卷积神经网络的最佳实践
应用于可视化文档分析

Patrice Y. Simard、Dave Steinkraus、John C. Platt
微软研究院，一种微软方式，雷德蒙德 WA 98052
{patrice,v-davste,jplatt}@microsoft.com

抽象的

神经网络是一种强大的技术，用于对文档产生的视觉输入进行分类。然而，在文献和工业中使用的不同的神经网络方法过多，令人困惑。本文描述了一组具体的最佳实践，文档分析研究人员可以使用这些实践来通过神经网络获得良好的结果。最重要的做法是让训练集尽可能大：我们通过添加一种新形式的扭曲数据来扩展训练集。下一个最重要的实践是卷积神经网络比全连接网络更适合视觉文档任务。我们建议使用灵活架构的简单“自己动手”卷积实现适用于许多视觉文档问题。这个简单

卷积神经网络不需要复杂的方法，例如动量、权重衰减、结构相关的学习率、平均层、切线道具，甚至不需要微调架构。最终结果是一个非常简单但通用的架构，可以为文档分析提供最先进的性能。我们说明了我们对 MNIST 英文数字图像集的主张。

一、简介

在 1990 年代初期非常流行之后，神经网络在过去 5 年中在研究中失宠。2000年，神经信息处理系统（NIPS）会议的组织者甚至指出，投稿标题中的“神经网络”一词与接受率呈负相关。相比之下，正相关与支持向量机 (SVM)、贝叶斯网络和变分方法形成了正相关。

在本文中，我们展示了神经网络在手写识别任务 (MNIST) 上取得了最佳性能。MNIST [7] 是分割手写数字图像的基准数据集，每个图像具有 28x28 像素。有 60,000 个训练示例和 10,000 个测试示例。

我们在带有神经网络的 MNIST 上的最佳表现与其他研究人员一致，他们发现神经网络继续在视觉文档分析任务中产生最先进的性能 [1] [2]。

MNIST 的最佳性能是通过两个基本实践实现的。首先，我们创建了一组新的、通用的弹性扭曲，极大地扩展了训练集的大小。其次，我们使用了卷积神经网络。第 2 节详细描述了弹性变形。第 3 节和第 4 节描述了一个易于实现的通用卷积神经网络架构。

我们相信这两种做法适用于 MNIST 之外的文档分析中的一般视觉任务。应用范围从传真识别到扫描文档的分析以及 Tablet PC 中的草书识别（使用视觉表示）。

2. 通过弹性扭曲扩展数据集

合成数据的合理变换很简单，但“逆”问题——变换不变性——可以任意复杂。幸运的是，学习算法非常擅长学习逆问题。给定一个分类任务，可以应用变换来生成额外的数据，并让学习算法推断变换不变性。这种不变性嵌入在参数中，因此在某种意义上它是自由的，因为识别时的计算是不变的。如果数据稀缺并且要学习的分布具有变换不变性，则使用转换甚至可以提高性能 [6]。在手写识别的情况下，我们假设分布不仅在仿射变换方面具有一定的不变性，而且在与手部肌肉不受控制的振荡相对应的弹性变形方面具有一定的不变性，受到惯性的抑制。

通过将仿射位移场应用于图像，可以生成简单的扭曲，例如平移、旋转和倾斜。这是通过为每个像素计算相对于原始位置的新目标位置来完成的。位置 (x,y) 处的新目标位置是相对于先前位置给出的。例如，如果 $\Delta x(x,y)=1$，并且 $\Delta y(x,y)=0$，这意味着每个像素的新位置向右移动 1。如果

位移场为：Δx(x,y)= αx，并且 Δy(x,y)= αy，图像将从原点位置 (x,y)=(0,0) 缩放 α。由于 α 可能是非整数值，因此需要进行插值。

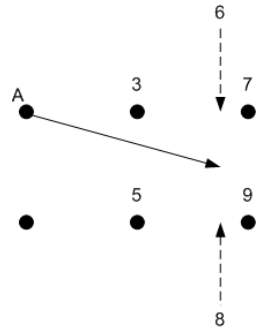

图 1. 如何计算 A 的新灰度级，在位置 (0,0) 给定位移 Δx(0,0) = 1.75 和 Δy(0,0) = -0.5。双线性插值产生 7.0。

图 1 说明了如何应用位移场来计算每个像素的新值。在这个例子中，假设 A 的位置是 (0,0) 并且数字

3、7、5、9分别是要变换的图像在(1,0)、(2,0)、(1,-1)和(2,-1)位置的灰度级。A 的位移由 Δx(0,0) = 1.75 和 Δy(0,0) = -0.5 给出，如图箭头所示。新（扭曲）图像中 A 的新灰度值是通过评估原始图像位置 (1.75,-0.5) 处的灰度级来计算的。评估灰度的一种简单算法是原始图像像素值的“双线性插值”。尽管可以使用其他插值方案（例如，双三次和样条插值），但双线性

