变分的进化方法 Autoencoders

Jeff Hajewski 爱荷华大学计算机科学系
爱荷华市，IA，USA

Suely Oliveira 爱荷华大学计算机科学系
爱荷华市，IA，USA

abstract -变分自动编码器是生成数据模型领域的一个重要工具，然而，由于缺乏关于如何最好地设计相应的编码器和解码器神经网络的直觉，除了确定潜在维度的大小外，它们仍然很难设计。此外，对于一个人应该如何构造这些网络，也没有明确的指导。设计一个有效的变分自编码器通常需要对不同的神经网络架构和潜在维度进行微调和实验，对于大型数据集来说，这在时间和金钱上都可能是昂贵的。在这项工作中，我们提出了一种基于进化神经架构搜索的变分自编码器设计方法。我们的技术是高效的，避免了冗余计算，并且可扩展。我们探索了在神经架构搜索期间使用的 epoch 数量如何影响由此产生的变分自编码器的属性，以及研究学习的潜流形的特征。我们发现，进化搜索能够找到高性能的网络，即使在仅经过两个 epoch 的训练后对网络进行评估。利用这一见解，我们能够显著降低应用于变分自编码器的神经架构搜索系统的整体计算需求。

索引词—— ***variational autoencoder, neural architecture search, evolutionary algorithm, distributed system***

我的介绍。

尽管最近许多深度学习技术取得了成功，但其中许多都依赖于标记数据。不幸的是，这种标记数据的创建成本可能很高，因为它需要专家手动标记每一个单独的数据。无监督学习能够在不需要标记数据的情况下找到模式并学习数据的潜在分布，这是无监督学习如此吸引人的部分原因。

变分自编码器[1]是一种生成式无监督学习技术，它在表示学习方面取得了成功。变分自编码器仅在设计上与传统的自编码器[2]-[4]相似，由两个神经网络组成，分别称为编码器网络和解码器网络。典型的自编码器学习输入数据的简化表示，变分自编码器学习输入数据的潜在概率分布，可以用来生成新的数据样本。变分自编码器有

978-1-7281-3783-4/20/$31.00 c 2020 IEEE

也被用于图像描述、教育数据挖掘[5]、[6]，甚至异常值去除[7]等领域。

尽管它们获得了广泛的成功，但在如何设计组成变分自编码器的神经网络以及该设计应如何根据任务而改变方面，几乎没有工作或指导——这包括确定潜空间的维度。训练和评估一个给定的变分自编码器是一个时间和资源密集的任务，这使得每次迭代的成本很高，这一事实使这变得更加困难。

神经架构搜索(Neural architecture search)试图通过将其视为优化问题来解决这个问题:找到给定问题的验证损失最小化的神经网络架构。它在设计与当前最先进的结果[8]-[11]相匹配或击败的神经网络方面取得了巨大的成功。神经架构搜索的两种主要方法是强化学习和进化算法;本文重点介绍进化算法方法。

在这项工作中，我们提出了一种高效的进化方法来构建变分自编码器。我们算法中的效率来自于缓存先前看到的架构和对基因型(神经网络架构的编码表示)施加不变性约束，这两者都在很大程度上受到函数式编程的启发。我们在并行训练和评估候选神经网络架构的分布式系统上运行这个算法。

本文其余部分的组织如下。我们在第二节中介绍变分自编码器，然后在第三节中简要概述进化神经架构搜索和描述我们的进化算法。在第四节中，我们讨论了相关工作，在第五节中，我们描述了我们的实验并展示了他们的结果。

2 变分 AUTOENCODERS

变分自动编码器是一种建立数据生成模型的无监督学习技术。给定 Rn 中 m 个数据点的数据集 X∈Rm×n，我们假设数据是由分布 p(X |z)生成的，其中 z 是从参数化的潜分布 p(z)中采样的。z 被采样的空间被称为潜空间，通常是一个比 x 低维的空间。我们可以

