数据感知神经架构搜索

Emil Njor
emjn@dtu.dk 丹麦技术大学
孔恩斯灵比，丹麦

Jan Madsen
jama@dtu.dk 丹麦技术大学
孔恩斯灵比，丹麦

Xenofon Fafoutis
xefa@dtu.dk 丹麦技术大学 丹麦孔恩斯灵比

摘要

神经架构搜索（NAS）是一种用于自动生成神经网络（NN）架构的流行工具。在早期的神经架构搜索研究中，这些工具通常针对单一指标（例如准确率）对神经网络架构进行优化。然而，在资源受限的机器学习场景中，单一指标不足以评估神经网络架构。例如，如果一个神经网络模型无法装入特定系统的闪存中，那么即使它具有很高的准确率，也毫无实用价值。因此，近期针对资源受限系统的神经架构搜索研究探索了多种针对多个指标进行优化的方法。在本文中，我们提出，除了这些方法之外，在针对资源受限系统进行神经架构搜索优化时，考虑输入数据粒度也可能带来益处。我们将这种系统命名为“数据感知神经架构搜索”，并通过将其与传统 NAS 进行比较，提供了证明其优势的实验依据。

关键词

神经架构搜索、资源受限机器学习、微型机器学习、卷积神经网络、数据粒度

ACM 引用格式：

Emil Njor、Jan Madsen 和 Xenofon Fafoutis。2023年。数据感知神经架构搜索。载于《*tinyML 研究研讨会论文集（tinyML 研究研讨会'23）*》。ACM，纽约州，美国，7页。

1 引言

针对给定问题，手动设计一个性能良好的机器学习（ML）系统是一项艰巨的任务。机器学习工程师必须依靠知识和经验，组建他们认为能够良好运行的系统。之后，必须通过实验测试所设计系统的实际性能。这些实验可能耗时较长，因此，神经网络（NN）系统的手动设计过程通常需要投入大量工程工作。

另一种替代方案是自动化机器学习的概念。自动化机器学习旨在自动完成机器学习系统的设计步骤。自动化机器学习有望提升生产力，并让非机器学习工程师也能够设计机器学习系统。神经架构搜索（NAS）是研究最深入的自动化机器学习子领域之一。在神经架构搜索中，计算机程序会自动搜索合适的神经网络架构。

设计性能良好的资源受限机器学习系统（例如微型机器学习系统）更具挑战性。微型机器学习指的是在超低功耗系统中运行机器学习的概念。在这些功耗限制下，硬件能力极其有限。例如，最常用的微型机器学习设备之一 Arduino Nano 33 BLE Sense，其处理器频率仅为 64 MHz，随机存取存储器仅为 256 KB。此类资源受限的机器学习系统无法针对单一指标进行优化。例如，如果神经网络模型不符合资源受限设备的约束，那么模型的高准确率便毫无意义。因此，针对资源受限机器学习提出的自动化机器学习方法通常会尝试以各种方式优化多个指标。

本文研究了面向资源受限机器学习的神经架构搜索方法。具体而言，我们认为，对于资源受限系统，神经架构搜索方法若能同时搜索合适的数据粒度，将会有所助益。“数据粒度”指的是，数据可以以不同的粒度级别输入机器学习模型。例如，音频样本可以按不同采样率输入机器学习模型。在这种情况下，我们会认为，以 24 kHz 采样率提供的音频样本，其数据粒度高于以 12 kHz 采样率提供的同一音频样本。同样，图像可以按不同分辨率输入机器学习模型。传感器的数量和类型（例如单声道与立体声音频）也是数据粒度的另一个例子。

我们的一个主要见解是，神经网络模型并非整个机器学习系统中唯一消耗资源受限设备资源的部分。此外，提供给模型的数据本身也可能消耗大量本已有限的资源。这些数据的资源消耗可以通过其粒度进行调节。同样，模型架构和数据粒度都会对机器学习系统的性能产生显著影响。因此，模型和数据都构成了可供探索的搜索空间，从而能够针对给定问题找到性能最佳的资源受限机器学习系统。例如，如果降低数据粒度，从而为更复杂的神经网络架构留出更多可用资源，就可能获得性能更优的机器学习系统。

因此，为了对此进行研究，我们提出了数据感知神经架构搜索（Data Aware NAS）：一种面向资源受限机器学习的扩展型神经架构搜索方法，该方法考虑统一的模型与数据粒度搜索空间，以便为资源受限设备生成近乎最优的机器学习系统。

2 数据感知神经架构搜索

在本节中，我们将进一步探讨数据感知神经架构搜索的理念。我们首先讨论神经架构搜索中的单目标优化和多目标优化。然后，我们将阐述为何将数据粒度纳入神经架构搜索的搜索空间能够提升系统性能。

2.1 单目标与多目标优化

要使用神经架构搜索，必须将“什么样的模型才算好”形式化为一项指标。无论是在资源受限还是非资源受限的神经架构搜索场景中，情况都是如此。这项指标将是神经架构搜索要优化的

