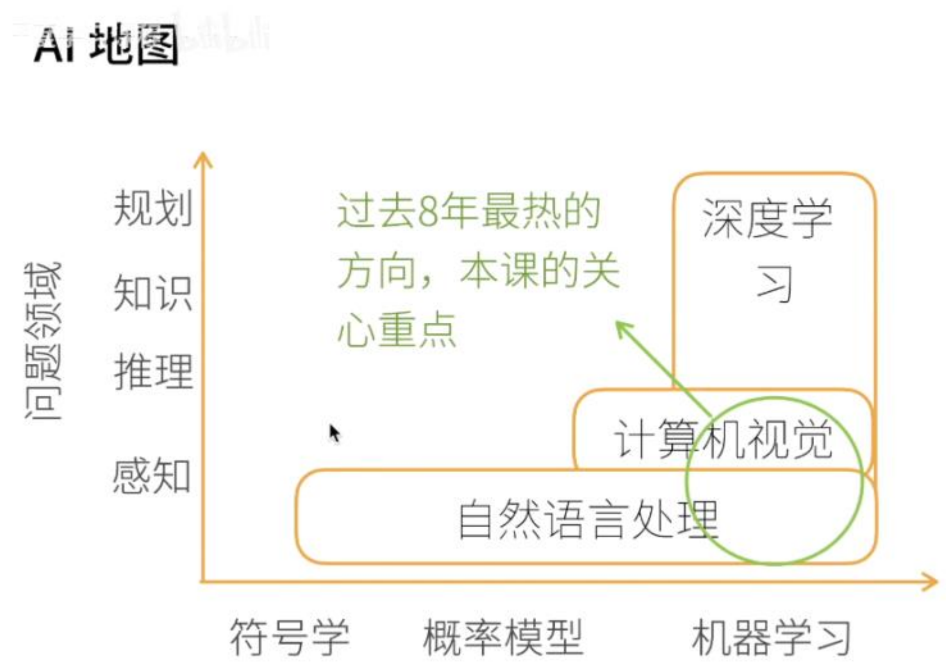
深度学习导读

2026年1月30日　　11:14



2017年后几乎所有团队的错误率都能实现在5%以内
深度学习的应用领域：
图片分类
物体检测和分割
样式迁移——不通过风格的图片合成
人脸合成
文字生成图片
文字生成模型
无人驾驶

案例研究：
广告点击

预测与训练

完整的故事

机器学习应用内有三类人：

名称	主要职责
领域专家	实现广告的展现、用户点击、应用设计、模型的使用、了解机器学习的作用（明白如何将模型与应用相拟合？结合）
数据科学家	将数据转化为模型
AI专家	提升模型精度和性能

一些问题：

Q 机器学习的可解释性是否有定论？

A 机器学习来说对于简单的模型有一些理解，但是对于复杂模型几乎已经放弃了解释，深度学

习的图片解释更如此。有效性和可解释性是两件事情。对于为什么有效，当然可以解释。但是人能否理解模型，为什么工作、什么时候不工作、有哪些地方有偏差等等是不同的。什么样的模型考虑空间或者时间信息、什么模型考虑泛化性能等等，这些目前的理论上限。至于之后则是理论突破后的才能考虑的事情。

Q：什么是领域专家？
A：比如说农业领域的模型，数据科学家是不知道树苗长大的不同阶段、时间等等具体需求的，需要有农业领域的专家来提出需求（类似甲方），然后让数据科学家翻译为机器语言，从而获取数据、实现模型。

Q：深度学习可以用数学规范表述吗？
A：可以进行数学规范表述，但是暂时无法用数学进行理论说明。

Q：符号学可以和机器学习融合起来吗？
A：符号学在深度学习有一些新的进展。模型足够复杂的情况下确实可以实现一定的推理工作

Q：数据科学家和AI专家的主要区别是什么？
A：没有太多区别。数据科学家更加关心领域专家的实际业务问题变成数据、机器模型，并且使用。AI专家主要是关心模型的精度和性能。你可以理解为资深的数据科学家就是AI专家。其实就是广度和深度，广度是让更多领域使用到机器模型，深度则是专研于某个领域使用模型，从而称为AI专家

Q：自然语言仅仅在感知层面似乎不太合适，语言的产出涉及到知识、规划等的
A：当然如此。自然语言处理其实进展一般，不如在深度学习在图片上的应用。2021年

Q：如何寻找自己领域的paper？
A：怎么样找到适合自己的论文，后面可能会跟大家分享一下。

Q：以无人驾驶的实例，误判率在下降，但是误判的影响很严重，有可能实现通过误判案例来进行改进吗？
A：一般来说，无人驾驶对于误判率是几乎最深入的，会通过大量的技术，不同模型的融合（投票）、改进雷达或传感器，等等融合来降低误差。竞赛会说明一些如何通过多个模型提升精度的做法。

