arXiv:2611.30081v1 [cs.LG] 23 Nov 2026

增长路径是神经网络缺失的另一半

Tingan Jin*[1] 和 Haosong Li*[2]

[1]加州大学洛杉矶分校[2]独立研究者

摘要

人工神经元能够计算和学习，却无法存活：网络中的单元在设计时便已固定，无论是否被使用，每个单元都会被保留并为其支付计算量。我们为每个神经元增加第二条路径。除了其任务路径（即通常的计算和梯度学习）之外，双通路神经元 还具有一条增长路径：营养物质从输出单元沿网络反向流动，每一层都获得相同的固定预算，而每个单元获得的供给量则与其上方单元对它的使用量成正比。获得充足供给的单元会分裂，而缺乏供给的单元会死亡。这种分配并非独立的分析或剪枝步骤：它在网络训练期间持续运行，作为一条只读取激活值和权重的常规规则。任务仍然通过标签和普通损失进行学习，训练过程无需增加阶段、调度计划或目标规模。由于预算守恒，竞争是零和的，只需一个数字便可设定网络规模。在预训练网络部署到更窄领域时启用增长路径，网络会在持续学习的同时缩小规模。使用 Adam 时，与以普通方式训练的同一网络相比，该网络在该领域的准确率提高了 1.0 至 1.2 个百分点，而计算量仅为其 24% 至 60%；即使仅使用 12% 的计算量，表现仍然领先。随机选择相同数量的单元则不会带来任何提升。让饥饿的预训练单元休眠而非死亡，能在每种设置下保留更多稀有内容，提升幅度为 1.2 至 4.6 个百分点。在由不断变化的任务组成的长数据流中，带有增长路径的网络在 800 个任务上保持其准确率，而普通网络则持续下降。使用普通随机梯度下降时，提升幅度较小：在部署阶段，该方法需要在计算量和稀有内容之间进行权衡；在两条持续学习流中的一条上，它的表现仅与同等规模的网络持平。我们并列报告两种优化器的结果。实验规模较小（三层感知器，使用类似 MNIST 的数据）；实验表明，一条守恒且无损失的第二路径能够自行决定网络需要哪些单元。

1 引言

人工神经元只有一项任务：计算激活值，并调整自身的权重以降低损失。它是否应该存在，并不由它决定：单元数量由设计者在训练开始前选定，此后网络的整个生命周期中，每个单元都会被保留，无论是否使用。一个已经不再发挥作用的单元仍会占用内存，而且每一步仍需进行一次乘法运算。在我们下文研究的一个网络中，三个隐藏层中有两个隐藏层的九成以上单元处于非激活状态，但每一步都会计算所有这些单元。

生物神经元还有第二项工作：存活。在发育过程中，产生的神经元远多于最终保留下来的数量，它们会争夺靶细胞以有限数量供给的营养因子；获得的营养因子太少的神经元就会死亡 [27, 34, 38]。学习与存活在同一个细胞中进行，

*贡献相同。

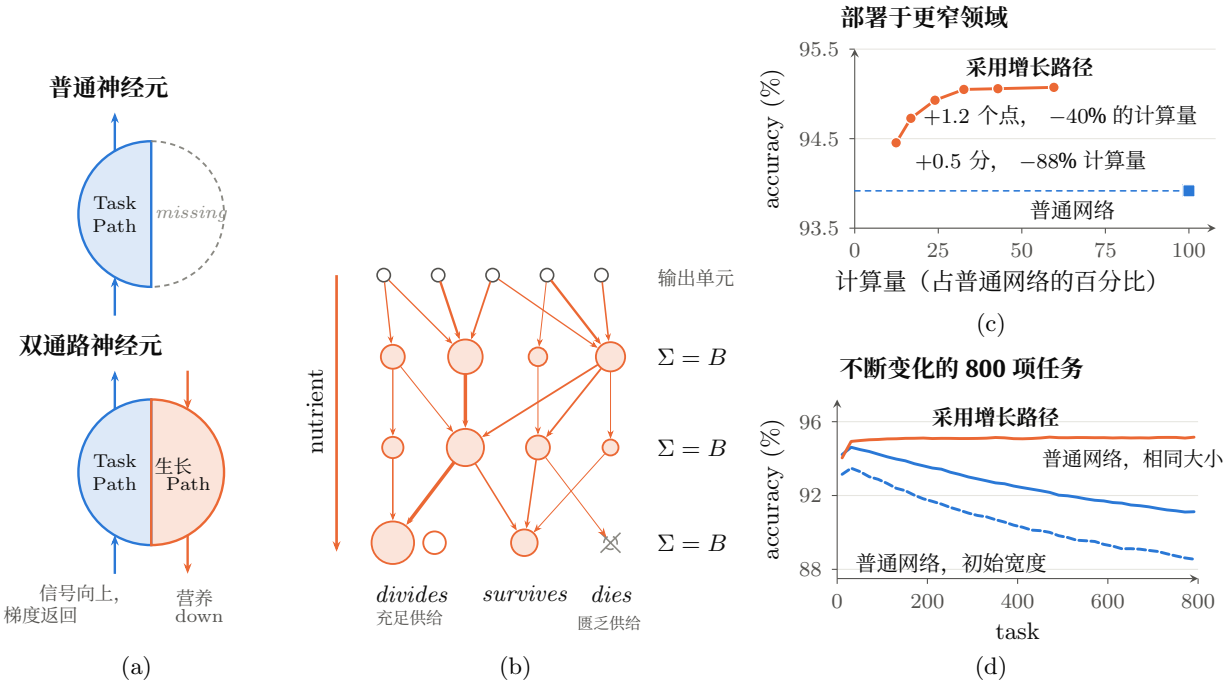

图 1：增长路径是每个神经元中的第二条路径。(a) 普通单元只有任务路径：信号向上传递，梯度则传回来。双通路神经元还会接收来自其上方单元的营养。(b) 营养从输出单元进入并向下传递；每一层接收相同的预算 B，并根据各单元被上一层使用的比例，在这些单元之间进行分配。营养充足的单元会分裂，营养匮乏的单元会死亡，其余单元则存活。网络训练期间，每 100 次更新就会重新进行一次分配，而且不会读取损失。(c) 在预训练网络部署到更窄领域时启用增长路径，它会在网络持续学习的同时缩小网络。每个点代表一个预算（$B = 25$ 至 100）。当计算量为 12% 时，该网络在此领域上的准确率仍高于满尺寸的普通网络。(d) 在不断重连输入的 800 项任务上从头开始训练时，带有增长路径的网络保持了其准确率，而同等尺寸及初始宽度的普通网络准确率则有所下降。(c) 和 (d) 面板使用 Adam（五个随机种子的均值）；使用随机梯度下降时的结果在部署时表现较差，详见第4 节和5节。

但依赖不同的信号。人工神经元网络具备这种安排的前半部分，却缺少后半部分。确实存在改变网络结构的方法（增长、剪枝、单元重置），但这些方法从外部作用于网络：它们是独立的流程，由损失、梯度或效用分数驱动，并采用由设计者选定的调度计划、配额或目标规模（第 7节）。

缺失的另一半。 我们提出将第二项工作交给神经元本身。一个 双通路神经元 （图 1a）有两条通路。它的任务路径 负责普通单元所做的一切。它的 增长路径 则是新增的。一种 营养 从输出单元进入网络，并沿层向后流动。每一层获得相同数量的营养，即 预算 B，每个单元则按照自身对下方单元的使用量，将收到的营养按比例传递给它们（图1b）。得到充足供给的单元会分裂：它会把部分权重和营养分给一个子单元。挨饿的单元会死亡。其他单元则会存活。这种分配并非只在网络训练完成后进行一次分析。只要网络持续训练，每经过 100 次更新就会重复进行一次，单元也会据此存活下来。它与通常采用的增长和剪枝方式有三点不同。

- **它在内部运行　常规训练。** 这条规则读取激活值和权重；它看不到标签　　，

没有损失值，也没有梯度。任务仍通过标签和常见的交叉熵损失来学习，因此权重由一种信号决定，而结构则由另一种信号决定。网络训练时，无需额外关注任何事项：没有剪枝阶段、没有重新训练，也没有调度计划（算法 1）。

- **这是守恒的**。每次检查时，每一层获得相同的预算，因此层内竞争是零和的：一个单元获得的，恰好是其他单元失去的。没有替换配额，也没有目标规模。若不满足守恒，同一规则就无法控制规模（第5节）。

- **一个数决定规模**。预算是决定网络规模的唯一调节项，也由此决定计算量。它只设置一次。

营养传递的方式有所借鉴：这种形式的分配能够解释训练好的神经网络的预测 [2]，并曾被用于剪枝 [51],，而单元此前也曾以多种方式生长、剪枝和替换。据我们所知，新的做法是将这种跨层守恒分配转化为贯穿整个训练过程的生长规则，同时决定分裂与死亡（第 7节）。

它带来的好处。最清晰的检验是部署（第4节）。一个网络先在五种内容上预训练，然后部署到一个领域：该领域保留其中两种，很少呈现第三种，并舍弃其余内容。我们在预训练网络中启用增长路径，预算设为低于其宽度，其他条件一概不变。网络在大约 200 个任务内缩小，并继续学习（图 1c）。使用 Adam 时，它在该领域上的准确率比以常规方式训练的同一网络高出 1.0 至 1.2 个百分点，而计算量仅为后者的 24% 至 60%；即使计算量仅为 12%，准确率仍领先 0.5 个百分点。单元的选择决定了这一结果：若对随机选取的单元施加相同数量的死亡与分裂，收益便会消失。死亡的单元中，有相当一部分曾负责处理该领域不再呈现的内容，尽管规则从未看到标签。将饥饿的预训练单元设为休眠而非移除后，在所有设置下都能保留更多较少使用的内容（使用 Adam 时提高 3.8 至 4.6 个百分点，使用随机梯度下降时提高 1.2 至 2.2 个百分点）；在需要它们的任务中，休眠单元的唤醒频率也提高了 10 至 20 倍。

同一规则也能帮助网络持续学习（图 1d；第 5 节）。在旧知识迅速过时的 800 个任务的数据流中，采用增长路径的网络从第一个任务到最后一个任务，表现始终维持在 94% 至 95%；而规模相同的普通网络则从 94% 降至 91%。在旧知识仍然有用的数据流中，使用 Adam 时，它比规模相同的普通网络高出 0.8 至 1.6 个百分点。

本文未展示的内容。使用普通随机梯度下降（SGD）时，我们的数据流中的普通网络性能下降很少，情况也随之不同。在部署时，增长路径使用三分之一的计算量，便能在常见内容上达到普通网络的水平，但会丢失稀有内容；而在第二个持续学习流中，它的表现也仅与规模相同的普通网络持平。所有实验均使用 MNIST 类数据上的三层感知器，每项比较采用五个随机种子。我们在每张表中并列报告两种优化器的结果，并在第8节汇总局限。标题阐明了本文的立场，而这些实验是对这一立场的初步检验：它们表明，一条不受损失影响、得以保留的第二路径能够决定网络需要哪些单元；而普通训练浪费单元之处，反而有利于网络。

贡献。（i）双通路神经元及其增长路径：在各层之间进行守恒分配，并将其转化为贯穿整个训练过程的增长规则，逐个单元决定分裂、存活和死亡，同时像以往一样学习任务（第2节）。（ii）部署结果：一个预训练网络在学习某个领域的同时缩减至给定预算，并提供完整的成本—准确率曲线

算法 1 训练双通路神经元网络。任务路径的步骤与常规训练相同。增长路径额外执行标记为 ▶ 的步骤；这些步骤读取激活值和权重，但绝不读取 y 或损失。

对数据流的每个小批量（x, y）**执行**
  前向传播；损失 = 为输出单元与 y之间的交叉熵 ▷任务路径
  反向传播；更新存活单元之间的权重 ▷任务路径
  ▶根据每个存活单元的激活值 $\bar{a}_i$ 更新其活动值 ▷ 增长路径
  如果更新次数是 K 的倍数**则** ▷结构检查
    ▶ 将预算 B 从输出单元向下分配，（1）；更新层级 $\bar{n}_i$
    ▶ 已建立单元以 $\bar{n}_i < \theta_{\text{die}}$ 死亡；获得最多预算的单元以 $\bar{n}_i > \theta_{\text{div}}$ 分裂，（3）–（4）结束条件判断
结束循环