在不收取费用的前提下，允许为个人或课堂使用而制作本作品部分或全部内容的数字副本或纸质副本，但不得为营利或商业利益而制作或分发副本，且副本首页须附上本声明及完整引文。本作品中第三方组件的版权必须得到尊重。其他用途请联系所有者/作者。*tinyML 研究研讨会'23，2023年3月*，加利福尼亚州伯林盖姆© 2023年版权归所有者/作者所有。

神经网络模型的目标。传统上，这通常是一项单一指标，例如在能够代表使用场景 [8] 的数据集上的准确率。这称为单目标神经架构搜索。在单目标神经架构搜索中，目标很简单：返回能够优化这项单一指标的神经网络模型。

近期关于神经架构搜索的研究开始探讨如何在优化多个指标时进行神经架构搜索。这称为多目标优化神经架构搜索 [3]。与单目标优化相比，多目标优化更具挑战性，因为优化目标分散在多个指标上。这不仅引出了如何在内部比较神经网络模型的问题，也涉及多目标优化神经架构搜索应输出哪些模型作为结果。多目标优化神经架构搜索在内部比较模型、以决定继续处理哪些模型的一种方法，是通过对各项指标进行加权求和，创建一个综合指标 [14]。另一种方法是根据解决方案是否位于帕累托前沿来评估它们 [3]。无论多目标优化神经架构搜索内部采用何种比较标准，通常都会将帕累托前沿作为最终结果输出。返回一组帕累托前沿上的解决方案，可以根据具体使用场景评估应优先考虑多项优化指标中的哪种权重。在资源受限机器学习场景中，多目标优化通常优于单目标优化。这是因为，为资源受限设备生成机器学习模型时，多个指标之间的权衡不可避免。

2.2 数据感知神经架构搜索的优势

在设计资源受限机器学习系统时，充分利用可用资源至关重要。根据设备的不同，设备上的可用资源可能涉及多个方面。不过，对于资源受限机器学习，我们通常会关注一些特定资源，例如内存消耗、能耗以及模型运行推理所需的时间。本文将聚焦于内存消耗，因为它通常可以很好地反映其他资源的使用情况。例如，占用内存较少的模型平均而言能耗更低，完成推理所需的时间也更短。同时，对于各种设备，内存消耗也容易评估。

在基于神经网络的资源受限机器学习中，通常有两类数据占用了大部分内存。一类是神经网络模型，可以在多目标优化神经架构搜索中进行优化。另一类是作为该模型输入的数据。以音频数据为例。通常，麦克风传感器会以一定的采样率生成这些数据。随后，数据会被发送到机器学习设备，并保存在其内存中。此时，常见的做法是对样本进行预处理，例如使用短时傅里叶变换（STFT）将其转换为频谱图，然后将其输入机器学习模型。这样的预处理步骤需要一定数量的样本，这一数量对应于机器学习模型单次运行所需分析的时间窗。因此，在最佳情况下，该缓冲区的最小大小在某个时刻为：

$$Size = Sample_{size}\text{（位）} * Sample_Rate\text{（赫兹）} * Window\text{（秒）}$$

我们来看一个示例场景：每个样本的位宽为 8 位，采样率为 6kHz，采集时长为 5 秒。在这种情况下，消耗的内存约为 30 kB——对于小型微控制器而言，这可能是相当可观的。此外，数据和神经网络模型都可能对整个机器学习系统的性能产生重大影响。例如，无论输入什么数据，小型神经网络模型都可能无法生成准确的预测结果。在同

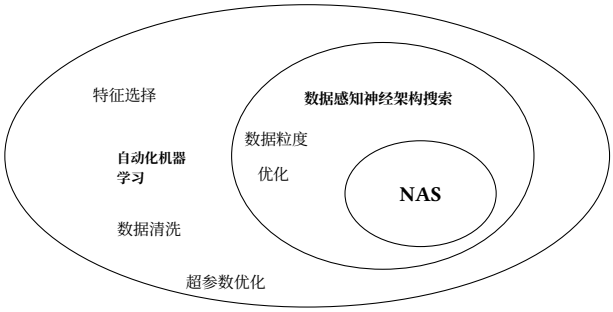

图 1：自动化机器学习、数据感知神经架构搜索与神经架构搜索之间关系的直观表示。

样的情况下，即使是大型神经网络模型，在数据粒度极低时也可能无法生成准确的预测结果。因此，在资源受限的机器学习系统中，平衡数据和模型所消耗的资源与带来的性能提升非常重要。这正是我们提出数据感知神经架构搜索的目标。图 1 概述了数据感知神经架构搜索与自动化机器学习和神经架构搜索的对比。

值得注意的是，一篇发表于2016年的论文研究了加速度计数据集，发现文献中使用的采样率最高比实际所需高出57% [12]。这些结果可能适用于资源受限机器学习领域，以及其他类型的数据和数据参数。如果确实如此，降低数据粒度或许能显著提升资源受限机器学习系统的性能。