利用条件分布和先验分布 p(x|z)和 p(z)推导 x 的分布。

$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \quad (1)$$

潜在分布通常被假定为多元高斯，N(μ， σ2 • I)，其中 I∈Rd×d 是潜在维度 d 大小为 d×d 的单位矩阵。我们可以通过神经网络[12]和采样 z ~ N(μ， σ2 • I)来近似后验分布 p(x|z)。当然，这种潜在分布是双向工作的-如果我们可以通过 p(x|z)从潜分布中生成给定样本 z 的样本 x，那么我们应该能够通过分布 p(z|x)从数据分布中生成给定样本 x 的潜样本 z。与分布 p(x|z)一样，我们可以用神经网络来近似 p(z|x)。我们把这种近似分布称为 q(z|x)。在实践中，我们假设 z ~ N(μ， σ2 • I)，并使用神经网络生成 μ 和 σ，而不是 z 本身。学习过程试图最小化后验概率分布 p(x|z)和真实分布 p(x)之间的距离。总损失被定义为衡量两个概率分布之间相似性的 Kullback-Leibler (KL)散度[13]和期望的重构误差之和，由式(2)给出。

$$L = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[-\log p(\mathbf{x}|\mathbf{z})] - KL(q(m\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \quad (2)$$

3 进化神经架构搜索

神经体系结构搜索(Neural architecture search, NAS)是一组用于设计神经网络的算法和技术。最流行的两种方法是强化学习和进化技术。我们在这项工作中使用进化搜索算法，因为与基于强化学习的同类算法相比，进化算法实现和理解起来更简单，而且通常需要更少的计算资源。进化算法需要克服的主要挑战是探索与利用的对比。我们需要确保我们给每个架构一个公平的评估，从这个意义上说，它已经训练了足够长的时间，以便在评估阶段，在验证数据上测试它，准确地评估它在手头任务上的有效性。另一方面，给定有限的计算资源，我们希望最大化探索的架构的数量。进化神经架构搜索的这两个方面是相互矛盾的——训练时间越长消耗计算资源，留给剩余搜索的资源就越少。

形式上，我们用神经架构搜索解决的问题由式(3)给出，其中 L 是损失函数，ψ 是网络架构，a 是所有神经网络架构的空间。

$$\Psi^* = \min_{(\psi_e,\psi_d)\in\mathcal{A}} L(\mathbf{X}, \psi_d(N(\psi_e(\mathbf{X})))) \quad (3)$$

哪里 Ψ = (Ψe, ψd 和 N(ψe(X))表示样本 z ~ N(μ， σ2 • I)对于 μ， σ2 = ψe(X).请注意，搜索实际上是针对两个网络架构，编码器和

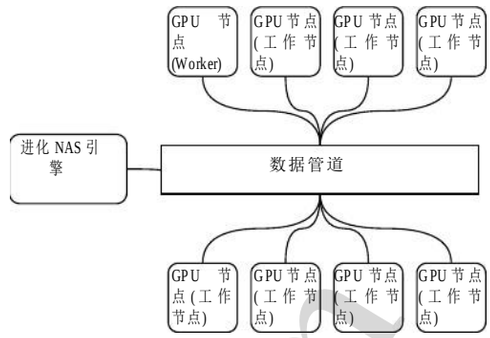


图 1:系统架构高层概览。

译码器的架构。使用 Kingma 等人建议的重新参数化技巧[1]，我们使用的损失函数由式(4)给出。

$$L(\theta_x, \theta_z; \mathbf{x}) = \frac{1}{L}\sum_{i=1}^{L} \log p(\mathbf{x}|\mathbf{z}) + \frac{1}{2}\sum_{j=1}^{J}(1 + 2\log(\sigma j) - \mu 2j - \sigma j2) \quad (4)$$

这个技巧简单地将 z 的先验分布从 N(μ， σ2 • I)重新参数化到

Z = μ + σ2 •

式中，其中，~ N(0,1).训练过程由算法 1 描述。第 7 和 8 行是 z 的采样和重新参数化。函数 AdamUpdate 通过随机梯度下降封装了梯度的计算和编码器和解码器神经网络的参数更新。