涵盖两种优化器，以及将休眠作为保留稀少使用知识的一种方式（第4节）。（iii）有关机制的证据：随机周转、无守恒和同规模对照组，并直接考察哪些单元会死亡（第4–6节）。（iv）一套实验方案：按规模调整基线，在运行检查种子之前固定规则，并将不利结果与有利结果并列报告（第3节）。

2 双通路神经元

双通路神经元是在普通隐藏单元上添加了第二条通路。 任务路径 是普通单元原本就有的部分：其输入权重和偏置、激活值，以及由任务损失驱动的梯度更新。 增长路径 是每个单元保有的一种小型状态，以及一条根据该状态决定单元是否 分裂、存活或死亡的规则。它只读取激活值和权重，从不读取损失、标签或梯度。两条通路在一个地方交汇：当单元分裂或死亡时，任务路径训练的权重集合会发生变化。算法 1展示了增长路径如何进入训练循环。损失、标签、数据和训练顺序都与普通网络相同；增长路径为每个单元添加一个运行平均值，并且每经过 K 次更新，就从输出单元向下执行一轮遍历。生长或剪枝没有单独的阶段，也无需在任何步骤停止、检查或重新启动训练。

我们描述本文所用网络的方法：由 $L = 3$个包含 ReLU 单元的隐藏层和 $M = 10$ 个输出单元组成的多层感知机。隐藏层 ℓ 包含一组由 N_ℓ 个存活 单元组成的集合 $\mathcal{A}_\ell$；$a_i(x) \geq 0$ 是隐藏单元 i的激活值，而 W_{ij} 是从单元 i 到下一层单元 j 的权重。

2.1 任务路径

任务路径采用普通学习：损失为交叉熵，存活单元之间的权重通过小批量随机梯度下降或 Adam 进行更新 [23]。不存活的单元不参与前向传播或反向传播，也不会获得更新。在两次结构事件之间，训练与由相同存活单元组成的普通网络相比，仅有一处不同：最近诞生的单元使用更大的步长（第 2.2节）。

2.2 增长路径：从输出路由而来的守恒营养 s

状态与结构检查。 每个隐藏单元 i 都保留三个数值：其活动值 $\bar{a}_i$，即其批次平均激活值的指数移动平均（时间常数为 1,000 次更新）；其营养水平 $\bar{n}_i$，定义如下；以及其年龄，即自其诞生以来的更新次数。每经过 $K = 100$ 次更新，增长路径都会执行一次结构检查：传递营养、更新层级，并执行死亡与分裂。

路由与守恒。每一层在每次检查时获得相同数量的营养，即预算 B。每个输出单元发出 B/M的供给。单元 j 会将其全部供给σ_j 传递给下方层的存活单元，分配比例取决于其使用量，即 $u_{ij} = \bar{a}_i\,|W_{ij}|$，也就是 i 对 j输入的平均绝对贡献。层 ℓ 中单元 i 的摄取量是

$$n_i = \sum_j \sigma_j \frac{u_{ij}}{\sum_{k\in\mathcal{A}_\ell} u_{kj}}, \tag{1}$$

其中，j 遍历层 $\ell + 1$的存活单元，或者在 $\ell = L$时遍历输出单元。隐藏单元的供给就是其在本次检查中的自身摄取量，即 $\sigma_i = n_i$，因此路由只需从输出单元向下传递至第一隐藏层。下游单元分配出的份额总和为一，因此每个隐藏层的摄取量总和等于预算：[1]

$$\sum_{i\in\mathcal{A}_\ell} n_i = B \qquad \text{for } \ell = 1, \ldots, L. \tag{2}$$

这种重新分配的形式符合逐层相关性传播的守恒规则 [2] 以及激励反向传播 [54]；不同之处在于它的起点（每个输出单元获得相等的供给，而不是某个类别的得分）和用途（第7节）。由此产生两个性质。层内的竞争是零和的：一个单元只有在同层其他单元的摄取量减少时，才能获得更多摄取量。营养是自上而下传递的：一个单元之所以获得供给，是因为使用它的单元本身也获得了供给；因此，如果其消费者处于营养匮乏状态，那么无论该单元多么活跃，它获得的营养都很少。

营养水平。 该水平是摄取量随时间平滑变化的历史记录：每次检查 $\bar{n}_i \leftarrow \bar{n}_i + \beta\,(n_i - \bar{n}_i)$ 时，时间常数为 τ 次更新，且 $\beta = 1 - e^{-K/\tau}$。每次检查时路由的营养量才是守恒的；这些水平则是对其持续累积的记忆。

分裂、存活或死亡。 单元的年龄达到保护期 P后，即成为成熟单元 。每次检查时，成熟单元若满足 $\bar{n}_i < \theta_{\text{die}}$ 就会死亡 ；若满足 $\bar{n}_i > \theta_{\text{div}}$，则可能分裂 ；否则存活 。处于保护期内的单元会存活。首先执行死亡。随后，每层最多有 $\max(1, \lceil \rho N_\ell \rceil)$ 个单元分裂，优先选择层级最高的单元。死亡数量没有上限，也没有最小规模或替换配额：一层的规模由竞争结果决定。

死亡。 死亡的单元会被移除：其权重、偏置和优化器状态均设为零，所占位置也会被释放。它仍持有的层级不会传递下去，而是随之消失，因此死亡事件发生后，该层的总层级会下降，并在之后的检查中逐渐恢复。

[1]只要每个正供给的下游单元至少使用一个下层的存活单元，这一规则就成立。若不满足，该单元的供给在本次检查时不会被路由，实现会对这一事件进行计数。

分裂。分裂单元 p（亲代）会保留下来，并在同一层的空闲位置创建一个子单元 c。子单元会获得亲代的输入权重和偏置 w_p

$$w_c = \lambda \left(w_p + \varepsilon \, s_p \, \xi \right) + (1 - \lambda) \, w_0, \tag{3}$$

其中，$\lambda \in [0, 1]$ 是继承比例，ξ 是标准高斯噪声，s_p 是 w_p的均方根元素，ε 是相对噪声尺度，而 w_0 是从用于初始化网络的分布中重新采样得到的值。亲代通过其输出权重和层级为子单元支付代价：

$$W_{cj} = \tfrac{\lambda}{2} \, W_{pj}, \quad W_{pj} \leftarrow \left(1 - \tfrac{\lambda}{2}\right) W_{pj}, \qquad \bar{n}_c = \tfrac{\lambda}{2} \, \bar{n}_p, \quad \bar{n}_p \leftarrow \left(1 - \tfrac{\lambda}{2}\right) \bar{n}_p. \tag{4}$$

因此，分裂会将层级从亲代转移给子单元，而层的总层级保持不变。在 $\lambda = 1$ 时，子单元是一个带噪声的副本，与亲代共享输出权重，如同 Net2Net 的加宽步骤 [5]。在 $\lambda = 0$ 时，它是一个全新初始化的单元，输出权重和层级均为零；在受到保护期间，它必须通过努力赚取自己的摄取量。两种情况下，分裂时网络的功能都保持不变（在 $\lambda = 1$ 时仅有噪声造成的差异）；而在两者之间，网络功能会略有变化。

保护、容量与设置。新生单元在 P 次更新期间既不能死亡，也不能分裂；在此期间，其输入权重和偏置的步长会乘以 $\kappa \geq 1$，以便它在接受评判前找到用途。存储容量上限为每层 C 个单元；如果某一层达到 C、失去所有单元或产生非有限值，则该次运行无效，不予采用。所有部署实验均使用 $\theta_{\text{die}} = 0.05$、$\theta_{\text{div}} = 1.5$、$\tau = P = 3$、750 次更新、$\kappa = 3$、$\varepsilon = 0.1$、$\rho = 0.01$、$\lambda = 0.5$ 和 $C = 768$；这些常数是在另一数据流的调优随机种子上选定的（第3节）。附录A介绍其余细节。

2.3 预算决定规模

阈值是固定数值，根据（2），层中的平均摄入量为 B/N_ℓ。在拥挤层中，份额较小，更多单元会低于 θ_{die}；在稀疏层中，更多单元会高于 θ_{div}。因此，在阈值固定的情况下，预算是决定层规模的唯一因素。从头开始训练网络时，我们将 B 设为其初始宽度 N_0，这样平均摄入量初始为 1，阈值也就表示为均等份额的倍数。在宽度为 N_0 的预训练网络中启用增长路径时，我们选择 $B < N_0$：每个预训练单元都以成熟单元的身份开始，层级为 B/N_0，网络则必须适应小于其宽度的预算。它能否做到这一点，以及准确率会因此付出怎样的代价，将在第4节讨论。

守恒也限制了规模。层的总层级永远不会超过 B：平滑步骤会使其趋向于 B，分裂不会改变它，而死亡会使其降低。每次检查后，每个已建立单元至少持有 θ_{die}，因此无论数据流有多长，一个层最多有 B/θ_{die} 个已建立单元。这个上限较宽松，无法预测实际规模；但若没有守恒，上限便不复存在（第3节中的对照）。

2.4 两种变体：两个亲代与休眠

双亲分裂。由单亲产生的子单元在 $\lambda > 0$ 开始时几乎是一个副本。在双亲变体中，第二个亲代从该层的其他成熟单元中抽取，概率与它们的层级成正比；子单元的输入权重结合两个亲代的权重，方式可以是随机地从其中一个亲代复制每个权重，也可以取平均值。每个亲代贡献其四分之一的输出权重和层级；而单亲情况下则各贡献一半，因此该层的总层级再次保持不变。

休眠而非死亡。死亡是不可逆的：一个仅偶尔被需要的单元可能在两次使用之间陷入饥饿状态，随后它所学到的内容也会丢失。因此，对于预训练网络，我们还研究了一种变体：原本会死亡的饥饿预训练单元会变为休眠。它不参与前向传播或反向传播；其权重、优化器状态和 Growth Path 状态均原样保存。在每次检查时，都会询问每个休眠单元 d：如果仅将它重新加入网络，它会获得什么结果；其中，$\hat{}a_d$ 表示它在最近一个训练批次中的平均激活值，

$$\hat{n}_d = \sum_j \sigma_j \frac{\hat{a}_d \, |W_{dj}|}{\sum_{k \in \mathcal{A}_\ell} u_{kj} + \hat{a}_d \, |W_{dj}|}. \tag{5}$$

如果 $\hat{}n_d \geq B/N_\ell$（即其所在层存活单元的平均摄入量）达到条件，该单元就会醒来，并且在一个保护期内不能再次进入休眠。此测试不增加任何参数。启动后诞生的单元仍会像之前一样死亡；若要将休眠机制扩展到这些单元，就需要制定一条规则来删除永不返回的单元，这一点我们暂不讨论。

2.5 成本计数

我们用参与计算的参数来统计成本。网络的计算成本是其存活单元之间的权重和偏置数量；对于休眠单元，还包括唤醒测试，并在每次检查时计数一次。存储成本则将存活单元和休眠单元一并计算。我们以普通网络为基准报告成本：部署时，以普通的 784–300–300–300–10 网络的百分比表示；对于从头开始训练的数据流，则以 784–100–100–100–10 网络（99,710 个参数）为单位表示。该计数不包括增长路径本身的记账，也不是实际运行时间测量（第 6 节）。

3 实验方案

以下比较、指标和决策规则均在将该方法应用于用于得出结论的随机种子之前确定。附录B 提供了完整细节。

基线：一个普通网络。本文始终以一个普通网络作为基线：采用相同结构的感知机，在相同的数据流上使用相同的优化器进行反向传播训练，并针对每种使用规模分别调优步长。只有相对于该网络取得优势，结果才算支持该方法。当网络从头开始训练时，方法会自行选择规模，因此普通网络会以两种匹配规模运行，即该方法的最终规模和峰值规模，而该方法必须同时胜过这两种规模。部署时，基线是同一个预训练网络，在其固定规模上进行训练。

三种对照。随机更替会重放一次运行，在每次检查和每一层中保持死亡与分裂数量不变，但从符合条件的单元中随机抽取；这将哪些单元被替换与替换数量及替换时机区分开来。无守恒按每个单元自身的使用量成比例地馈送，不采用式（1）中的归一化，因此层的总量可以自由增减。一个从头训练的同等规模普通网络用于检验规模本身是否足以解释结果；在部署时，它仅作为参考，因为它需要预先知道目标规模。

