深度学习算法文献综述

李坤桐

目 录

一.研究意义

深度学习（Deep Learning，DL）是机器学习的分支，是一种试图使用包含复杂结构或由多重非线性变换构成的多个处理层对数据进行高层抽象的算法[2]。

深度学习是机器学习中一种基于对数据进行表征学习的算法，至今已有数种深度学习模型，如卷积神经网络、深度置信网络、递归神经网络等已被应用在计算机视觉、语音识别、自然语言处理、音频识别与生物信息学等领域，并获取了极好的效果。

得益于当前计算机架构更快的处理速度，这类模型有能力应对成千上万个特征。与早期的统计分析形式不同，深度学习模型中的每个特征通常对于人类观察者而言意义不大。这导致的结果就是该模型的使用难度很大或者难以解释。在德勤的调查中只有 34%的人在使用深度学习技术[3]。

深度学习模型使用一种称为反向传播的技术，通过模型进行预测或对输出进行分类。从在围棋大赛中击败人类专家到对互联网图像进行分类，都是使用反向传播的深度学习。在多伦多大学及谷歌任职的 Geoffrey Hinton 通常被称为深度学习之父，部分原因就在于他在反向传播方面的早期研究。

近年来，全世界在这一领域取得了许多重大突破。由于深度学习正快度发展，导致了它的进展很难被跟进，特别是对于新的研究者。在本文中，我们将简要讨论近年来关于深度学习的最新进展。

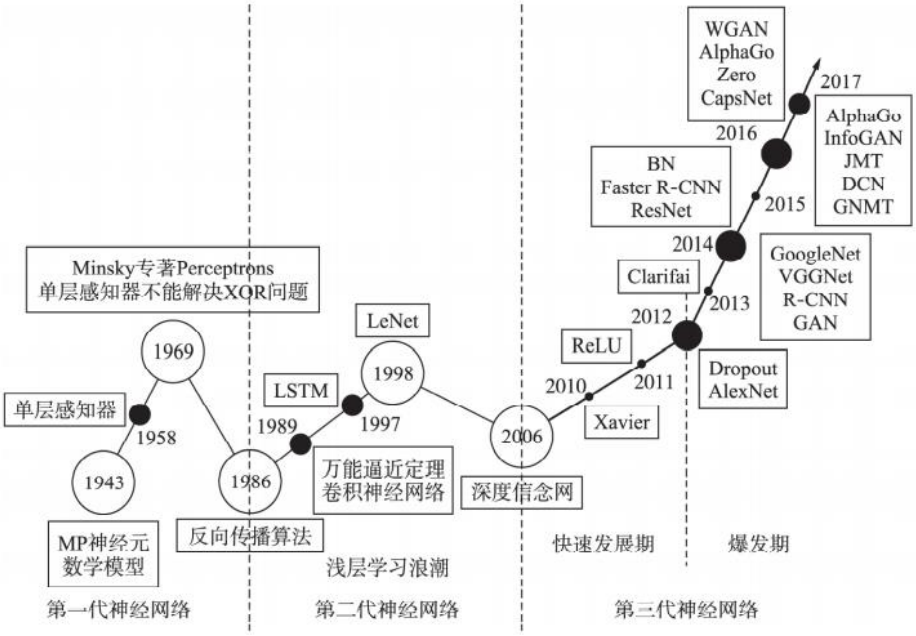

图 1-1 深度学习算法发展历程

二.模型原理描述

人工神经网络已经取得了长足的进步，同时也带来了其他的深度模型。第一代人工神经网络由简单的感知器神经层组成，只能进行有限的简单计算。第二代使用反向传播，根据错误率更新神经元的权重。此后支持向量机浮出水面，在一段时间内超越 ANN。为了克服反向传播的局限性，人们提出了受限玻尔兹曼机，使学习更加容易。此时其他技术和神经网络出现了，如前馈神经网络（FNN）、卷积神经网络（CNN）、循环神经网络（RNN）等，以及深层信念网络、自编码器等。从那时起，为实现各种用途，ANN 在不同方面得到了改进和设计。

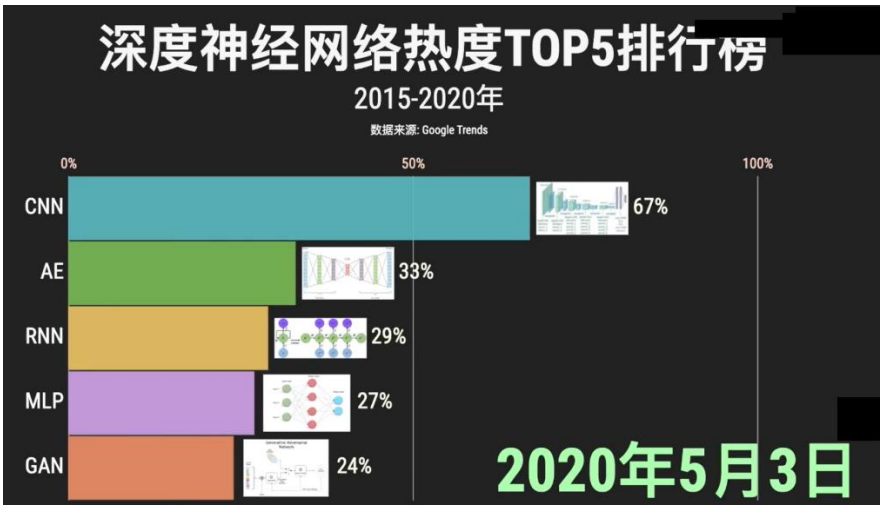

图 2-1 神经网络热度排行

上图是在 2020 年中旬基于 google trends 数据得到的 5 个热门神经网络模型，这里就主要介绍一下这 5 种模型。

2.1 CNN

卷积神经网络(Convolutional Neural Networks，CNN)有四个基本概念，即局部连接、共享权值、池化和多层使用。CNN 的前一部分由卷积层和池化层组成，后一部分主要是全连接层。卷积层从特征中检测局部连接，池化层将相似特征合并为一个。CNN 在卷积层中使用卷积而不是矩阵乘法，2016 年，Goodfellow 等人在著书中详细介绍了 CNN 的基本架构和思想[4]。

2.1.1 AlexNet

第一个典型的 CNN 是 LeNet5 网络结构，但是第一个引起大家注意的网络却是 AlexNet。该网络在 2012 年提出，模型训练了当时最大的卷积神经网络之一，并在数据集上获得了当时的最佳结果。这是深度学习的一个重大突破，论文专注于，展示即使一个简单的 CNN 模型，在有足够的规模、深度之后，也有极其强大的能力。

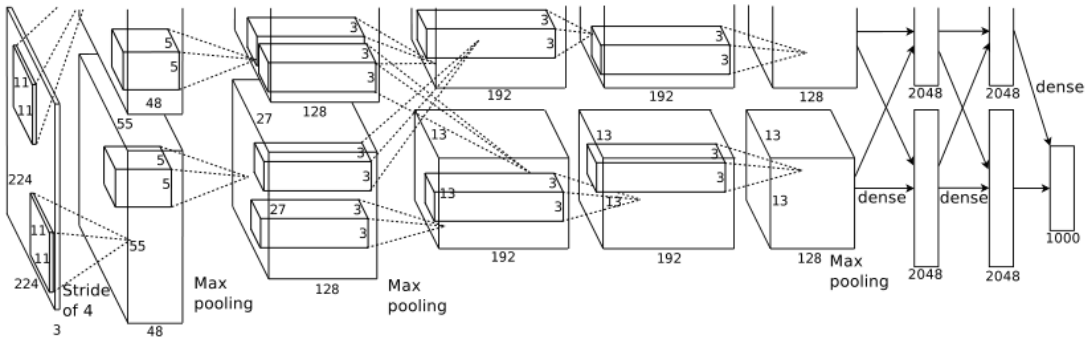

图 2-2 AlexNet 网络结构

如图所示，该网络包含有权重的 8 层。前五个是卷积的，其余三个是完全连接的。最后一个全连接层的输出被馈送到一个 1000 路的 softmax，该 softmax 产生超过 1000 个类标签的分布。该网络最大限度地实现了多项逻辑回归目标，即在预测分布下，最大限度地提高了正确标签的对数概率的平均训练案例。第二、第四和第五卷积层的内核只与前一层中驻留在同一图形处理器上的内核映射相连。第三卷积层的核连接到第二层中的所有核映射。完全连接层中的神经元与前一层中的所有神经元相连。响应标准化层遵循第一个和第二个卷积层。

最大池层遵循响应标准化层和第五个卷积层。ReLU 非线性应用于每个卷积和全连接层的输出。第一卷积层对 224×224×3 的输入图像进行滤波，其中 96 个核的大小为 11×11×3，步长为 4 个像素。第二卷积层将第一卷积层的输出作为输入，并用 256 个大小为 5×5×48 的核对其进行滤波。第三、第四和第五卷积层相互连接，没有任何中间的汇集或标准化层。第三个卷积层有 384 个大小为 3×3×256 的核，连接到第二个卷积层的输出。第四卷积层有 384 个大小为 3×3×192 的核，第五卷积层有 256 个大小为 3×3×192 的核。完全连接的层各有 4096 个神经元。

该网络架构在当时有一些新颖的特点。首先是引入 ReLU 作为激活函数，标准的 L-P 神经元的输出一般使用 tanh 或 sigmoid 作为激活函数，但是这些饱和的非线性函数在计算梯度的时候都要比非饱和的现行函数慢很多。在深度学习中使用 ReLU 要比等价的 tanh 快很多。

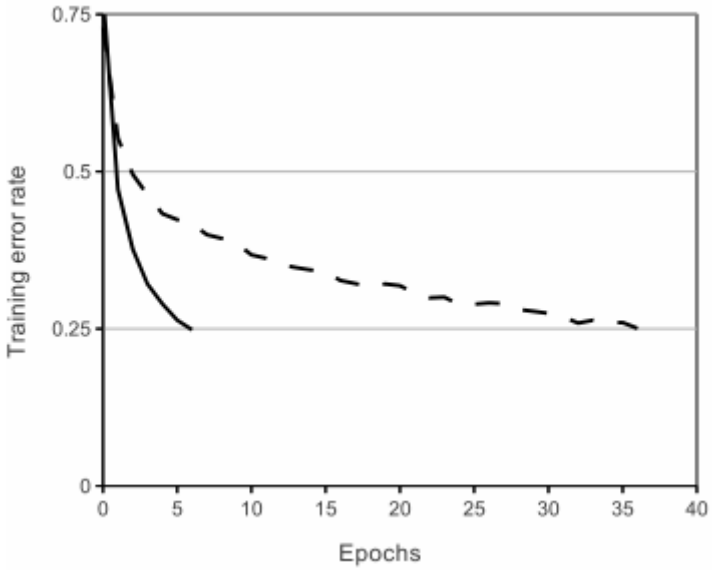

图 2-3 收敛曲线

上图是使用 ReLU 和 tanh 作为激活函数的典型四层网络的在数据集 CIFAR-10s 实验中，error rate 收敛到 0. 25 时的收敛曲线，可以很明显的看到收敛速度的差距。虚线为 tanh，实线是 ReLU。

