实验1 卷积+遗传算法

一、实验原理

① 遗传算法

遗传算法（Genetic Algorithm，GA）是一类受自然选择和生物进化机制启发的全局优化方法。其核心思想是：在候选解的集合中，通过“生存竞争”不断保留更优秀的个体，使解逐步逼近最优区域。与基于梯度的优化不同，遗传算法不依赖目标函数的可导性，也不需要显式的几何结构，因此特别适合处理搜索空间复杂、超参数维度较高、目标函数具有多峰或噪声干扰的问题。

典型的遗传算法流程包括三个关键步骤：
（1）**选择（selection）**：根据适应度（fitness）从当前种群中挑选表现较好的个体，使高质量解有更大概率传递到下一代；
（2）**交叉（crossover）**：将两个或多个父代个体的参数组合生成新的解，以此探索搜索空间中未被访问的区域；
（3）**变异（mutation）**：对个体的部分参数施加随机扰动，使算法保持多样性并避免陷入局部最优。

在多轮迭代后，种群逐渐向高适应度区域集中，从而实现对模型结构、超参数或训练策略的自动优化。

② 正则化

对于给定的有限数据集 $\{x_1, \ldots, x_n\}$ 以及对应的 $\{t_1, \ldots, t_n\}$，往往存在无限多种条件分布都能与这些数据兼容，仅靠数据本身无法唯一确定模型。因此，机器学习通常需要引入某种**先验结构**来缩小可行解空间，使模型在这些候选分布中选出更合理的一类。在许多实际任务里，我们希望输入的微小扰动不会造成输出的剧烈波动，也就是说，模型应当倾向于**平滑的函数族**。为了鼓励这种特性，我们通常会在优化过程中加入某种形式的正则化，把解“推向”变化更平滑、更具结构的区域。

权重衰减（weight decay）是机器学习中应用最广的一类正则化方法，其典型形式为

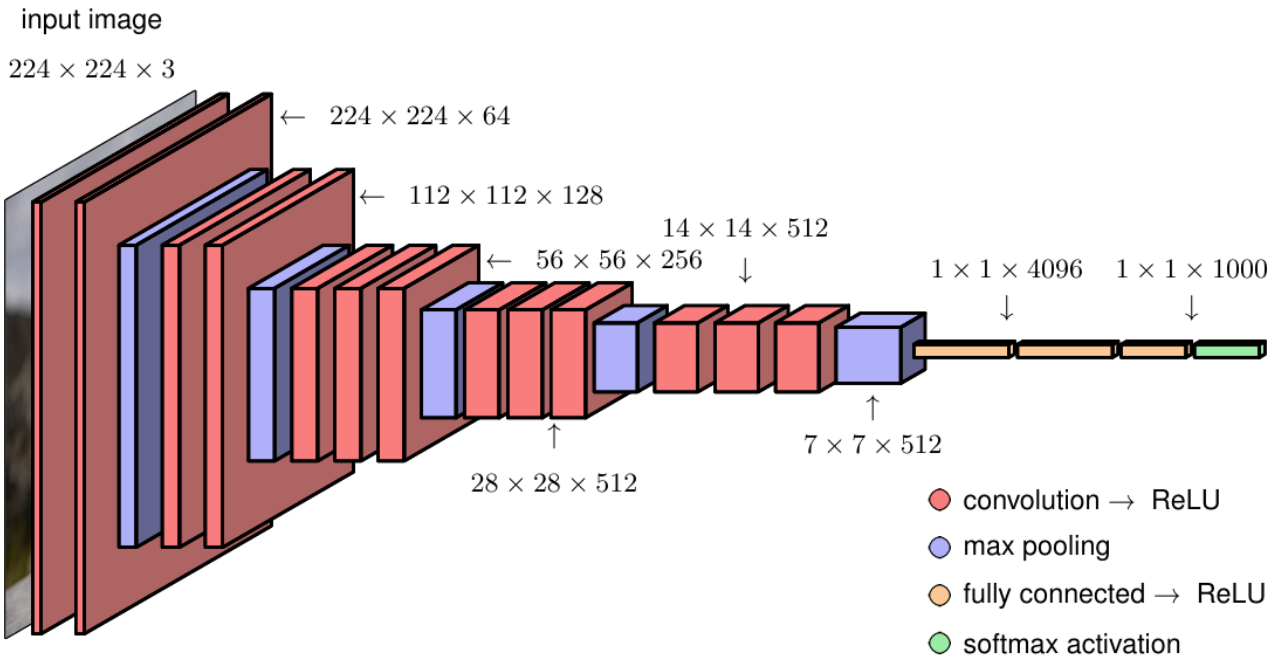
$$\tilde{E}(\mathbf{w}) = E(\mathbf{w}) + \frac{\lambda}{2}, \mathbf{w}^\top \mathbf{w}, .$$