最后，值得注意的是，数据粒度与神经网络模型大小之间存在有趣的相互作用。对于典型的神经网络，输入层的大小与输入数据的大小相同。因此，降低数据粒度会减小模型大小。对于某些专用神经网络模型，这种影响更为显著。例如，对于自编码器，其输入层和输出层的大小都与输入数据的大小相同。

3 相关工作

本文的相关工作可分为三类：

资源受限神经架构搜索： 与在给定资源约束下使用神经架构搜索寻找优质神经网络架构相关的工作。**资源受限自动化机器学习：** 与创建自动化流程以构建资源受限机器学习系统相关的工作。**手动数据粒度优化：** 作者手动对数据粒度进行优化的相关工作。

3.1 资源受限神经架构搜索

资源受限神经架构搜索领域最早的研究之一由 Elsken 等人开展，[7], 他们提出了一种使用进化算法的多目标优化神经架构搜索。其方法的输出是针对多个参数（例如参数数量或延迟）的可能配置所构成的帕累托前沿 [7]。大约在同一时期，Hsu

等人提出了一种基于强化学习的多目标优化神经架构搜索方法，同时优化准确率和乘加运算次数 [11]。

Lu 等人提出的神经架构搜索系统对分类错误率和浮点运算次数进行多目标优化 [15]。Fedorov 等人的论文结合神经架构搜索与神经网络剪枝，设计出体积小但性能出色的模型 [9]。Wang 等人 [20] 针对资源受限机器学习，研究了扩展形式的神经架构搜索。他们扩展后的神经架构搜索同时考虑神经架构、量化和剪枝策略。为缩短搜索时间，他们使用预测器来预测模型准确率，而不是针对所有组合重新训练模型。Cassimon 等人 [4] 提出了一种基于强化学习的多目标优化神经架构搜索。在这项工作中，作者将硬约束和软约束加入强化学习模型的奖励函数，以纳入多个目标。硬约束的一个例子是模型应能在可用的随机存取存储器（RAM）容量内运行。软约束的一个例子是神经网络的准确率。Bakhtiarifad 等人的一项研究描述了一种基于表格型基准测试的神经架构搜索系统，该系统能够感知模型的能耗 [3]。

尽管这些工作都提出了有用的资源受限神经架构搜索变体，但没有一项考虑到本文提出的数据粒度优化。

3.2 资源受限自动化机器学习

Edge Impulse 是微型机器学习领域一家知名的公司，已发布一款名为 EON Tuner 的自动化机器学习工具。EON Tuner 会探索输入粒度和神经网络架构的不同组合 [2]。由于其专有性质，我们无法得知 EON Tuner 如何提出解决方案。不过，根据我们对该软件的使用体验和了解，我们认为它会在初始化阶段生成一些组合，并根据若干指标评估这些架构。因此，该软件似乎没有利用性的一面，不会尝试改进最初表现良好的架构。

Doyo 等人最近提出了一种名为微型机器学习即服务（tinyMLaaS）的范式 [6]。借此，他们旨在应对异构性问题：多个机器学习框架需要与多个硬件制造商协同工作，而每个制造商都有各自用于部署微型机器学习模型的工具。他们提议创建一个平台，为所有可用的框架和硬件组合提供机器学习编译器。这样一来，该平台便可为任意一组异构设备生成微型机器学习系统。此外，该平台还应能够利用空中软件更新（SOTA）技术，将这些系统部署到异构设备上。我们设想，本文提出的数据感知神经架构搜索系统可用于此类系统。

3.3 手动数据粒度优化

在我们此前的论文中，我们研究了如何在预测性维护场景中，手动降低微型机器学习应用的数据粒度。这项研究表明，在保持模型性能并改善推理时间 [17] 的同时，降低采样率仍有很大的空间。类似地，Zalewski 等人发表的一篇论文 [21] 描述了如何手动设计一个利用不同数据粒度的微型机器学习系统。该架构首先使用低数据粒度模型进行分类，随后使用一个更高数据粒度的模型。

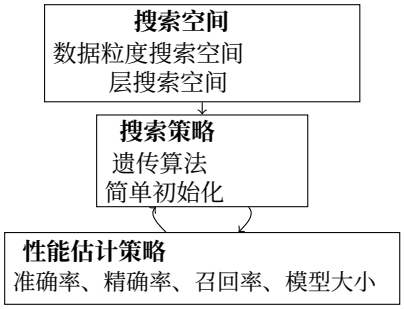

图 2：我们的系统设计概览。

如果较低数据粒度的模型对其结果缺乏信心，则会触发该机制 [21]。尽管这些论文证明了数据粒度优化的应用价值，但并未提出将这一概念自动化的方法。

4 系统设计

本节将介绍数据感知神经架构搜索原型的系统设计。该设计的总体理念并不是尝试创建一个最优的数据感知神经架构搜索系统，而是努力提供一个简单的概念验证系统。因此，我们通常倾向于采用简单的设计，而不是可能带来更高性能的复杂设计。此外，请注意，我们设计的系统用于生成卷积神经网络（卷积神经网络）。这是因为卷积神经网络是大多数神经架构搜索研究的目标 [14]。图 2 展示了我们系统设计的快速概览。

