基于神经架构搜索的多目标认知自动化系统

Eric Ke Wang, Ship Peng Xu, Chien-Ming Chen, and Neeraj Kumar

摘要–
***目前，*基于深度学习的工业信息学决策认知自动化是认知计算领域的新热点，其中多目标架构优化是该研究领域的一大难点。现有算法在面对多目标认知模型问题时，往往需要花费大量时间不断设置不同的搜索偏好参数来生成新的搜索过程。本文主要是为了解决多目标神经架构搜索过程中的问题，关键问题是如何在架构搜索过程中适应用户的偏好。我们提出了一种新的算法：线性-偏好共同进化算法。与原来的用户约束方法和帕累托主导的NSGA-II算法相比，我们的适应时间更快，适应质量更好。同时，在推理阶段，它能以相对较快的速度响应用户的需求。基于大量的对比测试结果，我们的算法在搜索质量上优于传统的多目标问题的认知自动化算法。**

***Index Terms-认知*自动化，进化算法，多目标，神经结构搜索（NAS），帕累托主导。**

I. 简介

COGNITIVE自动化源于模拟人脑的计算机系统的人工智能（intelligence）。它通过人与自然环境的互动和不断学习，帮助决策者从不同类型的海量数据中揭示出非凡的洞察力，从而实现不同程度的感知、记忆、学习和其他认知活动[1]。在大数据时代，数据的规模、类型、速度和复杂性远远超过了人脑的认知能力。如何有效实现对大数据的认知，也给传统的认知性计算。

2020年1月28日收到稿件；2020年6月4日修订；2020年6月7日接受。出版日期为2020年6月29日；当前版本日期为2021年6月16日。这项工作得到了国家重点研发计划的部分支持。本工作得到了中国国家重点研发计划2018YFB1003800和2018YFB1003805的支持，部分得到了广东省自然科学基金2016A030313660和2017A030313365的支持。以及部分由深圳市科技创新项目资助的JCYJ20160608161351559、KQJSCX70726103044992、JCYJ20170811155158682和JCYJ20170811155158682。JCYJ20160428092427867.*(通讯作者: Neeraj Kumar.)*

Eric Ke Wang和Ship Peng Xu在哈尔滨工业大学（深圳）计算机科学系，哈尔滨518055，中国（电子邮件：wk_hit@hit.edu.cn；19s051059@stu.hit.edu.cn）。

陈建明在山东科技大学，山东266510，中国（电子邮件：chienmingchen@ieee.org）。

Neeraj Kumar在台湾台中41354亚洲大学计算机科学与信息工程系，同时也在印度Patiala 147004的Thapar工程与技术学院（电子邮件：neeraj.kumar@thapar.edu）。

数字对象标识符10.1109/JSYST.2020.3002428

深度学习（DL）作为一种新的机器学习方法，已经成为大数据时代认知计算的一个优秀解决方案。通过建立一个基于表现形式的多层机器学习模型，DL对海量数据进行训练，学习有用的特征以提高识别、分类或预测的准确性[2]。一个优秀的DL认知模型将获得优秀的认知效果。这里最关键的问题是深度神经网络模型的选择。

随着各领域工业应用的快速发展，大量的数据被积累起来。处理和分析数据内容以获得有价值的信息已成为一项关键任务。越来越多的DL和神经网络模型已被应用于工业应用。然而，为工业应用设计和验证一个合适的、优秀的基于DL的认知自动化模型，往往需要大量的人力劳动，如选择算法、超级参数调整、迭代建模、模型评估等。因此，如果有一种方法能够自动找到当前问题的正确解决方案，可以有效地节省科学家的人工劳动，解放研究人员的创造力。神经结构搜索（NAS）是一个新的研究方向，也是自动化机器学习的热门话题之一[3], [4]。通过设计一种高性价比的搜索方法，可以自动获得具有较强泛化能力和硬件友好性的神经网络结构，节省了大量的人力成本。其部署和使用更适合于工业化的推广和应用。

NAS的设计是为了解决设计DL网络时严重依赖专家的深刻知识的问题，目的是在实现神经网络架构设计过程自动化的同时，达到有竞争力的性能。对于一些任务，如图像识别，NAS已经产生了非常有希望的结果。

NAS的一个非常重要的分支问题是如何有效地执行多目标NAS过程。由于历史原因，关于多目标NAS的相关工作很少。然而，许多神经网络的结构往往不仅需要高精确度，而且需要低资源消耗。因此，多目标NAS是一个需要研究的领域。相比之下，目前的许多算法往往需要用户事先设定准确度和资源消耗之间的权衡。因此，不同的搜索偏好参数将被设置为生成一个新的搜索过程以适应用户的新请求。由于用户的偏好漂移，以前关于多目标NAS的算法经常在适应用户的偏好上浪费大量的时间。

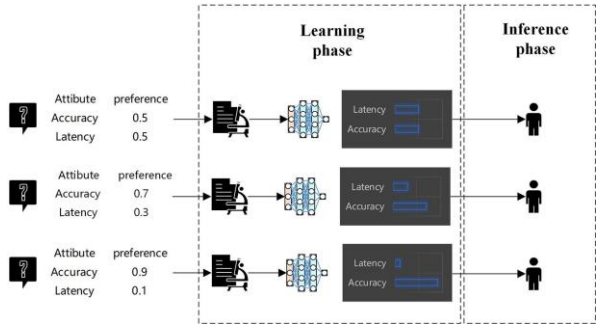

图1. 给定的偏好情景。

如图1所示，给定偏好后，平台首先获取用户的偏好，然后在用户的偏好条件下进行学习。在获得满足用户需求的网络结构后，网络被返回给用户。在这种情况下，每次推断用户的偏好都意味着一个耗时的学习过程。在单次推断的过程中，每个用户需要花费更多的资源。此外，面对需要快速部署的用户场景，服务器仍然需要很长的搜索时间，这往往不能满足用户快速部署的要求。

