翻译：www.gwylab.com 原文来源：https://arxiv.org/pdf/1410.8516.pdf

arXiv:1410.8516v6 [cs.LG] 10 Apr 2015

NICE：非线性独立成分估计

Laurent Dinh David Krueger Yoshua Bengio
Departement´ d'informatique et de recherche operationnelle´
Universite´ de Montreal´
Montreal,´ QC H3C 3J7

摘 要

我们提出了一种深度学习框架，用于建模复杂的高维密度，称为非线性独立分量估计（NICE）。它基于这样的想法：良好的表示是指数据具有易于建模的分布。为此目的，学习数据的非线性确定性变换，将其映射到潜在空间，以使变换数据符合分解分布，即，产生独立的潜变量。我们对这种变换进行参数化，以使得计算雅可比行列式和逆雅可比行列式是简易的，并且我们通过构造简单的构建块来保持学习复杂的非线性变换的能力，每个构建块都基于深度神经网络。训练标准只是精确的对数似然，它是易处理的。无偏的原始采样也较为容易。我们证明这种方法可以在四个图像数据集上产生良好的生成模型，并可用于修复。

1 介绍

无监督学习的核心问题之一是如何捕获具有未知结构的复杂数据分布。 深度学习方法（Bengio，2009）依赖于学习数据的表示，这些数据将捕获其最重要的变异因素。 这引出了一个问题：什么是好的表现？ 与最近的工作一样（Kingma 和 Welling，2014; Rezende 等，2014; Ozair 和 Bengio，2014），我们认为良好的表示是一种数据分布易于建模的表示。 在本文中，我们考虑这样一种特殊情况，即我们要求学习者将数据的变换 h = f（x）投射到一个新的空间中，以便得到的分布能够分解，并且组件 hd 是独立的:

$$p_H(h) = \prod_d p_{H_d}(h_d).$$

建议的训练标准直接来自对数似然。 更具体地，我们考虑变量 h = f（x）的变化，假设 f 是可逆的并且 h 的维度与 x 的维度相同，以便适合分布 PH。 变量规则的变化给出了

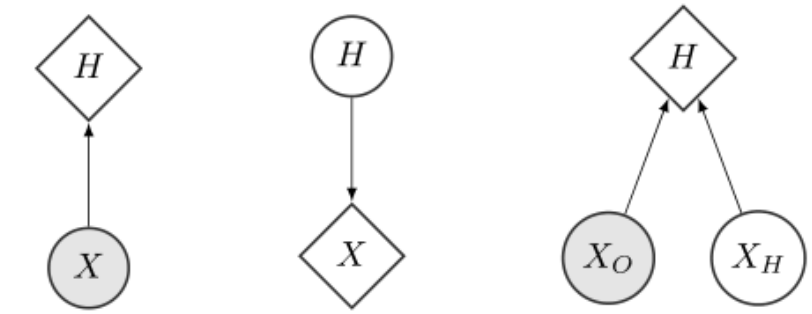
$$p_X(x) = p_H(f(x))|\det \frac{\partial f(x)}{\partial x}|. \tag{1}$$

其中 $\frac{\partial f(x)}{\partial x}$ 是 x 处函数 f 的雅可比矩阵。 在本文中，我们选择 f 使雅可比行列式能简单地获得。此外，还可以简单地获得其逆 f^{-1}，从而允许我们如下容易地从 Px(x)进行采样:

$$\begin{aligned} h &\sim p_H(h) \\ x &= f^{-1}(h) \end{aligned} \tag{2}$$

本文的一个关键新颖之处在于设计了这样一个变换 f，它产生了“雅可比的简易行列式”和“易逆”这两个属性，同时允许我们拥有尽可能多的容量

Yoshua Bengio 是 CIFAR 的高级研究员。



(a) Inference (b) Sampling (c) Inpainting

图 1：概率模型的计算图，使用以下公式.

(a) 推论: $\log(p_X(x)) = \log(p_H(f(x))) + \log(|\det(\frac{\partial f(x)}{\partial x})|)$

(b) 采样: $h \sim p_H(h), x = f^{-1}(h)$

(c) 修复:

$$\max_{x_H} \log(p_X((x_O, x_H))) = \max_{x_H} \log(p_H(f((x_O, x_H)))) + \log(|\det(\frac{\partial f((x_O, x_H))}{\partial x})|)$$

根据需要学习复杂的转换。 这背后的核心思想是我们可以将 x 分成两个块（x1; x2）并作为构建块应用从（x1; x2）到（y1; y2）形式的转换:

$$\begin{aligned} y_1 &= x_1 \\ y_2 &= x_2 + m(x_1) \end{aligned} \tag{3}$$

其中 m 是任意复杂的函数（在我们的实验中是 ReLU MLP）。 这个构建块具有任何 m 的单位雅可比行列式，并且从那时起是可逆的:

$$\begin{aligned} x_1 &= y_1 \\ x_2 &= y_2 - m(y_1). \end{aligned} \tag{4}$$

详细信息，周围讨论和实验结果如下。

2 学习连续概率的双射转换

我们考虑从密度为 $\{p_\theta, \theta \in \Theta\}$的参数族中学习概率密度的问题，在有限数据集 D 上选取 N 个例子，每个都存在于空间 X 中; 通常 $\mathcal{X} = \mathbb{R}^D$。