NAS 系统主要由三个组件构成：搜索空间、搜索策略和性能估计策略 [8]。

在我们的系统中，我们将搜索空间划分为数据粒度搜索空间和 CNN 中卷积层的搜索空间。数据粒度搜索空间在很大程度上取决于系统用例。我们的系统设计通过使搜索空间可配置来考虑这一点。默认情况下，我们假设数据为音频。由于原型系统的目的仅仅是证明数据感知神经架构搜索的可行性，因此我们将搜索空间限制为包含采样率和预处理类型的取值。层搜索空间对使用场景的依赖较小，但仍可进行配置。默认情况下，我们选择将卷积层中的滤波器数量、滤波器大小以及该层使用的激活函数纳入其中。

对于搜索策略，我们使用遗传算法——这是神经架构搜索中常见的选择 [8]。我们的系统支持两种算法初始化方式。默认情况下，我们会生成由平凡解组成的初始种群；根据 [19] ，这种方式在其他基于遗传算法的神经架构搜索中表现良好。或者，我们也可以生成随机初始种群。随后，我们的搜索策略会使用下一段所述的性能估计策略评估该种群。基于锦标赛选择 [14], ，我们从该种群中选出一些适应度最高的个体，作为下一代种群的基线。最后，我们对这些基线应用各种交叉算子和变异算子，以生成下一代种群。请注意，这些算子需要根据配置的搜索空间进行实现。

表 1：默认搜索空间参数及其变异算子。

搜索空间参数	可能值	变异算子
采样率（SR）	48 000 Hz、24 000 Hz、12 000 Hz、6000 Hz、3000 Hz、1500 Hz、750 Hz、375 Hz	将数值增加或减少一档。
预处理类型（PT）	频谱图（SP）、梅尔频谱图（MS）、梅尔频率倒谱（MFCC）	更改为另一个值。
层数 (L)	1,2,3,4,5	增加或减少一个值。
滤波器数量 (F)（每层）	2, 4, 8, 16, 32, 64, 128	增加或减少一个值。
滤波器大小 (FS)（每层）	3, 5	增加或减少一个值。
激活函数（AF）（每层）	修正线性单元 (R)、Sigmoid 函数 (S)	更改为其他值。

我们设计的默认交叉算子会从基线中选择两个个体，并从二者中随机选取值来创建一个新个体。我们针对默认搜索空间设计了若干变异算子。概览请参见表 1。

由于我们正在为资源受限机器学习设计一种数据感知神经架构搜索，因此需要在性能估计策略中采用多个指标。在这一概念验证设计中，我们选择优化准确率（Acc）、精确率（Pre）、召回率（Rec）和模型大小（MS）。我们选择使用这些指标，是因为准确率、精确率和召回率都是相对容易评估的指标，能够很好地反映系统的预测性能。模型大小是另一个易于评估的资源约束指标，也可以作为其他资源约束的合理替代指标。

这些指标的评估方式并不相同。因此，我们的性能估计策略由两个子系统组成。第一个子系统通过对给定的卷积神经网络进行可配置数量的训练轮次，来评估准确率、精确率和召回率指标。资源受限机器学习中使用的模型相对较小，因此我们不会遭遇极长的模型训练时间；这也促使其他神经架构搜索系统间接评估这些指标。第二个子系统通过将模型导出为二进制格式，并记录模型占用的字节数，来评估模型大小。我们将这些指标相加，以比较两个机器学习系统。为确保没有任何一个指标主导其他指标，我们会先将这些指标归一化到 0 到 1 的范围内，再进行求和。准确率、精确率和召回率本身就在这个范围内。另一方面，模型大小是一个没有上限的正整数。因此，我们使用以下公式将模型大小转换为 0 到 1 之间的值：

$$e^{-\frac{Model_{size}}{Approximate_Model_Size_Range}}$$

作为最终结果，我们输出生成的机器学习系统的帕累托前沿。

5 实验设置

本节介绍了用于对第 4 节所述系统进行实验的环境配置。我们使用 Python 3.10 编程语言实现了该系统。有关托管此实现代码的 GitHub 代码仓库链接，请参见附录 A。Python 使我们能够与常用的机器学习和神经网络库进行交互。在这些库中，我们使用 TensorFlow (TF) [5] 库来实现卷积神经网络架构，并使用 TensorFlow Lite (TFL) 库来计算这些架构的模型大小。此外，我们还使用了常用的 Python 库，例如用于高效数据结构的 NumPy [10] 、用于.....的 librosa [16] 。

表 2：我们的实验中使用的可配置参数值（以自然语言描述）。

参数	值
输出类别数量	2
损失函数	分类交叉熵
全连接层宽度	10
正常文件数量	900
异常文件数量	200
帧大小（STFT）	2048
跳跃长度（STFT）	512
梅尔滤波器组数量	80
梅尔频率倒谱系数数量	13
最大层数	5
优化器	Adam
每个模型的训练轮次数	20
批量大小	32
近似模型大小范围	100000
种群规模	10
每代更新的种群比例	0.5
交叉与变异的比例	0.2

