变分的进化方法
自编码器

Jeff Hajewski
爱荷华大学计算机科学系 美国爱荷华州
爱荷华市 jeffrey-
hajewski@uiowa.edu

Suely Oliveira
爱荷华大学计算机科学系 美国爱荷华州
爱荷华市 suely-
oliveira@uiowa.edu

摘要 变分自动编码器是生成数据模型领域的重要工具,但由于缺乏直觉,除了确定潜在的大小之外,如何最好地设计相应的编码器和解码器神经网络,它们仍然很难设计。尺寸。

此外,对于应该如何构建这些网络,没有明确的指导。设计一个有效的变分自动编码器通常需要微调和试验不同的神经网络架构和潜在维度,对于大型数据集,这在时间和金钱上都是昂贵的。

在这项工作中,我们提出了一种基于进化神经架构搜索设计变分自动编码器的方法。

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过程中使用的时期数如何影响所得变分自动编码器的属性,并研究了学习到的潜在流形的特征。我们发现进化搜索能够找到高性能网络,即使在仅经过两个时期的训练后对网络进行评估也是如此。利用这种洞察力,我们能够显着降低应用于变分自动编码器的神经体系结构搜索系统的总体计算要求。

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

一、引言

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个单独的数据。Unsu 监督学习无需标记数据即可找到模式并了解数据的潜在分布的能力是无监督学习如此吸引人的部分原因。

变分自动编码器 [1] 是一种生成式无监督学习技术,在表示学习方面取得了成功。变分自编码器与传统自编码器 [2]-[4] 的相似之处仅在于它们的设计,由两个神经网络组成,称为编码器和解码器网络。

典型的自动编码器学习其输入数据的简化表示,而变分自动编码器学习输入数据的潜在概率分布,可用于生成新的数据样本。变分自动编码器有也被用于图像字幕、教育数据挖掘[5]、[6],甚至异常值去除[7]等领域。

尽管它们取得了广泛的成功,但关于如何设计构成变分自动编码器的神经网络以及该设计应如何根据任务而改变的工作或指导很少 这包括确定潜在空间的维度。由于训练和评估给定的变分自动编码器是一项时间和资源密集型任务,这使得每次迭代的成本都很高,因此这变得更加困难。

神经架构搜索试图通过将其视为优化问题来解决此问题:找到使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结果相匹配或击败的神经网络方面取得了巨大成功 [8]-[11]。神经结构搜索的两种主要方法是强化学习和进化算法;我们在本文中关注进化算法方法。

在这项工作中,我们提出了一种有效的进化方法来构建变分自动编码器。我们算法的效率来自缓存缓存以前见过的架构和对基因型(神经网络架构的编码表示)施加不变性约束,这两者在很大程度上都受到函数式编程的启发。我们在分布式系统上运行该算法,该系统并行训练和评估候选神经网络架构。

本文的其余部分安排如下。我们在第二部分介绍了变分自动编码器,然后简要概述了进化神经架构搜索,并在第三部分描述了我们的进化算法。在第 IV 节中,我们讨论了相关工作,在第 V 节中,我们描述了我们的实验并展示了他们的结果。

二。变分自动编码器

变分自动编码器是一种无监督学习技术,可构建数据生成模型。给定 Rn中 m 个数据点的数据集 X ∈ Rm×n ,我们假设数据是从分布 p(x|z) 生成的,其中 z 从参数化潜在分布 p(z) 中采样。从中采样 z 的空间称为潜在空间,通常是比 x 的空间维数更低的空间。我们可以

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

使用条件分布和先验分布 p(x|z) 和 p(z) 导出 x 的分布。

$$p(x) = p(x|z)p(z)dz \quad (1)$$

潜在分布通常假设为多变量高斯分布,N(μ, σ2 · I),其中 I ∈ Rd×d是潜在维度 d 的大小为 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的 如果我们可以通过 p(x|z) 从潜在分布中给定样本 z 生成样本 x,那么我们应该能够在给定样本 x 的情况下生成潜在样本 z通过分布 p(z|x) 的数据分布。与分布 p(x|z) 一样,我们可以用神经网络近似 p(z|x)。我们将这种近似分布称为 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。