算法 1 变分自动编码器训练算法。

1:程序 TRAINVAE(X， Ψ，n, m)

2:(ψe, ψd)← Ψ　　变性 VAE

3:对于 I = 1 到 n 做　　N - #个时代

4:对于 j = 1 到|X|/m 做　　样本 mini-batch

5:

6: xs←sampleMB(X,m)

(μ， σ 2)← ψe(x)

7:

Z←μ+ σ2 •

8:

9: ←n (0, i)

Xs^ ← ψd(z)

10:AdamUpdate(Ψ， L(x s, xs))+

11: end for

12:结束

13:结束

A.系统架构

图 1 显示了我们用于执行此搜索的分布式系统的架构。系统分离模型

使用很少计算资源的生成，以及使用大量计算资源的模型评估，跨越不同的节点类型。模型生成不需要任何专门的硬件，而模型评估则需要 GPU。演化 NAS 引擎负责处理网络体系结构的演化。生成的网络结构被编码在协议缓冲区[14]中，并通过 gRPC[15]，[16]发送给 GPU 节点(称为 worker)。这种方法的一个优点是，数据管道对它传输的数据类型是不可知的，这使得整个系统在其应用中更加灵活。

候选神经网络架构的训练和评估是由工人异步完成的。一个中央工作队列(数据管道的一部分)将任务反馈给工人，工人完成训练和评估并返回结果。我们能够在 worker 的数量上实现接近线性的扩展，因为系统工作量是计算上有限的(而不是受通信速度的限制)，并且因为 worker 是无状态的，允许系统在 worker 故障后恢复。工作人员通过数据管道将其结果发回给 NAS 引擎。引擎使用这些评估来生成下一批候选架构。系统基础结构是用 Go 编写的，而工作程序是用 Python 编写的。代码可以在 GitHub 上免费获取，网址是 github.com/j-haj/brokered-deep-learning。

B.进化算法

我们使用(μ + λ)选择，其中 μ 父母生成 λ 后代，我们选择顶级 μ 个体以在下一代中生存。适应度被定义为损失的倒数，赋予它良好的属性，即损失很容易从适应度计算出来，高适应度对应于小的损失。可能的变异操作是向网络添加新层或修改现有层。我们将每个神经网络的层数限制为 5 层，并将层类型限制为线性层。这就产生了不超过 5 层隐藏层的全连接神经网络。如果一个基因型(即表示给定神经网络架构的编码)试图在拥有最大允许层数时追加一层，它会退回到修改随机选择的现有层。层的大小从 50 到 1000(包括)，步骤为 50。这个过程同时发生在编码器和解码器网络中。此外，基因型可以突变潜在维度，这允许搜索探索不同的潜在空间。隐维度的范围从 10 到 100(包括在内)，步骤为 10。由于编码器和解码器网络独立演化，总共有 $10 \cdot 2010 \approx 1014$ 或 100 万亿不同的架构。

我们系统的部分效率来自于驱动 NAS 引擎的进化算法。所见的集合跟踪了之前见过的架构。如果搜索碰巧找到以前见过的架构(第 8 行和第 9 行)，它将适应度设置为-1，这导致基因型在选择过程中被丢弃。因为选择总是保留排名靠前的 μ 个体，一个以前见过的

算法 2代际进化神经架构搜索。

```
1:程序 EVONASSEARCH(n, m)
2:     maxgeneration←m
3:     seen←∅ track seen architectures
4:     p←InitializePopulationOfSize(n) for i = 1 to
5:     maxGenerations do p←p∪p. produceoffspring () for o∈p do
6:
7:         如果 o.is evaluate()或 o∈seen 则如果
8:         o∈seen 则 o。适应度←−1 下降基因型结束 if
9:             继续跳过评价 end if
10:        发送 o 到任务队列
11:        seen.add
12:        (o)
13:    结束了
14:
19:    p←SelectTopLambda(r, λ)          (μ + λ)
20:结束
21:结束程序
```