随机种子和规则预先固定。 每项选择（结构设置、步长）都依据随机种子 0–4 上的调优结果，按照固定规则作出，随后即予以冻结。每项比较都在五个检查种子上运行一次，这些随机种子从未用于任何调优运行。数据流长度、主要指标和判定阈值均已预先写明

，之后才运行检查种子；未预先固定的读数会在出现处标明。在部署时，我们没有对该方法进行任何调优：其设置是在混合流上选定的，步长则采用预训练网络的步长。我们尚未在新种子上进行进一步确认；每项结果都基于其比较中的五个检查种子。我们使用不带动量的随机梯度下降和 Adam [23] 及其默认设置，批大小为 16，并且只比较使用相同优化器的运行。差值按随机种子配对，并以均值和 95% 置信区间（Student 的 t，4 个自由度）报告；只有当区间不包含零时，我们才将差值称为正或负。

三种任务流。数据流是由一系列前后相接的十分类任务组成的序列，任务边界处没有任何提示；所有网络都有 784 个输入和三层隐藏层。在像素置换数据流中，每个任务都是 MNIST [26]，但像素采用新的随机置换 [18]（普通版本有 200 个任务，长版本有 800 个）。得分是在线准确率，主要指标是最后十分之一任务的在线准确率均值。每个任务都会重新连接输入，因此，该数据流用于测试网络能否持续学习，却不会因保留已学内容而给予奖励。混合流包含 2,000 个任务，每个任务有 24,000 张图像；每个任务都会重新将十个类别分配给输出单元。其主内容会反复出现：手写数字、字母 [7] 和 Kuzushiji 字符 [6]。两种无关类型各出现一次，持续 200 个任务：服装 [50] 以及第二个传感器，即采用固定像素置换的主图像。一个任务的得分是其区块得分，即该任务四个时点的留出准确率均值；主要指标是最后十分之一主任务的平均区块得分。部署数据流（图2，上）包含相同格式的任务。在其通用阶段（任务 1–500）中，五种类型出现的概率相同。在其部署阶段（任务 501–2,000）中，数字和字母是常见内容，Kuzushiji 是稀有内容，每 20 个任务组成的区块中恰好出现一个任务；服装和第二个传感器则不再出现。一个具有三层、每层 300 个单元的网络，在通用阶段作为普通网络进行训练；在任务 500 结束时，在该网络中启用增长路径，预算设为 25 至 100 之间。

指标与判定规则。在从头开始训练的数据流上，如果该方法的主要指标高于两个规模匹配的普通网络，且两个区间均高于零，则判定为通过。对于部署，常见内容准确率是最后十分之一常见任务的平均区块分数，而后期稀有内容准确率是第 1,501 至 2,000 个任务中的 25 个 Kuzushiji 任务的平均区块分数；计算量与存储量按常见内容窗口取平均。以下三条规则用于判定：(a) 常见内容表现不差于普通网络，与普通网络相比的差值区间下限高于 -0.5 个百分点；(b) 更小，计算量区间上限不超过普通网络的 75%；(c) 保留稀有内容，后期稀有内容差值区间下限高于 -1.0 个百分点。规则 (a) 和 (b) 在该方法于此数据流上运行前就已确定。规则 (c) 则是在得到死亡的首批结果后、运行休眠之前加入的；休眠需依据全部三条规则进行判定。

4 部署：一个在学习过程中不断缩小的网络 s

实际使用时，通常不需要完整的通用模型。一个在五种内容上训练的网络被部署到一个领域中：其中两种内容仍会出现，第三种只偶尔出现，另外两种则不再出现（图 2，上方）。普通网络会让所有单元都经历这一变化。我们在部署时刻，于同一个预训练网络中启用增长路径，并为每一层设定低于其宽度的预算。其他一切都不作改动，也不会在这个数据流上进行调优（第 3节）。

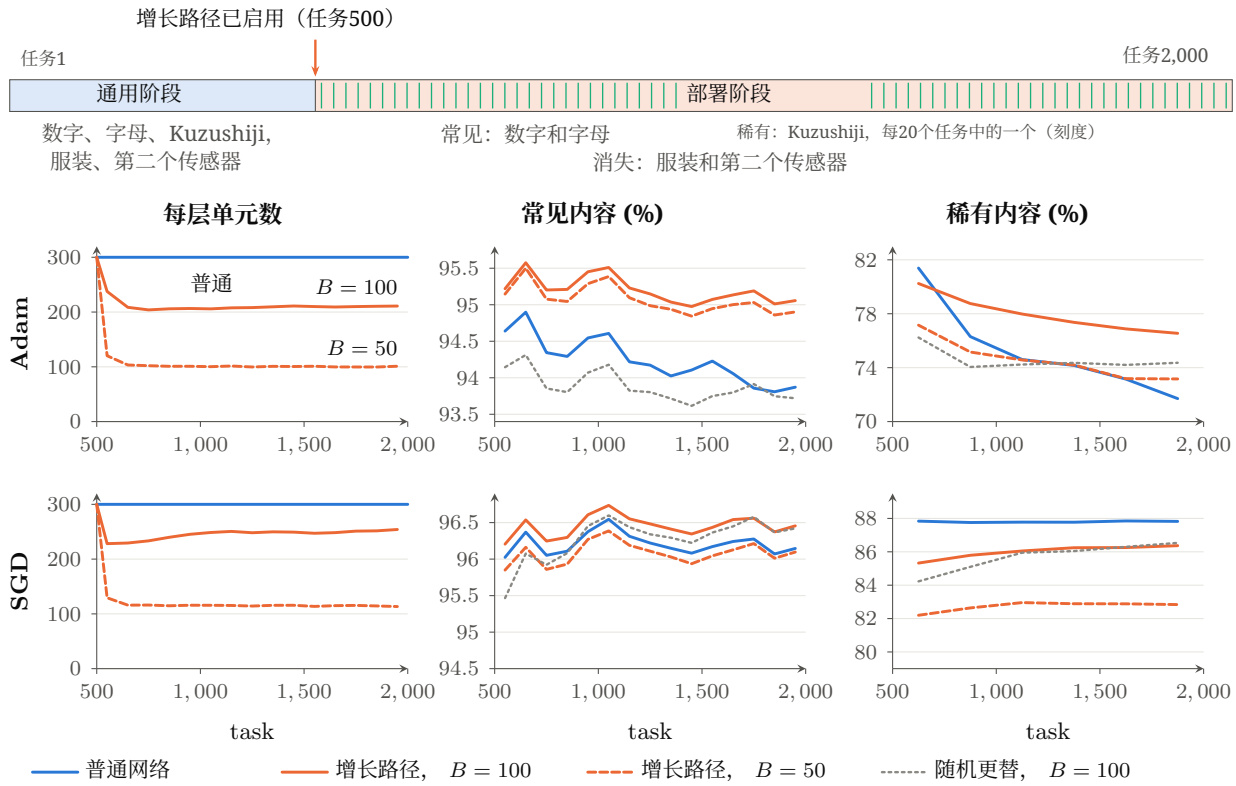

图2：在部署时启动后，增长路径会缩小网络，同时让它持续学习。上：部署数据流。下：每个隐藏层的存活单元均值、常见内容上的准确率（每 100 个任务为一个窗口）和稀有内容上的准确率（每 250 个任务为一个窗口）；数值为五个随机种子的均值。网络规模会在约 200 个任务内稳定下来，达到由预算 B决定的层级。使用 Adam 时，这个较小的网络在两类内容上的准确率都高于普通网络，而随机选择同样数量的单元则做不到。使用随机梯度下降时，它在常见内容上的表现与普通网络持平，在稀有内容上的表现则较差。

4.1 普通网络放任自流

两种优化器给出的基线大不相同，以下每项结果都必须结合各自的基线来解读。使用 Adam 时，普通网络在常见内容上的准确率达到 93.9%，而稀有内容的表现自行下降：Kuzushiji 准确率从通用阶段结束时的 87.3% 降至部署后期的 72.4%。它的大多数单元都已停止工作：部署后期，在留出图像上，第 1、2、3 层中非活动单元的比例分别为 30%、94% 和 92%。使用随机梯度下降时，普通网络的准确率达到 96.1%，并保留了对稀有内容的表现（88.7% 降至 87.8%），非活动单元的比例则为 9%、60% 和 35%。因此，在 Adam 下，可获益之处很多而损失很小；在随机梯度下降下，普通网络是一个强有力的基线，没有损失。

4.2 更小，而且使用 Adam 时表现更好

图2 展示了启动之后发生的变化，表1给出了判定结果。大约在 200 个任务内，各层会缩减到由预算决定的规模，大致为每单位预算对应两个单元（ $B = 100$ 时每层约 200 个单元，而 $B = 50$时每层约 100 个单元），之后便保持这一规模。网络始终在持续学习。

Adam。 在 $B = 100$ 时，该网络在常见

网络	计算量 (%)	常见内容 (点)	后期稀有内容 (点)	规则 (a)	(b)	(c)
Adam						
普通网络	100	93.92	72.41			
增长路径, $B = 100$	60	$+1.16$ [0.76, 1.55]	$+4.3$ [1.1, 7.5]	✓	✓	✓
增长路径, $B = 50$	24	$+1.01$ [0.69, 1.33]	$+0.8$ [-2.7, 4.3]	✓	✓	×
增长路径, $B = 25$	12	$+0.54$ [0.21, 0.86]	-3.5 [-7.0, -0.1]	✓	✓	×
带有休眠, $B = 100$	71	$+1.20$ [0.87, 1.53]	$+8.1$ [5.1, 11.2]	✓	✓	✓
具有休眠, $B = 50$	31	$+1.04$ [0.63, 1.44]	$+5.4$ [2.2, 8.5]	✓	✓	✓
具有休眠, $B = 35$	22	$+0.84$ [0.55, 1.13]	$+3.8$ [1.3, 6.4]	✓	✓	✓
SGD						
普通网络	100	96.15	87.84			
增长路径, $B = 100$	77	$+0.31$ [0.16, 0.45]	-1.5 [-2.0, -1.0]	✓	×	×
增长路径, $B = 50$	31	-0.05 [-0.17, 0.07]	-5.0 [-5.5, -4.4]	✓	✓	×
增长路径, $B = 25$	15	-0.85 [-0.92, -0.78]	-11.7 [-12.8, -10.5]	×	✓	×
在休眠条件下, $B = 100$	85	$+0.31$ [0.20, 0.42]	-0.3 [-0.7, 0.1]	✓	×	✓
在休眠条件下, $B = 50$	39	$+0.04$ [-0.09, 0.17]	-2.8 [-3.4, -2.2]	✓	✓	×
在休眠条件下, $B = 35$	27	-0.17 [-0.31, -0.04]	-5.5 [-5.8, -5.1]	✓	✓	×

表1：部署：与普通网络的差值（种子 15–19，按随机种子配对，95% 置信区间）。计算量是计入成本占普通网络的百分比。每个区块的第一行给出普通网络的准确率（百分比）。预先固定的规则（第 3 节）：(a) 常见内容的表现不得差超过 0.5 个百分点；(b) 计算量最多为 75%；(c) 后期稀有内容的表现不得差超过 1.0 个百分点。每项均根据区间末端进行判断，因此，均值为正但区间较宽时，仍可能不符合规则 (c)。无休眠的增长路径根据 (a) 和 (b) 进行判断；规则 (c) 是针对休眠新增的，现列出所有行。附录 C 列出了全部六种预算。

内容上的表现比普通网络高出 1.16 个百分点（95.07% 对比 93.92%），而计算量仅为其 60%。在 $B = 50$ 时，它的表现高出 1.01 个百分点，而计算量仅为其 24%。预先设定的两条规则都成立，五个随机种子的结果也都一致。稀有内容也没有丢失：$B = 100$ 时为 $+4.3$ 个百分点，$B = 50$时则没有可测量的差值。在 $B = 100$时，三层中的非活动单元占比分别降至 14%、6% 和 44%：留下来的单元都发挥着作用。

随机梯度下降。在 $B = 50$ 条件下，网络在常见内容上的表现与普通网络持平（-0.05 个百分点），而计算量仅为普通网络的 31%，因此符合规则 (a) 和 (b)。代价体现在稀有内容上，其最终低了 5.0 个百分点。在 $B = 100$ 条件下，常见内容的表现高出 0.31 个百分点，但计算量仅下降了 23%，未达到规则 (b) 要求的 25%，而稀有内容低了 1.5 个百分点。