A. 贡献

为了解决多目标神经网络架构搜索过程中的关键问题，即搜索架构时如何自适应地设置用户偏好，我们提出了一种用户偏好延迟的方法。

1) 我们将传统的NAS过程分成两个相对独立的阶段：学习阶段和推理阶段。
2) 在学习阶段，我们使用具有偏好和最优个体的协同进化算法来探索满足用户偏好的解决方案集。
3) 在推理阶段，我们可以在短时间内使用这个解决方案集对用户的新偏好做出反应。
4) 主要目的是为了满足用户可以更新搜索偏好，而系统可以快速适应偏好的问题。在学习阶段

在本文中，我们使用平均适应性质量和平均适应性速度作为指标。本文的算法在这两个指标上都优于其他算法。

B. 组织机构

本文的其余部分组织如下。第二节介绍了相关工作。第三节从多目标优化、延迟偏好方案和线性-
偏好协同进化（LCNAS）算法方面描述了所提方案的工作。第四节强调了实验结果。最后，第五节是本文的结论。

II. 相关工作

我们的模型与这两个领域有关：多目标优化和NAS。在下面的相关工作中，我们主要从两个方面对其进行介绍。

A. 神经结构搜索

NAS可以分为以下几个子问题：搜索空间的设计，搜索策略的选择，以及搜索过程的加速。

搜索空间可分为链式搜索空间、多分支搜索空间[5]，[6]，基于基本单元构造的搜索空间[7]，以及搜索空间的分层表示法[8]。NAS的主要方法可以分为三类，使用强化学习和进化学习-
ing，以及其他算法方法。在强化学习方面，Zoph等人[7]将循环神经网络（RNN）作为控制器，将RNN产生的序列转化为模型，通过强化学习学习神经结构。在进化算法方面，Liu等人[9]和Real等人[10]、[11]为进化算法设计了一个搜索空间，并使用遗传算法在这个搜索空间中搜索神经架构。在其他算法中，Hutter等人[12]使用基于模型的顺序优化作为搜索算法，Liu等人[13]提出使用不同的NAS来发现硬件感知的高效卷积网络。当涉及到NAS过程的加速时，有一些方法被证明是有效的，如参数共享[14]，网络变形[15]，以及一次性架构搜索[16]。

B. 多目标NAS

目前，多目标NAS中常用的算法有两类：一类是约束条件下的搜索算法[17]，[18]，另一类是帕累托边界下的多目标搜索算法[19]-[24]。

在用户约束算法中，用户往往需要事先设定一系列的约束条件或偏好，然后将神经结构搜索过程构建为特定约束条件下的一次性搜索过程。这导致了两个主要问题：1）搜索过程只能找到一个满足单一约束条件或用户偏好的神经网络结构。当用户的约束条件或偏好发生变化时，需要对搜索过程进行研究；2）约束条件或偏好的选择通常与问题有关，所以在没有问题特征信息的情况下，搜索过程很难进行。

一方面，帕累托主导策略算法（如NSGA-
II[27]）适用于低维多目标优化问题。然而，对于高维的多目标优化问题，使用帕累托主导策略往往会导致弱的选择压力。此外，在高维空间，使用帕累托算法的计算复杂性也相对较高。

为了在搜索时间内得到相对较多的满足用户偏好的搜索结果，同时降低搜索的计算复杂度，我们提出了LCNAS算法。

C. 延迟的线性偏好情况

Yang的工作[30]中介绍了延迟线性偏好的情况，用于建立多目标强化学习算法。我们把这种情况简化为两个阶段，用它来

重建NAS的过程，并介绍了这个方案的原始想法，如下所示。

一个偏好方程如下：$T:R^m \rightarrow R$ indi-
表示m，它将目标向量映射为一个单一的标量。
给定一个可行的解决方案$\mathbf{x}$及其目标向量$\mathbf{y}\hat{} = \mathbf{F}(\mathbf{x}) = (y_1, \ldots, y_m)$，该策略的效用可以定义为$t(y\hat{})$。这个可行的解决方案的一个可行的解决方案$\mathbf{x}$，其中偏好权重为$\omega$。$\omega^* \mathbf{x}$，用于$\omega = (\omega_1, \ldots, \omega_n)_{n=1}^{L-n}$ $\omega =$

1.在这种特殊情况下，
倾向性解决方案只能来自帕累托边界的凸覆盖集（CCS）。

In a multiobjective optimization problem, given its Pareto boundary P^*, this CCS can be defined as

$$\begin{aligned} \mathrm{CCS} := \{\hat{\mathbf{x}} \in P^* \mid \exists \omega \in \Omega \text{ s.t. } \omega^{\mathrm{T}} \mathbf{F}(\hat{\mathbf{x}}) \\ \geq \omega^{\mathrm{T}} \mathbf{F}(\mathbf{x}^{\wedge}), \ \forall x^{\prime} \in P^*\}。 \end{aligned} \tag{1}$$