插值是最简单的方法之一，适用于以所选分辨率 (29x29) 生成额外的扭曲字符图像。插入值

水平，然后垂直插入值，完成评估。为了计算水平插值，我们首先计算箭头相对于它结束的正方形的结束位置。在这种情况下，正方形中的坐标为 (0.75, 0.5)，假设该正方形的原点是左下角（值 5 所在的位置）。在本例中，新值是：3 + 0.75 × (7-3) = 6；和 5 + 0.75 × (9-5) = 8。这些值之间的垂直插值产生 8 + 0.5 × (6-8) = 7，这是像素 A 的新灰度值。像素。假定给定图像外的所有像素位置都具有背景值（例如 0）。

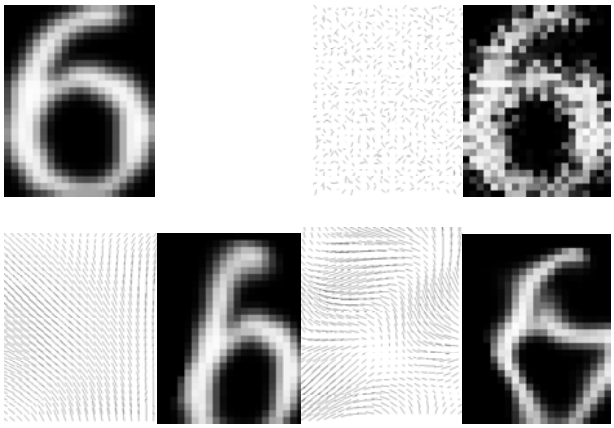

图 2. 左上角：原始图像。右和下：具有各种平滑的位移场对，以及当位移场应用于原始图像时产生的图像。

仿射失真极大地改善了我们在 MNIST 数据库上的结果。然而，我们最好的结果是
当我们使用弹性变形时获得。图像变形是通过首先生成随机位移场创建的，即 Δx(x,y) = rand(-1,+1) 和 Δx(x,y)=rand(-1,+1)，其中 rand (-1,+1) 是一个介于 -1 和 +1 之间的随机数，以均匀分布生成。然后，场 Δx 和 Δy 与标准偏差 σ（以像素为单位）的高斯卷积。如果 σ 很大，则结果值非常小，因为随机值的平均值为 0。如果我们将

位移场（范数为 1），则该场接近恒定，具有随机方向。如果 σ 很小，则该场在归一化后看起来像一个完全随机的场（如图 2 右上所示）。对于中间 σ 值，位移场看起来像弹性变形，其中 σ 是弹性系数。然后将位移场乘以控制变形强度的比例因子 α。

图 2 显示了纯随机场 (σ = 0.01)、对应于手的属性的平滑随机场 (σ=8) 以及对应于太多可变性 (σ=4) 的平滑随机场的示例。如果σ很大，位移接近仿射，如果σ很大，位移变成平移。

在我们的 MNIST 实验（29x29 输入图像）中，产生最佳结果的值是 σ=4 并且α=34。

3. 视觉任务的神经网络架构

我们为 MNIST 数据集考虑了两种类型的架构神经网络架构。最简单的架构是通用分类器，它是一个具有两层的全连接网络 [4]。更多

复杂的架构是卷积神经网络，已发现它非常适合视觉文档分析任务 [3]。标准神经网络的实现可以在教科书中找到，例如 [5]。第 4 节描述了卷积神经网络的一种新的、简单的实现。

为了测试我们的神经网络，我们试图让算法尽可能简单，以获得最大的可重复性。我们只尝试了两种不同的误差函数：交叉熵（CE）和均方误差（MSE）（更多细节见 [5，第 6 章]）。我们避免使用动量、权重衰减、依赖于结构的学习率、输入周围的额外填充以及平均而不是二次采样。（我们的动机是通过在各种架构/失真组合和