没有减少预算，就不会缩小。在 $B = 300$（预训练网络的宽度）条件下，两种优化器的每次运行都会持续增长，直到某一层达到 768 个单元的存储上限后停止。要让网络缩小，必须将预算设定在宽度以下；第 2.3节解释了原因。

4.3 计算量的单一调节旋钮

预算是我们为得到图 3 而改变的唯一数字。从 25 到 100 的六个预算给出六个网络，其计算量按顺序分别为：在 Adam 下占普通网络的 12% 到 60%，在随机梯度下降下占 15% 到 77%，且未改动任何其他设置。在 Adam 下，规则 (a) 和 (b) 在每一个预算下都成立。在 $B = 25$，该网络的三层分别使用 59、46 和 45 个单元，计算量为 12%，在常见内容上仍比宽度为 300 的普通网络高出 0.54 分。在随机梯度下降下，常见内容被压低至 $B = 50$，随后开始让步：-0.40 分，位于 $B = 35$，以及 -0.85，位于 $B = 25$。

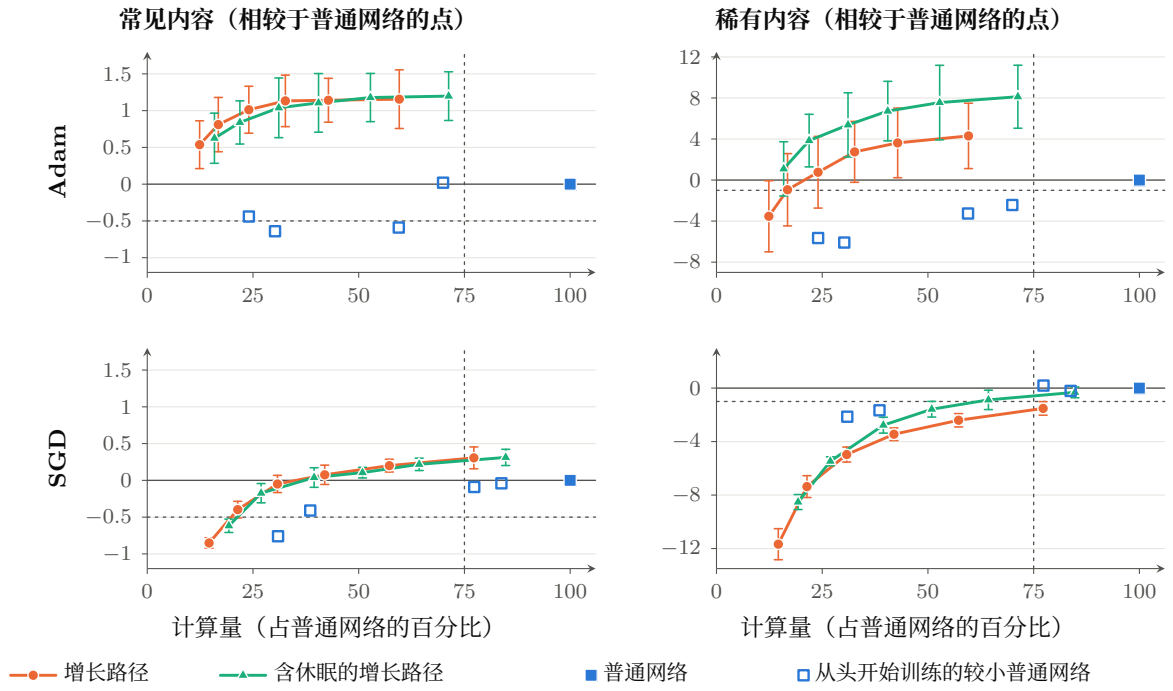

图3：预算沿着平滑曲线在计算量与准确率之间进行权衡。相对于普通网络的差值与计算量的关系，预算为 $B = 25$、35、50、65、80 和 100（每条曲线上从左到右）；柱状线表示配对差值的 95% 置信区间。虚线标出了表1中的规则。使用 Adam 时，在每个预算下，增长路径在常见内容上的表现都优于普通网络，即使计算量降至 12%；同时，休眠也保留了稀有内容。使用随机梯度下降时，常见内容可保持到约 30% 的计算量，而稀有内容则逐渐丢失。空心方框：达到 $B = 50$ 和 $B = 100$规模的普通网络，在完整数据流上从头开始训练。

4.4 哪些单元被替换很重要

一个拥有新单元的更小网络，其表现更好的原因可能与营养毫无关系。两个对照组对此进行检验。

随机更替在每次运行中重复相同数量的死亡与分裂，并在每次检查时随机抽取单元。在 Adam 下，它的表现不再优于普通网络（93.8% 和 93.9%，而普通网络为 93.9%），而增长路径在 $B = 100$ 高出 1.30 个百分点，在$B = 50$ 高出 1.04 个百分点（图2）。因此，Adam 下的增益来自替换哪些单元，而不是替换单元这一行为本身。在随机梯度下降下，两者表现接近： $+0.02$ 个百分点，在 $B = 100$ 以及 $+0.27$ 在 $B = 50$。

在 Adam 优化器下，在常见内容上的表现比增长路径低 1.74 和 1.45 个百分点；在随机梯度下降下，则低 0.40 和 0.71 个百分点。这是一个规模相同、在完整数据流上从头开始训练的普通网络，因此，规模本身也无法解释这一结果。

图4 关注的是单元本身。在启动之前，我们通过静默每个预训练单元对其进行分类：根据损失在哪类内容上升幅最大，将单元归类为对常见内容、稀有内容或已消失的内容重要；如果什么都没有变化，则归类为无。增长路径看不到这些信息；它读取的是激活值和权重。然而，经过 100 项任务后，它移除了 98% 对任何内容都无关紧要的单元、67% 服务于已消失内容的单元，以及 44% 服务于常见内容的单元（Adam， $B = 100$）。在 Adam 下，服务于已消失内容的单元比服务于常见内容的单元多死亡 23 个百分点；在随机梯度下降下，多死亡 19 个百分点；而随机选择时，这一差值与零无法区分。选择确实存在，但远不够干净：到

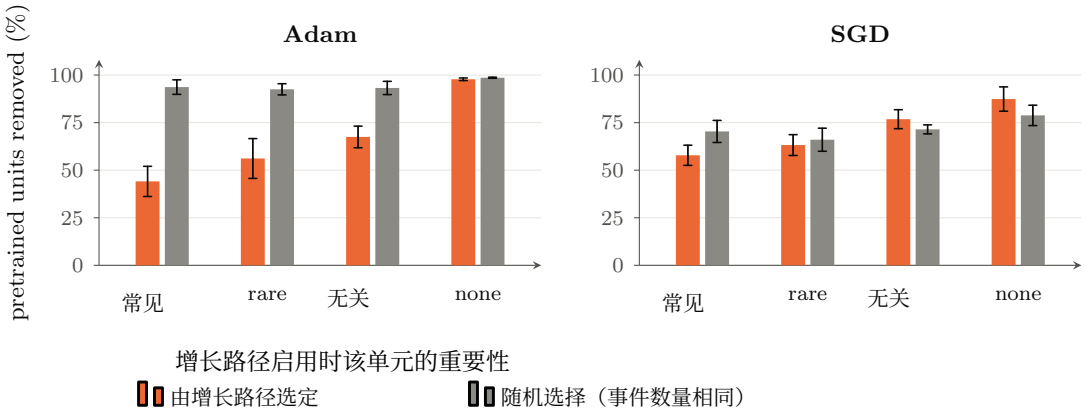

图4：该规则会移除领域不再需要的内容，且从未查看任何标签。启动增长路径后的前 100 项任务中被移除的预训练单元占比（$B = 100$），按增长路径启动时单元所关联的内容分类（静默该单元时损失增幅最大的内容；“无”：静默该单元不会改变任何内容）。柱状图表示五个随机种子的均值，误差范围为 95% 置信区间。随机移除相同数量的单元不会产生这种差值。

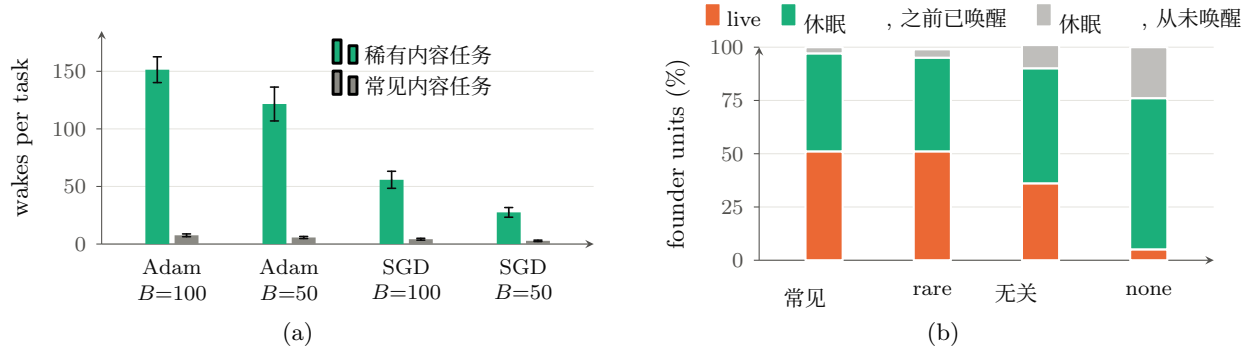

图 5：当休眠单元所服务的内容重新出现时，它们会被唤醒。(a) 稀有内容任务和常见内容任务中每项任务的唤醒次数（五个随机种子的均值，95% 置信区间）。(b) 数据流结束时的预训练单元，按其在启动时所关联的重要内容分类（Adam，$B = 100$）。

数据流结束时，大约 70% 对常见内容重要的单元也已被替换。网络是重建的，而不只是修剪（第6节）。

4.5 休眠保留稀有内容

死亡不可逆，而稀有内容也显现出代价：一个在二十项任务中只使用一次的单元，在其余十九项任务之间可能会挨饿。借助休眠（第 2.4 节），饥饿的预训练单元会连同它学到的一切一起被搁置，待其将获得的营养达到所在层的均值时再度返回。

在我们同时采用两种方法进行测试的四种设置中，休眠保留的稀有内容都多于死亡：$+3.8$ 以及 $+4.6$ 在 Adam 下的百分点（$B = 100$ 和 50），以及 $+1.2$ 和 $+2.2$ 在随机梯度下降下的百分点，且所有区间的数值均高于零。常见内容没有变化。代价是唤醒测试，以及任何时刻处于唤醒状态的单元，这会消耗 7 到 12 个百分点的计算量。在 Adam 下，从预算 35 到 100 的所有预算中，这三条规则现在都成立（表1，图3）：在 $B = 35$ 网络使用量为计算量的 22%，在常见内容上比普通网络高 0.84 个百分点，在稀有内容上高 3.8 个百分点。在计算量相同的情况下，休眠保留的稀有内容多出约 2.5 到 3.6 个百分点

在整个曲线上都比死亡多。

唤醒情况与内容有关（图 5a）。稀有内容任务在 Adam 下会触发 122 到 151 次唤醒，在随机梯度下降下会触发 28 到 56 次；常见内容任务则会触发 3 到 8 次。规则的任何部分都无法得知当前运行的是哪种任务。图 5b 展示了预训练单元最终的状态：对常见内容或稀有内容有用的单元中，约有一半最终处于存活状态，其余单元几乎都曾经处于唤醒状态；对任何内容都没有用的单元则大多处于休眠状态。

休眠节省的是计算量，而非存储。所有预训练单元都会保留，因此存储的网络大小是原来的 1.1 到 1.8 倍。

4.6 使用随机梯度下降：直截了当地说明权衡

在随机梯度下降下，同样的曲线体现的是一种权衡，而非免费收益（图 3，下图）。常见内容的计算量可降至约 30%。稀有内容在所有预算下都落后于普通网络：在 $B = 100$ 时落后 1.5 个百分点，到 $B = 25$时落后 11.7 个百分点；采用休眠时，落后幅度为 0.3 到 8.5 个百分点。在计算量相同的情况下，从约 40% 的计算量起，休眠的表现比死亡好约 1 个百分点；低于这一水平时则好不到半个百分点。没有任何预算能同时满足这三条规则：在 $B = 100$ 时，稀有内容的表现与普通网络持平，但计算量只降低了 15%；从 $B = 80$ 开始，稀有内容就会丢失。