第二个特点是使用两块 GPU 进行训练，可以提高运算的效率。第三个特点是进行局部响应归一化（Local Response Normalization），在神经网络中，我们用激活函数将神经元的输出做一个非线性映射，但是 tanh 和 sigmoid 这些传统

的激活函数的值域都是有范围的，但是 ReLU 激活函数得到的值域没有一个区间，所以要对 ReLU 得到的结果进行归一化。也就是 Local Response Normalization。局部响应归一化的方法如下面的公式：

$$b_{x,y}^{i}=a_{x,y}^{i}/\left(k+\alpha\sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)}(a_{x,y}^{j})^{2}\right)^{\beta}$$

第四个特点是覆盖的池化操作（Overlapping Pooling），一般的池化层因为没有重叠，所以 pool_size 和 stride 一般是相等的，例如 8×8 的一个图像，如果池化层的尺寸是 2×2，那么经过池化后的操作得到的图像是 4×4 大小，这种设置叫做不覆盖的池化操作，如果 stride<pool_size，那么就会产生覆盖的池化操作，这种有点类似于 convolutional 化的操作，这样可以得到更准确的结果。在 top-1，和 top-5 中使用覆盖的池化操作分别将 error rate 降低了 0.4%和 0.3%。论文中提到，在训练模型过程中，覆盖的池化层更不容易过拟合。

此外，关于防止神经网络过拟合问题，该模型引用了同年 Hinton 提出的“dropout”技术，使用该技术后，迭代速度提升了约两倍。

2.2 AE

自编码器（Autoencoders，AE）是以输出为输入的神经网络，采用原始输入，编码为压缩表示，然后解码以重建输入。在深度 AE 中，低隐藏层用于编码，高隐藏层用于解码，误差反向传播用于训练。

2.2.1 VAE

该模型提出于 2013 年，是一类重要的生成模型。2016 年 Carl Doersch 写了一篇关于 VAE 的教程，使模型更加易学易懂，最新的更新版本到 2021 年。

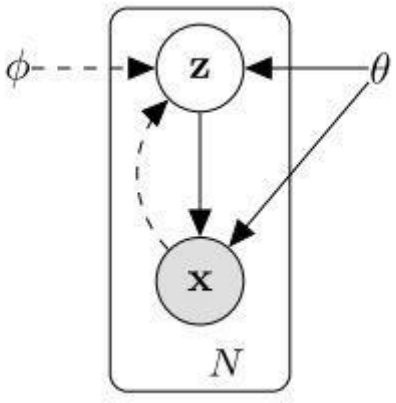

图 2-4 VAE 模型

上图是 VAE 的模型。我们能观测到的数据是x，而x由隐变量z产生，由$z \to x$是生成模型$p_\theta(x|z)$，从自编码器的角度来看，就是解码器；而由$x \to z$是识别模型$q_\varphi(z|x)$，类似于自编码器的编码器。VAE 现在广泛地用于生成图像，当生成模型$p_\theta(x|z)$训练好了以后，我们就可以用它来生成图像了。

下面对 VAE 模型进行一下推导。首先，假定所有的数据都是独立同分布的，两个观测不会相互影响。我们要对生成模型$p_\theta(x|z)$做参数估计，利用对数最大似然法，就是要最大化下面的对数似然函数：

$$\log p_\theta(\mathrm{x}^{(1)}, \mathrm{x}^{(2)}, \cdots, \mathrm{x}^{(N)}) = \sum_{i=1}^{N} \log p_\theta(\mathrm{x}^{(i)})$$

VAE 用识别模型$q_\varphi(z|x^{(i)})$去逼近真实的后验概率$p_\theta(z|x^{(i)})$，衡量两个分布的相似程度，我们一般采用 KL 散度，即：

$$\begin{aligned} KL(q_\phi(\mathrm{z}|\mathrm{x}^{(i)})||p_\theta(\mathrm{z}|\mathrm{x}^{(i)})) &= \mathbb{E}_{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})} \log \frac{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})}{p_\theta(\mathrm{z}|\mathrm{x}^{(i)})} \\ &= \mathbb{E}_{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})} \log \frac{q_\phi(\mathrm{z}|\mathrm{x}^{(i)}) p_\theta(\mathrm{x}^{(i)})}{p_\theta(\mathrm{z}|\mathrm{x}^{(i)}) p_\theta(\mathrm{x}^{(i)})} \\ &= \mathbb{E}_{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})} \log \frac{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})}{p_\theta(\mathrm{z}, \mathrm{x}^{(i)})} + \mathbb{E}_{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})} \log p_\theta(\mathrm{x}^{(i)}) \\ &= \mathbb{E}_{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})} \log \frac{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})}{p_\theta(\mathrm{z}, \mathrm{x}^{(i)})} + \log p_\theta(\mathrm{x}^{(i)}) \end{aligned}$$

于是：

$$\log p_\theta(\mathrm{x}^{(i)}) = KL(q_\phi(\mathrm{z}|\mathrm{x}^{(i)}), p_\theta(\mathrm{z}|\mathrm{x}^{(i)})) + \mathcal{L}(\theta, \phi; \mathrm{x}^{(i)})$$

其中：

$$\begin{aligned}\mathcal{L}(\theta,\phi;\mathrm{x}^{(i)}) &= -\mathbb{E}_{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})}\log\frac{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})}{p_\theta(\mathrm{z},\mathrm{x}^{(i)})}\\ &= \mathbb{E}_{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})}\log p_\theta(\mathrm{z},\mathrm{x}^{(i)}) - \mathbb{E}_{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})}\log q_\phi(\mathrm{z}|\mathrm{x}^{(i)})\end{aligned}$$

由于 KL 散度非负，当两个分布一致时（允许在一个零测集上不一致），KL 散度为 0。于是$\log p_\theta(\mathrm{x}^{(i)}) \geq \mathcal{L}(\theta,\phi;\mathrm{x}^{(i)})$。$\mathcal{L}(\theta,\phi;\mathrm{x}^{(i)})$称为对数似然函数的变分下界。

直接优化$\log p_\theta(\mathrm{x}^{(i)})$是不可行的，因此一般转而优化它的下界$\mathcal{L}(\theta,\phi;\mathrm{x}^{(i)})$。对应的，优化对数似然函数转化为优化$\mathcal{L}(\theta,\phi;\mathrm{X}) = \sum_{i=1}^{N}\mathcal{L}(\theta,\phi;\mathrm{x}^{(i)})$。

作者指出，$\mathcal{L}(\theta,\phi;\mathrm{x}^{(i)})$对$\varphi$的梯度方差很大，不适于用于数值计算。为了解决这个问题，假定识别模型$q_\varphi(z|x)$可以写成可微函数$g_\phi(\epsilon,\mathrm{x})$，其中，$\epsilon$为噪声，$\epsilon \sim p(\epsilon)$。于是，$\mathcal{L}(\theta,\phi;\mathrm{x}^{(i)})$可以做如下估计（利用蒙特卡罗方法估计期望）：

$$\tilde{\mathcal{L}}^{A}(\theta,\phi;\mathrm{x}^{(i)}) = \frac{1}{L}\sum_{l=1}^{L}[\log p_\theta(\mathrm{x}^{(i)},\mathrm{z}^{(i,l)}) - \log q_\phi(\mathrm{z}^{(i,l)}|\mathrm{x}^{(i)})]$$

其中，$\mathrm{z}^{(i,l)} = g_\phi(\epsilon^{(i,l)},\mathrm{x}^{(i)}),\quad \epsilon^{(i,l)} \sim p(\epsilon)$。

此外，$\mathcal{L}(\theta,\phi;\mathrm{x}^{(i)})$还可以改写为

$$\mathcal{L}(\theta,\phi;\mathrm{x}^{(i)}) = -KL(q_\phi(\mathrm{z}|\mathrm{x}^{(i)})||p_\theta(\mathrm{z})) + \mathbb{E}_{q_\phi(\mathrm{z}|\mathrm{x}^{(i)})}\log p_\theta(\mathrm{x}^{(i)}|\mathrm{z})$$

由此可以得到另外一个估计：

$$\tilde{\mathcal{L}}^{B}(\theta,\phi;\mathrm{x}^{(i)}) = -KL(q_\phi(\mathrm{z}|\mathrm{x}^{(i)})||p_\theta(\mathrm{z})) + \frac{1}{L}\sum_{l=1}^{L}\log p_\theta(\mathrm{x}^{(i)}|\mathrm{z}^{(i,l)})$$

其中，$\mathrm{z}^{(i,l)} = g_\phi(\epsilon^{(i,l)},\mathrm{x}^{(i)}),\quad \epsilon^{(i,l)} \sim p(\epsilon)$。

实际试验时，如果样本量N很大，我们一般采用 minibatch 的方法进行学习，对数似然函数的下界可以通过 minibatch 来估计：

$$\mathcal{L}(\theta,\phi;\mathrm{X}) \simeq \tilde{\mathcal{L}}^{M}(\theta,\phi;\mathrm{X}^{M}) = \frac{N}{M}\sum_{i=1}^{M}\tilde{\mathcal{L}}(\theta,\phi;\mathrm{x}^{(i)})$$

可以看到，为了计算$\mathcal{L}(\theta,\phi;\mathrm{X})$，我们用了两层估计。当$M$较大时，内层估计可以由外层估计来完成，也就是说，取$L=1$即可。实际计算中，作者取$M=100, L=1$。由上述推导得到 AEVB 算法：

Algorithm 1 Minibatch version of the Auto-Encoding VB (AEVB) algorithm. Either of the two SGVB estimators in section 2.3 can be used. We use settings $M = 100$ and $L = 1$ in experiments.