总损失定义为 Kullback-Leibler (KL) 散度 [13] 的总和,它衡量两个概率分布之间的相似性,以及预期的重建误差,由等式 (2) 给出。

$$L = E_{q(z|x)}[- \log p(x|z)] - KL(q(mz|x)||p(z)) \quad (2)$$

三、进化神经结构搜索

神经架构搜索 (NAS) 是一系列用于设计神经网络的算法和技术。两种最流行的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对应算法相比,进化算法除了通常需要更少的计算资源外,更易于实现和理解。进化算法需要克服的主要挑战是探索与开发。我们需要确保我们对每个架构进行公平的评估,因为它已经训练了足够长的时间,以便在评估阶段（在验证数据上对其进行测试）准确地评估其在手头任务上的有效性。另一方面,给定有限数量的计算资源,我们希望最大化探索架构的数量。进化神经架构搜索的这两个方面相互矛盾 更长的训练时间会消耗计算资源,为剩余搜索留下更少的资源。

形式上,我们用神经架构搜索解决的问题由等式（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。

$$\psi = \text{最小值}_{(\psi_e,\psi_d)\in A} L(X,\psi_d(N(\psi_e(X)))) \quad (3)$$

其中 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种网络架构,编码器和

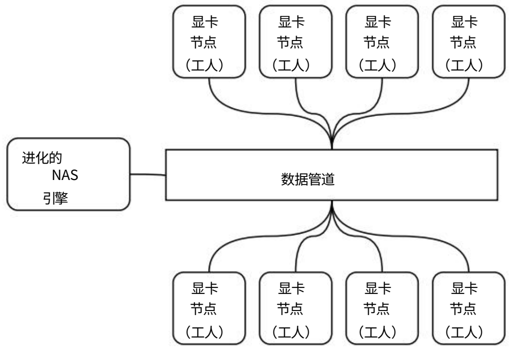


图 1:系统架构的高级概述。

解码器架构。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等式（4)给出。

$$L(\theta_x, \theta_z; x) = \frac{1}{N}\sum_{i=1}^{N} \text{对数 } p(x|z) + \frac{1}{2}\sum_{j=1}^{J}(1 + 2\log(\sigma_j) - \mu_j^2 - \sigma_j^2) \quad (4)$$

这个技巧简单地将 z 的先验分布从 N(μ, σ2 · I)重新参数化为

$$z = \mu + \sigma^2 \cdot$$

其中 N(0, 1)。训练过程由算法 1 描述。第 7 行和第 8 行显示了 z 的采样和重新参数化。 AdamUpdate 函数通过随机梯度下降封装了梯度的计算以及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) (ψe,ψd) ← ψ                解构 VAE n - # of epochs
do xs ←2sample MB(X, m) for j=1 to |X|/m
3:   -batch (μ, σ2) ← ψe(xs) ← N(0,I) z
4:        ← μ + σ2 · x^s ← ψd(z)
5:
6:
7:
8:
9:
10:           AdamUpdate(Ψ, L(xs, xs^ )) end
11:       for end for 13:结束程序
12:
```

A. 系统架构 图 1 显示了我们用来执行此搜索的分布式系统的架构。系统分离模型

使用非常少的计算资源的生成,以及跨不同节点类型使用大量计算资源的模型评估。模型生成不需要任何专用硬件,而模型评估需要 GPU。进化的 NAS 引擎处理网络架构的进化。生成的网络架构在协议缓冲区 [14] 中编码,并通过 gRPC [15]、[16] 发送到 GPU 节点（称为工作节点）。这种方法的优点之一是数据管道与其传输的数据类型无关,这使得整个系统在其应用中更加灵活。

候选神经网络工作架构的训练和评估由工作人员异步完成。

中央工作队列（数据管道的一部分)将任务提供给完成培训和评估并返回结果的工作人员。我们能够实现 worker 数量的近线性扩展,因为系统工作负载在计算上是有限制的（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 NAS 引擎。该引擎使用这些评估来生成下一批候选架构。系统基础设施是用 Go 编写的,而工人是用 Python 编写的。该代码可在 GitHub 上免费获得,网址为 github.com/j-haj/brokered-deep-learning。

B. 进化算法

我们使用 (μ + λ) 选择,其中 μ 父母产生 λ 后代,我们选择最优秀的 μ 个体在下一代中存活下来。适应度被定义为损失的倒数,赋予它很好的特性,即损失很容易从适应度中计算出来,并且高适应度对应于小损失。可能的变异操作是将新层附加到网络或修改现有层。我们将每个神经网络的层数限制为五个,并将层类型限制为线性层。这导致完全连接的神经网络具有不超过五个隐藏层。如果一个基因型（即表示给定神经网络架构的编码)试图在它具有最大允许层数时附加一个层,它会回退到修改随机选择的现有层。层大小范围从 50 到 1,000（含）,步长为 50。编码器和解码器网络都会发生此过程。此外,基因型可以改变潜在维度,从而允许搜索探索不同的潜在空间。潜在维度从 10 到 100（含）,步长为 10。由于编码器和解码器网络独立进化,因此总共有 $10 \cdot 2010 \approx 1014$或 100 万亿个不同的架构。

我们系统的部分效率来自驱动 NAS 引擎的进化算法。看到的集合跟踪以前看到的架构。

如果搜索碰巧找到以前见过的架构（第 8 和 9 行）,它会将适应度设置为 -1,这会导致基因型在选择过程中被丢弃。因为选择总是保留前 μ 个个体,一个以前看到的

算法 2 世代进化神经结构搜索。