有一项比较结果与该方法相悖，因此放在这里讨论。在相同计算量下，一个缩小规模的普通网络使用随机梯度下降，在完整数据流上从头开始训练，丢失的稀有内容更少：当计算量为 31% 时，分别丢失 2.1 和 5.0 个百分点；当计算量为 39% 时（启用休眠），分别丢失 1.7 和 2.8 个百分点。它丢失的常见内容更多（0.76 和 0.41 个百分点），而且这种做法预设最终规模事先已知，并且通用阶段可以在该规模下重复进行，但这并非实际部署的情况。不过，在随机梯度下降下，从未拥有额外单元的网络，比舍弃这些单元的网络更能保留稀有内容。

5 在长任务流上的持续学习

部署始于一个已经存在的网络。这里，网络在不断变化的数据流上从头开始训练，增长路径从第一次更新起就启用，预算等于初始宽度。网络自行选择规模，因此普通网络会按照该方法达到的规模运行（第3节）。我们使用两种数据流，它们对旧知识的价值判断不同。

5.1 旧知识何时失效：像素置换流

该数据流中的每个任务都会重新置换像素，因此关于输入所学的一切都无法延续，而普通网络训练时间越长，学习效果反而越差。图6 （左）展示了在 Adam 下进行的 800 个任务，初始宽度为 300。采用增长路径的网络从 94.0% 上升至 95.2%，并保持不变。初始宽度的普通网络从 93.1% 降至 88.6%，而采用该方法达到的尺寸的普通网络则分别从 93.8% 和 94.2% 降至 90.9% 和 91.1 %。与其中较强者相比，优势从前 20 个任务中的 -0.2 个百分点增长到后 20 个任务中的 $+4.0$ 个百分点。采用相同事件的随机更替也会下降（93.8% 至 92.8%），因此规则与随机选择之间的差距在整个数据流中不断扩大，最终达到 $+2.3$个百分点。在任务 100 左右从 900 个单元增长到 1,488 个单元后，该网络逐渐变小，最终为 1,230 个单元。

在随机梯度下降（中等；起始宽度为 100）下，该方法起初比更大的普通网络落后 2.1 个百分点，在约 200 个任务后追平，最终领先 2.3 个百分点；此时普通

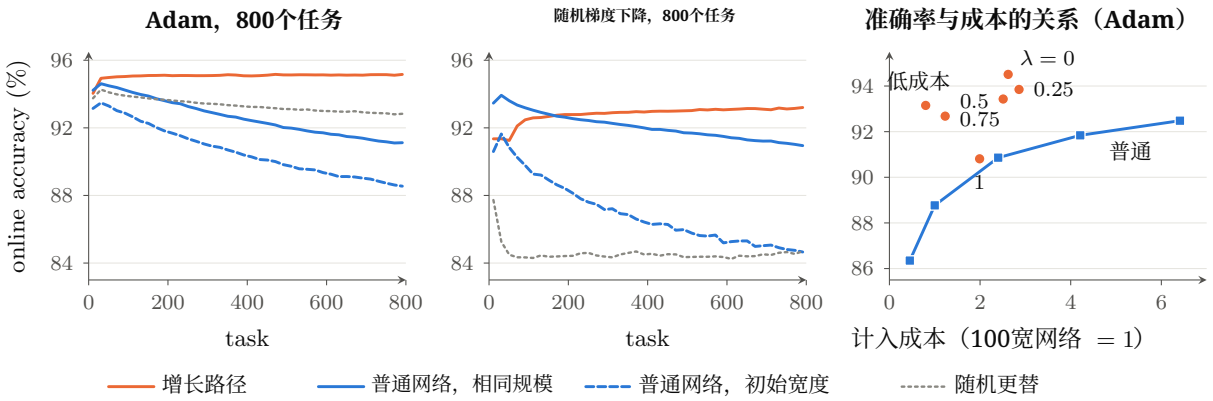

图 6：在输入持续重连的数据流上，具有增长路径的网络能够持续学习。左图和中图：在 800 个像素置换任务上的在线准确率（每 20 个任务为一组，5 个随机种子；$\lambda = 0$）。相同规模的普通网络是我们两个规模匹配基线中表现较强的一个（峰值规模）。使用 Adam 时，增长路径始终领先；使用随机梯度下降时，它起初落后于规模更大的普通网络，约经过 200 个任务后追平，并从此领先。右图：200 个任务后准确率与成本的关系（Adam）；每个橙色点表示为一个继承比例选取的设置 λ（标记为“低成本”的点是一个具有 $\lambda = 0.25$ 的第二种设置），蓝线表示宽度为 50 至 400 的普通网络。

	λ	准确率 (%)	高于普通网络 最终规模	峰值规模	高于随机 周转率	判定
Adam	0	93.21	+1.25 [0.92, 1.57]	+0.82 [0.53, 1.10]	+4.82 [4.60, 5.05]	通过
	0.5	93.41	+1.59 [1.35, 1.84]	+1.31 [1.19, 1.43]	+0.15 [0.10, 0.20]	通过
	1	93.16	+1.25 [1.18, 1.33]	+1.12 [0.93, 1.30]	+0.53 [0.24, 0.82]	通过
SGD	0	92.73	−1.17 [−1.45, −0.89]	−1.17 [−1.34, −1.00]	+6.61 [6.44, 6.79]	未显示
	0.5	93.91	−0.02 [−0.24, 0.20]	−0.06 [−0.24, 0.12]	+0.17 [−0.02, 0.36]	未显示
	1	93.50	−0.24 [−0.47, −0.00]	−0.42 [−0.65, −0.19]	+0.12 [−0.10, 0.35]	未显示

表 2：混合流（2,000 个任务；种子 15–19；配对差值以百分点计，并附 95% 置信区间）。准确率是主任务最后十分之一部分的平均区块分数。如果该方法的表现高于规模匹配的两个普通网络（每个网络都使用各自调优的步长），则判定为通过。

网络的表现正在下降。这里，网络反而不断增长，单元数量从 604 增至 1,217，而随机更替的表现远逊于其他所有情况。

继承。在这个数据流中，子单元不继承任何内容时表现最好（图 6，右）。使用 Adam 训练 200 个任务后，准确率从 $\lambda = 0$ 时的 94.5%，降至 $\lambda = 0.5$ 时的 93.4%，再降至 $\lambda = 1$时的 90.8%；在 $\lambda = 0$ 时，该网络比最终规模相同的普通网络高出 3.2 个百分点。每个任务都会重新连接输入，因此亲代对其输入的学习内容对子单元毫无用处；出现这种结果正是理所当然。（这项比较中的随机梯度下降实验因截止期限而中止，且未重新运行，因此继承曲线仅针对 Adam。）

成本。采用 $\lambda = 0.25$ 的设置最终稳定在 83、67 和 72 个单元，计入成本为 0.80 时达到 93.1%，低于其起始的 100 宽网络。普通网络每层需要 400 个单元，成本则是其八倍（6.41），才能达到 92.5%。

5.2 旧知识如何持续发挥作用：混合流

在混合流中，主内容会反复出现，因此值得保留单元，也有内容可供继承。表2 给出了三种继承比例下的判定结果。

Adam。 在三种继承比例下，增长路径在每个随机种子上都优于两种规模匹配的普通网络：与最终规模相比高出 1.2 至 1.6 个百分点，与峰值规模相比高出 0.8 至 1.3 个百分点。随机梯度下降。 则并非如此：它在 $\lambda = 0.5$ 处与普通网络持平，在 $\lambda = 0$和 $\lambda = 1$处略低于普通网络。整个比较中最高的准确率来自使用随机梯度下降训练的普通网络（最高达 94.0%）；使用随机梯度下降的增长路径（93.9%）与它们难以区分。在随机梯度下降下，普通网络在这条数据流上的性能几乎没有下降，因此该方法几乎无从修复；我们找不到任何能让它发挥作用的此类调度计划。

表格之外还有三个值得注意的事实。第一，在 $\lambda = 0.5$ 和 1 的情况下，使用随机更替也能获得 Adam 带来的大部分提升；这条规则还能额外提高 0.15 和 0.53 个百分点。与部署时相比，单元的选择在这里影响小得多；在部署时，全部提升都取决于单元的选择。只有在 $\lambda = 0$ 的情况下，也就是每个任务约替换 96 个单元时，随机选择才会造成显著损害。第二，在这里，死亡的单元同样是较不实用的单元：每次接触无关内容后，在 200 个任务内，最不实用的三分之一单元的死亡率比最实用的三分之一高 7 到 13 个百分点（六种情况中有五种是在随机梯度下降下），而使用随机更替时则没有差值；但最实用的三分之一单元中，也有一半到四分之三会死亡。第三，这种方法受到一次访问的干扰更大：第二个传感器访问期间，Adam 下它在主内容上的准确率会下降 1.9 到 2.4 个百分点，而普通网络只下降 0.4 到 0.7 个百分点；它需要 62 到 75 个任务才能恢复（随机梯度下降下最多需要 472 个任务）。

5.3 守恒决定规模

无守恒对照（第 3 节）根据每个单元自身的使用量进行供给，不进行分裂，因此一层中各单元的摄取量之和不会被限制在预算之内。在 Adam 下，面对混合流时，全部 15 次确认运行都不断增长，直到某一层填满 768 个单元的存储空间后停止；停止时任务数介于 44 至 930 之间。调优随机种子下尝试的 24 种设置，也都至少在一个随机种子上出现了同样的情况。采用守恒时，相同设置均能完成全部 2,000 项任务，每层有 175 至 256 个单元。在像素置换流上，结果取决于继承比例：网络在 $\lambda = 0$ 时缩减至 49、59 和 140 个单元，而在 $\lambda = 0.5$ 和 1 时增长至存储上限。由于无守恒的运行未能完成，我们无法断言守恒是否也能提高准确率。对照实验所揭示的结论范围更窄，却很明确：守恒是这条规则中限制规模的部分。

5.4 双亲移除重复单元

在 $\lambda > 0$ 进行分裂会创建一个副本，而副本可以持续存在。在混合流上使用随机梯度下降时，$\lambda = 1$ 第 2 层和第 3 层中已建立单元分别有 36% 和 42% 与另一个单元的响应几乎相同（相关性至少为 0.9）；使用 Adam 时，这一比例最高为 9%。采用双亲（第 2.4 节）时，子单元不是任何一方亲代的副本，各设置下此类重复单元的比例均降至最高 9%；若每个权重都从两个亲代之一复制而来，这一比例则最高为 3%。在八种设置中，有三种设置的准确率高于单亲情形（高出 0.2 至 0.6 个百分点），五种设置下无法区分，且没有任何一种设置更低。网络规模也大了 7% 至 27%，因为每个亲代提供的使用量较少，因而可以更早再次分裂，所以我们不会将差值仅归因于这一机制。与匹配规模的普通网络相比，判定结果如表 2 所示：在我们测试的全部四种设置中，使用 Adam 时均通过，高出 1.2 至 1.7 个百分点；使用随机梯度下降时则无一通过。

6 营养追踪什么，以及它在哪些方面失效

营养水平随单元的数值而变化。这条规则从未衡量过某个单元对损失有多大价值，因此我们亲自进行了测量。在像素置换流上，我们每次屏蔽一个单元，

并记录损失的上升幅度。在 $\lambda = 0$ 时，三层中这一上升幅度与营养水平的秩相关为 0.43 至 0.54；在 $\lambda = 0.5$时则为 0.30 至 0.42。移除某层中营养最充足的四分之一单元，会使第 1、2、3 层的准确率分别下降 21.5、7.9 和 2.7 个百分点；移除随机选取的四分之一单元，分别下降 5.3、2.8 和 1.1 个百分点；移除营养最匮乏的四分之一单元，降幅最多为 0.65 个百分点。因此，一个通过活动值和权重绝对值从输出单元反向传递、且从未接触损失的量，可以用来估计哪些单元承担了相应功能。

有用的单元也会死亡。 这个估计并不精确。在部署时，对于常见内容而言发挥过作用的预训练单元中，约有 70% 到数据流结束时已被替换（第 4.4 节）；而在混合流中，在 200 个任务内，最有用的三分之一单元中有一半到四分之三会死亡。一个可能的原因是，死亡阈值作用于单元在其层中的占比，而在预算减少后启动时，每个占比都很小。网络大概是因为替代某个单元的子单元继承自得到充分供给的亲代，才得以保持其准确率；如果采用一条能保留更多有用单元的规则，效果可能会更好，而我们尚未寻找这样的规则。