延迟的线性偏好的另一个关键特征是延迟。在最初的文章中，杨将延迟的线性偏好情景分为三个连续的阶段：学习阶段、分析阶段和执行阶段。被延迟意味着线性偏好最初是不明确的，直到情景的后期阶段才会有具体的线性偏好。

D. 使用自适应权重的基于分解的多目标进化算法（MOEAs）

为了在学习阶段有效地研究CCS，我们使用了偏好共同进化策略。事实上，用户偏好协同进化的思想源于使用自适应权重的基于分解的MOEAs。偏好权重可以看作是用来激励算法达到更好结果的 权重，也就是说，它可以引导进化算法沿着给定的方向优化。

权重适应的方法有很多种。Zhang等人[25]提出将动态权重设计方法纳入MOEA/D（表示为DMOEA/D），以使MOEA/D在具有不同几何形状的问题上表现良好。Wang等人[26]提出了一种称为偏好启发的使用权重的协同进化算法，在搜索过程中，权重与候选解共同进化。

然而，大多数使用自适应权重算法的基于分解的MOEA都试图使用协同进化算法来获得帕累托边界，而忽略了基于权重的方法在解决凸问题时自然具有优势的事实。因此，我们在NAS任务中应用基于权重的算法进行近似。

III. 方法论

在这一节中，我们说明了如何将NAS形成为延迟的线性偏好情景问题，并介绍了基于给定偏好的情景和基于延迟偏好的情景之间的区别。在提出这个问题后，我们提出了一种新的算法，称为线性偏好协同进化算法，用于在学习阶段有效地搜索CCS。

A. 多目标优化

首先，我们将给出一个多目标优化问题的一般描述。对于一个有n 个 决策变量和m 个目标变量的多目标优化问题，我们可以表达如下。

$$\begin{cases} \text{最大值 } \mathbf{y} = \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})) \\ \text{s.t.} \quad g_i(\mathbf{x}) \leq 0, i = 1, 2, \ldots, q \\ h_j(\mathbf{x}) = 0, j = 1, 2, \ldots, p. \end{cases} \tag{2}$$

$\mathrm{x} = (\mathrm{x}_1, \ldots, \mathrm{x}_n) \in \mathrm{X} \subset \mathrm{R}^n$是n维决策向量，$X$是n维决策空间。$y = (y_1, \ldots, y_m) \in Y \subset R^m$是m维目标向量。YY是$m$维的目标空间。目标函数$\mathbf{F}(x)$定义了从决策空间到目标空间的映射函数；$g_i(x) \leq 0 (i = 1, 2, \ldots, q)$定义了$q$个不平等约束；$h(x) = 0 (j = 1, 2, \ldots, p)$定义了$p$个平等限制因素。

B. 延迟优惠的情况

通常情况下，用户根据自己的用户需求上传到平台，而平台对用户的需求进行处理，然后进行一次性搜索，找到合适的架构并返回给用户。不同用户的需求往往是不同的，而且用户的需求会随着时间的推移而改变。

它用数学语言对$\mathbf{M}_L$，描述如下。

$$\mathbf{m} \in \mathbf{M}_L \Rightarrow \exists \omega \in \omega \text{ s.t.} \forall m^t \in \mathbf{M}, \omega^* \mathbf{F}(\mathbf{m}) \geq \omega^* \mathbf{F}(\mathbf{m}^t)。 \tag{3}$$

只有在这个偏好方程下至少有一个线性偏好f_ω，$\mathbf{M}$才包含在$\mathbf{M}_L$；没有其他个体$\mathbf{m}$可以产生更高的实际效用。

在学习阶段，搜索的计算资源相对充足，但此刻的线性偏好是未知的。对于多目标进化算法，我们重点是我们是否可以用$\mathbf{M}_L$来近似CCS。

在这个阶段，多目标进化学习的目标是获得一个$\mathbf{M}_L$来近似CCS。对于任何相关的偏好向量ω，$\mathbf{M}_L$中总有一个是最好的。
一般来说，学习阶段是为处理未来的各种偏好做准备的。

在推理阶段，一个线性偏好方程$f_\omega(-) = \omega$将被给予。将用较少的时间从已经学到的政策集$\mathbf{M}_L$，给出一个最佳的个体$\mathbf{m}_\omega$，作为回应。我们把这个选择最佳策略的过程称为一个特定的偏好作为偏好的适应。因为需要学习的CCS可能非常大，所以它不是很很容易通过之前学到的$\mathbf{M}_L$，有效地选择当前偏好下的最佳个体。

在基于图2的延迟偏好的情况下，平台只需要在一个学习过程中记录最佳的个体集$\mathbf{M}_L$。在推理阶段，可以以非常有限的资源消耗来适应每个用户。换句话说，在这种情况下，平台可以高效、快速地推断出当前用户偏好下的最优方案。因此，基于延迟偏好的情况下，虽然在学习过程中可能会带来更多的资源消耗