个人要么仍然是最优秀的个体，要么已经被表现更好的个体取代。无论哪种方式，我们都可以安全地丢弃重新发现的基因型。

通过使基因型不可变，我们实现了算法效率的额外改进。一旦基因型被评估——通过构建、训练和评估相应的变分自编码器——它的适应度将不会改变。这是算法的一个简单方面，但避免了冗余评估，这对于较大的神经网络规模尤为重要。一个很自然的问题是，如果基因型是不可变的，那么在进化算法中基因型是如何突变的——我们创建了一个基因型的克隆，并在创建过程中对其进行修改，包括基因突变和交叉。

Iv.相关工作

虽然已经有大量的工作探索变分自编码器[7]，[17]-[19]的不同方面，但据我们所知，没有之前的工作探索进化神经架构搜索技术来设计变分自编码器。之前的工作与这个主题最相似的是[20]，[21]探索了进化神经架构搜索技术对自编码器的使用。除了我们关注的是变分自编码器而不是传统的自编码器外，我们的工作的不同之处在于它使用了一种进化算法，总是生成有效的网络架构，并且不要求编码器和解码器网络是相同的架构。

在神经架构搜索领域已经做了很多工作，大多应用于分类。诸如[8]，[11]，[22]等工作集中在基于强化学习的神经架构搜索方法上。与我们的工作更相关的是

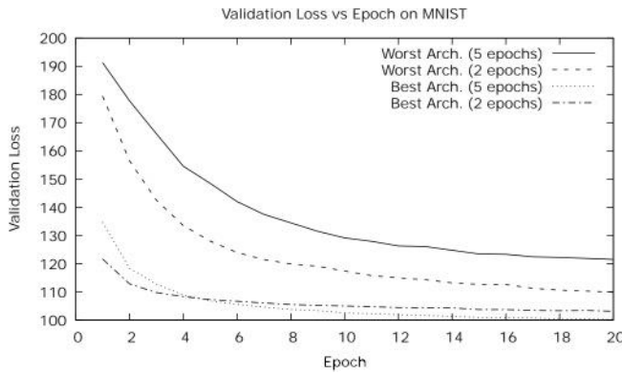


图 2:MNIST 数据集上最好和最差架构的验证损失与 epoch 数量的比较对网络进行评估

经过两个或五个时期的架构搜索。

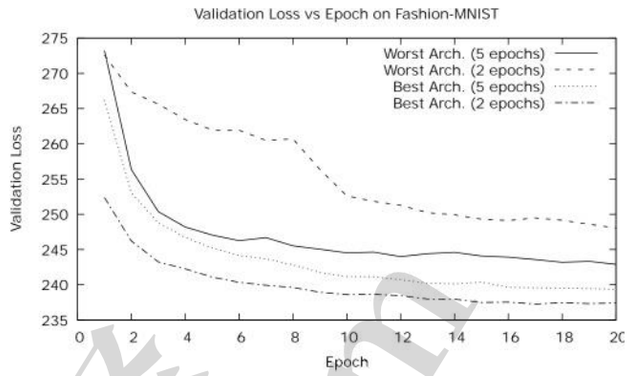


图 3:验证精度与次数的比较

时尚主义者数据集上最好和最差的架构。的

该网络在经历了两个或五个时期之后进行了评估

架构搜索。

[9]，[23]，[24]，侧重于基于进化的方法。之前在强化学习和基于进化算法的神经架构方法方面的大部分工作都集中在图像分类上，而不是学习生成模型。尽管与我们的工作在应用上存在差异，但这些技术中的许多都适用于其他应用领域，例如应用于变分自编码器的神经架构搜索。

V。E肉苁蓉

我们的实验探索了进化神经架构搜索寻找有效的变分自编码器的能力。我们通过探索其潜空间来评估已发现的体系结构，并说明在体系结构搜索期间使用的 epoch 数量对已发现体系结构的训练速度的影响。