被唤醒的单元无法保住自己的位置。 当一个休眠单元本应获得的摄取量达到其所在层的均值时，它就会苏醒。苏醒后，它实际获得的摄取量约为该值的十分之一（本应获得的摄取量中位数为 0.47 至 0.63；500 次更新后的实际摄取量中位数为 0.04 至 0.08），而在其受保护时间结束时，36% 至 58% 的唤醒单元再次低于死亡阈值。一个可能的原因是，该测试使用单元在当前批次上的激活值，而单元返回时携带的是其入睡时的活动均值，这正是路由所读取的值。尽管如此，休眠仍能保留稀有知识（第 4.5 节）。我们认为，最直接的改进是采用一条能让返回单元获得其应有份额的唤醒规则。

Adam 和随机梯度下降为何不同。我们没有经过验证的解释，只有观察结果。在我们的数据流上用 Adam 训练的普通网络会累积非活动单元（在混合流中，第 2 层和第 3 层的比例分别为 68% 和 51%，而用随机梯度下降时分别为 32% 和 23%；在部署时超过 90%），其输出权重会增长十倍，而且随着数据流持续进行，它们对每个新任务的适应速度越来越慢；而使用随机梯度下降时，适应速度会越来越快。在混合流中，使用增长路径时，两种优化器下永久非活动单元的比例都不超过 4%。一种解读是，增长路径主要修复了 Adam 造成、而随机梯度下降不会造成的浪费，因此它在 Adam 下有更大的改进空间。另一个观察结果也与此吻合：使用随机梯度下降时，亲代的副本会保留下来（第 5.4 节），这表明在两种优化器下，新单元的使用方式有所不同。这些都只是假设。我们提出这些假设，是因为优化器之间的差异是本研究尚待解决的核心问题。

时间成本。本文中的所有成本均以参数数量衡量。就实际运行时间而言，在调整优化器记账方式且确保结果位级一致后，我们对采用增长路径的网络进行一次更新所需的时间，约为具有相同数量存活单元的普通网络的 1.2 倍（在一个随机种子和五个任务上测得）；增长路径本身及其观测和结构检查约占十分之一的时间。本文报告的实验耗时为初始宽度普通网络的 1.6 至 2.1 倍。

7 相关工作

路由规则从何而来。 单元将营养向下传递所遵循的规则（1）并非新创。它采用了逐层相关性传播中的保守重新分配形式 [2]以及激励反向传播 [54], 的形式；这些方法通过将类别得分从输出单元向下传递

回传至输入，使每一层的总量保持相同。Yeom等人 [51] 利用此类相关性来剪枝经过调整、用于专门任务的预训练网络。其他从输出单元 [53] 向下传递 [46] 或在各层间守恒的得分 [43, 44]也曾用于剪枝，且要移除的数量预先固定。在这些用途中，分配是一个独立步骤，在网络训练之前或之后进行：要么解释一次预测，要么根据带标签的参考样本计算剪枝过程，随后进行微调。最接近的例外是 Shibata等人提出的剪枝方法|：在反向传播训练过程中，按当前神经元数量成比例计算的奖励从输出单元向下传递至输入，并按 ×权重 |输出的比例拆分；泄漏累积奖励低于阈值的神经元则会被删除。该方法只能移除单元，而且其奖励会随网络缩小而减少；它是一项剪枝标准，而不是单元赖以存活的资源。我们将分配转变为一种贯穿整个训练过程的生长规则。我们将其应用于活动值的运行平均值和权重绝对值；从每个输出单元获得的等量供给开始，而不是从标注类别的得分开始；并且每经过 100 次更新就重复执行一次，将固定预算设为各层相同的数值，供单元赖以存活并相互竞争，同时设置分裂和死亡的阈值。我们认为，这项工作的主要原创之处，正是将分配从分析或剪枝标准转变为训练过程中的生长规则。

单元的增长、剪枝与替换。 网络可以通过添加单元 [12, 16, 52]以及复制或分裂单元 [5, 15, 29, 48, 49]来实现生长；我们在 $\lambda = 1$ 处进行的分裂，是将 Net2Net 的加宽步骤应用于一个单元。网络也可以根据幅值或重要性进行剪枝 [17, 20, 22],，在训练期间以固定数量的参数进行重新连接 [4, 14, 31], ，交替进行增长与剪枝 [8, 9],，以及在对预训练模型进行微调时进行剪枝 [40]。在这些方法中，增长和移除由损失、梯度或幅值决定，按照调度计划进行，并趋向于预先设定的规模或稀疏度。单隐藏层网络中的一个例外是 Qiao、Han及其同事提出的基于敏感度的设计 [19, 37]：每个隐藏单元对输出敏感度的贡献，在层内归一化后，如果较高就分裂该单元，如果较低就删除该单元，因此宽度无需预先固定。该贡献是根据单一输出在每次检查时重新计算的分数；它不会经过多层传递，不由单元保留，也不是单元彼此竞争的预算。

关于可塑性丧失的研究会替换或重置固定规模网络中的单元。持续反向传播 [10, 11] 以固定速率重新初始化效用最低的成熟单元。效用是一种不会造成损失的度量，与我们的使用量一样，由激活值和输出权重构成；其成熟度阈值对应于我们的保护期，而其重新初始化单元对应于我们在 $\lambda = 0$的子单元。其他方法会回收非活动单元 [45], ，或扰动、正则化或重置权重 [1, 13, 24, 33]。（Sokar等人将非活动单元称为休眠单元；我们的休眠单元是指存储起来、未参与计算的单元。）与这些方法相比，增长路径没有替换率，也没有固定规模；子单元从亲代继承，而且同一个守恒量决定出生与死亡。Lillo和Cheney [28] 发现，新生长的单元往往“前向活跃但反向匮乏”；我们采用较大步长的保护期应对同一难题，而唤醒单元的表现表明，这一问题尚未解决。

单元经济学。 让单元依托一种守恒的货币而存活，是强化学习中的一个古老理念。在 Holland 的桶队列 [21],Schmidhuber 的神经桶队列 [41]以及 Baum 的哈耶克机器 [3], （Kwee 等人 [25], 对此有过描述）中，智能体沿着一条链传递由奖励产生的货币；破产的智能体会被移除，而在哈耶克机器中，富有的智能体会付费创造子单元。在这些系统中，货币是外部奖励，智能体则是规则或分类器。这里的货币是一笔固定预算，既不读取奖励，也不读取损失，而是经由网络的权重进行分配；该网络通过反向传播进行训练。Peng 和 Braun [35] 让神经元争夺输入激活，并在没有误差信号的情况下死亡；他们的网络并非通过梯度训练。神经进化曾在一代代网络中选择单个神经元，

普及代网络 [32]；我们的选择则在单个网络的生命周期内进行。

生物学。 增长路径受到神经营养假说的启发，但并非该假说的模型：发育中的神经元竞争存活因子，这些因子由其目标细胞以有限的量 [27, 34,36, 47], 供给；一般而言，细胞会死亡，除非其他细胞向它们发出信号使其存活 [38]。

创新之处。 据我们所知，这是首个学习规则：它将一种守恒量以每层相同的预算从输出单元传递而来，并据此决定梯度训练网络中的隐藏单元何时分裂、存活或死亡，且不读取损失、标签或梯度。通过按任务冻结或门控参数来保留旧知识的方法 [30, 39, 42] 依赖于已知任务；据我们所知，目前还没有与休眠相对应、并基于同一种营养进行唤醒测试的方法。我们的唯一基线是普通网络：我们没有运行上述任何方法，也不对增长路径与它们相比的表现作出任何判断。

8 局限性

结果取决于优化器。 使用 Adam 时，增长路径在三个数据流的主要指标上都优于普通网络。使用随机梯度下降时，它无法提高混合流上的准确率；部署时，它会在计算量与使用稀少的内容之间权衡；而从头开始训练的较小普通网络则能更好地保留稀有内容（第 4.6 节）。Adam 下的出色结果是有条件的：在这些数据流上，普通 Adam 网络的大多数单元都被浪费了，而且网络本身会退化。使用自适应优化器训练的大型模型是否也会以同样的方式浪费单元，仍是一个我们尚未探讨的经验性问题。

规模。 所有网络都是具有三个隐藏层的感知机，所有数据都是 28×28 灰度图像。该规则针对全连接层的单元定义；卷积通道和注意力头需要各自的使用量定义。每项比较都基于五个随机种子，而且我们没有使用新种子重复实验（第3节）。

调优。结构设置是在混合流上选定的，并在部署时原样沿用。这是部署结果的一项优势，因为没有针对它进行拟合；但这也是一种局限：其他阈值可能使图3 中的曲线朝任一方向偏移。预算必须小于宽度；据我们所知，除了尝试之外，并没有规则可以确定在保持给定准确率的前提下所需的最小预算。网络训练时，增长路径无需人工干预，但这并不意味着它不需要预先作出选择：预算、结构常数和存储容量都需手动设定；如果一次运行所需的存储空间超出容量，就必须舍弃该次运行。它还在一处改变了优化器：新生单元采用了更大的步长。

未被守恒、未被保留，也未被测量的内容。 守恒适用于每次检查时传递的营养，而不适用于层级；在大量死亡后，层级会下降。有用单元会大量死亡，而唤醒单元很少能保持其位置（第 6 节）。休眠会减少计算量，而不会减少存储；只有预训练单元可以休眠。若要让后续单元休眠，就需要制定规则，删除那些永远不会返回的单元。成本按参数计数，而不是按测量的时间或能量计算。在混合流上，不采用守恒的运行无法完成，因此守恒对准确率的影响尚不清楚。

致谢

我们披露，实验代码和分析脚本是在 AI 编程代理（OpenAI Codex）的帮助下编写的，论文文本、图表和表格是在 AI 助手（Anthropic Claude）的帮助下起草的。作者设计了这项研究，指导了实验，依据实验记录核实了所有报告的数字，并对内容承担全部责任。

参考文献

[1] J. T. Ash 和 R. P. Adams. 关于神经网络训练热启动. 收录于神经信息处理系统进展（*NeurIPS*），2020. arXiv:1910.08475.[2] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. M¨uller 和 W. Samek. 关于通过逐层相关性传播对非线性分类器决策进行逐像素解释. *PLOSONE*，10(7):e0130140，2015. doi: 10.1371/journal.pone.0130140.[3] E. B. Baum. 朝着将智能建模为智能体经济的方向. 机器学习，35 (2):155–185，1999. doi: 10.1023/A:1007593124513.[4] G. Bellec, D. Kappel, W. Maass 和 R. Legenstein. 深度重连：训练极稀疏深度网络. 收录于国际学习表征会议（*ICLR*），2018. arXiv:1711.05136.[5] T. Chen, I. Goodfellow 和 J. Shlens. Net2Net：通过知识迁移加速学习. arXiv 预印本 arXiv:1511.05641，2015.[6] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto 和 D. Ha. 日本古典文学的深度学习. *arXiv* 预印本 *arXiv:1812.01718*，2018.[7] G. Cohen, S. Afshar, J. Tapson 和 A. van Schaik. EMNIST：将 MNIST 扩展到手写字母. 收录于国际神经网络联合会议（*IJCNN*），2017.[8] D. Dai, Z. Yu, Y. Hu, W. Cao 和 M. Luo. 代谢神经网络. arXiv 预印本 arXiv:1809.00837，2018.[9] X. Dai, H. Yin 和 N. K. Jha. NeST：基于生长与剪枝范式的神经网络合成工具. arXiv 预印本 arXiv:1711.02017，2017.[10] S. Dohare, R. S. Sutton 和 A. R. Mahmood. 持续反向传播：具有持久随机性的随机梯度下降. arXiv 预印本 arXiv:2108.06325，2021.[11] S. Dohare, J. F. Hernandez-Garcia, Q. Lan, P. Rahman, A. R. Mahmood 和 R. S. Sutton. 深度持续学习中的可塑性丧失. *Nature*，632(8026):768–774，2024. doi: 10.1038/s41586-024-07711-7.[12] T. J. Draelos, N. E. Miner, C. C. Lamb, J. A. Cox, C. M. Vineyard, K. D. Carlson, W. M. Severa, C. D. James 和 J. B. Aimone. 神经发生深度学习. 收录于国际神经网络联合会议（*IJCNN*），2017. arXiv:1612.03770.[13] M. Elsayed 和 A. R. Mahmood. 解决持续学习中的可塑性丧失与灾难性遗忘. 收录于国际学习表征会议（*ICLR*），2024. arXiv:2404.00781.