我们的特殊方法包括使用以下变量公式，通过最大似然学习数据的连续分布，从而学会如何把可微分的几乎无处不在的非线性变换 f 变换到更简单的分布的方法:

$$\log(p_X(x)) = \log(p_H(f(x))) + \log(|\det(\frac{\partial f(x)}{\partial x})|)$$

其中 pH（h），即先验分布，是预定义的密度函数，例如一个标准的等向性高斯函数。 如果先验分布是阶乘的（即具有独立维度），那么我们获得以下非线性独立分量估计（NICE）标准，这是我们的数据生成模型下的最大似然作为因子分布的确定性变换:

$$\log(p_X(x)) = \sum_{d=1}^{D} \log(p_{H_d}(f_d(x))) + \log(|\det(\frac{\partial f(x)}{\partial x})|)$$

其中$f(x) = (f_d(x))_{d \leq D}$。

我们可以将 NICE 视为学习数据集的可逆预处理变换。 可逆的预处理可以简单地通过收缩数据来任意增加概率。 我们使用的变化

[1]请注意，此先前分布不需要是常量，也可以学习

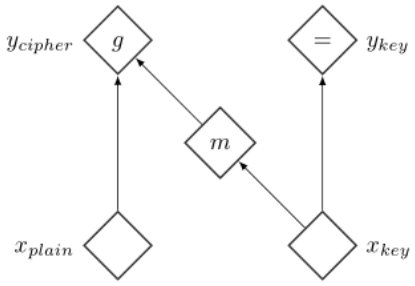


图 2：耦合层的计算图

变量公式（公式 1）准确地抵消这种现象，并使用先前 pH 的分解结构来鼓励模型在数据集中发现有意义的结构。在该公式中，变换 f 的雅可比矩阵的行列式判断收缩并且根据需要促进高密度区域（即在数据点处）的扩展。如 Bengio 等人所述（2013），表示学习倾向于扩大输入中更“有趣”区域的空间的体积（例如，在无监督学习情况下的高密度区域）。

与先前使用自动编码器的工作一致，特别是变分自动编码器（Kingma 和 Welling，2014; Rezende 等，2014; Mnih 和 Gregor，2014; Gregor 等，2014），我们称之为编码器并且它的倒数 f^{-1} 是解码器。给定 f^{-1}，模型的采样可以通过有向图形模型 H → X 中的原始采样非常容易地进行，如等式 2 中所述。

3 架构

3.1 三角结构

该模型的结构对于获得一系列双射是至关重要的，其中这些双射的雅可比行列式是易处理的并且计算是直截了当的，不论是前向（编码器 f）还是后向（解码器 f^{-1}）。如果我们使用分层或组合变换 $f = f_L \circ \ldots \circ f_2 \circ f_1$，前向和后向计算是其层计算的组合（按照适当的顺序），其雅可比行列式是其层的雅可比行列式的乘积。因此，我们首先要着眼于定义那些更基本的组成部分。

首先，我们考虑仿射变换。 （Rezende 等，2014）和（Kingma 和 Welling，2014）在使用对角矩阵或具有 秩-1 校正的对角矩阵作为变换矩阵时，提供了逆和行列式的公式。另一类具有易处理行列式的矩阵是三角矩阵，其决定因素只是其对角线元素的乘积。在测试时反转三角矩阵在计算方面是合理的。许多平方矩阵 M 也可以表示为上三角矩阵和下三角矩阵的乘积 M = LU。由于可以组成这样的变换，我们看到这些组合物的有用组分包括雅可比对角线，下三角形或上三角形的组合物。

使用这种观察的一种方法是建立具有三角形权重矩阵和双射激活函数的神经网络，但这高度约束了该体系结构，将设计选择限制为深度和非线性选择。或者，我们可以考虑具有三角雅可比矩阵的函数族。通过确保雅可比行列式的对角元素易于计算，雅可比行列式的行列式也易于计算。

3.2 耦合层

在这一小节中，我们描述了一个具有三角形雅可比矩阵因此易处理的雅可比行列式的双射变换族。 这将成为转型 f 的基石。

一般耦合层 设 x ∈ X，I1,I2 是$[\![1, D]\!]$的分区使得 d =|I1|和 m 是在 Rd 上定义的函数，我们可以定义 y =（yI1，yI2）其中:

$$y_{I_1} = x_{I_1}$$
$$y_{I_2} = g(x_{I_2}; m(x_{I_1}))$$

其中$g : \mathbb{R}^{D-d} \times m(\mathbb{R}^d) \to \mathbb{R}^{D-d}$是耦合定律，是关于第二个参数的第一个参数的可逆映射。 相应的计算图如图 2 所示。如果我们考虑 I1 = $[\![1, D]\!]$和 I2 = $[\![d, D]\!]$，雅各比这个功能是:

$$\frac{\partial y}{\partial x} = \begin{bmatrix} I_d & 0 \\ \frac{\partial y_{I_2}}{\partial x_{I_1}} & \frac{\partial y_{I_2}}{\partial x_{I_2}} \end{bmatrix}$$