```
1:程序EVONASSEARCH(n, m) 2:
maxGeneration ← m
3:看过← ∅                                   跟踪看到的架构 p ←
4:      InitializePopulationOfSize(n) for i = 1 to maxGenerations
5:      do
6:          p ← p ∪ p.produceOffspring() for o ∈ p
7:          do if o.isEvaluated() or o ∈ seen then if
8:              o ∈ seen then o.fitness ← −1 Drop genotype
9:                  end if continue end if 发送 o 到任务队列
10:                     seen。 r 的 add(o) 结束 ← ReceiveResults()
11:
12:                                          跳过评估
13:
14:
15:
16:
17:
18:         ProcessResults(r) p ←
19:         SelectTopLambda(r, λ) end for                (μ + λ)
20:     21:结束程序
```

个人将仍然处于最高个人集合中,或者将被表现更好的个人所取代。

无论哪种方式,我们都可以安全地丢弃重新发现的基因型。

我们通过使基因型不可变来进一步提高算法效率。一旦对基因类型进行了评估　通过构建、训练和评估相应的变分自动编码器　它的适应度就不会改变。这是该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。一个自然的问题是,如果基因型是不可变的,那么它们如何在进化算法中发生突变　我们创建了一个基因型的克隆,并在基因突变和交叉的创建过程中对其进行了修改。

四、相关工作

尽管已经有大量工作探索变分自动编码器的不同方面 [7]、[17]-[19],但据我们所知,之前没有探索进化神经架构搜索技术来设计变分自动编码器的工作。最相似的先前工作是 [20]、[21] 的工作探索了进化神经架构搜索技术在自动编码器中的应用。除了我们关注变分自动编码器而不是传统自动编码器之外,我们的工作不同之处在于它使用一种进化算法,该算法始终生成有效的网络架构,并且不需要编码器和解码器网络是相同的架构。

在神经结构搜索领域已经做了很多工作,主要应用于分类。 [8]、[11]、[22] 等工作侧重于基于强化学习的神经架构搜索方法。与我们的工作更相关的是

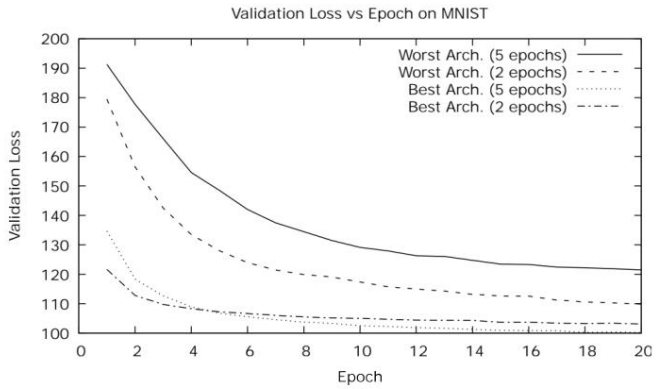


图 2:MNIST 数据集上最佳和最差架构的验证损失与时期数的比较。在架构搜索期间,网络在两个或五个时期后进行评估。

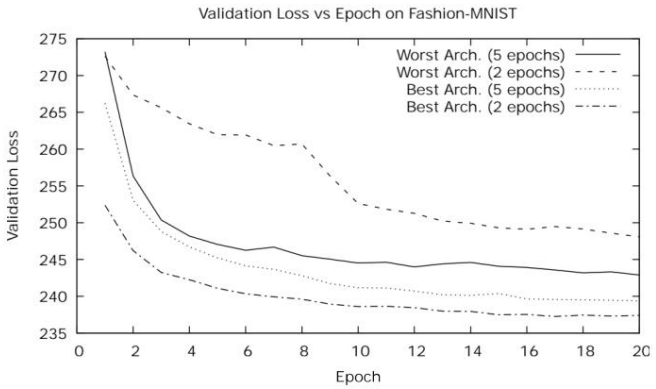


图 3:Fashion-MNIST 数据集上最佳和最差架构的验证准确性与轮数的比较。在架构搜索期间,网络在两个或五个时期后进行评估。

[9]、[23]、[24] 的研究重点是基于进化的方法。强化学习和基于进化算法的神经架构方法的大部分先前工作都集中在图像分类上,而不是学习生成模型。尽管与我们的工作的应用有所不同,但其中许多技术都适用于其他应用领域,例如应用于变分自编码器的神经体系结构搜索。

实验_

我们的实验探索了进化神经架构搜索寻找有效变分自动编码器的能力。

我们通过探索它们的潜在空间来评估找到的架构,并说明在架构搜索期间使用的时期数对找到的架构的训练速度的影响。