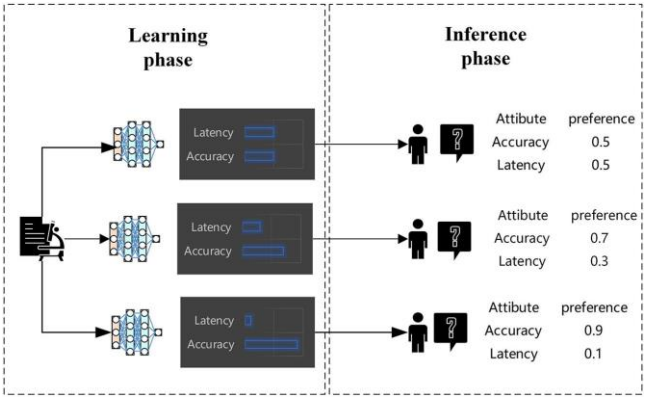

图2. 延迟偏好的情况。

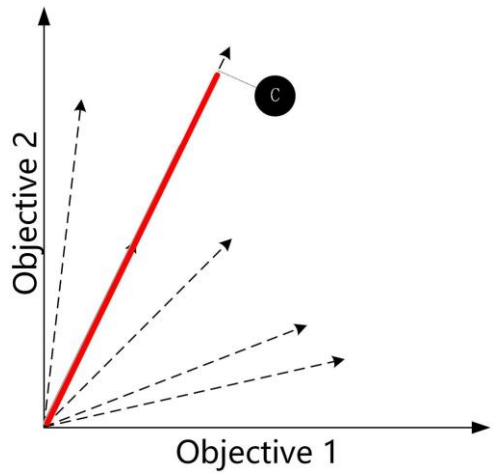

图3. 最大效用个体函数的描述。

相，如果考虑到大量的用户和这些用户不断变化的需求，延迟偏好的方案对神经网络搜索有很大意义。它可以节省更多的计算资源，更迅速地响应用户的部署需求，并且可以更好地处理不断变化的用户偏好。

C. LCNAS算法

在学习阶段，一个重要的问题是如何有效地使用进化算法在给定的偏好集上近似CCS。为了解决这个问题，我们提出了一种线性偏好的分层排序算法。通过这种算法，我们可以有效地选择在线性偏好下占优势的个体。另一个重要的问题是，多目标进化算法往往只在给定的偏好方向上进行优化，这导致解决方案不能有效覆盖整个CCS。为了解决这个问题，我们采用了一种偏好共同进化的策略。在这个策略中，我们在每次迭代中都会产生一组新的偏好。然后，我们将选择能使所选个体获得最大效用的偏好，如图3所示。

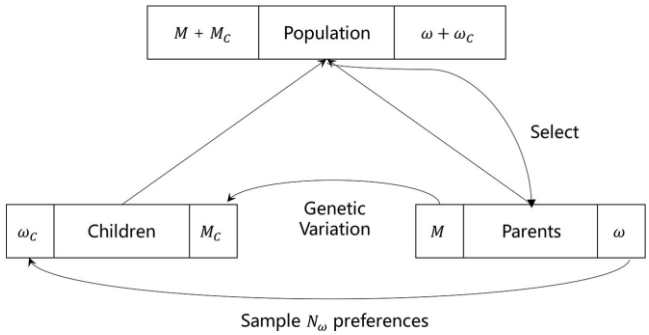

图4. LCNAS的实施框架。

算法1：LCNAS算法。

输入：偏好抽样分布D_ω，迷你批大小偏好N_Ω，每一代的个体数量N，初始种群M_0

$PMF^* = \emptyset$

$M = M_0$

F_M = **objectiveFuction** (M)

PMF^* = **updatePMF**(PMF^*, F_M)

抽取N个Ω，其中$\omega \sim D_\omega$，因为ω

对于每个$t = [1, 2, \ldots, T]$**做**

M_C = **generate-new-pop**(M)

F_{M_C} = **objectiveFuction** (M_C)

$M = M \cup M_C$

$F_M = F_M \cup F_{M_C}$

抽取N个Ω，其中$\omega \sim D_\omega$，因为ω_C $\omega = \omega \cup \omega_C$

(F_M, M, Ω) = **coEvolve** (F_M, M, Ω)

PMF^* = **updatePMF** (PMF^*, M, F_M, Ω)

结束，用于输出。矩阵（PMF^*

具体来说，我们在图4所示的框架内实现了LCNAS算法。

M和ω是固定大小的N和N_Ω向量，代表候选个体和偏好的群体。在每一代t，遗传变异算子被应用于亲代M，产生N个子代M_C。同时，产生N个Ω，新的权重向量ω_C。M和M_C，ω和ω_C，将分别被加入。然后将使用线性偏好算子对种群进行排序，最好的N个个体将被选为新的候选种群M。N_Ω的偏好将被选为新的偏好向量Ω。此外，我们将使用离线字典PMF^*来存储候选解决方案。LCNAS的伪代码在算法1中呈现。该算法LCNAS的核心部分在于函数**coEvolve**，这一点将在下文中阐述。

*1) Coevolve算法。*函数**coEvolve**评估候选个体和偏好的性能，然后分别选择新的父群体M和Ω。父群体M将被排序，使用线性偏好前排序。在对选定的候选方案进行排名时，将选择一个偏好作为最佳方案。

函数coEvolve的伪代码如下。

算法2：coEvolve算法。

输入：偏好向量Ω，候选方案M，以及其目标值F_M
F = **线性主导-前排序**(M, Ω)
$M_{c+1} = \emptyset$，j = 1
而 $|M_{t+1}| + |F_d| \leq N$**做**
$M_{t+1} = M_{t+1} \cup F_4$
i = i + 1
结束时
$m_{t+1} = m_{t+1} \cup f_l[1 : (n - [m_{t+1}|)]$
$M = M_{t+1}$
对于$m \in M$**做**
ω = **selectP**(m, ω)
结束
输出。Ω, M , F_M