训练/验证拆分数据并发现它们没有帮助。）

我们的初始权重设置为小的随机值（标准偏差 = 0.05）。我们使用的学习率从 0.005 开始，每 100 个 epoch 乘以 0.3。

3.1. MNIST 的整体架构

如第 5 节所述，我们发现卷积神经网络在 MNIST 上表现最好。我们认为这是视觉任务的一般结果，因为卷积神经网络 [3] 很好地捕获了空间拓扑，而标准神经网络忽略了输入的所有拓扑属性。也就是说，如果在所有输入像素都经过固定排列的数据集上重新训练和重新测试标准神经网络，则结果将是相同的。

我们用于 MNIST 数字识别的卷积神经网络的整体架构如图 3 所示。

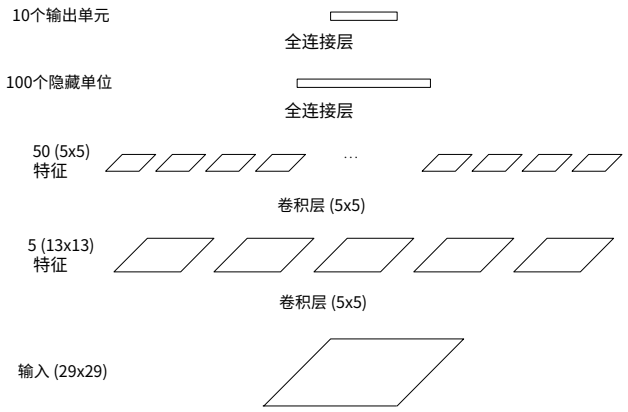

图 3. 用于手写识别的卷积架构

卷积网络的一般策略是以更高的分辨率提取简单的特征，然后

将它们以更粗的分辨率转换为更复杂的特征。最简单的是生成更粗的
分辨率是按因子 2 对层进行子采样。这反过来又是卷积核大小的线索。内核的宽度被选择为以一个单位为中心（奇数大小），以有足够的重叠不会丢失信息（只有一个单位重叠时 3 会太小），但又不会有冗余计算（7 会太大，有 5 个单位或超过 70% 的重叠）。一个大小为 5 的卷积核如图 4 所示，空心圆单元对应子采样，不需要计算。填充输入（使其变大，以便

有以边界为中心的特征单元）并没有显着提高性能。在没有填充、子采样为 2 和内核大小为 5 的情况下，每个卷积层将特征大小从 n 减小到 (n-3)/2。由于初始 MNIST 输入大小为 28x28，因此在 2 层卷积后生成整数大小的最近值是 29x29。2层卷积后，5x5的特征尺寸对于第三层卷积来说太小了。第一个特征层提取非常简单的特征，经过训练后看起来像边缘、墨水或交叉点检测器。我们发现使用少于 5 个不同的特征会降低性能，而使用超过 5 个并没有提高性能。同样，在第二层，我们发现少于 50 个特征（我们尝试了 25 个）会降低性能，而更多（我们尝试 100 个）并没有提高性能。

这个神经网络的前两层可以看作是一个可训练的特征提取器。我们现在以 2 个全连接层（通用分类器）的形式向特征提取器添加一个可训练的分类器。隐藏单元的数量是可变的，通过改变这个数量，我们可以控制整个分类器的容量和泛化能力。对于 MNIST（10 个类别），使用 100 个隐藏单元达到最佳容量。

4. 让卷积神经网络变得简单

多年来，卷积神经网络已被提出用于视觉任务 [3]，但在工程界并不流行。我们认为这是由于实现卷积神经网络的复杂性。本文介绍了实现此类网络的新方法，这些方法比以前的技术更容易，并且易于调试。

4.1. 卷积的简单循环

全连接神经网络通常使用以下
执行规则 前进和后退
传播：

$$x_j^{l+1} = \sum_{i} w_{ij,i}^{l+1} x_i^{l} + \theta_j^{l} \qquad (1.1)$$

$$G_{升,一世} = \sum_j 瓦_{升+1,j,一世} G_{升+1,j} \quad (1.2)$$

在哪里 $X^{升}_{一世}$和 $G^{升}_{一世}$分别是单元的激活和梯度 *一世* 在层 *升*，和 $瓦_{升+1}$是重量

连接单元 *一世* 在层 *升* 为单位 *j* 在层 *升* +1.