$\theta, \phi \leftarrow$ Initialize parameters
repeat
  $\mathbf{X}^M \leftarrow$ Random minibatch of M datapoints (drawn from full dataset)
  $\epsilon \leftarrow$ Random samples from noise distribution $p(\epsilon)$
  $\mathbf{g} \leftarrow \nabla_{\theta,\phi}\tilde{\mathcal{L}}^M(\theta,\phi;\mathbf{X}^M,\epsilon)$ (Gradients of minibatch estimator (8))
  $\theta, \phi \leftarrow$ Update parameters using gradients $\mathbf{g}$ (e.g. SGD or Adagrad [DHS10])
until convergence of parameters (θ, ϕ)
return θ, ϕ

图 2-5 AEVB 算法

上面给的 AEVB 算法是一个算法框架，只有给定了$\epsilon, p_\theta(\mathrm{x}|\mathrm{z}), q_\phi(\mathrm{z}|\mathrm{x}), p_\theta(\mathrm{z})$分布的形式以及$g_\phi(\epsilon,\mathrm{x})$，我们才能启动算法。实际应用中，作者取：

$$\begin{aligned} p(\epsilon) &= \mathcal{N}(\epsilon;0,\mathrm{I}) \\ q_\phi(\mathrm{z}|\mathrm{x}^{(i)}) &= \mathcal{N}(\mathrm{z};\mu^{(i)},\sigma^{2(i)}\mathrm{I}) \\ p_\theta(\mathrm{z}) &= \mathcal{N}(\mathrm{z};0,\mathrm{I}) \\ g_\phi(\epsilon^{(l)},\mathrm{x}^{(i)}) &= \mu^{(i)} + \sigma^{(i)} \odot \epsilon^{(l)} \end{aligned}$$

而$p_\theta(\mathrm{x}|\mathrm{z})$根据样本是实值还是二元数据进行选择，若样本为二元数据，则选择：

$$p_\theta(x_i|\mathrm{z}) = \mathcal{B}(x_i;1,y_i) = y_i^{x_i}\cdot(1-y_i)^{1-x_i}, \quad i=1,2,\cdots,D_\mathrm{x}(D_\mathrm{x}=\dim(\mathrm{x}))$$

若样本是实值数据，则选择：

$$p_\theta(\mathrm{x}^{(i)}|\mathrm{z}) = \mathcal{N}(\mathrm{x}^{(i)};\mu'^{(i)},\sigma'^{2(i)}\mathrm{I})$$

实验中，作者选择多层感知器对$p_\theta(\mathrm{x}|\mathrm{z}), q_\phi(\mathrm{z}|\mathrm{x})$进行拟合，具体来说，对$p_\theta(\mathrm{x}|\mathrm{z})$，参数为$\theta=(\mu',\sigma')$，若样本为二元数据，则

$$\log p(\mathrm{x}|\mathrm{z}) = \sum_{i=1}^{D_\mathrm{x}} x_i \log y_i + (1 - x_i) \cdot \log(1 - y_i)$$
$$\mathrm{y} = \mathrm{sigmoid}(\mathrm{W}_2 \tanh(\mathrm{W}_1 \mathrm{z} + \mathrm{b}_1) + \mathrm{b}_2)$$

若样本为实值数据，则

$$\mu' = \mathrm{W}_4 \mathrm{h}' + \mathrm{b}_4$$
$$\sigma' = \mathrm{W}_5 \mathrm{h}' + \mathrm{b}_5$$
$$\mathrm{h}' = \tanh(\mathrm{W}_3 \mathrm{z} + \mathrm{b}_3)$$

对$q_\phi(\mathrm{z}|\mathrm{x})$，参数为$\phi = (\mu, \sigma)$，

$$\mu = \mathrm{W}_7 \mathrm{h} + \mathrm{b}_7$$
$$\sigma = \mathrm{W}_8 \mathrm{h} + \mathrm{b}_8$$
$$\mathrm{h} = \tanh(\mathrm{W}_6 \mathrm{x} + \mathrm{b}_6)$$

根据以上假设的分布，不难计算

$$\mathcal{L}(\theta, \phi; \mathrm{x}^{(i)}) \simeq \frac{1}{2} \sum_{j=1}^{D_\mathrm{z}} (1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2) + \frac{1}{L} \sum_{l=1}^{L} \log p_\theta(\mathrm{x}^{(i)}|\mathrm{z}^{(i,l)})$$

其中，$\mathrm{z}^{(i,l)} = \mu^{(i)} + \sigma^{(i)} \odot \epsilon^{(l)}$，$\epsilon^{(l)} \sim p(\epsilon)$。最后，给出 VAE 训练时候的网络结构和训练完成后生成样本采用的网络结构。

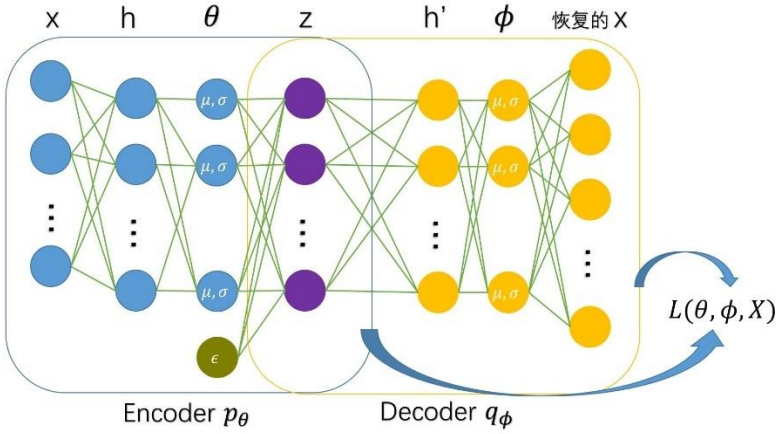

图 2-6 训练网络结构

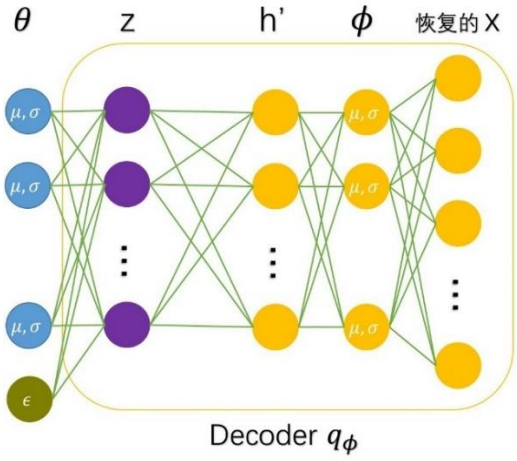

图 2-7 生成后网络结构

2.3 RNN

循环神经网络（Recurrent Neural Networks，RNN），更适合于序列输入，如语音、文本和生成序列。一个重复的隐藏单元在及时展开时可以被认为是具有相同权重的非常深的前馈网络。由于梯度消失和维度爆炸问题，RNN 曾经很难训练。为了解决这个问题，后来许多人提出了改进意见。

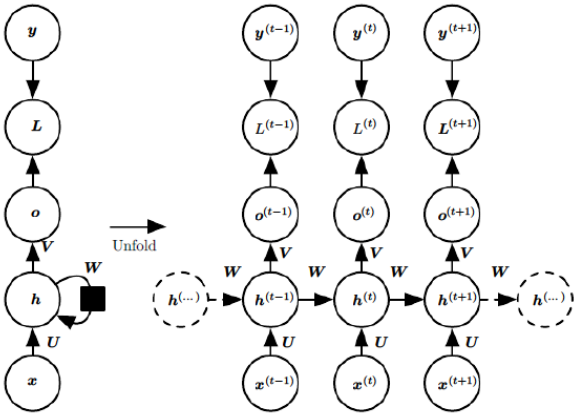

图 2-8 RNN 结构图

这是一个标准的 RNN 结构图，图中每个箭头代表做一次变换，也就是说箭头连接带有权值。左侧是折叠起来的样子，右侧是展开的样子，左侧中 h 旁边的箭头代表此结构中的“循环”体现在隐层。

在展开结构中我们可以观察到，在标准的 RNN 结构中，隐层的神经元之间也是带有权值的。也就是说，随着序列的不断推进，前面的隐层将会影响后面的隐层。图中 O 代表输出，y 代表样本给出的确定值，L 代表损失函数，我们可以看到，“损失”也是随着序列的推荐而不断积累的。

除上述特点之外，标准 RNN 的还有以下特点：

1、权值共享，图中的 W 全是相同的，U 和 V 也一样。

2、每一个输入值都只与它本身的那条路线建立权连接，不会和别的神经元连接。

以上是 RNN 的标准结构，然而在实际中这一种结构并不能解决所有问题，例如我们输入为一串文字，输出为分类类别，那么输出就不需要一个序列，只需要单个输出。如图。

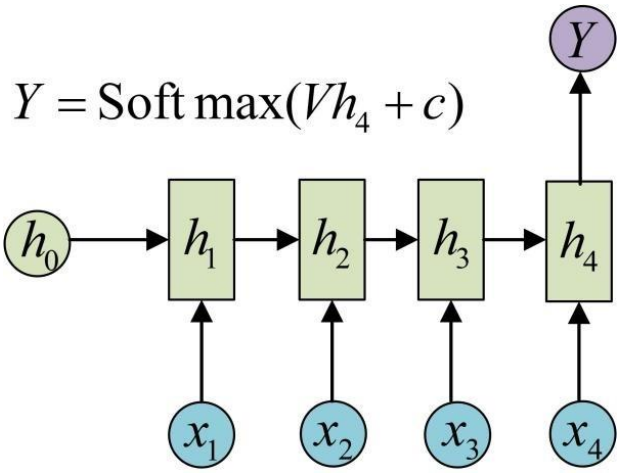

图 2-9 RNN 结构

同样的，我们有时候还需要单输入但是输出为序列的情况。那么就可以使用如下结构：

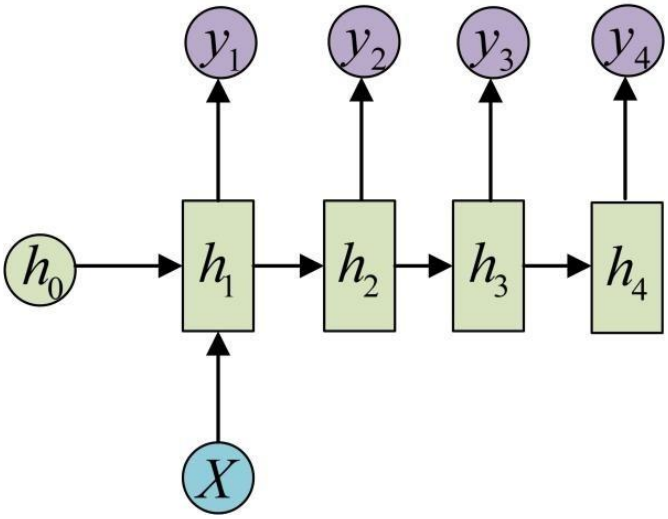

图 2-10 RNN 结构

还有一种结构是输入虽是序列，但不随着序列变化，就可以使用如下结构：

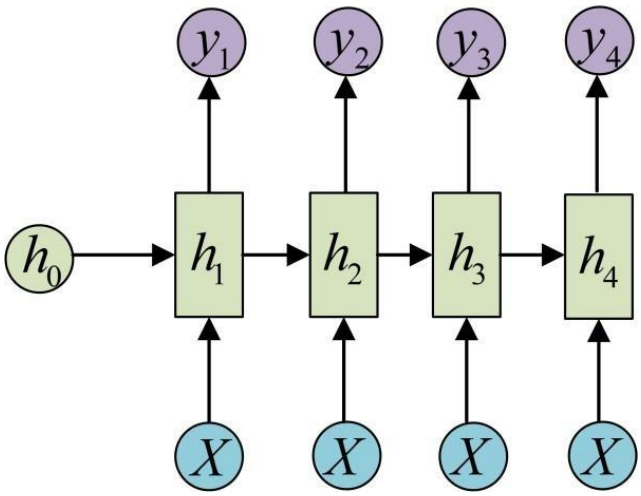

图 2-11 RNN 结构

原始的 RNN 要求序列等长，然而我们遇到的大部分问题序列都是不等长的，如机器翻译中，源语言和目标语言的句子往往并没有相同的长度。

下面我们来介绍 RNN 非常重要的一个变种：NvsM。这种结构又叫 Encoder-Decoder 模型，也可以称之为 Seq2Seq 模型。

（1） $c = h_4$

（2） $c = q(h_4)$

（3） $c = q(h_1, h_2, h_3, h_4)$

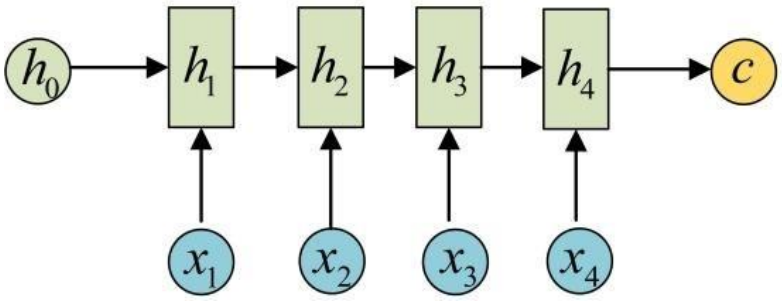

图 2-12 Encoder-Decoder

从名字就能看出，这个结构的原理是先编码后解码。左侧的 RNN 用来编码得到 c，拿到 c 后再用右侧的 RNN 进行解码。得到 c 有多种方式，最简单的方法就是把 Encoder 的最后一个隐状态赋值给 c，还可以对最后的隐状态做一个变换得到 c，也可以对所有的隐状态做变换。