1) 第1行应用函数**linear-dominate-front-sort**来对个体进行排序，使用linear-dominate算子。linear-prefer-front-sort的伪代码在算法3中表示。我们将**线性主导**算子定义为

linear-dominate($m, w,$ Ω)，其中个体$m, w \in M$，Ω是偏好向量

$$\text{线性主导}(m, w, \Omega) = \begin{cases} \text{True}, & \omega^T \mathbf{F}(m) \geq \omega^T \mathbf{F}(w), \forall \omega \in \Omega \\ \text{为假，否则} & \end{cases}.$$

2) 第2行应用函数**selectP**从W中选择最合适的偏好向量，为每一个幸存的candi--。日期的解决方案。我们将个人最大效用函数定义为**selectP**($m,$ ω)，其中个人$m \in M$，ω是对偏好分布D的偏好ω。

$$selectP(M, \omega) = \max_{\omega} \omega^T F(m) \ \omega \in \omega_{\circ} \quad (4)$$

IV. 实验和结果

A. 实验设置

本节实验是为了探索LCNAS在给定数据集CIFAR-10下的性能。

在我们的工作中，我们使用NASBench来评估Performance数据。NASBench是一个表格式的数据集，它将卷积神经网络架构与它们在CIFAR-10上的训练和评估的性能相联系。

对于这两种算法，我们设置初始种群数量N=100和T=10次的遗传过程，其中p_M =0.8，q_M =0.1，p_C =0.2，q_C =0.3。对于偏好适应的环境，我们将偏好数设定为N_ω = 100。我们对两种算法的有效性的测试是测试由固定的训练轮次、固定的种群数量和遗传过程的数量得到的模型的（$acc(m)$，$trainingTime(m)$，和$param(m)$。($acc(m)$、$trainingTime(m)$和$param(m)$，分别对应于神经网络模型的三个指标。

算法3：线性占优前排序算法。

输入。M，及其目标值F_M，偏好向量Ω
对于$m \in M$**做**
$S_m = \emptyset$
$n_m = 0$
对于$w \in M$**做**
如果*Lineardominate*(m, w)，**那么**
$S_m = S_m \cup \{w\}$
否则，如果*lineardominate*((w, m) **then**
$n_m = n_m + 1$
结束 如果
如果$n_m = 0$，**那么**
$m_{rank} = 1$
$F_1 = F_1 \cup \{m\}$
结束 如果
j = 1
虽然$F_i /= \emptyset$**做**
W = ∅
for $m \in M$ **do for**
$w \in M$ **do**
$n_m = n_m$ C1
如果$n_m = 0$，**那么**
$w_{tank} = i + 1$
$W = W \cup \{w\}$
结束 如果
结束时
，**结束**
时
结束时
结束
结束
输出。Ω, M , F_M

准确度，训练模型的收敛时间，以及参数的数量[31]。对于普通遗传算法，我们采取三次取最优组的方法。

B. 评价指标

在我们的工作中，我们引入了两个指标来评估LCNAS在多目标NAS下的定量性能。第一个指标是适应的质量，用于评估不同算法在同一用户偏好下的平均效用。第二个指标是适应速度，它是学习阶段和推理阶段的平均时间，对应于一组用户偏好。

1) 自适应效用（AU）。对于来自n个用户偏好的$\Omega_u = (\omega_1, \ldots, \omega_n)$的偏好集，我们使用多目标进化算法$A$来学习个体集，在$\Omega_u$下近似于CCS。在推理阶段，我们可以得到一组响应个体$M_u = m_1, \ldots, m_n$ 然后，我们可以定义AU为