通过对权重向量的每个分量施加二次惩罚，这种方法会倾向于把较大的参数“拉回”到接近零的位置，从而**限制模型的自由度，避免过度拟合。**

在机器学习中还有不少能够有效抑制过拟合的技巧。后来的一系列理论分析表明，它们**在某些条件下与正则化具有等价或相似的效果。**例如：早停法（early stopping）可以看作是一种隐式的 L^2 正则；批归一化（batch normalization，后续简称 BN）通过稳定激活分布减少参数敏感性；丢弃法（dropout）则等价于对模型结构进行随机扰动并施加惩罚。这些方法虽然表面上并不显式添加正则项，但在效果上也能起到“收缩模型复杂度”的作用。

参考补充证明，我们可以认为 dropout 等价于一种加入特殊惩罚项的正则化，而 BN 中也包含了隐式的正则。

③ 卷积神经网络中的正则化



常见的卷积神经网络通常由逐层减小空间尺寸、逐层增加通道数的**卷积层**堆叠而成，最后通过展平操作连接到高维的**全连接层**。卷积部分和全连接部分在表达能力、参数规模以及特征结构上存在显著差异，因此它们对正则化的需求也并不相同。

卷积层

卷积层具有显著的空间结构与参数共享特性，同一通道上的均值与方差可从 $B×H×W$ 个激活中估计，因此 BN 的统计量稳定，能够有效缓解深层卷积中的激活分布漂移，提升训练稳定性。卷积还天然具有平移不变性，池化操作进一步增强了对局部噪声的鲁棒性；在这种结构下，随机删除 feature map 上的几个点并不会破坏整体特征，因此 Dropout 引入的随机置零反而削弱了卷积层的空间一致性，不能提供有效的正则化。因此卷积部分更适合使用 BN，而不适合使用 Dropout。

全连接层

全连接层不具备空间维度，统计量只能来自 batch 维度，BN 的均值和方差估计较不稳定，甚至会干扰线性层对高维特征的全局组合。另一方面，全连接层更容易出现神经元之间的共适应，使模型泛化能力下降。Dropout 的随机屏蔽机制能有效打破这种耦合，同时不会破坏特征结构。因此在线性层中，Dropout 比 BN 更适合作为主要正则化方式。

在本次实验中，**我们将分别考察 BN 和 dropout 对模型训练与泛化的影响，并进一步借助遗传算法搜索不同超参数组合，通过实验验证它们在卷积神经网络重点合适使用位置。

二、实验方法

1. 网络基本架构

本实验固定使用一个结构简洁且可控的卷积神经网络作为搜索对象，其整体架构由三个部分构成：

（1）**卷积特征提取模块（CNNBlock）**
包含 3 个卷积层，每层通道数依次为 64、128、256。卷积核大小固定为 (3× 3)，步长为 1，padding 为 1。每个卷积层包含 ReLU 激活，并根据个体基因决定是否启用 Batch Normalization，以及是否在卷积输出处应用 Dropout。卷积层末尾使用固定的池化方式（max pooling），用于减小特征空间尺寸。

（2）**展平层（Flatten）**
将卷积输出的 ((C,H,W)) 张量展平成一维向量，为全连接部分提供输入。

（3）**全连接分类模块（LinearStack）**
由 2 层隐藏线性层（维度 128、64）构成，层间加入 ReLU 激活。根据基因决定是否在全连接层启用 BatchNorm，以及是否添加 Dropout。输出层为线性层，输出维度为 10，对应 CIFAR-10 分类任务。

2. 染色体/基因设计

为了让遗传算法能够同时搜索训练超参数与正则化机制，本实验设计了 7 维基因向量。一个个体 Ind 的结构为：

$$\mathrm{Ind} = [g_1, g_2, g_3, g_4, g_5, g_6, g_7]$$

基因意义如下：

基因编号	符号	含义	取值方式 / 范围
1	g_1	batch size	从 $\{16, 32, 64\}$ 中随机选择
2	g_2	学习率（log-uniform）	在区间 $[10^{-5}, 10^{-2}]$ 以对数均匀分布抽样
3	g_3	dropout rate	在 $[0, 0.7]$ 内均匀抽样，用于 CNN 与全连接层
4	g_4	CNN 是否启用 BatchNorm	0 或 1
5	g_5	CNN 是否启用 Dropout	0 或 1
6	g_6	全连接层是否启用 BatchNorm	0 或 1
7	g_7	全连接层是否启用 Dropout	0 或 1