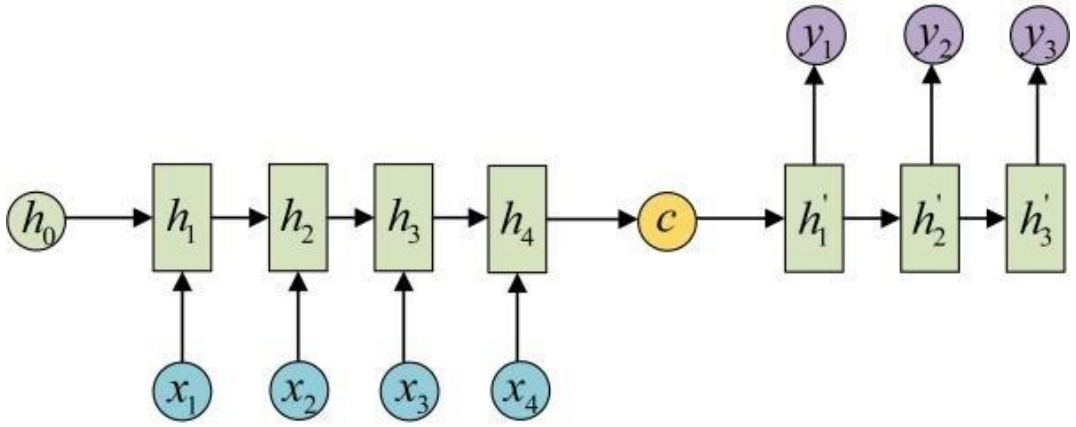

图 2-13 Encoder-Decoder

除了以上这些结构以外 RNN 还有很多种结构，用于应对不同的需求和解决不同的问题。但相同的是循环神经网络除了拥有神经网络都有的一些共性元素之外，它总要在一个地方体现出“循环”，而根据“循环”体现方式的不同和输入输出的变化就形成了多种 RNN 结构。

下面推导一下标准 RNN 的前向传播过程。x 是输入，h 是隐层单元，o 为输出，L 为损失函数，y 为训练集的标签。这些元素右上角带的 t 代表 t 时刻的状态，其中需要注意的是，因策单元 h 在 t 时刻的表现不仅由此刻的输入决定，还受 t 时刻之前时刻的影响。V、W、U 是权值，同一类型的权连接权值相同。

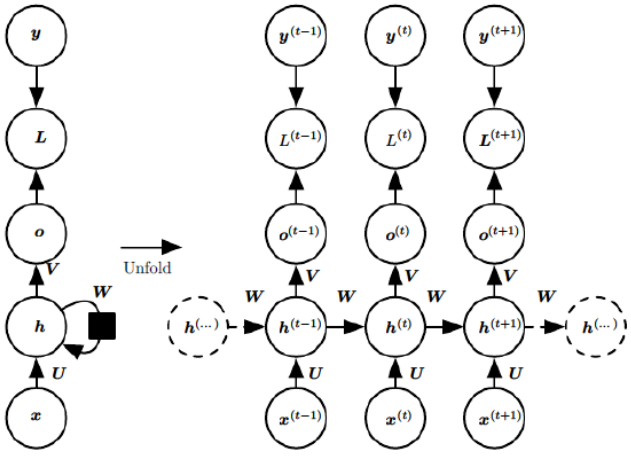

图 2-14 RNN 模型

有了上面的理解，前向传播算法就比较简单，对于 t 时刻：

$$h^{(t)} = \phi(Ux^{(t)} + Wh^{(t-1)} + b)$$

其中 ϕ 为激活函数，一般来说会选择 tanh 函数，b 为偏置。

t 时刻的输出就更为简单：

$$o^{(t)} = Vh^{(t)} + c$$

最终模型的预测输出为：

$$y^{(t)}=\sigma(o^{(t)})$$

其中 σ 为激活函数，通常 RNN 用于分类，故这里一般用 softmax 函数。

2.4 MLP

多层感知器(multilayer perception，MLP)也叫前向传播网络、深度前馈网络，是最基本的深度学习网络结构。MLP 由若干层组成，每一层包含若干个神经元。激活函数采用径向基函数的多层感知器被称为径向基网络(radial basis function network)。多层感知器的前向传播如图所示.

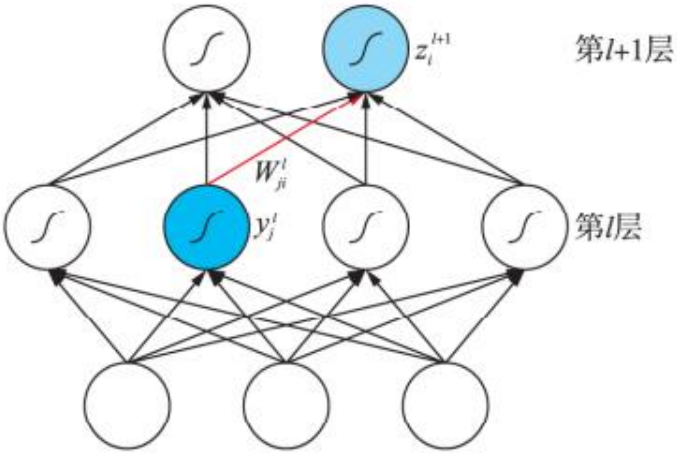

图 2-15 MLP 网络

MLP 的前向传播公式：

$$z_i^{l+1} = \sum_j W_{ji}^l y_j^l + b_i^l$$

$$y_i^{l+1} = f(z_i^{l+1})$$

其中，y_j^L是第l层的第j个神经元的输出，z_i^{l+1}是第 $l+1$层 的第i个神经元被激活函数作用之前的值，W_{ji}^l是第l层的第j个神经元与第$l+1$层的第i个神经元之间的权重，b_i^l是偏置，$f(\cdot)$ 是非线性激活函数，常见的有径向基函数、ReLU、PReLU、Tanh、Sigmoid 等。

如果采用均方误差(mean squared error) ，则损失函数为 $J = \frac{1}{2}\sum_i (y_i^L - y_i)^2$ 。其中，y_i^L是神经网络最后一层第i个神经元的输出，y_i是 第i个神经元的真实值. 神经网络训练的目标是最小化损失函数，优化方法通常采

用批梯度下降法。

2.5 GAN

生成对抗网络（Generate Adversarial Nets，GAN）提出于 2014 年，用于通过对抗过程来评估生成模型。GAN 架构是由一个针对对手（即一个学习模型或数据分布的判别模型）的生成模型组成。

通过对抗过程来估计生成网络，同时训练两个模型：一个生成模型 G，用来捕捉数据分布，一个判别模型 D，用来估计来自训练数据而不是模型 G 中的样本的可能性。关于模型 G 的训练程序是最大化模型 D 犯错的可能性。这个框架与极大极小双方博弈相关。在任意函数 G 和 D 的距离内，存在一个唯一的解决方法，G 恢复训练数据的分布，D 在任何地方都等于 1/2。在这种情况下，G 和 D 被定义为多层感知器，整个系统可以用反向传播来训练。在整个训练和样本生成阶段，这不需要任何的马尔科夫链或者展开的近似推理网络。

接下来说明 GAN 生成图片的流程。

GAN 有两个网络，一个是 generator，一个是 discriminator，从双人零和博弈中受启发，通过两个网络互相对抗来达到最好的生成效果。首先，有一个一代的 generator，它能生成一些很差的图片，然后有一个一代的 discriminator，它能准确的把生成的图片，和真实的图片分类，简而言之，这个 discriminator 就是一个二分类器，对生成的图片输出 0，对真实的图片输出 1。

接着，开始训练出二代的 generator，它能生成稍好一点的图片，能够让一代的 discriminator 认为这些生成的图片是真实的图片。然后会训练出一个二代的 discriminator，它能准确的识别出真实的图片，和二代 generator 生成的图片。以此类推，会有三代，四代……n 代的 generator 和 discriminator，最后 discriminator 无法分辨生成的图片和真实图片，这个网络就拟合了。

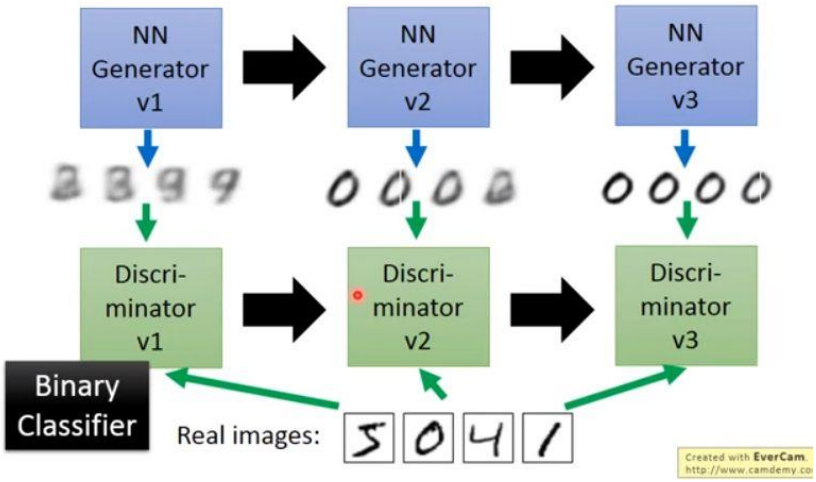

图 2-16 GAN 生成图片

接下来对 GAN 的模型原理进行推导。

首先我们知道真实图片集的分布$P_{data}(x)$，x 是一个真实图片，可以想象成一个向量，这个向量集合的分布就是P_{data}。

我们现在有的 generator 生成的分布可以假设为$P_G(x;\theta)$，这是一个由θ控制的分布，θ是这个分布的参数（如果是高斯混合模型，那么θ就是每个高斯分布的平均值和方差）。