音频预处理、joblib（便于实现并行化），以及用于评估准确率、精确率和召回率的 scikit-learn [18] 。

在实验中，我们使用了 ToyADMOS 数据集的一个子集 [13]。ToyADMOS 数据集包含玩具在正常和异常运行条件下的音频录音。所用子集包含玩具传送带第一个案例的音频通道一的录音。异常运行条件的例子之一是传送带上附着了一个金属物体。我们的数据感知神经架构搜索所生成的机器学习系统旨在学习将未见过的音频录音分类为正常或异常。我们还将数据集中包含的噪声录音与正常和异常音频录音混合，以增加分类难度。

为运行实验，我们使用丹麦技术大学提供的高性能计算集群。我们使用的计算节点配备 XeonGold6226R 或 XeonGold6326 服务器 CPU，并分别配备 40 GB 和 80 GB 的 Nvidia A100 GPU。其他配置选项的参数信息请参见表 2。

在此配置下，我们进行两组实验。每组实验运行五次，以确保结果不受特定随机种子的影响。每次运行实验时，我们都会为实现中使用的随机数生成器指定一个随机种子。该随机数生成器会选择例如

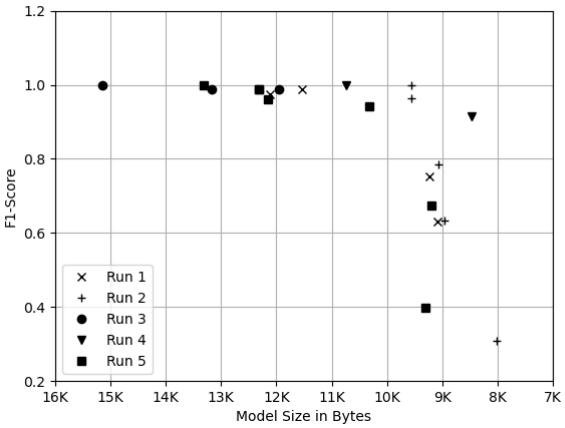

图 3：数据感知实验的帕累托前沿。

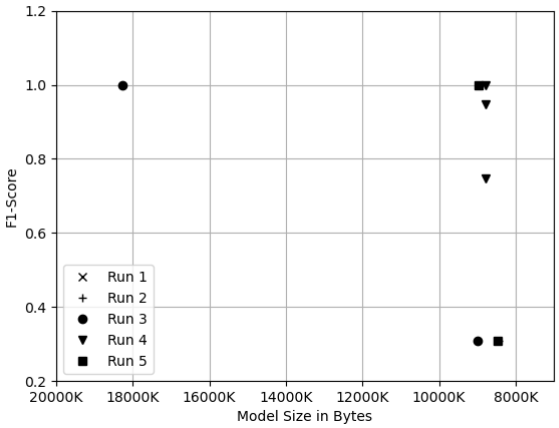

图 4：固定数据实验的帕累托前沿。

在种群更新过程中要应用的变异算子类型。我们将每次实验运行配置为生成并评估 300 种数据粒度与模型架构的组合。

在第一组实验中，我们使用表 1 列出的所有可能值构成的完整搜索空间运行我们的实现。我们将这组实验称为"数据感知"实验。本实验旨在为我们的数据感知神经架构搜索实现所能生成的资源受限机器学习系统建立一个基线。

在第二组实验中，我们将数据粒度固定为一种静态配置。对于该静态配置，我们选择 6千赫兹的采样率，并采用能够生成频谱图的数据预处理步骤。我们将这组实验称为"固定数据"实验。通过将本实验生成的机器学习系统与数据感知实验生成的系统进行比较，我们可以判断数据感知神经架构搜索能否生成优于同等神经架构搜索的机器学习系统。我们认为，6千赫兹的采样率和频谱图数据预处理步骤，很可能是非数据感知神经架构搜索系统也会采用的配置。这是因为 6千赫兹的采样率能在内存消耗（见第 2 节）和数据质量 [17], 之间取得平衡，而且频谱图预处理是音频数据的一种常见预处理步骤，适用于非人耳可听的音频。请注意，由于硬件限制，我们无法在本实验中生成滤波器数量为 128 的模型，因为采用此配置时，模型大小对于我们的处理器而言过大。

6 结果

在本节中，我们展示了实验结果。实验中找到的机器学习系统的帕累托前沿见表 3。我们还在图 3 中绘制了每个数据感知实验的帕累托前沿上的模型，并在图 4 中绘制了每个固定数据实验的帕累托前沿上的模型。

从这些结果可以清楚地看出，数据感知实验和固定数据实验都找到了准确的机器学习系统。不过，我们的数据感知实验找到的机器学习系统，其模型大小比固定数据实验找到的机器学习系统小约三个数量级。由于模型大小这一指标并未计入机器学习系统输入数据缓冲区的内存消耗，因此这两个机器学习系统的实际内存消耗差距会更大。这表明，我们关于使用数据感知神经架构搜索而非常规神经架构搜索，能够找到更好的资源受限模型这一假设是正确的。