安装实践

2026年1月30日 11:59

读书、实践、讲学（这是学习的不同境界）

数据操作——N维数组

2026年2月3日 6:38

三维数组——RGB图片（宽*高*通道）
四维数组——一个RGB图片批量 batch?(批量大小*宽*高*通道)
五维数组——一个视频批量（批量大小*时间*宽*高*通道）

1. 如何创建数组

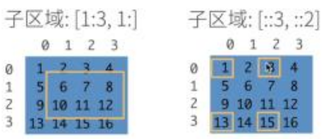
   a.

形状	例如3*4矩阵
每个元素的数据类型	例如32位浮点
每个元素的值	全0或随机数

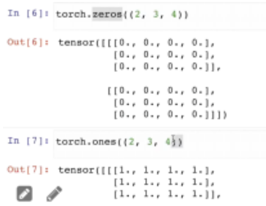
2. 访问元素

   a.

一个元素	[行，列]
一行	[1, :]
一列	[:, 1]
子区域	[1:3,1:]
子区域(跳跃访问)	[::3,::2]

   b. 

张量	表示一个数值组成的数组，可能有多个维度?
x.shape	访问 张量的形状
x.numel( )	张量中元素的总数
x.reshape()	改变张量的形状，不改变其中的值
x.zeros()	创建全0元素的张量
x.ones()	创建全1元素的张量
x.tensor(list[])	可以套用列表形成特定值的张量



标准算术运算符(+, -, * , /, **)均升级为按对应元素运算（相乘、相除）

张量合并：
torch.cat((X, Y) , dim=n)

dim = 0 按行合并	X与Y的第N行一同合并为新的第N行
dim = 1 按列合并	X与Y的第N列一同合并为新的第N列

逻辑运算符构建二元张量
X == Y
对张量中所有元素求和——产生一个一个元素的张量
X.sum()

广播机制
即使形状不同也可以执行按元素操作：
分析两个张量维度大小，然后该维度维度数最大值进行复制

元素访问：索引机制，与numpy规则一致，0为第一元素，第二位数减一取值，-1为倒数第一元素
元素修改：索引+赋值

关于内存的问题：

id(X)	python的id类似于指针

Y = Y + X 操作会导致为新结果分配内存（即Y不等于Y）
执行原地操作可以解决该问题：新建全0的张量Z，并使用修改方式获得操作的结果（并非单纯赋值）
Z = torch.zeros_like(Y)
Z[:] = X + Y
如果后续计算不需要重复使用Y，也可以直接修改或复合赋值运算符进行操作：

张量修改值	Y[:]= X+Y
复合赋值运算符	Y += X

torch.tensor	可以转化numpy为张量
x.item float(x) int(x)	可以将大小为1的张量转化为标量

数据预处理

2026年2月3日 7:20

使用pandas库

pd.read_csv()	读取文件

处理缺失数据

data.fillna(data.mean())	插值法	插入该列的均值
data.dropna(axis = 0)	删除法	直接删除含有缺失值的行（axis = 1为列）
pd.get_dummies(data, dummy_na = True)	独热编码	针对类别值或离散值，0或1
data.interpolate(method='linear')	插值法	用线性插值填充缺失值

张量转化

x = torch.tensor(data.values)	转化张量	数值类型的数据可以转化为张量格式

自动求导

2026年2月3日 15:49

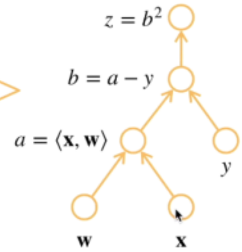
何为自动求导?

自动求导计算函数在指定值上的倒数，有别于符号求导和数值求导

计算图:

将代码一步步分解为操作子（最小操作），将计算展开为无环图

$$z = \left(\langle \mathbf{x}, \mathbf{w} \rangle - y\right)^2$$



显示构造 数学上常用

```
from mxnet import sym

a = sym.var()
b = sym.var()
c = 2 * a + b
# bind data into a and b later
```

隐式构造函数

```
from mxnet import autograd, nd

with autograd.record():
    a = nd.ones((2,1))
    b = nd.ones((2,1))
    c = 2 * a + b
```