其中 Id 是大小为 d 的单位矩阵。 这意味着$\det \frac{\partial y}{\partial x} = \det \frac{\partial y_{I_2}}{\partial x_{I_2}}$。 此外，我们观察到我们可以使用以下方法反转映射

$$x_{I_1} = y_{I_1}$$
$$x_{I_2} = g^{-1}(y_{I_2}; m(y_{I_1}))$$

我们将这种变换称为具有耦合函数 m 的耦合层。

加性耦合层 为简单起见，我们选择加法耦合定律 g（a; b）= a + b 作为耦合定律，以便通过取 a = xI2 和 b = m（xI1）：

$$y_{I_2} = x_{I_2} + m(x_{I_1})$$
$$x_{I_2} = y_{I_2} - m(y_{I_1})$$

因此，计算这种变换的逆矩阵与计算变换本身一样昂贵。 我们强调，对耦合函数 m 的选择没有限制（除了具有适当的域和陪域）。 例如，m 可以是具有 d 个输入单元和 D-d 个输出单元的神经网络。

此外，由于$\det \frac{\partial y_{I_2}}{\partial x_{I_2}} = 1$，附加耦合层变换除了其平凡的逆之外还具有单位雅可比行列式。 人们还可以选择其他类型的耦合，例如多重耦合定律$g(a; b) = a \odot b,\ b \neq 0$或仿射耦合定律$g(a; b) = a \odot b_1 + b_2,\ b_1 \neq 0$如果$m : \mathbb{R}^d \to \mathbb{R}^{D-d} \times \mathbb{R}^{D-d}$。出于数值稳定性原因，我们选择了加性耦合层，因为当神经网络是整流神经网络时，耦合函数 m 变换成分段线性

组合耦合层 我们可以组合多个耦合层以获得更复杂的分层变换。 由于耦合层使其输入的一部分保持不变，我们需要在交替层中交换分区中两个子集的作用，以便两个耦合层的组合修改每个维度。 检查雅可比行列式，我们观察到至少需要三个耦合层才能使所有尺寸相互影响。 我们一般使用四个。

3.3 允许重新缩放

由于每个附加耦合层具有单位雅可比行列式（即，体积保持），它们的成分也必然具有单位雅可比行列式。 为了解决这个问题，我们将对角缩放矩阵 S 作为顶层，将第 i 个输出值乘以 Sii：$(x_i)_{i \leq D} \to (S_{ii} x_i)_{i \leq D}$。这允许学习者在某些维度上给予更多权重（即，模型更多变化）而在其他维度上更少。

在 Sii 对于某些 i 变为+∞的极限中，数据的有效维数已减少 1。只要 f 在数据点周围保持可逆，这是可能的。对于其余的这种缩放的对角线最后阶段以及其余的下三角形或上三角形阶段（具有对角线中的标识），NICE 标准具有以下形式:

$$\log(p_X(x)) = \sum_{i=1}^{D} [\log(p_{H_i}(f_i(x))) + \log(|S_{ii}|)].$$

翻译：www.gwylab.com 原文来源：https://arxiv.org/pdf/1410.8516.pdf

我们可以将这些缩放因子与 PCA 的本征谱相关联，显示每个潜在维度中存在多少变化（Sii 越大，维度 i 越不重要）。 可以将频谱的重要维度视为算法学习的多样性。 前一个术语鼓励 Sii 很小，而决定性项 log Sii 阻止 Sii 达到 0。

3.4 先验分布

如前所述，我们选择先前的分布为阶乘，即:

$$p_H(h) = \prod_{d=1}^{D} p_{H_d}(h_d)$$

我们通常在标准分布系列中选择此分布，例如 高斯分布：

$$\log(p_{H_d}) = -\frac{1}{2}(h_d^2 + \log(2\pi))$$

或者逻辑斯谛分布:

$$\log(p_{H_d}) = -\log(1 + \exp(h_d)) - \log(1 + \exp(-h_d))$$

我们倾向于使用逻辑分布，因为它倾向于提供更好的行为梯度。

4 相关方法

在生成模型方面取得了重大进展。由于这些模型允许的有效近似推理和学习技术，像 Boltzmann 深层机器（DBM）（Salakhutdinov 和 Hinton，2009）这样的无向图形模型已经非常成功并且是一个激烈的研究课题。然而，这些模型需要用于训练和采样的马尔可夫链蒙特卡罗（MCMC）采样程序，并且当目标分布具有尖锐模式时，这些链通常缓慢混合。此外，对数似然是难以处理的，最著名的估计程序，退火重要性抽样（AIS）（Salakhutdinov 和 Murray，2008），可能会产生过于乐观的评估（Grosse 等，2013）。