3. 遗传算法超参数选择

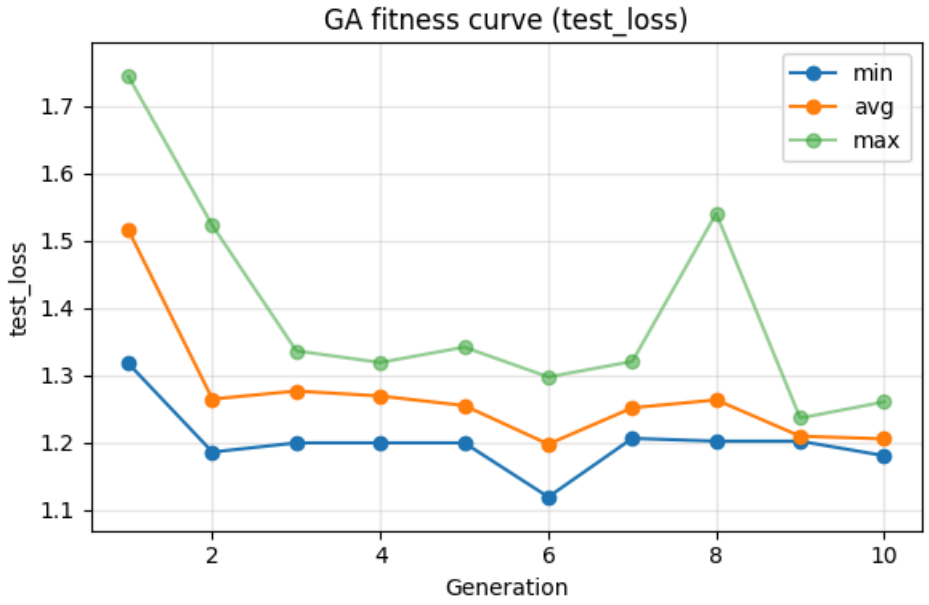
实验采用 DEAP 库实现标准遗传算法流程，主要超参数如下：

- **种群规模**：12
- **进化代数**：10
- **选择策略**：锦标赛选择，锦标赛规模为 2
- **交叉算子**：双点交叉（cxTwoPoint），交叉概率 0.8
- **变异算子**：按基因位点突变，基因突变概率 `indpb=0.2`，个体整体突变概率 0.5
- **去重机制**：对出现完全重复的个体强制改变某个基因，以保持多样性
- **适应度函数**：训练 15 epoch 后的 **测试集 loss**（越低越好）

为了降低评估时间，训练集与测试集分别从 CIFAR-10 中抽取固定数量的样本：

- 训练集：2000 张随机子集
- 测试集：500 张随机子集
- 抽样使用确定性随机种子，使同一基因在不同评估中保持一致性

三、实验结果与分析



本实验通过遗传算法（GA）对卷积神经网络的训练超参数与正则化手段进行联合搜索。上图给出了 10 代进化过程中 min/avg/max 三条 test_loss 曲线。据此，我们对 GA 在当前实验条件下的表现与最终模型的结构特点进行了系统分析。

1. 遗传算法整体性能：快速粗筛，但收敛有限

（1）**初始两代出现大幅下降，说明 GA 成功过滤了初始随机种群中的差解。**

- 从 Gen1 到 Gen2：
  - min 从 1.3169 → 1.1854
  - avg 从 1.5151 → 1.2642
- 这是典型的 GA “粗筛阶段”：极端不良组合（例如过大学习率、关掉 BN 且使用不合适的 dropout）迅速被淘汰。

（2）**Gen2–Gen5 进入停滞区，min 基本维持在 1.19 附近不再显著下降。**

- avg 在 1.25–1.27 区间小幅震荡。
- 说明 GA 已进入一个局部区域，当前交叉与变异未能进一步找到显著更优的超参数组合。

（3）**Gen6 出现一次“跳跃式改进”，min 降到 1.1187，但随后 GA 未能围绕更优区域继续优化。**

- 这是一次典型的 “lucky mutation”：随机突变恰好采到了一个结构更优的超参组合。
- 但接下来几代没有持续改善，说明 GA **没有有效捕捉并固化这一良性方向**，且周围超参空间的形状复杂，使得 GA 难以继续开发此区域。

（4）**Gen7–Gen10 min、avg 整体稳定，但没有出现进一步下降，收敛并不明显。**

- min 在 1.18–1.21 之间波动。
- avg 维持在 1.20–1.26 之间。
- 说明 GA 最终只能“保持一片中等质量的区域”，但无法更深地探索复杂超参组合。

2. 最终最优模型的基因组合明显偏向 BatchNorm-only

最终最优个体（HOF）为：

$$[16,\ 7.78 \times 10^{-5},\ 0.6093,\ 1,\ 0,\ 1,\ 0]$$

对应的结构如下：

- batch size：16（较小）
- lr ≈ 7.78×10^{-5}（非常小）
- dropout_rate = 0.609（但卷积与线性层的 dropout 均关闭）
- 卷积：BN=1，Dropout=0
- 全连接：BN=1，Dropout=0