所有实验都是在亚马逊网络服务服务器上使用 Nvidia K80 gpu 进行的。我们使用 PyTorch[25]来实现神经网络。使用 MNIST[26]和 Fashion-MNIST[27]对变分自动编码器结构进行了评估。我们使用 128 的小批量大小进行训练，Adam[28]用于更新神经网络参数，除非另有说明，结果是基于 20 个 epoch 的训练。由于资源限制，我们无法探索更大的数据集，如 CIFAR-10[29]。

表 1 显示了在对 MNIST 和 Fashion-MNIST 数据集进行的 2 次和 5 次 epoch 搜索中发现的最好和最差的架构。架构描述由由垂直线分隔的三部分组成。第一部分代表编码器网络架构，其中线性层大小由逗号分隔。第二部分(在竖条之间)代表潜在维度，最后逗号分隔的列表代表解码器架构。

A. epoch对搜索的影响

虽然在进化搜索算法中使用的 epoch 的数量看起来像是一个小细节，得到这个

超参数正确对搜索过程的进展和结果有真实的影响。更复杂的神经网络——即参数更多的神经网络——通常需要对数据进行更多的遍历，才能达到其损失开始趋于平稳的点。这一点在图 2 和图 3 中都很明显。挑战在于，这个平台开始的时期是未知的，并随着神经网络架构的变化而变化。

图 2 显示了在进化神经架构搜索过程中发现的最佳和最差架构的验证损失与 epoch 的比较。在搜索过程中，神经网络架构在两个 epoch 或五个 epoch 之后进行评估。正如预期的那样，通过两次 epoch 选择搜索发现的体系结构以更低的损失开始，它们的损失下降得更快。这是两个 epoch 之后的早期评估所产生的选择压力的结果。发现的架构采用了他们在训练早期获得更好性能的特征。人们会期望，在选择之前给予更多 epoch 来训练的体系结构可能具有更好的整体性能，因为搜索能够更好地评估体系结构，而不是简单地选择训练更快的网络。这在性能最好的体系结构中可以看到，但在性能最差的体系结构中没有表现出来。

图 3 显示了与图 2 相同的比较，但针对的是时尚 MNIST 数据集。时尚 MNIST 实验显示了与 MNIST 实验相似(如果不是更极端的话)的结果。这些结果的有趣之处在于，在两次 epoch 搜索中找到的最佳架构总是优于在五次 epoch 搜索中找到的最佳架构。这个结果在两个层面上是令人惊讶的。如上所述，我们预计更长的训练时间可以让搜索过程更好地评估候选网络架构。此外，即使在五次 epoch 搜索中获得的额外训练对搜索没有帮助，它也不应该损害搜索。这很可能是两次 epoch 搜索碰巧找到了一个理想架构的结果，五次 epoch 搜索通过

表一:发现的架构

数据集	搜索时代	排名	体系结构
MNIST	2	最好的	850||1000 年
	5	最糟的	200、500|20|1000、650、50
		最好的	800、900|30|150、1000
		最糟的	850,700,400 |30|850、50、1000,900、50
时尚 MNIST	2	最好的	1000 ||350 年,1000 年
	5	最糟的	950、850、250、200|10|150、500、750
		最好的	1000 ||700, 750
		最糟的	900、50、700|60|50、750,600、950

机会，没有发现。

B.潜空间流形

我们可以通过将潜在维度降为 2，并从单位平方[0,1]×[0,1]上的高斯累积分布函数的逆采样中可视化学习到的流形。图 4 显示了从两个时代搜索中发现的最佳 VAE 架构中学习到的流形，如图 4a 所示，以及从五个时代搜索中发现的最佳 VAE 架构中学习到的流形，如图 4b 所示。需要注意的是，这个分析是纯定性的，并没有说明哪种模型更可取。虽然这两个流形之间有相似之处，但令人惊讶的是，它们是截然不同的。图 4a 所示的流形由约 50%的鞋类组成，而图 4b 所示的流形只有约 15%的鞋类。我们指出这一点是因为，尽管我们预计流形彼此不同(它们是完全不同的网络架构)，但有趣的是，学习到的流形如此不同。