定向图形模型缺乏允许 DBM 有效推理的条件独立结构。然而，最近，变分自动编码器（VAE）的发展（Kingma 和 Welling，2014; Rezende 等，2014; Mnih 和 Gregor，2014; Gregor 等，2014） - 在训练期间允许有效的近似推断。与 NICE 模型相比，这些方法使用随机编码器 q（h | x）和不完美解码器 p（x | h），需要成本中的重构项，确保解码器近似反转编码器。这将噪声注入自动编码器循环，因为 h 是从 q（h | x）采样的，这是对真实后验 p（h | x）的变分近似。得到的训练标准是数据对数似然的变分下界。指导图形模型的通常快速的原始采样技术使这些模型具有吸引力。此外，保证对数似然的重要性抽样估计器在期望中不乐观。但是使用下限标准可能会产生关于真实对数似然的次优解。这种次优的解决方案可能例如在生成过程中注入大量非结构化噪声，导致不自然的样本。在实践中，我们可以输出 p（x | h）的统计量，如期望值或中位数，而不是实际样本。确定性解码器的使用可以通过消除低级噪声的目的来推动，低级噪声在诸如 VAE 和 Boltzmann 机器的模型中被自动地添加到生成的最后阶段（可见被认为是独立的，给定隐藏的）。

NICE 标准与变分自动编码器的标准非常相似。更具体地说，由于变换及其逆可以被视为完美的自动编码器对（Bengio，2014），因此重构项是可以忽略的常数。这留下了变换标准的 Kullback-Leibler 发散项：$log(p_H(f(x)))$可以看作先前项，这迫使代码 h = f（x）可能相对于先验分布和$\log(|\det \frac{\partial f(x)}{\partial x}|)$可以看作是熵项。该熵项反映了数据周围的局部体积扩展（对于编码器），其转化为解码器 f^{-1} 中的收缩。以类似的方式，熵

变分标准中的术语鼓励近似后验分布占据体积，这也转化为来自解码器的收缩。完美重建的结果是我们还必须在顶层 h 对噪声进行建模，而在这些其他图形模型中它通常由条件模型 p（x | h）处理。

我们还观察到，通过将变分标准与重新参数化技巧相结合，（Kingma 和 Welling，2014）在具有两个仿射耦合层的 NICE 模型中有效地最大化了对（x; ϵ）的联合对数似然（其中 ϵ 是辅助的）噪声变量）和高斯先验，见附录 C.

概率密度函数的变量公式的变化主要用于逆变换采样（这实际上是用于此处采样的过程）。独立分量分析（ICA）（Hyvarinen 和 Oja，2000），更具体地说是其最大似然公式，学习数据的正交变换，需要在参数更新之间进行昂贵的正交化过程。在（Bengio，1991）中提出了学习更丰富的变换族，但是所提出的变换类，神经网络，一般缺乏使推理和优化实用的结构。 （Chen 和 Gopinath，2000）学会了分层转换为高斯分布，但是以贪婪的方式，它无法提供易处理的采样程序。

（Rippel 和 Adams，2013）重新引入了学习这些转换的想法，但由于缺乏生物学约束，因此被迫进入正则化自动编码器设置作为对数似然最大化的代理。通过集成学习（Roberts 和 Everson，2001; Lappalainen 等，2000），在非线性独立分量分析（Hyvarinen 和 Pajunen，1999）中更成功地使用了对数似然的更有原则的代理 - 变分下界。 （Kingma 和 Welling，2014; Rezende 等，2014）使用一种亥姆霍兹机器（Dayan et al。，1995）。生成对抗网络（GAN）（Goodfellow 等，2014）也训练生成模型以将简单（例如因子）分布转换为数据分布，但不需要编码器进入另一个方向。 GAN 通过学习区分 GAN 样本和数据的二级深度网络来回避推理的困难。该分类器网络向 GAN 生成模型提供训练信号，告诉它如何使其输出与训练数据的区别较小。

与变分自动编码器一样，NICE 模型使用编码器来避免不一致的困难，但其编码是确定性的。对数似然是易处理的，并且训练过程不需要任何采样（除了对数据进行反量化）。 NICE 中用于获得易处理性的三角形结构也出现在另一个易处理密度模型家族中，即神经自回归网络（Bengio 和 Bengio，2000），其中包括神经自回归密度估计（NADE）的最新成功例子。 （Larochelle 和 Murray，2011 年）。实际上，NADE 定向图形模型中的邻接矩阵是严格的三角形。然而，逐元素自回归方案使得原始采样过程计算成本高且对于高维数据（例如图像数据）上的生成任务而言不可并行化。使用一个耦合层的 NICE 模型可以看作具有两个块的 NADE 的块版本。

5 实验

5.1 对数似然和生成

我们在 MNIST（LeCun 和 Cortes，1998），多伦多面部数据集 2（TFD）（Susskind 等人，2010），街景房屋数字数据集（SVHN）（Netzer 等，2011）和 CIFAR-10 上培训 NICE（Krizhevsky，2010）。 按照（Uria et al，2013）的规定，我们使用数据的反量化版本：我们向数据添加 $\frac{1}{256}$ 的均匀噪声并将其重新调整为 $[0, 1]^D$ 去量化后。 我们添加 $\frac{1}{128}$ 的均匀噪声并将数据重新为 CIFAR-10 调整为 $[-1, 1]^D$。

使用的体系结构是四个耦合层的堆栈，最后阶段具有对角线正缩放（参数化指数）exp（s），并且在 SVHN 和 CIFAR-10 上具有近似的 TFD 白化和精确 ZCA。 我们通过分离奇数（I1）和偶数（I2）来划分输入空间

[2] 我们训练此数据集的未标记数据.