这可以看作是更高层的激活单元“拉动”了所有连接到它们的单元的激活。同样，下层的单元也在拉动与其相连的所有单元的梯度。然而，在计算卷积网络的梯度时，拉动策略实施起来既复杂又痛苦。原因是在一个

卷积层，由于边界效应，离开每个单元的连接数不是恒定的。

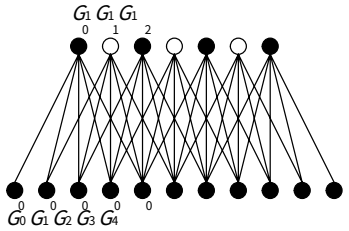

图 4. 卷积神经网络 (1D)

这在图 4 中很容易看到，其中所有单位都标记为 $G^{0}_{一世}$具有可变数量的传出连接。相比之下，上层的所有单元都有一个固定的
传入连接数。为了简化
计算，而不是从下层拉梯度，我们可以从上层“推”梯度。结果方程为：

$$G_{升,一世} = \sum_j 瓦_{升+1,一世,j} G_{升+1,j} \quad (1.3)$$

每个单元 *j* 在上层，我们更新固定数量的（传入）单元 *一世* 从下层（图中， *一世* 介于 0 和 4 之间）。因为在卷积中权重是共享的，*瓦* 不依赖 *j*. 请注意，推比拉慢，因为梯度在内存中累积，而不是在拉中，梯度累积在寄存器中。根据架构的不同，这有时可能会慢 50%（这相当于整体性能下降不到 20%）。然而，对于大卷积，推梯度可能更快，并且可用于利用英特尔的 SSE 指令，因为所有内存访问都是连续的。从一个

从实现的角度来看，拉动激活和推动梯度是迄今为止实现卷积层的最简单方法，值得在速度上稍作妥协。

4.2. 模块化调试

反向传播有一个很好的特性：它允许以模块化方式表达和调试神经网络。例如，我们可以假设一个模块 *米*

有一个前向传播函数来计算它的输出 *M(I,W)* 作为其输入的函数 *一世* 及其参数 *宽*. 它也有反向传播

函数（相对于输入）计算输入梯度作为输出梯度的函数，梯度函数（相对于权重），计算相对于输出梯度的权重梯度，以及权重更新函数，它使用一些更新规则（批量、随机、动量、权重衰减等）将权重梯度添加到权重中。根据定义，函数的雅可比矩阵 *米* 被定义为

成为 $J_{演} \equiv \frac{\partial 米_{克}}{\partial X_{一世}}$(见 [5] 页。148 了解更多信息

神经网络的雅可比矩阵）。使用反向传播函数和梯度函数，可以直接计算两个雅可比矩阵

$\frac{\partial 一世}{\partial 米(一世,宽)}$和 $\frac{\partial 宽}{\partial 米(一世,宽)}$ 通过简单地喂食

（梯度）单位向量 $\triangle 米_{克}$（*我，W*）对于这两个函数，其中 *克* 索引所有输出单元 *米*，和
唯一单位 *克* 设置为 1，所有其他设置为 0。相反，我们可以生成任意准确的估计

雅可比矩阵 $\frac{\partial 米(一世,宽)}{\partial 一世}$ 和 $\frac{\partial 米(一世,宽)}{\partial 宽}$ 经过

将小的变化 ε 添加到 *一世* 和 *宽* 并调用
M(I,W) 功能。使用等式：

$$\frac{\partial 一世}{\partial 米} = F\left(\frac{\partial 米}{\partial 一世}\right)^{吨} \text{ 和 } \frac{\partial 宽}{\partial 米} = F\left(\frac{\partial 米}{\partial 宽}\right)^{吨}$$

在哪里 *F* 是一个函数，它采用矩阵并反转其每个元素，可以自动验证前向传播是否准确对应于后向和梯度传播（注意：反向传播计算 *F（∂ 一世/ ∂ M(I,W)）* 因此只需要一个转置就可以将其与通过前向传播计算的雅可比行列式进行比较）。换句话说，如果上面的等式被验证为机器的精度，那么学习是正确实施的。这对于大型网络特别有用，因为不正确的实现有时会产生合理的结果。事实上，学习算法往往对错误也很健壮。在我们的实现中，每个神经网络都是一个 C++ 模块，并且是更多基本模块的组合。模块测试程序以双精度实例化模块，设置 $\varepsilon=10^{-12}$ (double 的机器精度是 10^{-16})，生成随机值 *一世* 和 *宽*，并执行正确性测试，精度为 10^{-10}. 如果较大的模块未通过测试，我们将测试每个子模块