C.潜空间采样

图 5 比较了两次和五次 epoch 搜索中发现的最佳架构的样本。从图 2 中回想一下，与两次 epoch 搜索中发现的最佳架构相比，五次 epoch 搜索在 20 次 epoch 后取得了更好的损失。图 5a 中的数字比图 5b 中的数字更加模糊。例如，与图 5b 的 8 相比，图 5a 中的很多 8 更难辨别，后者的线条更实，中心更明显。一般来说，图 5b 中的大多数数字的线条较粗，使其更加清晰易读。通过对比这两幅图可以清楚地看出，在五次 epoch 搜索中发现的变分自编码器比在两次 epoch 搜索中发现的架构学到了更精确的数据分布。这可能是由于更大的学习能力，因为五次 epoch 架构中的神经网络更大，有更多的参数。更大的网络是更灵活的模型，因为可学习参数的数量增加了，但它们也更有可能过拟合数据。

d .的观察

我们注意到的搜索的一个方面是，与其他架构相比，包含信息瓶颈的架构，即由两个高节点计数层围绕大小为 50 的层组成的三层结构，往往表现不佳。

例如，在 5 个 epoch 搜索中，表现最差的时尚 MNIST 网络，如表 I 所示，有 900 个节点的一层，然后是 50 个节点的一层，然后是 700 个节点的一层。当数据到达 700 节点层时，从 900 节点层传输到 50 节点层的大部分信息都丢失了，只是因为 50 节点层没有能力传输它。一旦这些信息丢失，就无法恢复，网络的整体性能也会因此受到影响。这在很大程度上并不奇怪——在设计神经网络时，一个常见的启发式方法是轻轻地降低层的维度或轻轻地增加层的维度。

六。结论

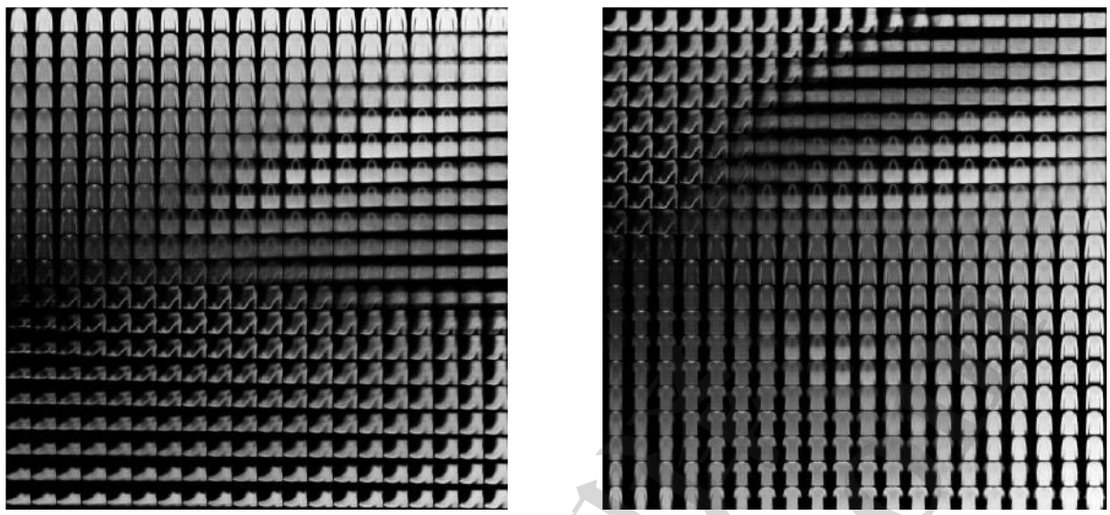
本文提出了一种进化变分自动编码器算法，并在 MNIST 和 Fashion-MNIST 数据集上验证了该算法的有效性。我们的算法能够高效地为变分自编码器找到高性能的架构。未来工作的一个领域是将我们使用的世代算法与锦标赛式的，或稳态[30]，[31]进行比较，这将通过减少工人的空闲来提高系统的吞吐量;然而，它也会影响选择过程，可能会导致一些原本会从种群中移除的基因型幸存下来。未来工作的另一个领域是在更复杂的数据集上研究这种方法;由于计算资源的限制，我们无法探索更大的数据集。