$$\text{AU}(A) = \sum_{i=1}^{n} \omega_i F(m_i^*)_{\circ} \quad (5)$$

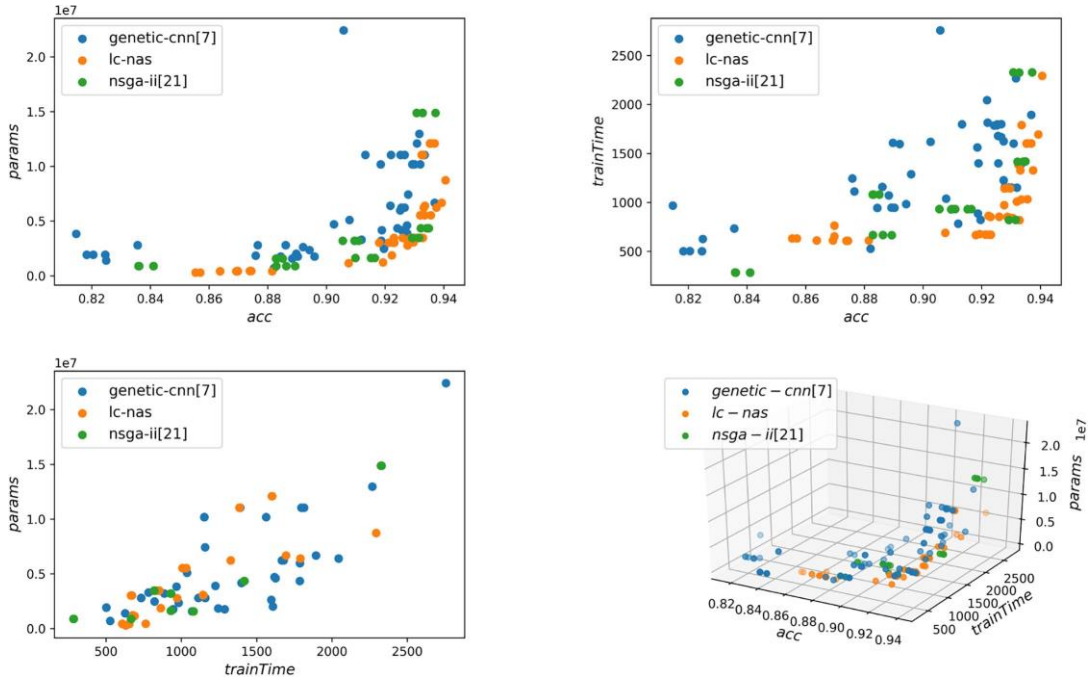

图5. 解决方案集的可视化。

其中，F是目标函数，表示NAS中结构的准确性和资源消耗指数。

2) 适应性时间。对于来自n个用户偏好的$\Omega_u = (\omega_1, \ldots, \omega_n)$的偏好集，我们使用多目标进化算法$A$来学习这组偏好，用$t_L$来表示学习阶段所用的时间。在推理阶段，我们可以使用响应每个偏好的时间成本为$t_1 \ldots t_n$。然后，我们可以定义适应性时间为

$$\mathrm{AT}(A) = t_L + \sum_{i=1}^{n} t_i 。 \quad (6)$$

3) 覆盖率。覆盖率是衡量多目标算法找到所有潜在最优解决方案的能力。对于一个多目标进化算法A，我们用M代表算法得到的最优种群集，用M_{target}代表期望的目标种群集。我们定义，覆盖率可以用F1分数的形式来定义。

$$\mathrm{CR}_A(M^*, M_{\text{target}}) = 2 - \frac{\text{精度} \cdot \text{召回率}}{\text{精度} + \text{召回率}} \quad (7)$$

其中精度$=|M^* \cap M_{\text{目标}}|/|M^*|$，这代表了解集中的最优解的比率，以及召回率$=|M^* \cap M_{\text{target}}|/|M_{\text{target}}|$，表示所获得的最优解与整个最优解的比率。

C. 实验结果和实验分析

首先，在图5中，我们直观地看到了使用遗传-CNN[4]、LCNAS和NSGA-II[27]算法经过100代迭代后最终得到的解集。它是一对指标的比较，包括准确度和参数对，准确度和训练时间对。

表一
AU指标

	AU	
N_u	genetic-cnn[4]	**lc-nas**
1	0.190	**0.191**
2	0.274	**0.275**
4	0.353	**0.355**
8	0.290	**0.291**
16	0.313	**0.314**
32	0.341	**0.342**
64	0.307	**0.308**

从直观的角度来看，我们可以推测，使用LCNAS获得的解集质量更高，也就是说，使用LCNAS获得的解集更准确，参数和训练时间更短。为了定量评价解集，我们将引入AU的概念来评价算法获得的解集的质量。然而，如果我们想用AU来比较这两种NAS算法，我们必须进行归一化操作。其原因是

在上一节中，提到的NAS的三个参数，即准确率、训练时间和
参数的数量，往往有不同的数量大小。这里。
我们选择了这两个算法的最后一代的数据。

归一化的算法，归一化的结果如图6所示。

使用归一化操作后，我们比较了两种算法的AU值，如表I所示。从表中可以看出，随着用户偏好数的增加，两种算法的平均效用值呈现出先上升、后下降、再稳定的趋势。然后，在每个用户偏好数下，LCNAS的平均效用都高于genetic-
CNN算法。因此，给定一组偏好，LCNAS可以实现更大的平均效用。