假设我们在真实分布中取出一些数据，$\{x^1, x^2, \ldots, x^m\}$，我们想要计算一个似然$P_G(x^i;\theta)$，对于这些数据，在生成模型中的似然就是$L = \prod_{i=1}^{m} P_G(x^i;\theta)$，我们想要最大化这个似然，等价于让 generator 生成那些真实图片的概率最大。这就变成了一个最大似然估计的问题了，我们需要找到一个θ^*来最大化这个似然。

$$
\begin{aligned}
\theta^* &= arg\ \max_{\theta} \prod_{i=1}^{m} P_G(x^i;\theta) \\
&= arg\ \max_{\theta}\ log \prod_{i=1}^{m} P_G(x^i;\theta) \\
&= arg\ \max_{\theta} \sum_{i=1}^{m} logP_G(x^i;\theta) \\
&\approx arg\ \max_{\theta}\ E_{x\sim P_{data}}[logP_G(x;\theta)] \\
&= arg\ \max_{\theta} \int_x P_{data}(x) logP_G(x;\theta)dx - \int_x P_{data}(x) logP_{data}(x)dx \\
&= arg\ \max_{\theta} \int_x P_{data}(x)(logP_G(x;\theta) - logP_{data}(x))dx \\
&= arg\ \min_{\theta} \int_x P_{data}(x) log\frac{P_{data}(x)}{P_G(x;\theta)}dx \\
&= arg\ \min_{\theta}\ KL(P_{data}(x)||P_G(x;\theta))
\end{aligned}
$$

寻找一个θ^*来最大化这个似然，等价于最大化 log 似然。因为此时这 m 个数据，是从真实分布中取的，所以也就约等于，真实分布中的所有 x 在P_G分布中的 log 似然的期望。

真实分布中的所有 x 的期望，等价于求概率积分，所以可以转化成积分运算，因为减号后面的项和θ无关，所以添上之后还是等价的。然后提出共有的项，括号内的反转，max 变 min，就可以转化为 KL divergence 的形式了，KL divergence 描述的是两个概率分布之间的差异。

所以最大化似然，让 generator 最大概率的生成真实图片，也就是要找一个θ让P_G更接近于P_{data}。那如何来找这个最合理的θ呢？我们可以假设$P_G(x;\theta)$是一个神经网络。

首先随机一个向量 z，通过 G(z)=x 这个网络，生成图片 x，那么我们如何比较两个分布是否相似呢？只要我们取一组 sample z，这组 z 符合一个分布，那么通过网络就可以生成另一个分布P_G，然后来比较与真实分布P_{data}。

大家都知道，神经网络只要有非线性激活函数，就可以去拟合任意的函数，那么分布也是一样，所以可以用一直正态分布，或者高斯分布，取样去训练一个神经网络，学习到一个很复杂的分布。

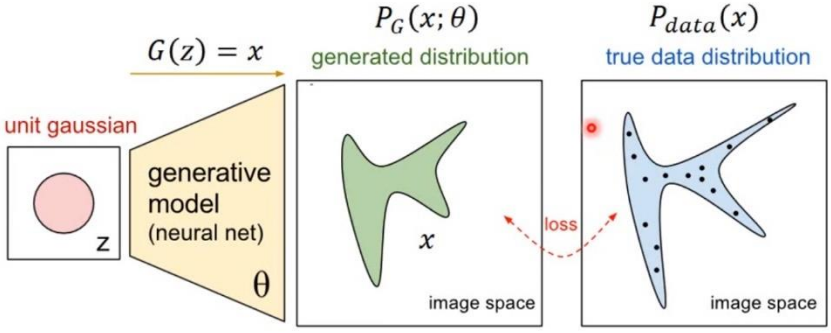

图 2-17 GAN

如何来找到更接近的分布，先给出 GAN 的公式：

$$V(G,D) = E_{x\sim P_{data}}[logD(x)] + E_{x\sim P_G}[log(1-D(x))]$$

这个式子的好处在于，固定 G，$\max V(G,D)$就表示P_G和P_{data}之间的差异，然后要找一个最好的 G，让这个最大值最小，也就是两个分布之间的差异最小。

$$G^* = arg \min_G \max_D V(G,D)$$

表面上看这个的意思是，D 要让这个式子尽可能的大，也就是对于 x 是真实分布中，D(x)要接近与 1，对于 x 来自于生成的分布，D(x)要接近于 0，然后 G 要让式子尽可能的小，让来自于生成分布中的 x，D(x)尽可能的接近 1

现在我们先固定 G，来求解最优的 D

- Given G, what is the optimal D* maximizing

$$V = E_{x\sim P_{data}}[logD(x)] + E_{x\sim P_G}[log(1-D(x))]$$

$$= \int_x P_{data}(x)logD(x)\,dx + \int_x P_G(x)log(1-D(x))\,dx$$

$$= \int_x [P_{data}(x)logD(x) + P_G(x)log(1-D(x))]\,dx$$

Assume that D(x) can have any value here

- Given x, the optimal D* maximizing

$$P_{data}(x)logD(x) + P_G(x)log\big(1 - D(x)\big)$$

a D b D

- Find D* maximizing: $f(D) = alog(D) + blog(1 - D)$

$$\frac{df(D)}{dD} = a \times \frac{1}{D} + b \times \frac{1}{1-D} \times (-1) = 0$$

$$a \times \frac{1}{D^*} = b \times \frac{1}{1-D^*} \qquad a \times (1 - D^*) = b \times D^*$$

$$a - aD^* = bD^*$$

$$D^* = \frac{a}{a+b} \Rightarrow \qquad D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_G(x)}$$

对于一个给定的 x，得到最优的 D 如上图，范围在(0,1)内，把最优的 D 带入 $\max_D V(G,D)$，可以得到：

$$\max_D V(G,D) = V(G,D^*) \qquad D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_G(x)}$$

$$= E_{x\sim P_{data}}\left[log\frac{P_{data}(x)}{P_{data}(x) + P_G(x)}\right] + E_{x\sim P_G}\left[log\frac{P_G(x)}{P_{data}(x) + P_G(x)}\right]$$

$$= \int_x P_{data}(x)log\frac{P_{data}(x)}{\frac{P_{data}(x) + P_G(x)}{2}}dx + \int_x P_G(x)log\frac{P_G(x)}{\frac{P_{data}(x) + P_G(x)}{2}}dx$$

$-2log2$

$$\max_D V(G,D) \qquad \mathrm{JSD}(P\parallel Q)=\frac{1}{2}D(P\parallel M)+\frac{1}{2}D(Q\parallel M) \qquad M=\frac{1}{2}(P+Q)$$

$$\max_D V(G,D) = V(G,D^*) \qquad D^*(x)=\frac{P_{data}(x)}{P_{data}(x)+P_G(x)}$$

$$= -2log2 + \int_x P_{data}(x) log \frac{P_{data}(x)}{(P_{data}(x)+P_G(x))/2} dx + \int_x P_G(x) log \frac{P_G(x)}{(P_{data}(x)+P_G(x))/2} dx$$

$$= -2\log 2 + \mathrm{KL}\left(\mathrm{P_{data}(x)}\middle|\middle|\frac{\mathrm{P_{data}(x)+P_G(x)}}{2}\right) + \mathrm{KL}\left(\mathrm{P_G(x)}\middle|\middle|\frac{\mathrm{P_{data}(x)+P_G(x)}}{2}\right)$$

$$= -2log2 + 2JSD\left(P_{data}(x)||P_G(x)\right)$$ Jensen-Shannon divergence

JS divergence 是 KL divergence 的对称平滑版本，表示了两个分布之间的差异，这个推导就表明了上面所说的，固定 G，$\max_D V(G,D)$表示两个分布之间的差异，最小值是-2log2，最大值为 0。现在我们需要找个 G，来最小化 $\max_D V(G,D)$，观察上式，当$P_G(x)=P_{data}(x)$时，G 是最优的。

三.国内外研究现状

深度学习已经使机器学习的许多实际应用成为可能，并延伸到人工智能的整个领域。与浅层学习相比，深层学习具有构建深层架构以学习更多抽象信息的优势。深度学习方法最重要的特性是它可以自动学习特征表示，从而避免大量耗时的工程。更好的芯片处理能力、机器学习算法的巨大进步以及可负担的计算硬件成本是深度学习蓬勃发展的主要原因。

2012 年 6 月，《纽约时报》披露了 Google Brain 项目，该项目拟计划在包含 16000 个中央处理单元的分布式并行计算平台上构建一种被称之为“深度神经网络”的类脑学习模型，其主要负责人为机器学习界的泰斗、来自斯坦福大学的 Ng 教授和 Google 软件架构天才、大型并发编程框架 MapReduce 的作者 Jeff Dean；2012 年 10 月，在天津举行的“21 世纪的计算大会”上，微软首席研究官 Rick Rashid 展示了一套全自动同声传译系统，演讲者的英文能够被实时、流畅地转换成与之对应的、音色相近的中文，其背后的关键技术深度神经网络也逐渐被人们

所知。

2013 年 1 月，作为百度公司创始人兼 CEO 的李彦宏在其年会上宣布了成立百度研究院的计划， 并且强调首当其冲的就是组建“深度学习研究所”．在 2015 年 3 月 9 日的两会期间，李彦宏又提议设立 “中国大脑”计划的提案，与 2013 年 1 月和 2013 年 4 月的“欧盟大脑计划”和“美国大脑计划”相呼应.2015 年 3 月，阿里巴巴公司的创始人马云通过支付宝的“刷脸支付”功能，在德国举行的 IT 博览会上成功购得了一款汉诺威纪念邮票.这一人脸识别技术在商业领域的应用雏形所采用的是基于神经网络的技术，其网络训练所使用的正是“深度学习算法”。

在学术界，以 Hinton、LeCun、Bengio 和 Ng 等为代表的神经网络大师们不断将深度学习的研究推向新的高峰，对包括计算机视觉、自然语言处理和机器学习在内的诸多领域带来了深远的影响．自 2006 年深度学习出现以来，关于深度学习理论和应用方面的研究文献在国际知名期刊和会议上不断涌现，如 narure、science、PAMI、NIPS、CVPR、ICML 等。同时，由 Bengio 等人编写的第一本关于深度学习的专著“Deep Learning”也由 MIT 出版社出版．包括斯坦福大学、卡内基梅隆大学、纽约大学、多伦多大学等在内的机构都提供了深度学习的公开课程，并公开了实验数据和源代码，为深度学习的进一步发展做出了贡献。

在国内，深度学习也受到了学术界的广泛关注，但目前主要是以深度学习的应用研究为主，在理论方面的工作相对较少．以北京大学、 浙江大学、上海交通大学、哈尔滨工业大学和西安电子科技大学等为代表的研究人员将深度学习算法应用到遥感图像分类、多媒体检索、交通流预测和盲图像质量评价等领域，取得了较传统方法更优的效果。

2020 年，是人工智能的盛夏，人工智能研究工作正处在爆炸增长期：2019 年全球发布了超过 12 万篇人工智能研究领域的同行评审论文。自 2000 年以来，人工智能领域论文在同行评审论文中的占比，从 0.8%一路攀升至 2019 年的 3.8%[11]。