令人惊讶的是，我们的数据感知神经架构搜索找到的机器学习系统，其采样率为 375 Hz 或 750 Hz。这些数值低于通过简单手动搜索发现的理想采样率 [17]。同样，对于这些机器学习系统而言，梅尔频率倒谱系数似乎是最佳的预处理方式。鉴于梅尔频率倒谱系数最初旨在帮助处理人类语音音频，而非工业机械音频，这一点也令人惊讶。因此，我们也认为，工程师若仅靠手动设计，难以得到与数据感知神经架构搜索同样准确且资源高效的机器学习系统。不过，这些令人鼓舞的结果，部分可能归因于默认系统设计中配置的搜索空间。考虑到我们为固定数据实验选择的数据粒度，该搜索空间可能不允许常规神经架构搜索生成较小的模型。此外，ToyADMOS 数据集似乎包含大量冗余数据，因为即使将采样率降至 375 Hz，仍能保持完美的准确率。因此，该数据集是数据感知神经架构搜索的理想案例，而模型大小的差异在其他数据集中可能不会这么大。

7 结论与未来工作

本文介绍了数据感知神经架构搜索这一构想——它是神经架构搜索的扩展形式，除了搜索神经网络模型的最优架构外，还会为生成的神经网络模型搜索最优数据粒度。我们认为，在设计受资源约束的机器学习系统时，这类系统尤其有用。我们设计并提供了数据感知神经架构搜索的概念验证实现，并开展实验，以实验性证据验证这一构想。结果证实了我们的假设：考虑数据粒度后，我们能够在资源约束下创建更好的模型。

本文显而易见的扩展方向是扩展当前系统，纳入与资源消耗相关的其他指标。

表 3：我们在实验中发现的机器学习系统的帕累托前沿。我们通过组件名称加层编号来描述神经网络架构的各个部分。例如，FS3 表示第 3 个卷积层的滤波器大小。帕累托前沿上的机器学习系统均未使用可能存在的第 5 个神经网络层。

实验	Run	SR	PT	F1	FS1	AF1	F2	FS2	AF2	F3	FS3	AF3	F4	FS4	AF4	Acc	Pre	Rec	MS
数据感知	1	375	MFCC	2	3	R	2	5	S	-	-	-	-	-	-	90%	71%	80%	9240 字节
数据感知	1	750	MFCC	2	5	S	2	5	R	2	5	R	-	-	-	100%	98%	100%	11 536 B
数据感知	1	375	MFCC	2	3	R	2	5	R	-	-	-	-	-	-	84%	55%	75%	9088 B
数据感知	1	750	MFCC	2	3	S	2	5	R	2	5	R	-	-	-	100%	100%	100%	16 528 B
数据感知	1	750	MFCC	4	5	S	2	5	R	2	5	R	-	-	-	99%	100%	95%	12 108 B
数据感知	2	375	MFCC	2	5	R	2	3	S	-	-	-	-	-	-	92%	79%	78%	9064 B
数据感知	2	375	MFCC	2	5	R	2	3	R	-	-	-	-	-	-	83%	52%	83%	8960 B
数据感知	2	750	MFCC	8	5	R	2	5	S	2	5	R	2	3	R	100%	100%	100%	9556 B
数据感知	2	750	MFCC	8	5	S	2	5	R	2	5	R	2	3	R	99%	93%	100%	9552 B
数据感知	2	750	MFCC	4	5	S	2	5	S	2	5	R	2	3	R	18%	18%	100%	8012 B
数据感知	3	750	MFCC	8	5	R	2	5	R	2	5	R	-	-	-	100%	100%	98%	13 172 B
数据感知	3	750	MFCC	8	5	R	4	5	R	2	5	R	-	-	-	100%	100%	100%	15 148 B
数据感知	3	750	MFCC	4	5	R	2	5	R	2	5	R	-	-	-	100%	98%	100%	11 956 B
数据感知	4	750	MFCC	4	5	R	2	5	R	2	3	R	4	5	S	100%	100%	100%	10 744 B
数据感知	4	750	MFCC	4	5	R	2	5	S	2	3	R	2	5	S	97%	90%	93%	8464 B
数据感知	5	375	MFCC	2	3	R	2	5	S	-	-	-	-	-	-	86%	60%	78%	9192 B
数据感知	5	375	MFCC	2	3	S	2	5	S	-	-	-	-	-	-	50%	26%	90%	9304 B
数据感知	5	750	MFCC	8	5	S	2	5	S	4	5	R	2	3	S	98%	89%	100%	10 324 B
数据感知	5	750	MFCC	2	5	R	4	5	R	2	5	S	-	-	-	100%	98%	100%	12 308 B
数据感知	5	750	MFCC	2	5	R	4	5	R	2	5	S	-	-	-	100%	100%	98%	12 308 B
数据感知	5	750	MFCC	2	5	R	4	5	R	2	5	R	-	-	-	99%	97%	95%	12 156 B
数据感知	5	750	MFCC	4	5	R	4	5	R	2	5	S	-	-	-	100%	100%	100%	13 316 B
固定数据	1	6000	SP	64	3	R	4	3	R	2	5	S	-	-	-	100%	100%	100%	8 966 120 B
固定数据	1	6000	SP	64	3	R	2	3	S	2	5	S	-	-	-	81%	0%	0%	8 961 264 B
固定数据	2	6000	SP	16	5	R	2	5	R	-	-	-	-	-	-	100%	100%	100%	8 957 760 B
固定数据	2	6000	SP	16	5	R	2	5	R	64	3	R	2	5	R	18%	18%	100%	8 438 432 B
固定数据	2	6000	SP	16	5	R	2	5	R	64	3	R	2	5	R	81%	0%	0%	8 438 432 B
固定数据	3	6000	SP	32	3	S	4	5	S	-	-	-	-	-	-	100%	100%	100%	18 278 100 B
固定数据	3	6000	SP	32	3	R	16	5	R	64	3	S	2	5	S	81%	0%	0%	8 697 324 B
固定数据	3	6000	SP	32	3	R	8	5	S	2	3	R	-	-	-	18%	18%	100%	8 981 060 B
固定数据	4	6000	SP	2	5	R	32	5	R	2	3	R	-	-	-	98%	100%	90%	8 782 444 B
固定数据	4	6000	SP	2	5	R	32	5	R	2	3	R	-	-	-	88%	60%	100%	8 782 444 B
固定数据	4	6000	SP	4	5	R	32	5	S	2	3	R	-	-	-	100%	100%	100%	8 789 220 B
固定数据	4	6000	SP	4	5	R	32	5	S	2	5	R	-	-	-	81%	0%	0%	8 613 956 B
固定数据	5	6000	SP	8	5	S	4	3	S	64	5	S	2	5	S	18%	18%	100%	8 456 952 B
固定数据	5	6000	SP	8	5	S	4	3	S	64	5	S	2	5	S	81%	0%	0%	8 456 952 B
固定数据	5	6000	SP	32	5	S	2	5	R	-	-	-	-	-	-	100%	100%	100%	8 962 744 B