Dataset	MNIST	TFD	SVHN	CIFAR-10
# dimensions	784	2304	3072	3072
Preprocessing	None	Approx. whitening	ZCA	ZCA
# hidden layers	5	4	4	4
# hidden units	1000	5000	2000	2000
Prior	logistic	gaussian	logistic	logistic
Log-likelihood	1980.50	5514.71	11496.55	5371.78

图 3：架构和结果。 # hidden 单位是指每个隐藏层的单位数。

Model	TFD	CIFAR-10
NICE	5514.71	5371.78
Deep MFA	5250	3622
GRBM	2413	2365

图 4：TFD 和 CIFAR-10 的对数似然结果。 请注意，深 MFA 编号对应于（Tang et al，2012）获得的最佳结果，但实际上是变化下限。

组件，所以等式是:

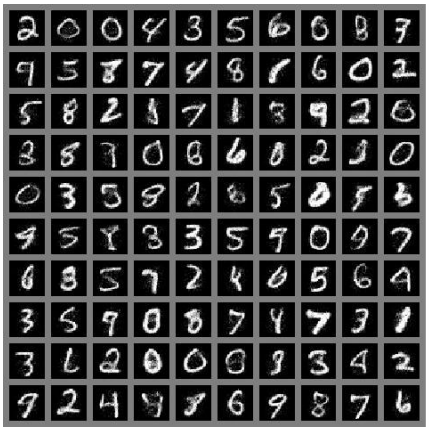
$$
\begin{aligned}
h^{(1)}_{I_1} &= x_{I_1} \\
h^{(1)}_{I_2} &= x_{I_2} + m^{(1)}(x_{I_1}) \\
\\
h^{(2)}_{I_2} &= h^{(1)}_{I_2} \\
h^{(2)}_{I_1} &= h^{(1)}_{I_1} + m^{(2)}(x_{I_2}) \\
\\
h^{(3)}_{I_1} &= h^{(2)}_{I_1} \\
h^{(3)}_{I_2} &= h^{(2)}_{I_2} + m^{(3)}(x_{I_1}) \\
\\
h^{(4)}_{I_2} &= h^{(3)}_{I_2} \\
h^{(4)}_{I_1} &= h^{(3)}_{I_1} + m^{(4)}(x_{I_2}) \\
\\
h &= \exp(s) \odot h^{(4)}
\end{aligned}
$$

耦合函数 m（1）,m（2）;用于耦合层的 m（3）和 m（4）都是具有线性输出单元的深度整流网络。我们对每个耦合函数使用相同的网络体系结构：MNIST 为 1000 个单位的五个隐藏层，TFD 为 5000 个四个，SVHN 和 CIFAR-10 为 2000 个。

标准逻辑分布用作 MNIST，SVHN 和 CIFAR-10 的先验。标准正态分布用作 TFD 的先验。

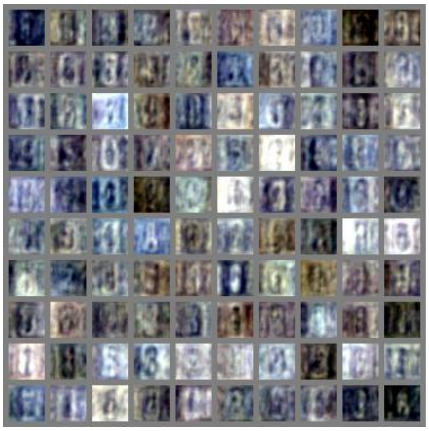
通过使用 AdaM（Kingma 和 Ba，2014）最大化对数似然对数 $\log(p_H(h)) + \sum_{i=1}^{D} s_i$ 来训练模型，学习率为 10-3，动量 0.9, β2= 0.01，λ=1 ，和ϵ= 10-4。我们在 1500 个时期之后根据验证对数似然选择最佳模型。

我们在 MNIST 上获得了 1980.50 的测试对数似然，在 TFD 上获得了 5514.71，对于 SVHN 获得了 11496.55，对于 CIFAR-10 获得了 5371.78。这与我们在对数似然方面所知的最佳结果相比：TFD 为 5250，CIFAR-10 为 3622，因子分析仪的深度混合（Tang 等，2012）（虽然它仍然是下限），由于连续 MNIST 的生成模型通常使用 Parzen 窗口估计进行评估，因此无法进行公平比较。由训练模型生成的样本如图 5 所示。