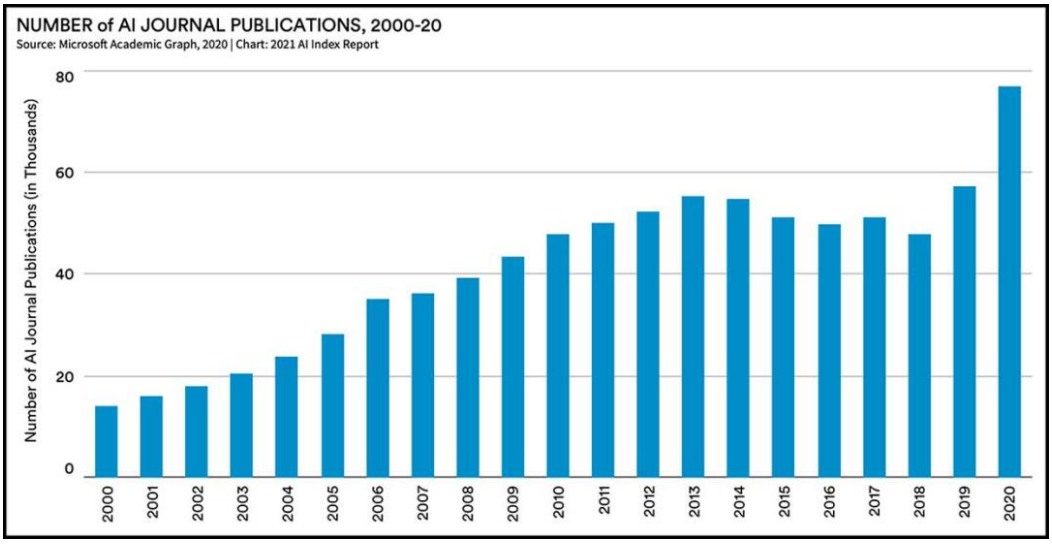

图 3-1 人工智能论文发表趋势

下面具体分析一下。

3.1 深度学习典型应用

3.1.1 语音处理

深度学习最先在语音处理领域取得突破性进展，无论是在标准小数据集上还是大数据集上。语音处理领域主要有两大任务: 语音识别和语音合成。深度学习广泛应用于语音识别中，Google 推出端到端的语音识别系统、百度推出语音识别系统 Deep Speech 2。2016 年，微软在日常对话数据上的语音识别准确率达到 5.9% ，首次达到人类水平。各大公司也都用深度学习来实现语音合成，包括 Google、Apple、科大讯飞等。Google DeepMind 提出并行化 WaveNet 模型来进行语音合成，百度推出产品级别的实时语音合成系统 Deep Voice 3。

3.1.2 计算机视觉

深度学习被广泛应用于计算机视觉各种任务，包括交通标志检测和分类、人脸识别、人脸检测、图像分类、多尺度变换融合图像、物体检测、图像语义分割、实时多人姿态估计、行人检测、场景识别、物体跟踪、端到端的视频分类、视频里的人体动作识别等。另外，还有一些很有意思的应用，如给黑白照片自动着彩色、将涂鸦变成艺术画、艺术风格转移、去掉图片里的马赛克等。牛津大学和 Google DeepMind 还共同提出了 LipNet 来读唇语，准确率达到 93% ，远超人类

52%的平均水平。

3.1.3 自然语言处理

NEC Labs America 最早将深度学习应用于自然语言处理领域。目前处理自然语言时通常先用 word2vec 将单词转化成词向量，其可以作为单词的特征。自然语言处理领域各种任务广泛用到了深度学习技术，包括词性标注、依存关系语法分析、命名体识别、语义角色标注、只用字母的分布式表示来学习语言模型、用字母级别的输入来预测单词级别的输出、Twitter 情感分析、中文微博情感分析、文章分类、机器翻译、阅读理解、自动问答、对话系统等。

3.1.4 其他

在生物信息学方面，深度学习能够被用来预测药物分子的活动，预测人眼停留部位，预测非编码 DNA 基因突变对基因表达与疾病的影响。金融行业积累了大量的数据，故深度学习在金融方面也有众多应用，包括金融市场预测、证券投资组合、保险流失预测等，也相应涌现出一批金融科技创业公司。另外，基于深度学习的实时发电调度算法能在满足实时发电任务的前提下，使机组总污染物排放量降低，达到节能减排的目的。深度学习还可以诊断电动潜油柱塞泵的故障，避免故障事故的发生，有效延长检泵周期。深度学习应用于强非线性、复杂的化工过程软测量建模中也能获得很好的精度。

3.2 经典深度学习网络

在模型原理描述环节，主要在五类深度网络模型中各选择一个具体模型进行详细推理，下面更详细的介绍一下这五类深度网络模型的研究发展。

3.2.1 CNN

LeNet 是一项开创性的工作，由 Yann LeCun 在成功迭代后命名为 Lenet-5。当时，LeNet 架构主要用于字符识别任务，如读取邮政编码、数字等。随着 LeNet 的流行，LeCun 等人也引入了 MNIST 数据库，该数据库被称为数字识别领域的标准基准。

ZFNet 获得 2013 年 ILSVRC。它是由马修·泽勒和罗布·弗格斯提出的。它被

称为 ZFNet。这是对 AlexNet 的改进，通过调整架构超参数，特别是通过扩展中间卷积层的大小，使第一层的步长和滤波器大小更小。

VGGNet 是 2014 年牛津大学 VGG 小组的亚军。它比 AlexNet 有所改进，总共有 19 层。它的主要贡献在于表明网络的深度或层数是良好性能的关键因素。尽管 VGGNet 在 ImageNet 数据集上实现了惊人的准确性，但它在即使是最中型的图形处理单元(GPU)上的部署也是一个问题，因为在内存和时间方面都有巨大的计算需求。由于卷积层的宽度大，它变得低效。

GoogLeNet 是 Szegedy 等人从获得 ILSVRC 2014 发明的。它的主要贡献是开发了一个初始模块，大大减少了网络中的参数数量。初始模块近似于一个稀疏的 CNN，具有正常的密集结构。由于如前所述，只有少量神经元是有效的，所以特定核大小的卷积滤波器的宽度/数量保持较小。此外，它使用不同大小的卷积来捕捉不同比例的细节。该模块的另一个显著特点是它有一个所谓的瓶颈层。这有助于大幅度降低计算要求。GoogLeNet 所做的另一个改变是用一个简单的全局平均池来代替末端的 FCLs，该池在最后一个卷积层之后对整个 2D 要素图的信道值进行平均。这大大减少了参数的总数。这可以从 AlexNet 中理解，其中 FCLs 包含大约 90%的参数。使用大的网络宽度和深度允许 GoogLeNet 在不影响准确性的情况下移除 FCLs。它在 ImageNet 上达到 93.3%的前 5 名准确率，比 VGG 快得多。

由何凯明等人开发的 ResNet (Residual Network)是 ILSVRC 2015 的获胜者。ResNet 是一个 152 层的网络，比它发明时通常看到的要深 10 倍。它的特点是特殊的跳过连接和大量使用批处理标准化。它使用全局平均池，然后是分类层。它比 VGGNet 和 googleet 获得更好的准确性，同时比 VGGNet 的计算效率更高。ResNet-152 达到 95.51%的 top-5 准确率。

DenseNet 由高煌等人发表，获得 CVPR 2017 最佳论文奖。它以前馈的方式将每一层直接连接到其他每一层。DenseNet 已被证明在四个高度竞争的目标识别基准任务(CIFAR-10、CIFAR-100、SVHN 和 ImageNet)上比以前的最先进的架

构获得了显著的改进。

CNN 常见应用场景：语音用户界面，自然语言处理，计算机视觉等；微软的 WaveNet 模型；分类情感；图像分类；指导人工智能代理玩游戏；无人机；图像中文字识别。

3.2.2 RNN

(1)Long Short Term Memory (LSTM)

长短记忆神经网络——通常称作 LSTM，是一种特殊的 RNN，能够学习长的依赖关系。他们由 Hochreiter＆Schmidhuber 引入，并被许多人进行了改进和普及。他们在各种各样的问题上工作的非常好，现在被广泛使用。

LSTM 是为了避免长依赖问题而精心设计的。记住较长的历史信息实际上是他们的默认行为，而不是他们努力学习的东西。

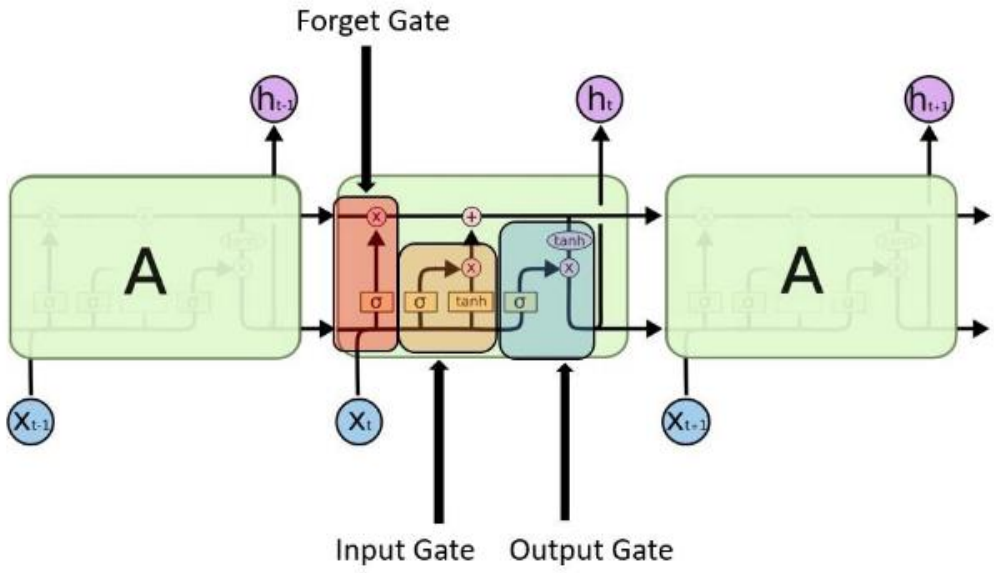

图 3-2 LSTM 模型

(2)Gated Recurrent Unit (GRU)

GRU 是 LSTM 网络的一种效果很好的变体，它较 LSTM 网络的结构更加简单，而且效果也很好，因此也是当前非常流形的一种网络。GRU 既然是 LSTM 的变体，因此也是可以解决 RNN 网络中的长依赖问题。

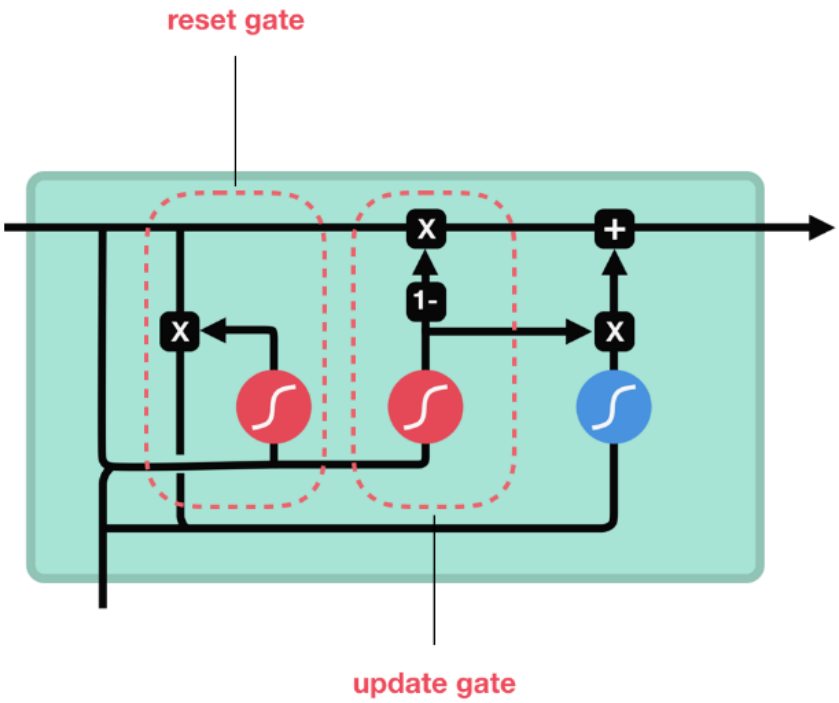

图 3-3 GRU 模型

(3)Convolutional LSTM (ConvLSTM)