例如，这些指标可以包括能耗、内存消耗或推理时间。我们预计，可以使用 CarbonTracker [1] 之类的工具估算能耗。内存消耗和推理时间的估算可以基于生成的表格型基准测试。不过，为了准确估算推理时间，必须针对每种设备单独生成表格型基准测试。

如第 6 节所述，由于 ToyADMOS 数据集存在冗余，它可能是数据感知神经架构搜索的理想案例。未来，可以研究数据感知神经架构搜索在其他冗余程度较低的数据集上的应用。同样，也可以研究扩大数据粒度和神经网络模型的搜索空间是否能构建出更好的机器学习系统。

此外，设计并实现一种更复杂但性能更好的数据感知神经架构搜索也可能很有意义。我们可以然后，将该系统生成的机器学习系统与使用其他面向资源受限系统的神经架构搜索解决方案生成的机器学习系统进行比较。例如，第 3 节中提出的一些系统。

将该系统纳入更大型的自动化机器学习流水线也可能很有意义，例如 tinyMLaaS [6] 中描述的系统。

A 资源

我们数据感知神经架构搜索系统的实现可在以下网址获取：https://github.com/Ekhao/DataAwareNeuralArchitectureSearch。

致谢

本研究由丹麦创新基金资助，资助项目为 DIREC（9142-00001B）。

参考文献

[1] Lasse F Wolff Anthony、Benjamin Kanding 和 Raghavendra Selvan。2020年。Carbontracker：追踪并预测深度学习模型训练的碳足迹。*arXiv* 预印本 *arXiv:2007.03051*（2020）。[2] Jenny Plunkett & Mathijs Baaijens。2022年。介绍 EON Tuner：Edge Impulse 面向嵌入式机器学习的新型自动化机器学习工具。https://www.edgeimpulse.com/blog/introducing-the-eon-tuner-edge-impulses-new-automl-tool-for-embedded-machine-learning [3] Pedram Bakhtiarifard、Christian Igel 和 Raghavendra Selvan。2022年。面向神经架构搜索的能耗感知表格基准测试。*arXiv* 预印本 *arXiv:2210.06015*（2022）。[4] Thomas Cassimon、Simon Vanneste、Stig Bosmans、Siegfried Mercelis 和 Peter Hellinckx。2020年。针对嵌入式设备，使用神经架构搜索设计资源受限神经网络。物联网 12（2020），100234。[5] TensorFlow 开发者。2022年。*TensorFlow*。https://doi.org/10.5281/zenodo. 6574269[6] Hiroshi Doyu、Roberto Morabito 和 Martina Brachmann。2021年。面向物联网机器学习的 tinyMLaaS 生态系统：概述与研究挑战。载于*2021*年超大规模集成电路设计、自动化与测试国际研讨会（*VLSI-DAT*）。电气电子工程师学会，1–5。[7] Thomas Elsken、Jan Hendrik Metzen 和 Frank Hutter。2018年。通过拉马克进化实现高效的多目标神经架构搜索。载于国际学习表征会议。[8] Thomas Elsken、Jan Hendrik Metzen 和 Frank Hutter。2019年。神经架构搜索：综述。机器学习研究期刊 20，1（2019），1997–2017。[9] Igor Fedorov、Ryan P Adams、Matthew Mattina 和 Paul Whatmough。2019年。Sparse：面向资源受限微控制器的卷积神经网络稀疏架构搜索。神经信息处理系统进展 32（2019）。[10] Charles R. Harris、K. Jarrod Millman、Stéfan J. van der Walt、Ralf Gommers、Pauli Virtanen、David Cournapeau、Eric Wieser、Julian Taylor、Sebastian Berg、Nathaniel J. Smith、Robert Kern、Matti Picus、Stephan Hoyer、Marten H. van Kerkwijk、Matthew Brett、Allan Haldane、Jaime Fernández del Río、Mark Wiebe、Pearu Peterson、Pierre Gérard-Marchant、Kevin Sheppard、Tyler Reddy、Warren Weckesser、Hameer Abbasi、Christoph Gohlke 和 Travis E. Oliphant。2020年。使用 NumPy 进行数组编程。