(a) Model trained on MNIST

(b) Model trained on TFD

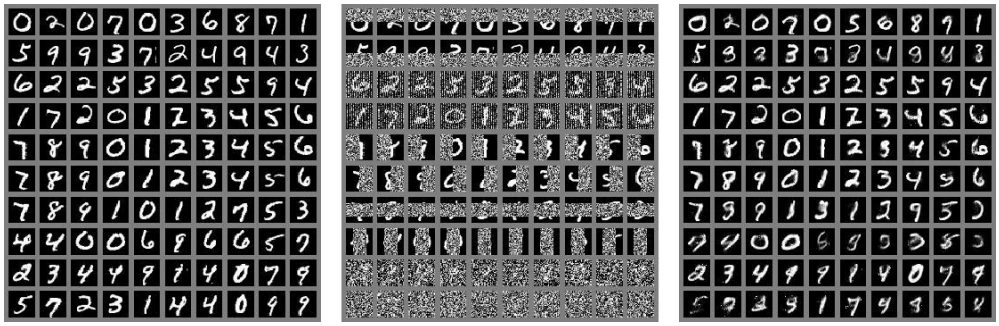


(c) Model trained on SVHN

(d) Model trained on CIFAR-10

图 5：来自训练有素的 NICE 模型的无偏样本。 我们采样 $h \sim p_H(h)$ 并输出:

$$x = f^{-1}(h).$$



(a) MNIST test examples (b) Initial state (c) MAP inference of the state

图 6：在 MNIST 上修复。 我们在下面列出了从上到下依次覆盖上图中间图的图像部分的类型：顶行，底行，奇数像素，偶数像素，左侧，右侧，中间垂直，中间水平，75 %随机，90%随机。 我们将未屏蔽的像素钳位到其基本事实值，并通过投影梯度上升的可能性来推断屏蔽像素的状态。 请注意，对于中间掩码，几乎没有关于数字的可用信息。

5.2 修复

在这里，我们考虑一个简易的迭代过程来实现训练的生成模型的修复。 对于修复，我们将观察到的尺寸（XO）钳制到它们的值，并使用投影梯度上升（以保持输入在其原始值的间隔中）使用步长$\alpha_i = \frac{10}{100+i}$的高斯噪声最大化相对于隐藏尺寸（XH）的对数似然性，其中 i 是迭代，随机梯度更新:

$$x_{H,i+1} = x_{H,i} + \alpha_i(\frac{\partial \log(p_X((x_O, x_{H,i})))}{\partial x_{H,i}} + \epsilon)$$
$$\epsilon \sim \mathcal{N}(0, I)$$

其中 XH，i 是迭代 i 处隐藏维度的值。 结果显示在 MNIST 的测试例子中，如图 6 所示。尽管该模型没有经过这项任务的训练，但修复程序似乎产生了合理的定性性能，但需注意偶尔会存在虚假模式。

6 结论

在这项工作中，我们提出了一种新的灵活架构，用于学习高度非线性的双射变换，将训练数据映射到其分布被分解的空间，以及通过直接最大化对数似然来实现这一目标的框架。 NICE 模型具有高效无偏的原始采样，并在对数似然方面取得了有竞争力的结果。

请注意，我们模型的架构可以使用其他能够发挥其优势的归纳原理进行训练，如环形子空间分析（TSA）（Cohen 和 Welling，2014）。

我们还简要地与变分自动编码器建立了联系。 我们还注意到，NICE 可以实现更强大的近似推理，从而允许在这些模型中更复杂的近似后验分布族，或更丰富的先验族。

致 谢

我们要感谢 Yann Dauphin，Vincent Dumoulin，Aaron Courville，Kyle Kastner，Dustin Webb，Li Yao 和 Aaron Van den Oord 的讨论和反馈。 Vincent Dumoulin 提供了可视化代码。我们感谢 Theano 的开发人员（Bergstra 等人，2011；Bastien 等人，2012）和 Pylearn2（Goodfellow 等人，2013），以及计算资源

由 Compute Canada 和 Calcul Quebec 提供，以及由 NSERC，CIFAR 和加拿大研究主席提供的研究经费。

参考文献

Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I. J., Bergeron, A., Bouchard, N., and Bengio, Y. (2012). Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.

Bengio, Y. (1991). Artificial Neural Networks and their Application to Sequence Recognition. PhD thesis, McGill University, (Computer Science), Montreal, Canada.

Bengio, Y. (2009). Learning deep architectures for AI. Now Publishers.

Bengio, Y. (2014). How auto-encoders could provide credit assignment in deep networks via target propagation. Technical report, arXiv:1407.7906.

Bengio, Y. and Bengio, S. (2000). Modeling high-dimensional discrete data with multi-layer neural networks. In Solla, S., Leen, T., and Muller,¨ K.-R., editors, Advances in Neural Information Processing Systems 12 (NIPS'99), pages 400–406. MIT Press.

Bengio, Y., Mesnil, G., Dauphin, Y., and Rifai, S. (2013). Better mixing via deep representations. In Proceedings of the 30th International Conference on Machine Learning (ICML'13). ACM.