这意味着 GA 倾向于选择 **“双 BN、不用 Dropout + 小学习率 + 小 batch size”** 的组合。

3. BN-only 模式在此设置下的优势：训练稳定、短期 test_loss 更好

（1）**BN 平滑了激活分布，使训练路径更稳定。**

- BN 的噪声属于“结构化噪声”，往往在 early-stage 带来更好的梯度方向。
- 在 15 epoch 的短窗口中，BN Networks 通常收敛得更快更稳，因此 test_loss 更容易保持较低且波动小。

（2）**较小的学习率与 BN 结合，使网络训练得“温和”，避免短期极端过拟合。**

- lr 接近 10^{-5} 级别，使训练阶段非常缓慢。
- 在 2000 张训练集、500 张测试集的规模下，**小 lr 可以人为地“压制过拟合速度”**，使得 15 epoch 内 test_loss 看起来非常好。
- GA 使用 test_loss 作为适应度 → 这样的“短期表现好看”组合自然更容易被选中。

（3）**BN-only 模型 train vs test 成绩差距明显，验证了实际过拟合问题。**

- 虽然 test_loss 在 15 epoch 内表现不错，但训练集上更容易过拟合。
- 这说明 BN 在当前规模下更像是“训练稳定器”而非真正的“泛化正则项”。

4. Dropout 在本实验中被 GA 系统性“低估”的原因

（1）dropout 的效果依赖隐式耦合 —— rate、位置、lr、batch size 必须同时匹配

- 开启 dropout 后，收敛速度显著变慢。
- 若 dropout_rate 不合适（过大或过小），test_loss 在 early stage 会变差。
- 在 15 epoch 的评估窗口中，**Dropout 的早期惩罚效应>晚期泛化收益**，导致 GA 更倾向于关闭 Dropout。

（2）dropout 会显著放大 test_loss 的方差，使 GA 难以判断优劣

- 每次 forward 都有随机掩蔽 → 测试阶段也依赖 moving statistics。
- 在当前数据规模和 epoch 数下，GA 收到的适应度信号噪声非常大。
- 噪声大的基因组合（dropout=1）更容易被 GA 误判为“不稳定”和“不好”，进而被淘汰。

（3）搜索空间中 dropout 的维度多（率 + 开关） → 需要更多代数才能找到合适组合

- dropout_rate 是连续空间
- 同时还要匹配：
  - CNN dropout 开关

    - linear dropout 开关
    - lr、batch size
- 这是一个强耦合（epistasis）结构，高维度且非线性。
- 在仅 12 个种群 × 10 代的预算下，GA 难以靠随机突变找到“刚好合适的组合”。

综上，**Dropout 在这种有限训练 + noisy fitness 的情况下被 GA 系统性低估，而 BN-only 被高估。**

5. 从 fitness 曲线看出的“适应度噪声”问题

（1）max 曲线下降幅度比 min 小，说明差解减少但最优区域未拓宽。
（2）Gen 8 出现异常 spike（max 突增），表明 fitness 噪声较强。
（3）min 的变化幅度小，说明 GA 局部探索被噪声干扰而变得迟缓。

这些都说明：**在子集数据 + 短训练窗口 + 随机梯度噪声下，test_loss 的适应度信号噪声较大，限制了 GA 的搜索效率。**

四、实验总结

综合来看，本实验中遗传算法的表现主要受到搜索能力和适应度评估方式的限制。由于种群规模小、代数少，而搜索空间又包含连续与离散混合的 7 个维度，GA 只能进行有限的浅层探索，很容易在局部区域停滞。此外，适应度信号噪声较大：mini-batch 随机性、Adam 的自适应动量以及 dropout 带来的梯度扰动都会让 15 个 epoch 的短期 test_loss 难以稳定反映真实优劣，使 GA 偏向选择训练更平稳的 BN-only 方案。

更关键的是，适应度基于短期表现，使 GA 倾向于挑选“早期下降快、短期看起来最优”的组合，而非真正泛化能力最好的模型。BN + 小学习率恰好在 early stage test_loss 上更占优势，因此被持续强化；而 dropout 虽然对长期泛化有益，但在短期训练中往往收敛更慢、噪声更大，于是在当前设置下被系统性低估。

因此，最终得到的 BN-only 最优个体并不代表其长期泛化性能最好，而是遗传算法在“有限预算 + 高噪声 + 短期评估”条件下的结构性偏置。

> **Thoughts**
>
> 深度学习的“炼丹”在理解了背后的数学原理后，其实并不玄学，也不是我一直认为的“直觉”。但各种 trick 都不是彼此独立的，它们必须结合网络结构、模型实际表现和具体任务场景一同考虑。包括优化算法在内的所有选择，都需要根据当前资源和目标判断是否合适，以及我们最终希望找到怎样的“解”。