化效果变得不一致且不可靠。

AdamW 的优势：通过将权重衰减从梯度更新中分离出来，AdamW 确保了权重衰减能够独立且一致地应用于每个权重，无论其自适应学习率如何。

这使得正则化效果更稳定，提高了模型的泛化能力，尤其是在大型模型和数据集上。

在训练 Transformer 模型、大型语言模型（LLMs）以及图像分类等任务中，

AdamW 常常表现出比标准 Adam 更好的性能。

核心区别总结 


特征 Adam 优化器 AdamW 优化器

权重衰减处理 耦合到梯度更新中 (L2 正则化项加到梯度上) 解耦的权重衰减 (直接从权重中减去)

正则化效果 在自适应学习率下可能不一致或减弱 更一致、更稳定、更有效  
泛化能力 良好, 但在某些情况下不如 AdamW 通常更好, 尤其在大型模型和 Transformer 中

应用场景 广泛适用, 作为默认优化器 大型模型、Transformer、SOTA 模型训练的首选

匯出到試算表

实践中的选择 

在现代深度学习实践中, AdamW 通常是带有权重衰减时的首选优化器, 因为它能提供更可靠和有效的正则化, 从而带来更好的模型泛化能力。如果你发现 Adam 模型的泛化能力不如预期, 或者在训练大型模型时遇到困难, 那么切换到 AdamW 往往是一个很好的尝试。

几乎所有主流的深度学习框架 (如 PyTorch 和 TensorFlow) 都提供了 AdamW 的实现。例如, 在 PyTorch 中, `torch.optim.Adam` 和 `torch.optim.AdamW` 是两个不同的类。