GRU 是 LSTM 的变体，不一样的地方在于减少了门控制，只有遗忘门和重置门，但是效果同样很好。ConvLSTM 层不仅保留了 FC-LSTM 的优点，而且由于其固有的卷积结构，也适用于时空数据。

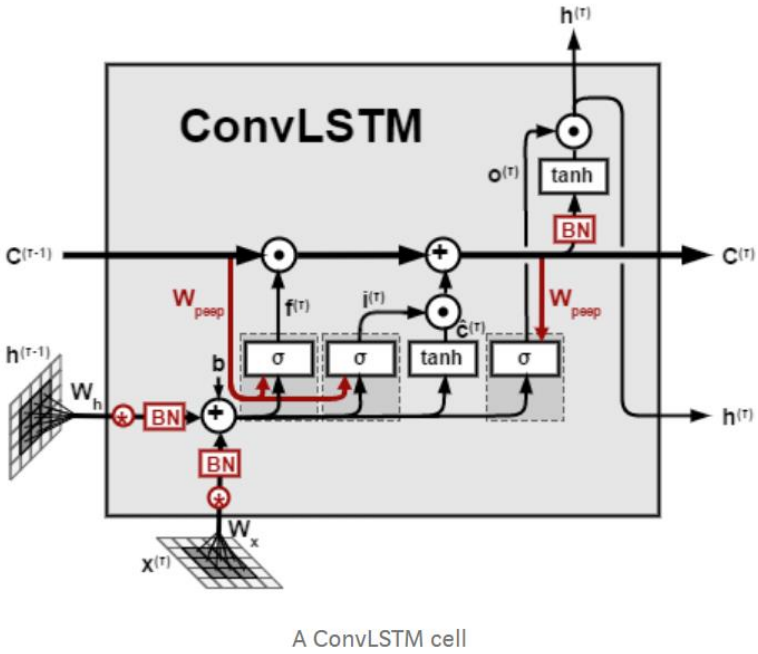

图 3-4 ConvLSTM 模型

(4)RNN 应用场景

包括 LSTM 和 GRU 在内的神经网络应用于张量处理。使用 RNN 技术的自然语言处理，包括母语教学和母语小组。基于多语言识别系统的卷积神经网络于 2017 年被提出。使用 RNNs 的时间序列数据分析。最近，基于预先训练的深度神经网络提出了时间网络用于时间序列分类。语音和音频处理，包括大规模声学建模的 LSTMs。使用卷积 RNNs 的声音事件预测。使用卷积语法的音频标记。早期心力衰竭检测是使用 RNNs 提出的。无线网络用于跟踪和监控:数据驱动的交通预测系统是使用图形卷积 RNN (GCRNN) 提出的。基于神经网络的模型提出了基于 LSTM 的网络流量预测系统。双向深度 RNN 应用于驾驶员行为预测。使用 RNN 的车辆轨迹预测。使用带有单词包的 RNN 进行动作识别。网络安全中的集合异常检测.

3.2.3 AE

(1)Split-Brain Auto-encoder

最近，由伯克利人工智能研究(BAIR)实验室提出的裂脑人工智能是对用于无监督表示学习的传统自动编码器的架构修改。在这种体系结构中，网络被分成不相交的子网络，其中两个网络试图预测整个图像的特征表示。

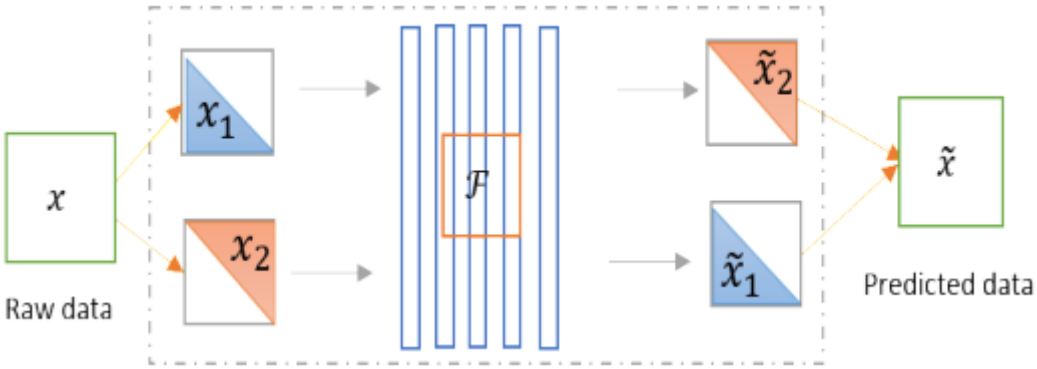

图 3-5 裂脑人工智能原理

(2)AE 应用场景

自动编码器应用于生物信息学和网络安全。我们可以应用自动进化算法进行无监督特征提取，然后应用赢家通吃算法对这些样本进行聚类，以生成标签。在过去的十年里，人工智能已经作为一种编码和解码技术被用于其他深度学习方法，

包括 CNN、DNN、RNN。

3.2.4 GAN

Yann LeCun 曾说过“GAN is the best concept proposed in the last ten years in the field of deep learning (Neural networks)”，GAN 是近十年来在该领域提出最好的理论。

GAN 用于使用超分辨率方法生成照片真实感图像。用半监督和弱监督方法进行语义分割的遗传算法。文本条件辅助分类器，用于根据文本描述生成或合成图像。多样式生成网络(MSG-Net)，它保留了基于快速优化方法的功能，该网络在多个尺度上匹配图像样式，并将计算负担放入训练中。大多数时候，视觉系统与雨、雪和雾作斗争。最近提出了一种使用氮化镓的单图像去雨系统。

使用生成层次神经网络模型的端到端对话系统。此外，GAN 还被用于语音分析领域。最近，GAN 被用于语音增强，称为 SEGAN，它结合了进一步的以语音为中心的设计，以逐步提高性能。甘为符号域和音乐生成，表现与 RNN 相当。

用于医学成像和医学信息处理的遗传神经网络，用于具有瓦瑟斯坦距离和感知损失的医学图像去噪的遗传神经网络。神经根也可以用于有条件神经根的脑肿瘤分割。提出了一种通用的医学图像分割方法，该方法使用一种称为 SegAN 的 GAN。在深度学习革命之前，压缩感知是最热门的话题之一。然而，深度感测用于自动磁共振成像的压缩感测。此外，由于隐私问题，GAN 也可用于健康记录处理，电子健康记录(EHR)仅限于或不像其他数据集一样公开。GANs 应用于合成 EHR 数据，可以降低风险。用递归神经网络和递归条件神经网络生成时间序列数据。LOGAN 由用于检测过拟合和识别输入的生成和鉴别模型的组合组成。

四.算法比较

下面是部分深度学习算法的比较表格。

年份	深度模型	提出者	网络结构	相关训练算法	模型特点及解决问题	存在问题
1969	Perceptron[26] 感知机	M.Minsky S.Papert	–	–	线性可分问题	线性不可分问题
1974	BP[31] 反向传播	P.J.Werbos	–	链式积分法	线性不可分问题;从简单神经网络到复杂神经网络的推广	局部最优解问题过拟合问题
1985	BM[37] 玻尔兹曼机	D.H.Ackley G.E.Hinton T.J Sejnowski	多层	–	统计力学中一种能量模型,随机神经网络实例	难于计算分布
1986	RBM[40] 受限玻尔兹曼机	P.Smolensky	2 层无向边	对比散度法	容易求得 BM 的概率分布;具有无监督学习能力	效率低
2002	CRBM[46] 连续受限玻尔兹曼机	H.Chen A.Murray	2 层	MCD;BP	能够对连续数据建模	参数调优困难
2010	SGRBM[47] 稀疏组受限玻尔兹曼机	Luo Heng Shen Ruimin Niu Cahngyong	2 层	稀疏惩罚对数似然;BP	稀疏表示符合生物学特征惩罚隐藏单元的损失	隐藏单元分组方式和依据尚不明确
2006	DBN[41] 深度置信网络	G.E.Hinton R.R.Salakhutdinov	多层有/无向边全连接	贪心逐层训练算法;BP	RBM 的堆叠;以无监督学习到的参数作为有监督学习的初始值,从而解决了 BP 的问题	可视层只能接收二值数值;优化困难
2006	DBM[39] 深度玻尔兹曼机	R.R.Salakhutdinov G.E.Hinton	多层无向边全连接	BP	BM 的特殊形式;自下而上生成结构;减少传播造成的误差	效率低
2009	CDBN[48] 卷积深度置信网络	Lee Honglak Grosse Roger Ranganath Rajesh Ng Andrew Y	多层无向边部分连接	贪心逐层训练算法;BP	采用概率最大池能够对高维图像做全尺寸衡量,并对输入的局部变换具有不变性	优化困难
2013	SDBN[49] 稀疏深度置信网络	X.Halkias S.Paris H.Glotin	多层有/无向边全连接	混淆范数;BP	混淆范数作为稀疏约束的 DBN 结构	优化困难
1986	AE[50] 自动编码器	D.E.Rumelhart G.E.Hinton R.J.Williams	3 层	贪心逐层训练算法	通过编码器和解码器工作完成训练;通过损失函数最小化求出网络的参数;	不能用于分类
2006	DAE[41] 深自动编码器	G.E.Hinton R.R.Salakhutdinov	多层	贪心逐层训练算法;BP	无监督逐层贪心训练算法完成对隐含层的预训练;并通过 BP 微调,显著降低了性能指数;	隐藏层数量和神经元的数量增多导致梯度稀释