所有实验都是在 Amazon Web Services 服务器上使用 Nvidia K80 GPU 执行的。我们使用 PyTorch [25] 来实现神经网络。使用 MNIST [26] 和 Fashion MNIST [27] 评估变分自动编码器架构。我们使用 128 的小批量进行训练,Adam [28] 用于更新神经网络参数,除非另有说明,否则结果基于 20 个训练周期。由于资源限制,我们无法探索更大的数据集,例如 CIFAR-10 [29]。

表 I 显示了在

两个和五个时期搜索 MNIST 和时尚 MNIST 数据集。架构描述由三部分组成,中间用一条竖线隔开。第一部分表示编码器网络架构,其中线性层大小由逗号分隔。第二部分(在竖线之间)表示潜在维度,最后的逗号分隔列表表示解码器架构。

A. 时代对搜索的影响

尽管进化搜索算法中使用的时期数可能看起来像是一个小细节,但得到这个

超参数是否正确对搜索过程的进度和结果有着实实在在的影响。更复杂的神经网络 即具有更多参数的神经网络 通常需要更多遍历数据才能达到损失开始趋于平稳的点。这在图 2 和图 3 中都很清楚。

挑战在于,这个高原开始的时代不是先验的,并且会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期间找到的最佳和最差架构的验证损失相对于纪元的比较。在搜索过程中,神经网络架构在两个时期或五个时期后进行评估。正如预期的那样,通过两个时期选择搜索发现的架构从较低的损失开始,并且它们的损失减少得更快。这是仅两个时期后的早期评估所产生的选择压力的结果。所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时期训练的架构可能具有更好的整体性能,因为搜索能够更好地评估架构,而不是简单地选择训练速度更快的网络。这在性能最好的架构中可以看到,但在性能最差的架构中却没有。

图 3 显示了与图 2 相同的比较,但针对的是 Fashion-MNIST 数据集。 Fashion-MNIST 实验显示了与 MNIST 实验相似(甚至更极端)的结果。这些结果的有趣之处在于,在两个时期搜索中找到的最佳架构总是优于在五个时期搜索中找到的最佳架构。

这一结果在两个层面上令人惊讶。如上所述,我们期望更长的训练时间允许搜索过程更好地评估候选网络架构。此外,即使在五轮搜索中接受的额外训练对搜索没有帮助,也不应该损害搜索。

这很可能是两个纪元搜索的结果,以找到一个理想的架构,而五个纪元搜索,通过

表一:发现的架构

数据集	搜索时代排名		建筑学
MNIST	2个	最佳	850|20|1000
		最差	200、500|20|1000、650、50
	5个	最好	800, 900|30|150, 1000
		最差	850, 700, 400|30|850, 50, 1000, 900, 50
时尚-MNIST	2个	最好	1000|20|350, 1000
		最差	950, 850, 250, 200|10|150, 500, 750
	5个	最好	1000|60|700, 750 900, 50,
		最差	700|60|50, 750, 600, 950

偶然,没有发现。

B. 潜在空间流形

我们可以通过将潜在维度减少到两个并从单位正方形 [0, 1]×[0, 1] 上的高斯累积分布函数的倒数采样来可视化学习流形。图 4 显示了从两个时期搜索中找到的最佳 VAE 架构（如图 4a 所示)和在五个时期搜索中找到的最佳 VAE 架构（如图 4b 所示)学习的流形。重要的是要注意,此分析纯粹是定性的,并没有说明哪种模型更可取。尽管这两个流形之间有相似之处,但它们却大相径庭。图 4a 中描绘的歧管由大约 50% 的鞋类组成,而图 4b 中的歧管仅包含大约 15% 的鞋类。我们指出这一点是因为尽管我们期望流形彼此不同　它们是完全不同的网络架构　但有趣的是学习流形是如此不同。