自动求导的两种方法：（基于链式法则）

- 链式法则：$\dfrac{\partial y}{\partial x} = \dfrac{\partial y}{\partial u_n}\dfrac{\partial u_n}{\partial u_{n-1}}\cdots\dfrac{\partial u_2}{\partial u_1}\dfrac{\partial u_1}{\partial x}$
- 正向累积 $\dfrac{\partial y}{\partial x} = \dfrac{\partial y}{\partial u_n}\left(\dfrac{\partial u_n}{\partial u_{n-1}}\left(\cdots\left(\dfrac{\partial u_2}{\partial u_1}\dfrac{\partial u_1}{\partial x}\right)\right)\right)$
- 反向累积、又称反向传递

$$\frac{\partial y}{\partial x} = \left(\left(\left(\frac{\partial y}{\partial u_n}\frac{\partial u_n}{\partial u_{n-1}}\right)\cdots\right)\frac{\partial u_2}{\partial u_1}\right)\frac{\partial u_1}{\partial x}$$

复杂度:

计算复杂度

- **正向求导**：与**输入维度**成正比（O(n)）。
- **反向求导**：与**输出维度**成正比（O(m)）。

内存复杂度

- **正向求导**：仅需存储当前计算状态，**内存消耗低**（O(1)）。
- **反向求导**：需存储整个前向计算过程的中间结果（计算图），**内存消耗高**（O(L)，L为计算深度）。

基础原理:

向量链式法则

- 标量链式法则

$$y = f(u),\ u = g(x) \qquad \frac{\partial y}{\partial x} = \frac{\partial y}{\partial u}\frac{\partial u}{\partial x}$$

- 拓展到向量

$$\underset{(1,n)}{\frac{\partial y}{\partial \mathbf{x}}} = \underset{(1,)}{\frac{\partial y}{\partial u}}\underset{(1,n)}{\frac{\partial u}{\partial \mathbf{x}}} \qquad \underset{(1,n)}{\frac{\partial y}{\partial \mathbf{x}}} = \underset{(1,k)}{\frac{\partial y}{\partial \mathbf{u}}}\underset{(k,n)}{\frac{\partial \mathbf{u}}{\partial \mathbf{x}}} \qquad \underset{(m,n)}{\frac{\partial \mathbf{y}}{\partial \mathbf{x}}} = \underset{(m,k)}{\frac{\partial \mathbf{y}}{\partial \mathbf{u}}}\underset{(k,n)}{\frac{\partial \mathbf{u}}{\partial \mathbf{x}}}$$

例子 1

假设 $\mathbf{x}, \mathbf{w} \in \mathbb{R}^n, \quad y \in \mathbb{R}$

$z = \left(\langle \mathbf{x}, \mathbf{w} \rangle - y\right)^2$

计算 $\dfrac{\partial z}{\partial \mathbf{w}}$

分解
$a = \langle \mathbf{x}, \mathbf{w} \rangle$
$b = a - y$
$z = b^2$

$$\begin{aligned}\frac{\partial z}{\partial \mathbf{w}} &= \frac{\partial z}{\partial b}\frac{\partial b}{\partial a}\frac{\partial a}{\partial \mathbf{w}} \\ &= \frac{\partial b^2}{\partial b}\frac{\partial a - y}{\partial a}\frac{\partial \langle \mathbf{x}, \mathbf{w} \rangle}{\partial \mathbf{w}} \\ &= 2b \cdot 1 \cdot \mathbf{x}^T \\ &= 2\left(\langle \mathbf{x}, \mathbf{w} \rangle - y\right)\mathbf{x}^T\end{aligned}$$

例子 2

假设 $\mathbf{X} \in \mathbb{R}^{m \times n}, \quad \mathbf{w} \in \mathbb{R}^n, \quad \mathbf{y} \in \mathbb{R}^m$

$z = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$

计算 $\dfrac{\partial z}{\partial \mathbf{w}}$

分解
$\mathbf{a} = \mathbf{X}\mathbf{w}$
$\mathbf{b} = \mathbf{a} - \mathbf{y}$
$z = \|\mathbf{b}\|^2$

$$\begin{aligned}\frac{\partial z}{\partial \mathbf{w}} &= \frac{\partial z}{\partial \mathbf{b}}\frac{\partial \mathbf{b}}{\partial \mathbf{a}}\frac{\partial \mathbf{a}}{\partial \mathbf{w}} \\ &= \frac{\partial \|\mathbf{b}\|^2}{\partial \mathbf{b}}\frac{\partial \mathbf{a} - \mathbf{y}}{\partial \mathbf{a}}\frac{\partial \mathbf{X}\mathbf{w}}{\partial \mathbf{w}} \\ &= 2\mathbf{b}^T \times \mathbf{I} \times \mathbf{X} \\ &= 2\left(\mathbf{X}\mathbf{w} - \mathbf{y}\right)^T\mathbf{X}\end{aligned}$$

未命名

2026年2月21日　　20:03