2007	SAE[51] 堆叠自动编码器	B.Yoshua L.Pascal P.Dan Hugo Larochelle	多层 有/无向边 全连接	梯度下降 算法;BP	将 DBN 中的 RBM 替换为 AE 后的生成模型;通过将第一层的贝努力分布的输入改为高斯分布,扩展成可输入任意值进行训练	同上
2007	SAE[52] 稀疏自动编码器	M.Ranzato, Y.Boureau Y.Lecun	3 层	梯度下降算法;BP	降维,学习稀疏的特征表达	同上
2008	dAE[53] 降噪自动编码器	P.Vincent H.Larochelle Y.Bengio A.Manzagol	3 层	梯度下降算法;BP	在破损数据的基础上训练;使训练得到的权重噪声较小,从而提高鲁棒性	同上
2010	SDAE[54] 堆叠消噪 自动编码器	P.Vincent H.Larochelle I.Lajoie Y.Bengio 等	多层	梯度下降 算法;BP	将多个 dAE 堆叠起来形成深度网络结构,用来提取特征表达	同上
2013	SSAE[55] 稀疏堆叠 自动编码器	Jiang Xiaojuan Zhang Yinghua Zhang Wensheng 等	多层	梯度下降 算法;BP	在 SAE 的损失函数上加入稀疏惩罚值形成的深层网络	同上
1998	CNN[56] 卷积神经网络	Y.Lecun L.Bottou Y.Bengio P.Haffner	多层 无向边 局部连接 共享权值	梯度下降 算法;BP	包含卷积层和子采样层;可以接受 2D 结构的输入;具有较强的畸变鲁棒性	要求较高计算能力的资源
2011	SCAE[57] 堆叠卷积 自动编码器	Masci Jonathan Meier Ueli Dan Cireşan 等	多层	梯度下降算法;BP	堆叠的 CAE 结构,每层采用没有正则项的传统梯度下降算法进行训练	同上
2012	DCNN[58] 深度卷积神经网络	A.Krizhevsky I.Sutskever G.Hinton	多层 局部连接 共享权值	梯度下降 算法;BP	CNN 的深层结构,采用纯监督学习广泛应用于图像识别	同上
1990	SRN[59] 简单循环网络	J.L.Elman	3 层	BPTT;梯度 下降算法	时间维度上的深层结构;上一时刻的输出是下一时刻的输入	长时间依赖问题
1995	RNN[60] 循环神经网络	S.E.Hihi M.Q.Hc-J Y.Bengio	多层	BPTT;梯度 下降算法	多层的时间维度上的深层结构;能够处理序列数据	梯度消失或梯度爆炸
1997	LSTM[61] 长短是记忆	S.Hochreiter J.Schmidhuber	多层	BPTT;梯度 下降算法	通过为每一个神经元引入 gate 和存储单元,能够解决 RNN 所面临的梯度消失或爆炸问题 由于具有记忆功能,能够处理较为复杂的序列数据	训练复杂度较高、解码时延较高

2014	GRU[62] 关口循环单元	K.Cho B.Van Merrienboer D.Bahdanau Y.Bengio	多层	BPTT;梯度下降算法	相比于 LSTM,只设置一个更新关口,运行比 LSTM 更快,更容易	表达能力相对弱
2014	Attention[63] 注意力机制	V.Mnih N.Heess A.Graves K.Kavukcuoglu	—	—	受人类的注意力机制的启发,每次处理注意力部分的数据,减少任务复杂度	增加了存储开销
2014	GAN[64] 生成对抗网络	Goodfellow Ian Pougetabadie Jean Mirza Mehdi Xu Bing Wardefarley David Ozair Sherjil Courville Aaron Bengio Yoshua	多层 无向边 局部连接 共享权值	BP;dropout	由不同网络组成,成对出现,协同工作 一个网络负责生成内容,另一个负责对内容进行评价多以前馈网络和卷积网络的结合为主	训练较难;训练过程不稳定
2015	DCGAN[65] 深度卷积 生成对抗网络	A.Radford L.Metz S.Chintala	多层 无向边 局部连接 共享权值	BP;梯度下降算法	GAN 基于 CNN 的扩展,可以从训练数据中学习近似的分布情况	训练过程不稳定

注:'—'表示尚不明确或不适用。

图 4-1 深度学习算法比较

五.未来发展方向

人工智能在以下这些领域十分受用，而深度学习在其中扮演重要角色[10]：

(1)缺乏人类专家（例如火星导航）；

(2)人们尚无法解释的专业知识（演讲、认知、视觉和语言理解）；

(3)问题的解决方案随时间不断变化（追踪、天气预报、偏好、股票、价格预测）；

(4)解决方案需要适应特定情况（生物统计学、个性化）；

(5)人类的推理能力有限，而问题的规模却很大（计算网页排名、将广告匹配到 Facebook、情感分析）。

在这里总结一下深度学习算法未来可能发展的方向。

5.1 训练方面

(1)缩短训练时间，节约训练计算资源

(2)解决梯度消失问题，降低训练难度

(3)降低对大规模带标签数据的依赖

(4)分布式训练问题

5.2 落地方面

(1)对抗样本攻击

(2)提升鲁棒性

(3)减少超参数

(4)加强可解释性

(5)减小模型

5.3 功能方面

(1)使用深度学习进行大数据分析

深度学习的兴起很大程度上归功于海量可用的数据。当前，实验神经科学与各个工程应用领域给我们带来了呈指数增长的海量复杂数据，通过各种不 同的形态被呈现出来（如文本、图像、音频、视频、基因数据、复杂网络等），且具有不同的分布，使得神经 网络所面临的数据特性发生了本质变化．这给统计学习意义下的神经网络模型的结构设计、参数选取、训练算法，以及时效性等方面都提出了新的挑战．因此，如何针对大数据设计有效的深度神经网络模型与学习理论，从指数增长的数据中获得指数增长的知识，是深度学习深化研究中必须面临的挑战。

(2)深度学习方法要有可扩展性

(3)在数据不可用于学习系统的情况下（尤其是对于计算机视觉任务，例如反向图形），生成数据的能力非常重要

(4)特殊用途设备的低能耗技术，如移动端智能，FPGA 等

(5)多任务和迁移学习（泛化）或多模块学习

(6)在学习中处理因果关系

深度学习三大巨头之一的 Bengio 在 2020 年底的一场讲座中提到，这是他们目前研究项目的核心。近期，Yoshua Bengio 团队发表的《Towards Causal Representation Learning》[12]，该论文已被《Proceedings of the IEEE》期刊接收。

一直以来机器学习和因果推理是两个相对独立的研究方向，各有优缺点。但在过去数年，两者开始互相借鉴，推进彼此的发展。如机器学习领域的蓬勃发展促进了因果推理领域的发展。采用决策树、集成方法、深层神经网络等强大的机器学习方法，可以更准确地估计潜在结果。于是，近年来，将两者结合起来的因果表示学习（Causal Representation Learning）吸引了越来越多的关注，成为人类迈向 Human Level AI 的潜在方向。

六.总结

上面在介绍时，我们选择的是 5 类经典的、热门的深度学习模型，在最后，我们对所有深度学习算法进行统一的分类。我们可以简单将深度学习分为深度监督学习、深度无监督学习、深度混合学习和深度强化学习。深度学习在这些方面都取得了很大的成功。

6.1 深度监督学习

监督学习应用在当数据标记、分类器分类或数值预测的情况。LeCun 等人对监督学习方法以及深层结构的形成给出了一个精简的解释。Deng 和 Yu 提到了许多用于监督和混合学习的深度网络，并做出解释，例如深度堆栈网络 (DSN) 及其变体。Schmidthuber 的研究涵盖了所有神经网络，从早期神经网络到最近成功的卷积神经网络 (CNN)、循环神经网络 (RNN)、长短期记忆 (LSTM) 及其改进。

6.2 深度无监督学习

当输入数据没有标记时，可应用无监督学习方法从数据中提取特征并对其进行分类或标记。LeCun 等人预测了无监督学习在深度学习中的未来。Schmidthuber 也描述了无监督学习的神经网络。Deng 和 Yu 简要介绍了无监督学习的深度架构，并详细解释了深度自编码器。

6.3 深度强化学习

强化学习使用奖惩系统预测学习模型的下一步。这主要用于游戏和机器人，解决平常的决策问题。Schmidthuber 描述了强化学习中深度学习的进展，以及深度前馈神经网络 (FNN) 和循环神经网络在 RL 中的应用。Li 讨论了深度强化学习 (Deep Reinforcement Learning, DRL)、它的架构 (例如 Deep Q-Network, DQN) 以及在各个领域的应用。

七.参考文献

[1]Musab COŞKUN et al. AN OVERVIEW OF POPULAR DEEP LEARNING METHODS[J]. European Journal of Technique, 2017, : 165-176.
[2]LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. Nature 521, 436–444 (2015).
[3]Thomas H.Davenport; Paul Michelman.The AI Advantage: How to Put the Artificial Intelligence Revolution to Work[M].MITP.2018.
[4]Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016.
[5]Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classifi-cation with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems – Volume 1, NIPS'12, pages 1097–1105, USA, 2012. Curran Associates Inc.
[6]Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only lookonce: Unified, real-time object detection. CoRR, abs/1506.02640, 2015.
[7] Kingma D P, Welling M. Auto-Encoding Variational Bayes[J]. stat, 2014, 1050: 10.
[8]Carl Doersch. Tutorial on variational autoencoders. CoRR, 1606.05908, 2016.
[9]Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative ad-versarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence,and K. Q. Weinberger, editors, Advances in Neural Information Process-ing Systems 27, pages 2672–2680. Curran Associates, Inc., 2014.
[10] Md Zahangir Alom and Tarek M. Taha and Christopher Yakopcic and Stefan Westberg and Paheding Sidike and Mst Shamima Nasrin and Brian C Van Esesn and Abdul A S. Awwal and Vijayan K. Asari, The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches, arXiv:1803.01164 [cs.CV],2018
[11]Daniel Zhang,Saurabh Mishra, Eril Brynjolfsson,John Etchemendy,Deep Ganguli,Barbara Grosz,Terah Lyons,James Manyika,Juan Carlos Niebles,Michael Selitto,Yoav Shoham,Jack Clark,and Raymond Perrault,"The AI Index 2021 Annual Report",AI Index Steering Committee,Human-Centered AI Institute,Stanford University,Stanford,CA,March 2021
[12] Bernhard Schölkopf , Francesco Locatello , Stefan Bauer , Nan Rosemary Ke , Nal Kalchbrenner,Anirudh Goyal, Yoshua Bengio, Towards Causal Representation

Learning, Proceedings of the IEEE ,2021
[13] Gao Huang, Zhuang Liu, Kilian Q. Weinberger, and Laurens van der Maaten, Densely Connected Convolutional Networks, 2016, arXiv:1608.06993v4.
[14] Yan Lecun B Boser, John S Denker, D Henderson, Richard E Howard, W Hubbard, and Lawrence D Jackel, Handwritten digit recognition with a back-propagation network, 1990, In Advances in neural information processing systems. Citeseer.
[15] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, Gradient-based learning applied to document recognition, 1998a, Proceedings of the IEEE, pp. 2278–2324.
[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, Imagenet classification with deep convolutional neural networks, In Advances in neural information processing systems, 2012, pages 1097–1105.
[17] Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. CoRR, abs/1311.2901, 2013, Published in Proc. ECCV, 2014.
[18] Karen Simonyan, and Andrew Zisserman, Very Deep Convolutional Networks For Large-Scale Image Recognition, 2014, arXiv preprint arXiv:1409.1556.
[19] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, Going Deeper with Convolutions, 2014, Computer Vision and Pattern Recognition (CVPR 2015).
[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, Deep Residual Learning for Image Recognition, 2015, arXiv:1512.03385v1.
[21] Zhang, Richard, Phillip Isola, and Alexei A. Efros. "Split-brain autoencoders: Unsupervised learning by cross-channel prediction." arXiv preprint arXiv:1611.09842 (2016).