神经架构搜索是深度学习的一个有前途的领域。持续努力提高其效率，拓宽其应用领域，将对深度学习领域产生积极影响。我们已经证明，神经架构搜索确实可以成为构建生成模型的一种有效方法。这很重要，因为生成模型可能会在未来的深度学习和人工智能系统中发挥重要作用。

参考文献

王文杰，“自编码变分贝叶斯”，第 2 届国际学习表征学术会议论文集，中国，2014 年 4 月，Y. Bengio 和勒坤主编，2014。

[2] T. Hrycej，神经网络模块化学习:一种模块化的神经网络分类方法。纽约，NY，USA:约翰·威利父子公司，1992 年第 1 版。



(a) (b)

图 4:(a)从两次 epoch 搜索中发现的最佳架构中学习到 manifold，经过 20 个 epoch 的训练。(b)学从 5 个 epoch 搜索中发现的最好的建筑中学习，经过 20 个 epoch 的训练。

(一) (b)

图 5:(a) p (x |z)的样本，来自两个 epoch 搜索中发现的最好的体系结构，经过 100 代，训练了 20 个 epoch。(b)样本 p (x |z)来自经过 100 代 5 次时代搜索发现的最好的建筑，训练了 20 个时代。

[3]P. Vincent，H. Larochelle，I. Lajoie，Y. Bengio 和 P.Manzagol，“堆叠去噪自动编码器:学习具有局部去噪准则的深度网络中的有用表示”J. Mach。学习。Res.， vol. 11，pp. 3371-3408，2010。

[4]P. Vincent、H. Larochelle、Y. Bengio 和 P. a。Manzagol，“基于去噪自动编码器的鲁棒特征提取与合成”，第 25 届机器学习国际会议论文集，2008，(纽约，NY，USA)，第 1096-1103 页，ACM，2008。

[5]M. Curi、G. Converse、J. Hajewski 和 S. Oliveira，“认知模型的可解释性变分自动编码器”，国际神经网络联合会议，2019 年。

[6]G. Converse、M. Curi 和 S. Oliveira，”教育评估的自动编码器”，2019 年国际人工智能教育会议。

戴斌，王赟，华国光，“变分自动编码器的隐藏能力”，中国科学技术杂志，vol. 614 - 614，2017。

[8]B. Zoph 和 Q. V.Le，“强化学习的神经架构搜索”，2017。

梁振英、梁振英、梅尔森、拉瓦尔、芬克、弗朗康、拉朱、沙赫扎德、纳夫鲁兹扬、杜菲、霍德加特，“深度神经网络的演化”，中国科学技术杂志，vol. abs/1703.00548，2017。

[10] J. Koutnik，G. Cuccu，J. Schmidhuber 和 F. Gomez，“演化的大规模神经网络用于基于视觉的强化学习”，第 15 届遗传与进化计算年会论文集，GECCO ' 13，(纽约，NY，USA)，第 1061-1068 页，ACM，2013 年。

[11] H. Pham，M. Guan，B. Zoph，Q.Le 和 J. Dean，“通过参数共享的高效神经结构搜索”，第 35 届会议论文集

机器学习国际会议(J. Dy 和 A. Krause 主编)，《机器学习研究录》第 80 卷，(Stock- holmsmassan，Stockholm Sweden)，第 4095-4104 页，PMLR，2018年 7月。