《自然》585，7825（2020年9月），357–362。https: //doi.org/ 10 1038 41586 020 2649 2 。/s - - - Chi-Hung Hsu、Shu-Huan Chang、[11] Jhao-Hong Liang、Hsin-Ping Chou、Chun-Hao Liu、Shih-Chieh Chang、Jia-Yu Pan、Yu-Ting Chen、Wei Wei 和 Da-Cheng Juan。2018年。Monas：使用强化学习进行多目标神经架构搜索。*arXiv* 预印本 *arXiv:1806.10332*（2018）。[12] Aftab Khan、Nils Hammerla、Sebastian Mellor 和 Thomas Plötz。2016年。优化基于加速度计的人体活动识别采样率。《模式识别快报》73（2016），33–40。[13] Yuma Koizumi、Shoichiro Saito、Hisashi Uematsu、Noboru Harada 和 Keisuke I moto。2019年。ToyADMOS：用于异常声音检测的微型机器运行声音数据集。载于*2019*年*IEEE*音频与声学信号处理应用研讨会（*WASPAA*）。电气电子工程师学会，313–317。[14] Yuqiao Liu、Yanan Sun、Bing Xue、Mengjie Zhang、Gary G Yen 和 Kay Chen Tan。2021年。进化神经架构搜索综述。*IEEE*神经网络与学习系统汇刊（2021）。[15] Zhichao Lu、Ian Whalen、Vishnu Boddeti、Yashesh Dhebar、Kalyanmoy Deb、Erik Goodman 和 Wolfgang Banzhaf。2019年。NSGA-Net：使用多目标遗传算法进行神经架构搜索。载于遗传与进化计算会议论文集。419–427。[16] Brian McFee、Alexandros Metsai、Matt McVicar、Stefan Balke、Carl Thomé、Colin Raffel、Frank Zalkow、Ayoub Malek、Dana、Kyungyun Lee、Oriol Nieto、Dan Ellis、Jack Mason、Eric Battenberg、Scott Seyfarth、Ryuichi Yamamoto、viktorandree- vichmorozov、Keunwoo Choi、Josh Moore、Rachel Bittner、Shunsuke Hidaka、Ziyao Wei、nullmightybofo、Adam Weiss、Darío Hereñú、Fabian-Robert Stöter、Lorenz Nickel、Pius Friesch、Matt Vollrath 和 Taewoon Kim。2022年。*librosa/librosa*: *0 9 2* 10 5281 6759664 ..。https://doi.org/ . /zenodo。Emil [17] Njor、Jan Madsen 和 Xenofon Fafoutis。2022年。tinyML 预测性维护入门：输入与模型优化。载于*IFIP*人工智能应用与创新国际会议。Springer，67–78。[18] F. Pedregosa、G. Varoquaux、A. Gramfort、V. Michel、B. Thirion、O. Grisel、M. Blondel、P. Prettenhofer、R. Weiss、V. Dubourg、J. Vanderplas、A. Passos、D. Cournapeau、M. Brucher、M. Perrot 和 E. Duchesnay。2011年。Scikit-learn：Python 机器学习。机器学习研究期刊 12（2011），2825–2830。[19] Kenneth O Stanley、Jeff Clune、Joel Lehman 和 Risto Miikkulainen。2019年。通过神经进化设计神经网络。《自然·机器智能》1，1 2019 24 35( )，–。[20] Tianzhe Wang、Kuan Wang、Han Cai、Ji Lin、Zhijian Liu、Hanrui Wang、Yujun Lin 和 Song Han。2020年。APQ：联合搜索网络架构、剪枝和量化策略。载于*IEEE/CVF*计算机视觉与模式识别会议论文集。2078–2087。

[21] Pawel Zalewski、Letizia Marchegiani、Atis Elsts、Robert Piechocki、Ian Craddock 和 Xenofon Fafoutis。2020。《从数据比特到知识比特——可穿戴感知系统的板载分类框架》。*Sensors* 20, 6 (2020)。https://doi.org/10.3390/s20061655