C. 对潜在空间进行采样

图 5 比较了在两个和五个纪元搜索中找到的最佳架构的样本。从图 2 回想一下,与两个时期搜索的最佳架构相比,五时期搜索在 20 个时期后实现了更好的损失。图 5a 中的数字比图 5b 中的数字更模糊。例如,与图 5b 中的 8 相比,图 5a 中的许多 8 更难辨别,其中线条更实心,中心更明显。一般来说,图 5b 中的大多数数字都有较粗的线条,使它们更加清晰易读。比较这两个图很明显,在五个时期搜索中发现的变分自动编码器比在两个时期搜索中发现的体系结构更准确地了解了数据分布。这可能是由于更大的学习能力,因为五个时期架构中的神经网络更大并且具有更多参数。

更大的网络是更灵活的模型,因为可学习参数的数量增加,但它们也更有可能过度拟合数据。

D. 观察

我们注意到搜索的一个方面是包含信息瓶颈的架构,由两个高节点数层组成的三层层围绕一个大小为 50 的层,与其他架构相比往往表现不佳。

例如,Fashion MNIST 在五个时期搜索中表现最差的网络,如表 I 所示,有一层 900 个节点,其次是一层 50 个节点,然后是一层 700 个节点。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,这仅仅是因为 50 节点层没有传输它的能力。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令人意外　设计神经网络时的一种常见启发式方法是逐渐减小层的维度或逐渐增加层的维度。

六。结论

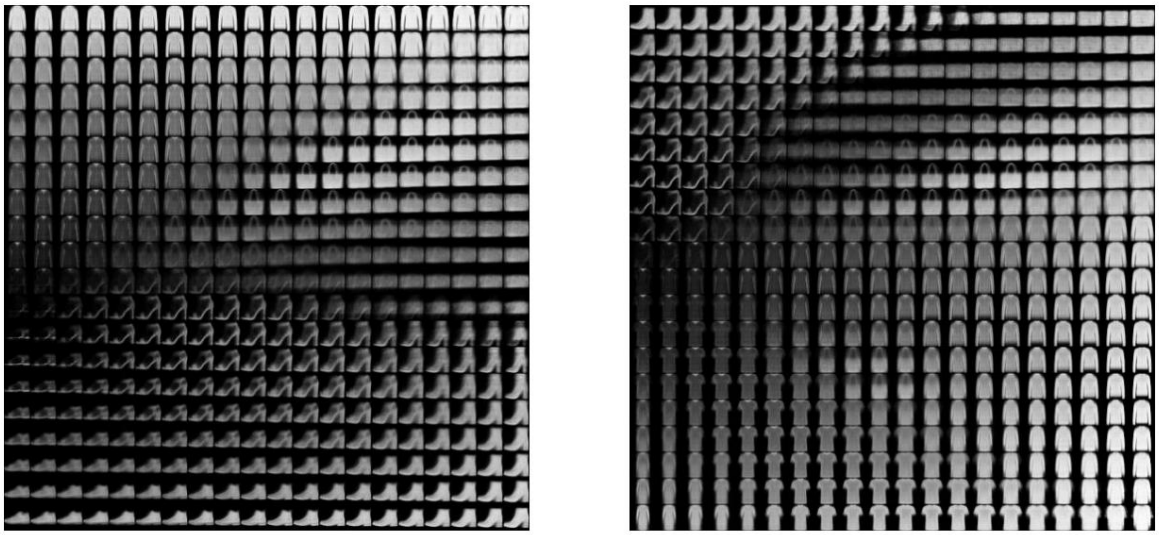
我们提出了一种用于进化变分自动编码器的进化算法,并证明了它在 MNIST 和 Fashion-MNIST 数据集上的有效性。我们的算法能够有效地为变分自编码器找到高性能架构。未来工作的一个领域是将我们使用的世代算法与锦标赛式或稳态 [30]、[31] 进行比较,这将通过减少工作人员的闲置来提高系统的吞吐量;然而,它也会影响选择过程,可能导致一些基因型幸存下来,否则这些基因型将被从种群中移除。未来工作的另一个领域是在更复杂的数据集上研究这种方法;由于计算资源的限制,我们无法探索更大的数据集。

神经架构搜索是深度学习的一个有前途的领域。在提高其效率和拓宽其应用领域方面的持续努力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

[1] DP Kingma 和 M. Welling,“自动编码变分贝叶斯”,第 2 届国际学习表示会议,ICLR 2014,班夫,AB,加拿大,2014 年 4 月 14-16 日,会议记录（Y. Bengio 和Y. LeCun 编）,2014 年。

[2] T. Hrycej,神经网络中的模块化学习:神经网络分类的模块化方法。美国纽约州纽约市:John Wiley & Sons, Inc.,第 1 版,1992 年。



(一种) (乙)

图 4:(a) 从两个时期搜索中找到的最佳架构中学习流形,训练了 20 个时期。 (b) 学到的
从五个时期搜索中找到的最佳架构中提取的流形,经过 20 个时期的训练。

(一种) (乙)

图 5:(a) p (x |z)的样本来自在 100 代后的两个时期搜索中找到的最佳架构,训练了 20 个时期。 (b) p(x|z)的样本来自在 100 代后的五个时期搜索中找到的最佳架构,训练了 20 个时期。

[3] P. Vincent、H. Larochelle、I. Lajoie、Y. Bengio 和 P. Manzagol，“堆叠式降噪自动编码器:使用局部降噪标准在深度网络中学习有用的表示”,J. Mach。学习。水库,卷。 11,第 3371–3408 页,2010 年。