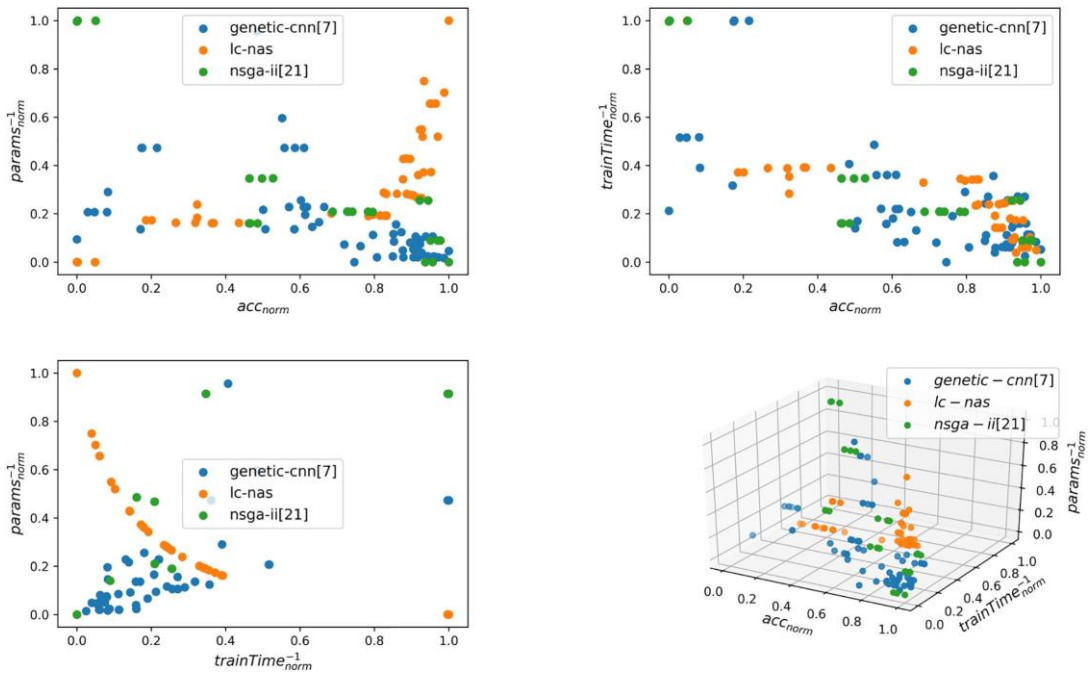

图6. 归一化的结果。

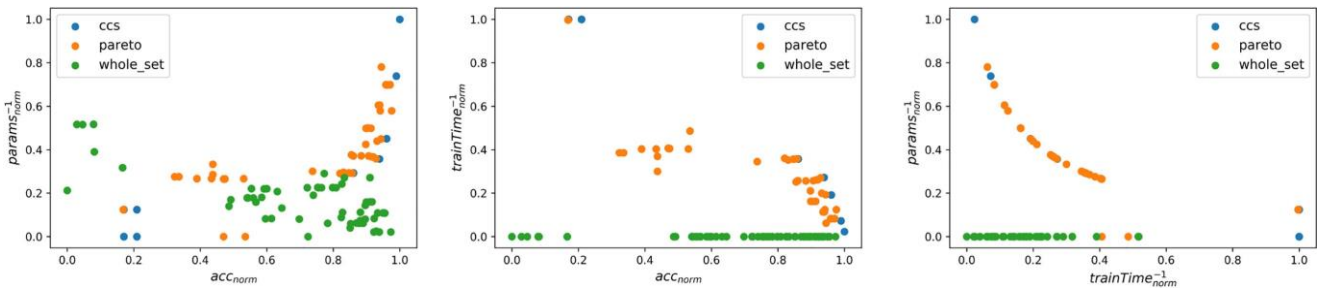

图7. 帕累托和CCS的可视化。

表二
与非线性主导算法的度量比较

	Precision		recall		Coverage Ratio	
	CCS	Pareto Front	CCS	Pareto Front	CCS	Pareto Front
genetic-cnn[4]	0.270	0.230	0.338	0.404	0.300	0.293
lc-nas	**0.520**	**0.510**	**0.863**	**0.930**	**0.649**	**0.659**
nsga-ii[27]	0.530	0.470	0.663	0.825	0.589	0.599

此外，为了比较两种算法在同一维度上的搜索能力，我们引入了覆盖率的概念。首先，我们将两种算法得到的最后一代解集进行合并和归一化，得到一个解集。然后，我们在这个解集中找到相应的帕累托解集和CCS解集。我们将合并后的帕累托解集和CCS解集可视化，如图7所示。

我们用覆盖率指数来评价两种算法得到的解集，结果如表二所示。其中，第一行显示了两种算法对CCS的近似效果。我们可以看到，LCNAS在召回率和覆盖率方面的表现更好。第二行显示的是两种算法对帕累托前沿的近似效果。LCNAS算法在一定程度上更好地接近帕累托边界，并赢得了比CNN算法更高的召回率和准确性。

D. 结果和与其他方法的比较

为了进一步评估我们提出的算法的性能，我们在CIFAR-10上进行了另一个实验。性能比较，其中主要关注的是参数的数量和准确性，在以下方法和人工模型之间进行了比较。

1) 架构模型LC-NET-V1，由LC-NAS方法在NASBENCH101搜索空间得到。
2) MobileNet V1,V2,[28],[29],它们是手动去签名的架构，旨在减少参数，同时重新保持高预测性能。
3) NASNet[7]，它是通过强化学习的NAS，之前在CIFAR-10上取得了最先进的性能。

表三 模型的比较

MODEL	PARAMS	ACC (%)	PARAMS/ACC
MOBILENET	834K	95.0	8.78
MOBILENET V2	850K	95.4	8.91
NASNET	926K	95.3	9.72
RANDOM SEARCH	1200K	94.7	12.67
DPP-NET	1.0M	95.3	10.5
LC-NAS	**873K**	**94.1**	**9.27**

4) DPP-net[19]，是由Dong等人提出的一种多目标NAS方法。

5) 一个随机搜索基线的NAS。

表三显示了上述方法和人工模型之间的性能比较实验的总结。

如表三所示，LC-NET-V1以相对较少的参数实现了相当高的精度。此外，与其他NAS方法相比，LC-NET-V1获得了最小的参数量，并且接近于MobileNet V1和MobileNet V2，后者是为移动性能手动设计的。

在NAS-BENCH101相对有限的搜索空间下，用LC-NAS方法搜索出了这个相对较好的模型。它与其他消耗巨大GPU模型的搜索方法在精度上处于同一水平，在参数上优于其他方法。

V. 总结

在这篇文章中，我们提出了一种偏好共进化算法和一种新的工业信息学多目标认知自动化的方案，将认知自动化过程分为两个步骤：学习阶段和推理阶段。在学习阶段，采用偏好共进化算法来探索满足用户偏好的解决方案集。在推理阶段，该算法可以在很短的时间内使用该解决方案集对用户的新偏好做出反应。实验结果表明，我们的算法优于其他算法。与原来的用户约束方法和帕累托主导的NSGA-II算法相比，它的适应时间更快，适应质量更好。

参考文献

[1] A.Fasth-Berglund and J. Stahre, "Cognitive automation strategy for reconfigurable and sustainable assembly systems," *Assembly Autom.*, vol. 33, no.3, pp. 294-303, 2013.