[14] U. Evci, T. Gale, J. Menick, P. S. Castro 和 E. Elsen. 操纵彩票：让所有彩票都成为赢家. 收录于国际机器学习会议（*ICML*），2020. arXiv:1911.11134.[15] U. Evci, B. van Merri¨enboer, T. Unterthiner, M. Vladymyrov 和 F. Pedregosa. GradMax：利用梯度信息增长神经网络. 收录于国际学习表征会议（*ICLR*），2022. arXiv:2201.05125.[16] S. Fahlman 和 C. Lebiere. 级联相关学习架构. 收录于神经信息处理系统进展第 *2* 卷，1989.[17]J. Frankle 和 M. Carbin. 彩票假设：寻找稀疏、可训练的神经网络. 收录于国际学习表征会议（*ICLR*），2019. arXiv:1803.03635.[18] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville 和 Y. Bengio. 基于梯度的神经网络中灾难性遗忘的实证研究. *arXiv* 预印本 *arXiv:1312.6211*，2013.[19] H.-g. Han, M. Li 和 J.-f. Qiao. 基于模型输出敏感性分析的动态神经网络设计. 《电子学报》，38(3):731–736，2010 年 3 月. 中文.[20] S. Han, J. Pool, J. Tran 和 W. J. Dally. 学习高效神经网络的权重与连接. 收录于神经信息处理系统进展（*NIPS*），2015. arXiv:1506.02626.[21] J. H. Holland. 桶式旅的性质. 收录于国际遗传算法会议论文集，Hillsdale，新泽西州，1985.[22] H. Hu, R. Peng, Y.-W. Tai 和 C.-K. Tang. 网络修剪：一种面向高效深度架构的数据驱动神经元剪枝方法. arXiv 预印本 arXiv:1607.03250，2016.[23] D. P. Kingma 和 J. Ba. Adam：一种随机优化方法. 收录于国际学习表征会议（*ICLR*），2015.[24] S. Kumar, H. Marklund 和 B. Van Roy. 通过再生正则化在持续学习中维持可塑性. arXiv 预印本 arXiv:2308.11958，2023.[25] I. Kwee, M. Hutter 和 J. Schmidhuber. 部分可观测世界中的基于市场的强化学习. 收录于国际人工神经网络会议（*ICANN*），*LNCS 2130*，2001. arXiv:cs/0105025.[26] Y. LeCun, L. Bottou, Y. Bengio 和 P. Haffner. 将基于梯度的学习应用于文档识别. *IEEE*会刊，86(11):2278–2324，1998.[27] R. Levi-Montalcini. 神经生长因子问世 35 年后. 《科学》，237(4819):1154–1162，1987. doi: 10.1126/science.3306916.[28] L. Lillo 和 N. Cheney. 结构可塑性中生长的稳定性：前向活跃但反向匮乏. arXiv 预印本 arXiv:2605.15435，2026.[29] Q. Liu, L. Wu 和 D. Wang. 用于增长型神经架构的分裂最速下降. 收录于神经信息处理系统进展（*NeurIPS*），2019. arXiv:1910.02366.

[30] A. Mallya 和 S. Lazebnik。《PackNet：通过迭代剪枝在单个网络中添加多个任务》。arXiv 预印本 arXiv:1711.05769，2017。[31] D. C. Mocanu、E. Mocanu、P. Stone、P. H. Nguyen、M. Gibescu 和 A. Liotta。《受网络科学启发、采用自适应稀疏连接的人工神经网络可扩展训练》。《Nature Communications》，2018。arXiv:1707.04780。[32] D. E. Moriarty 和 R. Miikkulainen。《通过共生进化实现高效强化学习》。《机器学习》，22(1–3):11–32，1996。doi: 10.1007/BF00114722。[33] E. Nikishin、M. Schwarzer、P. D'Oro、P.-L. Bacon 和 A. Courville。《深度强化学习中的首要性偏差》。发表于《国际机器学习会议》（ICML），2022。arXiv:2205.07802。[34] R. W. Oppenheim。《神经系统发育过程中的细胞死亡》。《Annual Review of Neuroscience》，14:453–501，1991。doi: 10.1146/annurev.ne.14.030191.002321。[35] Z. Peng 和 D. A. Braun。《通过竞争存活的神经元来开发神经网络》。发表于 IEEE发展与学习及表观遗传机器人联合国际会议（ICDL-EpiRob），第152–153页，2015。doi: 10.1109/DEVLRN.2015.7346133。[36] D. Purves。《身体与大脑：神经连接营养理论》。哈佛大学出版社，1988。[37] J.-F. Qiao 和 H.-G. Han。《径向基函数神经网络结构的最优设计》。《自动化学报》，36(6):865–872，2010。ISSN 0254-4156。doi: 10.3724/SP.J.1004.2010.00865。中文。[38] M. C. Raff。《细胞存活与细胞死亡的社会控制》。《Nature》，356(6368):397–400，1992。doi: 10.1038/356397a0。[39] A. A. Rusu、N. C. Rabinowitz、G. Desjardins、H. Soyer、J. Kirkpatrick、K. Kavukcuoglu、R. Pascanu 和 R. Hadsell。《渐进式神经网络》。arXiv 预印本 arXiv:1606.04671，2016。[40] V. Sanh、T. Wolf 和 A. M. Rush。《运动剪枝：通过微调实现自适应稀疏度》。发表于《神经信息处理系统进展》（NeurIPS），2020。arXiv:2005.07683。[41] J. Schmidhuber。《动态前馈与循环网络的局部学习算法》。《连接科学》，1(4):403–412，1989。doi: 10.1080/09540098908915650。[42] J. Serr`a、D. Sur´ıs、M. Miron 和 A. Karatzoglou。《通过任务硬注意力克服灾难性遗忘》。发表于《国际机器学习会议》（ICML），2018。arXiv:1801.01423。[43] A. Shibata。《多任务学习过程中的分层神经网络结构分析与创建》。博士论文，东京工业大学计算智能与系统科学系，2015。URL https://t2r2.star.titech.ac.jp/rrws/file/ CTT100683276/ATD100000413/。[44] A. Shibata、J. Lu、F. Dong 和 K. Hirota。《基于剪枝的神经网络结构分解及其可视化方法》。《先进计算智能与智能信息学杂志》，17(3):443–449，2013。doi: 10.20965/jaciii.2013.p0443。

[45] G. Sokar、R. Agarwal、P. S. Castro 和 U. Evci。《深度强化学习中的休眠神经元现象》。发表于《国际机器学习会议》（*ICML*），2023。arXiv:2302.12902。[46] H. Tanaka、D. Kunin、D. L. K. Yamins 和 S. Ganguli。《无需数据、通过迭代保持突触流来剪枝神经网络》。发表于《神经信息处理系统进展》（*NeurIPS*），2020。arXiv:2006.05467。[47] A. van Ooyen 和 D. J. Willshaw。《神经连接发育过程中对神经营养因子的竞争》。《伦敦皇家学会会刊》。*B*辑：生物科学，266(1422):883–892，1999。doi: 10.1098/rspb.1999.0719。[48] T. Wei、C. Wang、Y. Rui 和 C. W. Chen。《网络态射》。arXiv 预印本 arXiv:1603.01670，2016。[49] L. Wu、B. Liu、P. Stone 和 Q. Liu。《萤火虫神经架构下降：一种增长神经网络的通用方法》。arXiv 预印本 arXiv:2102.08574，2021。[50] H. Xiao、K. Rasul 和 R. Vollgraf。《Fashion-MNIST：用于机器学习算法基准测试的新型图像数据集》。*arXiv* 预印本 *arXiv:1708.07747*，2017。[51] S.-K. Yeom、P. Seegerer、S. Lapuschkin、A. Binder、S. Wiedemann、K.-R. M¨uller 和 W. Samek。《基于解释的剪枝：一种用于深度神经网络剪枝的新颖准则》。《模式识别》，2021。arXiv:1912.08881。[52] J. Yoon、E. Yang、J. Lee 和 S. J. Hwang。《利用动态可扩展网络实现终身学习》。arXiv 预印本 arXiv:1708.01547，2017。[53] R. Yu、A. Li、C.-F. Chen、J.-H. Lai、V. I. Morariu、X. Han、M. Gao、C.-Y. Lin 和 L. S. Davis。《NISP：利用神经元重要性分数传播剪枝网络》。arXiv 预印本 arXiv:1711.05908，2017。[54] J. Zhang、Z. Lin、J. Brandt、X. Shen 和 S. Sclaroff。《通过激励反向传播实现自上而下的神经注意力》。发表于《欧洲计算机视觉会议》（*ECCV*），2016。arXiv:1608.00507。

A 方法详述

保护与容量。 新生单元的初始年龄为零。在经过P 次更新之前，它既不会死亡，也不会分裂；在此期间，其输入权重和偏置的步长会乘以$\kappa \geq 1$。对于 Adam，其连接的动量估计从零开始，偏置校正则从该单元诞生时起计数。从头训练的网络中，所有初始单元的年龄都为零，层级为 B/N_ℓ，因此它们也受到保护，并在前 P 次更新中使用乘后的步长。存储容量最多为每层 C 个单元；如果某次运行中某一层达到 C、失去所有单元或产生非有限值，则该运行无效，不用于选择或比较。

设置。 上述常数构成一个结构设置；设置在调优随机种子（第 3 节）上选定，并与结果一同给出。所有部署实验均使用 $\theta_{\text{die}} = 0.05$、 $\theta_{\text{div}} = 1.5$、$\tau = P = 3$、750 次更新、 $\kappa = 3$、 $\varepsilon = 0.1$、 $\rho = 0.01$、 $\lambda = 0.5$ 和 $C = 768$。

A.1 双亲分裂

在双亲变体中，首先照常选择第一个亲代 p，然后从该层的其他已建立单元中抽取第二个亲代 q，其概率与它们的层级成正比。在（3）中，w_p 被 w_p 和 w_q的组合所替代：要么每个条目（每个输入权重以及偏置）以相等概率从两个亲代中的一个复制而来，要么对两个向量取平均。每个亲代提供四分之一，而单亲提供二分之一：子单元接收 $\frac{\lambda}{4}(W_{pj} + W_{qj})$，每个亲代保留其输出权重的 $1 - \frac{\lambda}{4}$，层级也按相同比例变动，因此该层的总层级再次保持不变。其他一切均保持不变；如果某层没有第二个已建立单元，分裂则退回到单亲方式。子单元通常不是任一亲代的副本，因此对于 $\lambda > 0$ 网络的功能会在分裂时发生变化。

A.2 休眠而非死亡

死亡是不可逆的：一个仅偶尔需要的单元可能会在两次使用之间因缺乏摄取量而死亡，而它所学到的内容也会随之丢失。因此，对于预训练网络，我们还研究了一种变体：让饥饿的预训练单元进入休眠状态。我们将启动增长路径时已存在的单元称为创始单元。如果一个已建立的创始单元的层级低于 θ_{die}，则它会进入休眠，而不是像原本那样死亡。它不参与正向传播或反向传播；其权重、偏置、优化器状态和 Growth Path 状态均在单独的存储区中保持不变，并且其位置仍被保留。启动后产生的单元则照常死亡。

每次检查时，都会询问每个休眠单元 d（层 ℓ）如果仅将其重新加入，它会接收到什么。其可能产生的活动值 $\hat{a}_d$ 是它在上一个训练批次上的平均激活值，根据其存储的输入权重和下方层的当前激活值计算得出。其可能产生的摄取量由（1）推得，其中将 d 加入活跃单元，

$$\hat{n}_d = \sum_j \sigma_j \frac{\hat{a}_d \, |W_{dj}|}{\sum_{k \in \mathcal{A}_\ell} u_{kj} + \hat{a}_d \, |W_{dj}|}, \tag{6}$$

连同其存储的输出权重 W_{dj} 和本次检查的供给量 σ_j。如果 $\hat{n}_d \geq B/N_\ell$，即其所在层的存活单元的平均摄入量，该单元就会醒来；此测试不增加任何参数。醒来的单元会带着其存储的权重和状态恢复，并像其他单元一样参与竞争，但在一个保护期（3,750 次更新）内不能再次进入休眠。由于它恢复时也带回了存储的层级，因此醒来后，某一层的总层级可能会短暂超过 B；每次检查时分配的营养仍保持守恒。本文中只有创始单元可以进入休眠：若要让之后诞生的单元也能休眠，就需要制定一条规则，删除已休眠很久的单元；这一问题留待以后解决。