[4] P. Vincent、H. Larochelle、Y. Bengio 和 P.-A。 Manzagol，“Extracting and composing robust features with denoising autoencoders”,第 25 届国际机器学习会议论文集,ICML 08，(美国纽约州纽约市),第 1096-1103 页,ACM,2008 年。

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

[6] G. Converse、M. Curi 和 S. Oliveira，“用于教育评估的自动编码器”,国际教育人工智能会议,2019 年。

[7] B. Dai、Y. Wang、J. Aston、G. Hua 和 DP Wipf，“变分自动编码器的隐藏天赋”,CoRR，卷。 abs/1706.05148, 2017.

[8] B. Zoph 和 QV Le，“使用强化学习进行神经架构搜索”,2017 年。

[9] R. Miikkulainen、JZ Liang、E. Meyerson、A. Rawal、D. Fink、O. Fran con、B. Raju、H. Shahrzad、A. Navruzyan、N. Duffy 和 B. Hodjat，“进化深度神经网络，”CoRR,卷。 abs/1703.00548, 2017.

[10] J. Koutník、G. Cuccu、J. Schmidhuber 和 F. Gomez，“为基于视觉的强化学习进化大规模神经网络”,第 15 届遗传和进化计算年会论文集,GECCO 13，(美国纽约州纽约市),第 1061–1068 页,ACM,2013 年。

[11] H. Pham、M. Guan、B. Zoph、Q. Le 和 J. Dean，“通过参数共享进行高效神经架构搜索”,第 35 届会议论文集

机器学习国际会议（J. Dy 和 A. Krause 编着），第一卷。 80 机器学习研究论文集，（Stock holmsmässan，瑞典斯德哥尔摩），第 4095–4104 页，PMLR，7 2018。

[12] K. Hornik、MB Stinchcombe 和 H. White，“多层前馈网络是通用逼近器”，神经网络，卷。 2，没有。 5，第 359–366 页，1989 年。

[13] S. Kullback，“致编辑的信：kullback-leibler 距离”，美国统计学家，卷。 41，第 340-341 页，1987 年第 11 页。

[14] K. Varda，“协议缓冲区：谷歌的数据交换格式”，tech。代表，Google，2008 年 6 月。

[15] 谷歌，“grpc. Accessed on March 2019”

[16] X. Wang、H. Zhao 和 J. Zhu，“Grpc：分布式系统中的通信协作机制”，SIGOPS Oper。系统。牧师，卷。 27，第 75-86 页，1993 年 7 月。

[17] CK Sønderby、T. Raiko、L. Maaløe、SK Sønderby 和 O. Winther，“阶梯变分自动编码器”，神经信息处理系统进展，第 3738–3746 页，2016 年。

[18] D. Kingma、T. Salimans、R. Josefowicz、X. Chen、I. Sutskever、M. Welling 等人，“使用逆自回归流改进变分自动编码器”，2017 年。

[19] X. Hou、L. Shen、K. Sun 和 G. Qiu，“深度特征一致变分自动编码器”，2017 年 IEEE 计算机视觉应用冬季会议 (WACV)，第 1133–1141 页，IEEE，2017。