Bergstra, J., Bastien, F., Breuleux, O., Lamblin, P., Pascanu, R., Delalleau, O., Desjardins, G., Warde-Farley, D., Goodfellow, I. J., Bergeron, A., and Bengio, Y. (2011). Theano: Deep learning on gpus with python. In Big Learn workshop, NIPS'11.

Chen, S. S. and Gopinath, R. A. (2000). Gaussianization.

Cohen, T. and Welling, M. (2014). Learning the irreducible representations of commutative lie groups. arXiv:1402.4437.

Dayan, P., Hinton, G. E., Neal, R., and Zemel, R. (1995). The Helmholtz machine. Neural Compu-tation, 7:889–904.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks. Technical Report arXiv:1406.2661, arxiv.

Goodfellow, I. J., Warde-Farley, D., Lamblin, P., Dumoulin, V., Mirza, M., Pascanu, R., Bergstra, J., Bastien, F., and Bengio, Y. (2013). Pylearn2: a machine learning research library. arXiv preprint arXiv:1308.4214.

Gregor, K., Danihelka, I., Mnih, A., Blundell, C., and Wierstra, D. (2014). Deep autoregressive networks. In International Conference on Machine Learning (ICML'2014).

Grosse, R., Maddison, C., and Salakhutdinov, R. (2013). Annealing between distributions by aver-aging moments. In ICML'2013.

Hyvarinen,¨ A. and Oja, E. (2000). Independent component analysis: algorithms and applications. Neural networks, 13(4):411–430.

Hyvarinen,¨ A. and Pajunen, P. (1999). Nonlinear independent component analysis: Existence and uniqueness results. Neural Networks, 12(3):429–439.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In Proceedings of the International Conference on Learning Representations (ICLR).

Krizhevsky, A. (2010). Convolutional deep belief networks on CIFAR-10. Technical report, University of Toronto. Unpublished Manuscript: http://www.cs.utoronto.ca/ kriz/conv-cifar10-aug2010.pdf.

翻译：www.gwylab.com 原文来源：https://arxiv.org/pdf/1410.8516.pdf

Lappalainen, H., Giannakopoulos, X., Honkela, A., and Karhunen, J. (2000). Nonlinear independent component analysis using ensemble learning: Experiments and discussion. In Proc. ICA. Citeseer.

Larochelle, H. and Murray, I. (2011). The Neural Autoregressive Distribution Estimator. In Pro-ceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AIS-TATS'2011), volume 15 of JMLR: W&CP.

LeCun, Y. and Cortes, C. (1998). The mnist database of handwritten digits.

Mnih, A. and Gregor, K. (2014). Neural variational inference and learning in belief networks. In ICML'2014.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. Deep Learning and Unsupervised Feature Learning Workshop, NIPS.

Ozair, S. and Bengio, Y. (2014). Deep directed generative autoencoders. Technical report, U. Montreal, arXiv:1410.0630.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. Technical report, arXiv:1401.4082.

Rippel, O. and Adams, R. P. (2013). High-dimensional probability estimation with deep density models. arXiv:1302.5125.

Roberts, S. and Everson, R. (2001). Independent component analysis: principles and practice. Cambridge University Press.

Salakhutdinov, R. and Hinton, G. (2009). Deep Boltzmann machines. In Proceedings of the Inter-national Conference on Artificial Intelligence and Statistics, volume 5, pages 448–455.

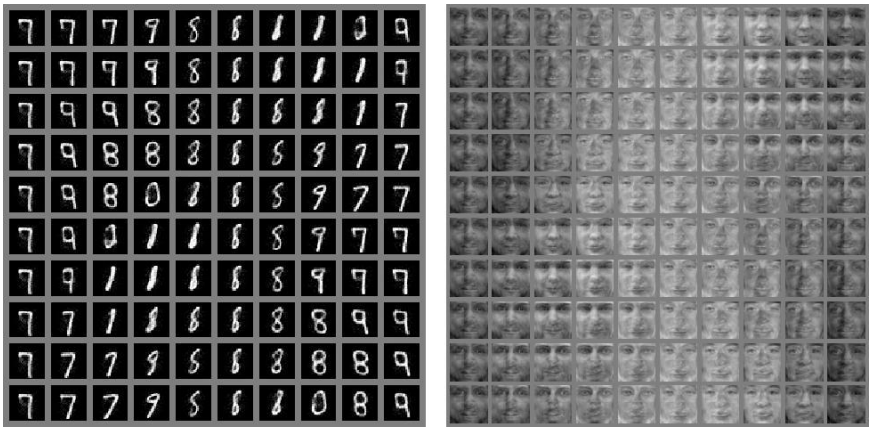
Salakhutdinov, R. and Murray, I. (2008). On the quantitative analysis of deep belief networks. In Cohen, W. W., McCallum, A., and Roweis, S. T., editors, Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08), volume 25, pages 872–879. ACM.

Susskind, J., Anderson, A., and Hinton, G. E. (2010). The Toronto face dataset. Technical Report UTML TR 2010-001, U. Toronto.