模块，直到我们找到罪魁祸首。这个极其简单和自动化的过程节省了大量的调试时间。

5. 结果

对于全连接和卷积神经网络，我们使用 MNIST 训练集的前 50,000 个模式进行训练，其余 10,000 个模式用于验证和参数调整。结果报告在测试集上，其中使用验证时最佳的参数值完成。本文中的两层多层感知器 (MLP) 有 800 个隐藏单元，而 [3] 中的两层 MLP 有 1000 个隐藏单元。

结果报告在下表中：

算法	失真	错误	参考
2层 MLP (MSE)	仿射	1.6%	[3]
支持向量机	仿射	1.4%	[9]
切线距离	仿射+厚	1.1%	[3]
Lenet5 (MSE)	仿射	0.8%	[3]
促进。Lenet4 MSE	仿射	0.7%	[3]
虚拟支持向量机	仿射	0.6%	[9]
2 层 MLP (CE) 2 层	没有任何	1.6%	这张纸
MLP (CE) 2 层 MLP	仿射	1.1%	这张纸
(MSE)	松紧带	0.9%	这张纸
2 层 MLP (CE) 简	松紧带	0.7%	这张纸
单转换 (CE)	仿射	0.6%	这张纸
简单转换 (CE)	松紧带	**0.4%**	这张纸

表 1. 各种算法之间的比较。

该表中有几个有趣的结果。最重要的是，弹性变形对 2 层 MLP 和我们的卷积架构的性能都有相当大的影响。据我们所知，0.4% 的错误是迄今为止在 MNIST 数据库上的最佳结果。这意味着 MNIST 数据库对于大多数算法来说太小而无法正确推断泛化，并且弹性变形提供了额外的和相关的先验知识。其次，我们观察到卷积网络与 2 层 MLP 相比表现良好，即使有弹性变形也是如此。即使使用弹性变形生成的大型训练集，MLP 也不容易推断出卷积网络中隐含的拓扑信息。最后，我们观察到，最近的实验比 8 年前进行的类似实验产生了更好的性能，并在 [3] 中报告。可能的解释是，硬件现在快了 1.5 个数量级（我们现在可以承受数百个 epoch），并且在我们的实验中，CE 的训练速度比 MSE 快。

6。结论

我们使用弹性失真和卷积神经网络在 MNIST 数据集上实现了迄今为止已知的最高性能。我们认为这些结果反映了两个重要问题。

训练集大小：学习系统的质量主要取决于训练集的大小和质量。这一结论得到了其他应用领域的证据的支持，例如文本[8]。对于视觉文档任务，本文提出了一种用于极大扩展训练集的简单技术：弹性扭曲。这些扭曲大大改善了 MNIST 的结果。

卷积神经网络：标准神经网络是最先进的分类器，其性能与对向量进行操作的其他分类技术差不多，无需了解输入拓扑。然而，卷积神经网络利用输入不是独立元素，而是来自空间结构的知识。

神经网络的研究已经放缓，因为神经网络训练被认为需要神秘的黑魔法才能获得最佳结果。我们已经证明最好的结果不需要任何神秘的技术：一些专门的技术可能源于计算速度的限制，这些限制不适用于 21英石 世纪。

7. 参考文献

[1] Y. Tay、P. Lallican、M. Khalid、C. Viard-Gaudin、S. Knerr，“离线草书
文字识别系统”，*过程 IEEE 区域 10 会议。*，
(2001)。

[2] A.辛哈， *一种改进的手写数字识别模块*, 硕士论文, 麻省理工学院,
(1999)。

[3] Y. LeCun, L. Bottou, Y. Bengio, P. “基于 哈夫纳，梯度的学习应用于 文档
承认” IEEE Proceedings of the IEEE, v. 86, pp. 2278-2324，1998。

[4] KM Hornik、M. Stinchcombe、H. White，“使用多层前馈网络对未知映射及其导数进行通用逼近”

神经网络, v. 3, pp. 551-560, (1990)。

[5] 厘米主教， *用于模式识别的神经网络*，牛津大学出版社，（1995 年）。

[6] L. Yaeger、R. Lyon、B. Webb，“用于单词识别的神经网络字符分类器的有效训练”， *国家知识产权局*, v. 9, pp. 807-813, (1996)。

[7] Y. LeCun，“手写数字的 MNIST 数据库”，http://yann.lecun.com/exdb/mnist。

[8] M. Banko、E. Brill，“缓解数据缺乏问题：探索训练语料库大小对自然语言分类器性能的影响

处理，”Proc。会议。人类语言技术，
(2001)。

[9] D. Decoste 和 B. Scholkopf，“训练不变支持向量机”，机器学习杂志，第 46 卷，第 1-3 期，2002 年。