[20] S. Lander 和 Y. Shang，“Evoae 一种用于深度学习网络自动编码器训练的新进化方法”，2015 年 IEEE 第 39 届年度计算机软件和应用大会，卷。 2，第 790–795 页，2015 年第 7 页。

[21] M. Suganuma、M. Ozay 和 T. Okatani，“通过进化搜索利用标准卷积自动编码器进行图像恢复的潜力”，第 35 届国际机器学习会议论文集（J. Dy 和 A.克劳斯编），卷。 80 机器学习研究论文集，（斯德哥尔摩，瑞典斯德哥尔摩），第 4771–4780 页，PMLR，2018 年第 7 期。

[22] B. Zoph、V. Vasudevan、J. Shlens 和 QV Le，“学习用于可扩展图像识别的可迁移架构”，载于 IEEE 会刊计算机视觉和模式识别会议，第 8697–8710 页，2018 年。

[23] H. Liu、K. Simonyan、O. Vinyals、C. Fernando 和 K. Kavukcuoglu，“高效架构搜索的层次表示”，CoRR，卷。 abs/1711.00436，2017。

[24] E. Real、S. Moore、A. Selle、S. Saxena、YL Suematsu、J. Tan、QV Le 和 A. Kurakin，“图像分类器的大规模演化”，第 34 届国际机器学习会议论文集，ICML 2017，澳大利亚新南威尔士州悉尼，2017 年 8 月 6 日至 11 日（D. Precup 和 YW Teh，编），卷。 70 机器学习研究论文集，第 2902–2911 页，PMLR，2017 年。

[25] A. Paszke、S. Gross、S. Chintala、G. Chanan、E. Yang、Z. DeVito、Z. Lin、A. Desmaison、L. Antiga 和 A. Lerer，“PyTorch 中的自动微分，”在 NIPS Autodiff Workshop，2017 年。

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

[27] H. Xiao、K. Rasul 和 R. Vollgraf，“Fashion-mnist：一种用于基准机器学习算法的新型图像数据集”，2017 年。

[28] DP Kingma 和 J. Ba，“亚当：一种随机优化方法”，第 3 届国际学习表征会议，ICLR 2015，美国加利福尼亚州圣地亚哥，2015 年 5 月 7 日至 9 日，会议记录（Y. Bengio 和 Y. LeCun 合编），2015 年。

[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. Hasenjäger，“并行架构的稳态和世代进化策略的比较”，来自自然的并行问题解决 - PPSN VIII（X. Yao， EK Burke、JA Lozano、J. Smith、JJ Merelo-Guervós、JA Bullinaria、JE Rowe、P. Tino、A. Kabán 和 H.-P. Schwefel，编），（柏林，海德堡），第 253 页–262，施普林格柏林海德堡出版社，2004 年。

[31] A.-C. Zavoianu、E. Lughofer、W. Koppelstätter、G. Weidenholzer、W. Amrhein 和 EP Klement，“针对计算密集型问题的生成和稳态异步多目标进化算法的性能比较”，Know.-Based Syst .，卷。 87，第 47-60 页，2015 年 10 月。