[2] W.Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomput- ing*, vol. 234, pp. 11-26, 2017.

[3] Y.他 等人，"AMC:用于移动设备上的模型压缩和加速的AutoML，" 在*Proc.Eur.Conf.Comput.Vis*, 2018, pp.815-832.

[4] T.Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search:A survey," *J. Mach.Learn.Res.*，第20卷，第1-21页，2019年。

[5] K.He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition, " *Proc. IEEE Conf.Comput.Vis.Pattern Recognit.*, 2016, pp.770-778.

[6] B.Li, K. Xu, X. Cui, Y. Wang, X. Ai, and Y. Wang, "Multi-scale DenseNet-based electricity theft detection," in *Intelligent Computing Theories and Application*.Cham, Switzerland:Springer-Verlag, 2018.

[7] B.Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE Conf.Comput.Vis.Pattern Recognit.*, 2018, pp.8697-8710.

[8] C.Liu 等人，"Auto-deepLab: Hierarchical neural architecture search for semantic image segmentation," in *Proc. IEEE Conf.Comput.Vis.Pattern Recognit.*, 2019, pp.82-92.

[9] H.Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, "Hierarchical representations for efficient architecture search," in *Proc. Int. Conf.Conf.Learn.Representations*, 2018, pp.13-25.

[10] E.Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proc. 33rd AAAI Conf.Artif.Intell.*, 2019, pp.4780-4789.

[11] Esteban Real 等人，"图像分类器的大规模演变"，在*Proc.34 Int.Conf.Mach.Learn.*, 2017, pp.2902-2911.

[12] F.Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Proc. Int. Conf.Conf.学习。Intell.Optim.*, 2011, pp.507-523.

[13] H.Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," in *Proc. Int. Conf.Conf.Learn.Representations*，2019，第106-118页。

[14] H.Pham, M. Y. Guan, B. Zoph, Q. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," in *Proc. 35th Int.Conf.Mach.Learn.*, 2018, pp.4095-4104.

[15] C.Liu 等人，"Progressive neural architecture search," *Proc.Eur.Conf.Comput.Vis.*, 2018, pp.19-35.

[16] M.Cho, M. Soltani, and C. Hegde, "One-shot neural architecture search via compressive sensing," 2019, *arXiv:1906.02869*.

[17] M.Tan 等人，"MnasNet:MnasNet: Platform-aware neural architecture search for mobile," in *Proc. IEEE Conf.Comput.Vis.Pattern Recognit.*, 2019。
pp.2815-2823.

[18] L.Tan, B. Brotherton, and T. Sherwood, "Bit-split string-matching engines for intrusion detection and prevention," *ACM Trans.Archit.Code Optim.*，第3卷，第3-34页，2006年。

[19] J.-D.Dong, A.-C.Cheng, D.-C.Juan, W. Wei, and M. Sun, "DPP-Net:设备感知渐进式搜索帕累托最优神经架构"，在*Proc.Eur.Conf.Comput.Vis.*, 2018, pp.517-531.

[20] N.Kumar, N. Chilamkurti, and S. C. Misra, "Bayesian coalition game for the Internet of Things:基于环境智能的评估"，*IEEE Commun.Mag.*，第53卷，第1期，第48-55页，2015年1月。

[21] N.Kumar, A. V. Vasilakos, and J. J. P. C. Rodrigues, "A multi-tenant cloud-based DC nano grid for self-sustained smart buildings in smart cities," *IEEE Communication.Mag.*, vol. 55, no.3, pp. 14-21, Mar. 2017.

[22] J.Lian 等人，"通过基于夸张局部变异的生成对抗网络进行基于深度学习的小表面缺陷检测"，*IEEE Trans.Ind.Inform.*, vol. 16, no. 2, pp. 1343-1351, Feb.

[23] A.Makkar和N. Kumar, "基于用户行为分析的网页排名智能能源管理。基于学习自动机的解决方案，" *Sustain.Comput:Inform.Syst.*, vol. 20, pp. 174-191, 2018.

[24] T.Elsken, J. H. Metzen, and F. Hutter, "Efficient multi-objective neural architecture search via Lamarckian evolution," in *Proc. Int. Conf.Conf.Learn.Representations*, 2019, pp.1442-1464.

[25] Q.Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans.Evol. Comput.*, vol. 11, no. 6,
pp.712-731, Dec. 2007.

[26] R.Wang, R. C. Purshouse, and P. J. Fleming, "Preference-inspired co evolutionary algorithms using weight vectors," *Eur.J. Oper.Res.*, vol.243, no.2, pp.423-441, 2015.

[27] K.Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm:NSGA-II", *IEEE Trans.Evol. Comput.*，第6卷，第2期，第182-197页，2002年4月。

[28] A.G. Howard 等人，"MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[29] M.Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2:Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf.Comput.Vis.Pattern Recognit.*, 2018, pp.4510-4520.

[30] R.Yang *et al.*, "A generalized algorithm for multi-objective reinforcement learning and policy adaptation," in *Proc.Conf.Adv. Neural Info.Process.Syst.*, 2019, pp.14636-14647.

[31] B.Wu 等人，"Shift:一个零FLOP、零参数的空间卷积替代方案，"在 *Proc. IEEE Conf.计算。Vis.Pattern Recognit.*, 2018,
pp.9127-9135.