K. Hornik, M. B. Stinchcombe和 H. White，“多层前馈网络是通用近似器”，《神经网络》，第 2卷，第 6期。第 5期，第 359-366页，1989。

[13] S. Kullback，“给编辑的信:Kullback -leibler距离”，《美国统计学家》，第 41卷，340-341 页，1987年 11月。

“协议缓冲器:谷歌的数据交换格式”，技术代表，《谷歌》，2008年第 6期。

[15]谷歌，”grpc。”Accessed on March 2019

分布式系统中的机制，”SIGOPS Oper.;系统。Rev，第 27卷，第 75-86 页，1993年 7月。

李志强，“阶梯变分自动编码器”，《神经信息处理系统的进展》，第 3738-3746页，2016。

陈志强，陈志强，陈志强，陈志强等，“基于逆自回归流的变分自动编码器设计”，2017。

侯晓明，沈林，孙凯，邱国强，“深度特征一致性变分自动编码器”，计算机视觉应用研究，2017 年 IEEE 计算机视觉冬季会议，pp. 1133-1141，IEEE，2017。

[20] S. Lander 和 Y. Shang，“Evoae -一种训练深度学习网络自动编码器的新进化方法”，2015 年 IEEE第 39届年度计算机软件与应用会议，第 2卷，790-795页，2015年 7月。

[21] M. Suganuma, M. Ozay和 T. Okatani，“利用进化搜索开发标准卷积自动编码器用于图像恢复的潜力”，《第 35届机器学习国际会议论文集》(J. Dy 和 A. Krause，主编)，《机器学习研究论文集》第 80卷，(Stockholm smassan，Stockholm Sweden)，第 4771-4780 页，PMLR，2018年 7月。

[22] B. Zoph, V. Vasudevan, J. Shlens 和 Q. V. Le，“学习可伸缩图像识别的可转移架构”，IEEE学报

Conference on computer vision and pattern recognition, 2018 年第 8697-8710页。

刘 H. Liu, K. Simonyan, O. Vinyals, C. Fernando, K. Kavukcuoglu，”高效架构搜索的层次表示”，CoRR, vol. abs/1711.00436, 2017。

[24] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan，Q. V. Le和 A. Kurakin，“图像分类器的大规模演化”，《第 34届机器学习国际会议论文集》，澳大利亚新南州悉尼，2017年 8月 6-11日(D. Precup 和 Y. W. Teh, eds.)，《机器学习研究论文集》第 70卷，2902-2911 页，PMLR，2017年。

林志强、林志强、A. Desmaison、L. Antiga 和 A. Lerer，《PyTorch 的自动分化》，中国科技期刊，2017年。

[26]Y. LeCun和 C. Cortes，“MNIST手写数字数据库”，2010。

[27] H. Xiao, K. Rasul 和 R. Vollgraf，”时尚 mnist:基准机器学习算法的新图像数据集”，2017。

D. P.金马和 J. Ba，“随机优化的方法”，第 3届国际会议学习表征，中国，2015年 5月，学术会议论文集(Y. Bengio 和 Y. LeCun主编)，2015。

[29] A. Krizhevsky, V. Nair和 G. Hinton，”Cifar-10(加拿大高级研究所)，”

[30] R. Enache, B. Sendhoff, M. Olhofer 和 M. Hasenjager，“并行架构的稳态和代际进化策略的比较”，《Nature的并行问题求解》- PPSN VIII (X. Yao, E. K. Burke, J. A. Lozano, J. Smith, J. J. Merelo-Guervos, J. A. Bullinaria, J. E. Rowe, P. Ti no, A. Kaban 和 h. P. P. Kaban)。(柏林，海德堡)，第 253-262页，施普林格柏林海德堡，2004年。

[31] a.c。Z avoianu, E. Lughofer, W. Koppelstatter, G. Weidenholzer, W. Amrhein 和 E. P. Klement，“代际和稳态异步多目标进化算法在计算密集型问题上的性能比较”，Know。的系统。，第 87卷，第 47-60 页，2015年 10月。