Tang, Y., Salakhutdinov, R., and Hinton, G. (2012). Deep mixtures of factor analysers. arXiv preprint arXiv:1206.4635.

Uria, B., Murray, I., and Larochelle, H. (2013). Rnade: The real-valued neural autoregressive density-estimator. In NIPS'2013.

翻译：www.gwylab.com 原文来源：https://arxiv.org/pdf/1410.8516.pdf



(a) Model trained on MNIST (b) Model trained on TFD

图 7：潜在空间中的球体。 这些图显示了模型学习的多种结构的一部分。

A 更多可视化

A.1 流行可视化

为了说明学习的流形，我们还在潜在空间中采用 3D 球体 S 的随机旋转 R 并将其转换为数据空间，结果 $f^{-1}(R(S))$ 如图 7 所示。

A.2 光谱

我们还检查了最后的对角线缩放层并查看了它的系数 $(S_{dd})_{d\leq D}$。如果我们联合考虑先验分布和对角缩放层，则 $\sigma_d = S_{dd}^{-1}$ 可以被认为是每个独立分量的尺度参数。 这向我们展示了模型对每个组件的重要性，以及最终模型在学习流形方面的成功程度。 我们对 $(\sigma_d)_{d\leq D}$ 进行排序并将其绘制在图 8 中。

B 近似白化

学习近似白化的过程是使用 NICE 框架，具有仿射函数和标准高斯先验。 我们有:

$$z = Lx + b$$

L 是下三角形和 b 是偏置矢量。 这相当于学习高斯分布。优化程序与 NICE 相同：RMSProp 具有早期停止和动量

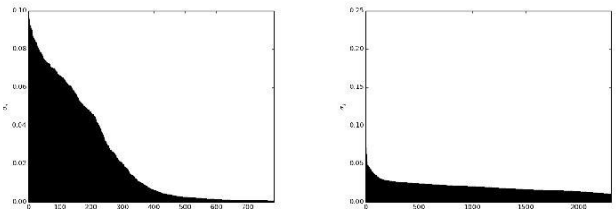
C 变异自动编码器类似 NICE

我们断言，随机梯度变分贝叶斯（SGVB）算法最大化了对上的对数似然（x,ϵ）.(Kingma 和 Welling，2014）定义了一个识别网络:

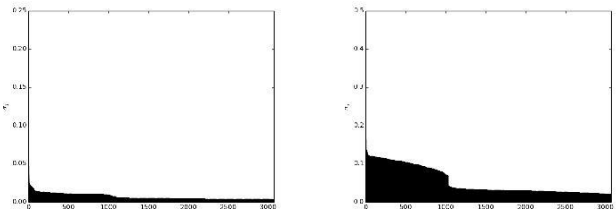
$$z = g_\phi(\epsilon \mid x),\ \epsilon \sim \mathcal{N}(0, I)$$

对于标准高斯先验 p（z）和条件 p（x | z），我们可以定义:

$$\xi = \frac{x - f_\theta(z)}{\sigma}$$



(a) Model trained on MNIST (b) Model trained on TFD



(c) Model trained on SVHN (d) Model trained on CIFAR-10

图 8：$\sigma_d = S_{dd}^{-1}$的衰减。 大值对应于模型选择具有较大变化的维度，从而从数据突出显示学习的流形结构。 这是 PCA 情况下的本征谱的非线性等价物。 在 x 轴上是按 d 排序的组件（在 y 轴上）。

如果我们在 h =（z,ξ）上定义标准高斯先验。 由此产生的成本函数是:

$$\log(p_{(x,\epsilon),(\theta,\phi)}(x,\epsilon)) = \log(p_H(h)) - D_X \log(\sigma) + \log(|\det \frac{\partial g_\phi}{\partial \epsilon}(\epsilon; x)|)$$

其中 D_X = dim(X). 这等价于:

$$\begin{aligned}
\log(p_{(x,\epsilon),(\theta,\phi)}(x,\epsilon)) - \log(p_\epsilon(\epsilon)) &= \log(p_H(h)) - D_X \log(\sigma) + \log(|\det \frac{\partial g_\phi}{\partial \epsilon}(\epsilon; x)|) - \log(p_\epsilon(\epsilon)) \\
&= \log(p_H(h)) - D_X \log(\sigma) - \log(q_{Z|X;\phi}(z)) \\
&= \log(p_\xi(\xi) p_Z(z)) - D_X \log(\sigma) - \log(q_{Z|X;\phi}(z)) \\
&= \log(p_\xi(\xi)) + \log(p_Z(z)) - D_X \log(\sigma) - \log(q_{Z|X;\phi}(z)) \\
&= \log(p_\xi(\xi)) - D_X \log(\sigma) + \log(p_Z(z)) - \log(q_{Z|X;\phi}(z)) \\
&= \log(p_{X|Z}(x \mid z)) + \log(p_Z(z)) - \log(q_{Z|X;\phi}(z))
\end{aligned}$$

这是蒙特卡洛对 SGVB 成本函数的估计（Kingma 和 Welling，2014）.