A.3 成本统计

我们以参与计算的参数来计算成本。计算成本是网络中活跃单元之间的权重和偏置数量，除以 99,710（即 784–100–100–100–10 网络的参数数量）；普通的 784–300–300–300–10 网络的成本为 4.2033。对于包含休眠单元的网络，计算成本还包括唤醒测试：对于每个休眠单元，将其与活跃单元之间的输入和输出连接以及其偏置计入每次检查一次，也就是除以 K。存储成本将活跃单元和休眠单元一并计算；没有休眠时，它等于计算成本。该计数不包括增长路径本身的记账，也不是实际运行时间测量（第8节）。

B 实验方案细节

以下的比较、指标和决策规则，均在运行该方法之前，依据用于得出结论的随机种子预先确定。第 4 节和第5 节将据此进行报告。

B.1 主要基线：普通网络

本文始终以普通网络作为基线：采用结构固定的同一多层感知机，在相同数据流上使用相同优化器进行训练，并通过反向传播更新参数。其步长针对调优随机种子进行调整；对于每一种使用规模，都单独进行调优（在混合流和部署数据流上，针对 Adam 使用从 10^{-4} 到 10^{-2} 、针对随机梯度下降使用从 10^{-2} 到 1 的 12 个对数间隔点）。只有相较于该网络表现更好，结果才算有利于该方法。

当网络从头开始训练时，方法会自行选择规模，因此普通网络会在两种匹配规模下运行：方法的最终规模 （评估窗口内每层存活单元数的均值）以及其峰值规模 （它在数据流中达到的最大规模）。该方法必须胜过这两种规模下的基线。在部署数据流中，基线是同一个预训练网络，按照常规方式以固定规模进行训练。

B.2 三种对照

随机更替会重放相同随机种子的真实运行：在每次检查和每一层中，它都应用相同数量的死亡与分裂（有休眠时，也包括休眠与唤醒），但在符合事件条件的单元中均匀随机抽取单元。它将替换哪些单元与替换多少单元以及何时替换区分开来。

无守恒只改变一件事：单元 i 接收$c_\ell\,\sigma_j\,u_{ij}$ 来自下游单元 j的量，但不再除以（1）中的总使用量。常数 c_ℓ 在初始保护期结束时、每层各自固定一次，使该时刻该层的摄取量总和为 B 。此后，单元的摄取量仅取决于其自身的使用量，而一层的总量可以自由升降。

一个从头训练的同等规模的普通网络采用该方法最终达到的宽度，并使用自身调优的步长，在完整数据流上进行训练。它要考察的是，某个结果是否仅由规模解释。在从头训练的两个数据流上，它本身就是基线。在部署数据流上，它仅作为参考：它需要预先知道目标规模并从头训练，而在部署预训练模型时，这两者都不可用。

B.3 调优随机种子、检查种子与预先固定的规则

种子 0–4 是调优随机种子。所有选择均按照固定规则基于这些种子做出，方法和每个普通网络皆如此：在有效设置中（所有调优随机种子上均未出现无效运行，见第2.2节），选择主要指标均值最高的设置。该方法的设置计入其调参预算。在训练任何检查种子之前，选择结果即已固定。随后，每项比较只在五个检查种子上运行一次，这些种子未用于任何调优运行：像素置换流上使用种子 5–9 和 10–14，混合流和部署数据流上使用种子 15–19。在部署数据流上，该方法的任何部分都未经过调优：结构设置和继承比例采用在混合流上选定的设置，步长则采用预训练网络的步长，因此，与基线相比，唯一的差异就是增长路径。

在使用检查种子运行该方法之前，我们已记录数据流长度、主要指标和判定阈值；未预先确定的读数会在出现之处标明。我们没有使用新种子进行进一步确认：本文中的每项结果都基于其比较所用的五个检查种子。

对于给定的随机种子，所有网络看到的都是相同的数据流。在从头开始训练的数据流上，该方法从初始宽度对应的普通网络的初始权重开始；在部署时，每种方法都从同一个预训练网络开始，包括其优化器状态。我们使用两种优化器：不带动量的随机梯度下降（SGD）和采用默认设置的 Adam，批大小为 16，不使用权重衰减，也不使用调度计划。比较始终是在使用相同优化器的运行之间进行，表格会将两种优化器的结果并列展示，并分别给出判定；若某项比较只使用了一种优化器，我们会注明。所有比较都按随机种子配对：我们报告五个逐种子差值的均值及其 95% 区间，该区间基于自由度为 4 的 Student t 分布；只有当区间不包含零时，我们才将差值判为正或负。

B.4 三种任务流

数据流由一系列前后相继的十分类任务组成，任务边界处没有任何提示信号；每个任务的所有训练图像只呈现一次。所有网络都有 784 个输入、三层隐藏层和 10 个输出单元，并且在所有数据流中，增长路径的时间常数均以更新次数计。

像素置换流。 每个任务包含 MNIST 的 60,000 张训练图像 [26]，像素经过新的随机置换 [18], 并按新的随机顺序排列（3,750 次更新）。该数据流包含 200 个任务，长版本则包含 800 个；网络的初始宽度为每层 100 个单元（第二组实验为 300 个），而 B 是初始宽度。评分指标为在线准确率：每个批次在用于训练前都会先进行预测。主要指标是最后十分之一任务（任务 181–200）的平均在线准确率。每个任务都会重新连接输入，因此该数据流测试网络能否持续学习；保留已学内容不计分。

混合流。该数据流包含 2,000 个任务。每个任务有 24,000 张训练图像（1,500 次更新），涵盖十个类别；每个任务都会重新将这十个类别分配给十个输出单元。主类别包括手写数字（MNIST）、手写字母（EMNIST Balanced 中 37 个字母类别里的十个 [7]）以及 Kuzushiji 字符（KMNIST [6]）；相邻的两个主任务类别绝不会相同。两种无关类别各出现一次，之后不再出现：服装（Fashion-MNIST [50]）出现在任务 401–600 中；而第二个传感器（对主任务图像应用同一种固定像素置换）出现在任务 1,201–1,400 中。在出现期间，每个任务有一半概率属于该类别。任务得分为其区块得分：在该任务训练 6,000、12,000、18,000 和 24,000 张图像后，分别测量 4,000 张留出测试图像（每类 400 张）的准确率，并取均值。主要指标是最后十分之一主任务的平均区块分数。网络初始每层有 100 个单元，且 $B = 100$。

混合流是在仅使用普通网络的情况下，从六个候选日程中依据预先确定的规则选出的，例如普通网络沿数据流的表现至少下降一个百分点。该选择是在 Adam 下确定的；在随机梯度下降下，没有候选方案满足该规则，第 5 节将再讨论这一点。

部署数据流。 任务的格式与混合流相同。在通用阶段 （任务1–500）中，每个任务都以相同概率从五类中抽取：数字、字母、Kuzushiji、服装和第二个传感器。在部署阶段 （任务501–2,000）中，内容分为三组。数字和字母是常见 内容。Kuzushiji是稀有 内容：每20个任务组成的区块中恰好有一个，位置随机。服装和第二个传感器已经消失：此后不再出现。一个包含三个隐藏层、每层300个单元的网络，作为普通网络在整个通用阶段进行训练。在任务500结束时，该网络启用增长路径，并设置

预算 B 介于25和100之间（以及 $B = 300$，即未缩减预算的情况），且所有单元均已建立（第2.3节）。基线是使用该方法训练的同一网络。

B.5 指标与判定规则

在从头开始训练的数据流上，如果该方法的主要指标高于两个同等规模的普通网络，且两者的 95% 置信区间都高于零，则该方法通过。每个准确率旁边都会报告第 2.5 节中的计算成本，即对数据流的所有更新取平均（训练成本），以及对评估窗口取平均（部署成本）。

对于部署，我们使用四个数值。*常见内容准确率* 是部署阶段最后十分之一的常见任务（142 个任务）的平均区块得分。*后期稀有内容准确率* 是第 1,501–2,000 个任务中 25 个 Kuzushiji 任务的平均区块得分。*计算成本* 和*存储成本* 均按常见内容窗口中的更新取平均。判定规则如下：

（a）*常见内容不劣*：与普通网络的差值区间下限高于 -0.5 个百分点；（b）*更小*：计算成本区间的上限至多为普通网络成本的 75%，即以 100宽网络为单位时至多为 3.1525；（c）*保留稀有内容*：与普通网络的后期稀有内容差值区间下限高于 -1.0 个百分点。

规则 (a) 和 (b) 在此数据流上运行该方法之前就已确定；对含有死亡的规则进行评估，并在其旁报告该规则的稀有内容准确率。规则 (c) 在首次获得死亡相关结果之后、运行休眠之前加入；休眠则依据全部三条规则进行评估。

C 补充表格

网络	计算量 (%)	常见内容 (点)	后期稀有内容 (点)	规则 (a)	(b)	(c)
Adam						
普通网络	100	93.92	72.41			
增长路径, $B = 100$	60	$+1.16$ [0.76, 1.55]	$+4.3$ [1.1, 7.5]	✓	✓	✓
增长路径, $B = 80$	43	$+1.14$ [0.84, 1.44]	$+3.6$ [0.2, 7.0]	✓	✓	✓
增长路径, $B = 65$	33	$+1.13$ [0.78, 1.48]	$+2.7$ [−0.2, 5.7]	✓	✓	✓
增长路径, $B = 50$	24	$+1.01$ [0.69, 1.33]	$+0.8$ [−2.7, 4.3]	✓	✓	×
增长路径, $B = 35$	17	$+0.81$ [0.44, 1.18]	-0.9 [−4.5, 2.6]	✓	✓	×
增长路径, $B = 25$	12	$+0.54$ [0.21, 0.86]	-3.5 [−7.0, −0.1]	✓	✓	×
含休眠, $B = 100$	71	$+1.20$ [0.87, 1.53]	$+8.1$ [5.1, 11.2]	✓	✓	✓
含休眠, $B = 80$	53	$+1.18$ [0.85, 1.51]	$+7.6$ [3.9, 11.2]	✓	✓	✓
含休眠, $B = 65$	41	$+1.11$ [0.71, 1.50]	$+6.7$ [3.8, 9.6]	✓	✓	✓
含休眠, $B = 50$	31	$+1.04$ [0.63, 1.44]	$+5.4$ [2.2, 8.5]	✓	✓	✓
带有休眠, $B = 35$	22	$+0.84$ [0.55, 1.13]	$+3.8$ [1.3, 6.4]	✓	✓	✓
带有休眠, $B = 25$	16	$+0.62$ [0.28, 0.97]	$+1.1$ [−1.5, 3.7]	✓	✓	×
SGD						
普通网络	100	96.15	87.84			
增长路径, $B = 100$	77	$+0.31$ [0.16, 0.45]	-1.5 [−2.0, −1.0]	✓	×	×
增长路径, $B = 80$	57	$+0.20$ [0.11, 0.29]	-2.4 [−2.9, −1.9]	✓	✓	×
增长路径, $B = 65$	42	$+0.08$ [−0.06, 0.21]	-3.5 [−3.9, −3.0]	✓	✓	×
增长路径, $B = 50$	31	-0.05 [−0.17, 0.07]	-5.0 [−5.5, −4.4]	✓	✓	×
增长路径, $B = 35$	21	-0.40 [−0.51, −0.28]	-7.4 [−8.2, −6.5]	×	✓	×
增长路径, $B = 25$	15	-0.85 [−0.92, −0.78]	-11.7 [−12.8, −10.5]	×	✓	×
含休眠状态, $B = 100$	85	$+0.31$ [0.20, 0.42]	-0.3 [−0.7, 0.1]	✓	×	✓
含休眠状态, $B = 80$	64	$+0.22$ [0.13, 0.30]	-0.9 [−1.6, −0.2]	✓	✓	×
含休眠状态, $B = 65$	51	$+0.11$ [0.03, 0.18]	-1.6 [−2.2, −1.0]	✓	✓	×
具有休眠, $B = 50$	39	$+0.04$ [−0.09, 0.17]	-2.8 [−3.4, −2.2]	✓	✓	×
具有休眠, $B = 35$	27	-0.17 [−0.31, −0.04]	-5.5 [−5.8, −5.1]	✓	✓	×
具有休眠, $B = 25$	19	-0.62 [−0.71, −0.53]	-8.5 [−9.1, −8.0]	×	✓	×

表 3： 部署，全部六种预算（种子 15–19；列和规则同表 1).