



西安电子科技大学

# 人工智能概述

Overview of Artificial Intelligence

主讲教师：谢涌

邮 箱：yxie@xidian.edu.cn

2024.9

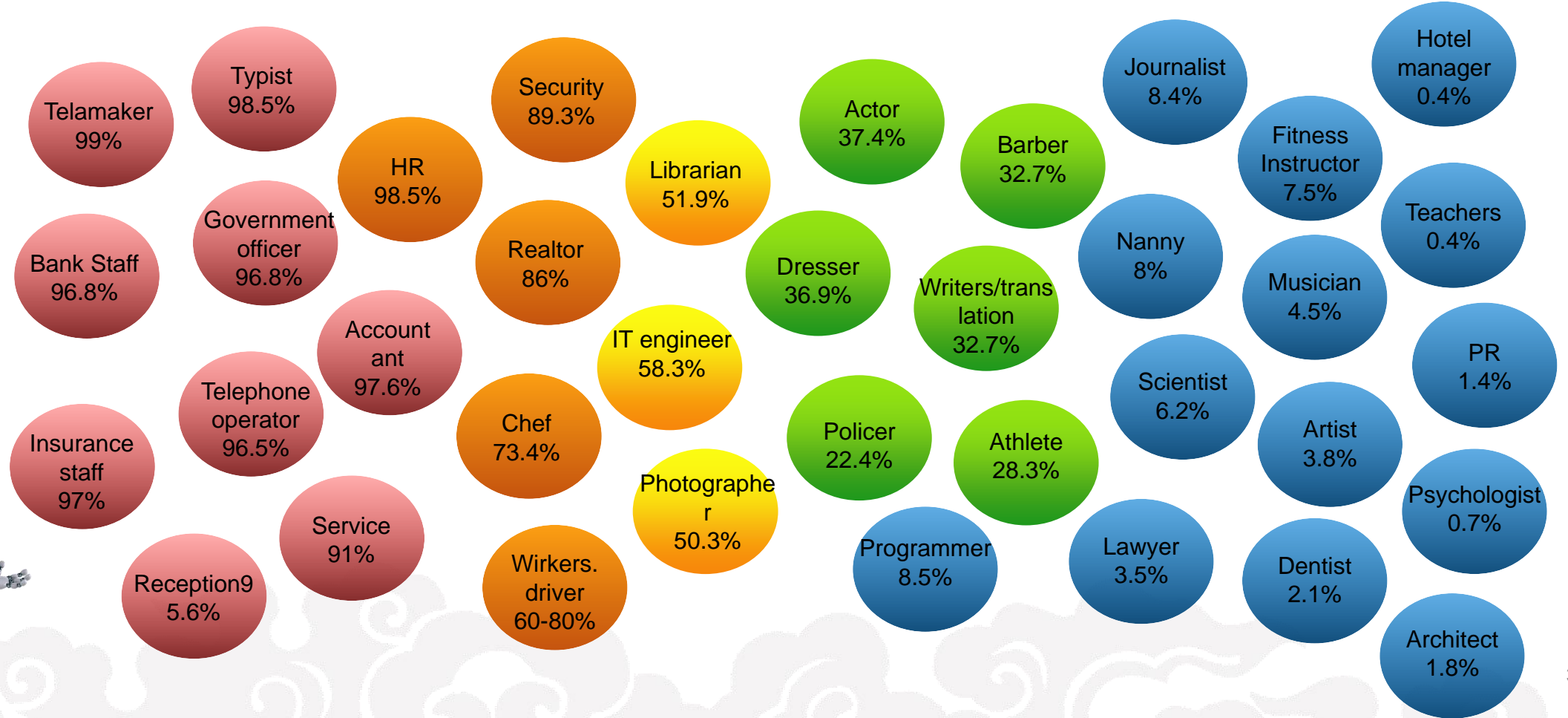


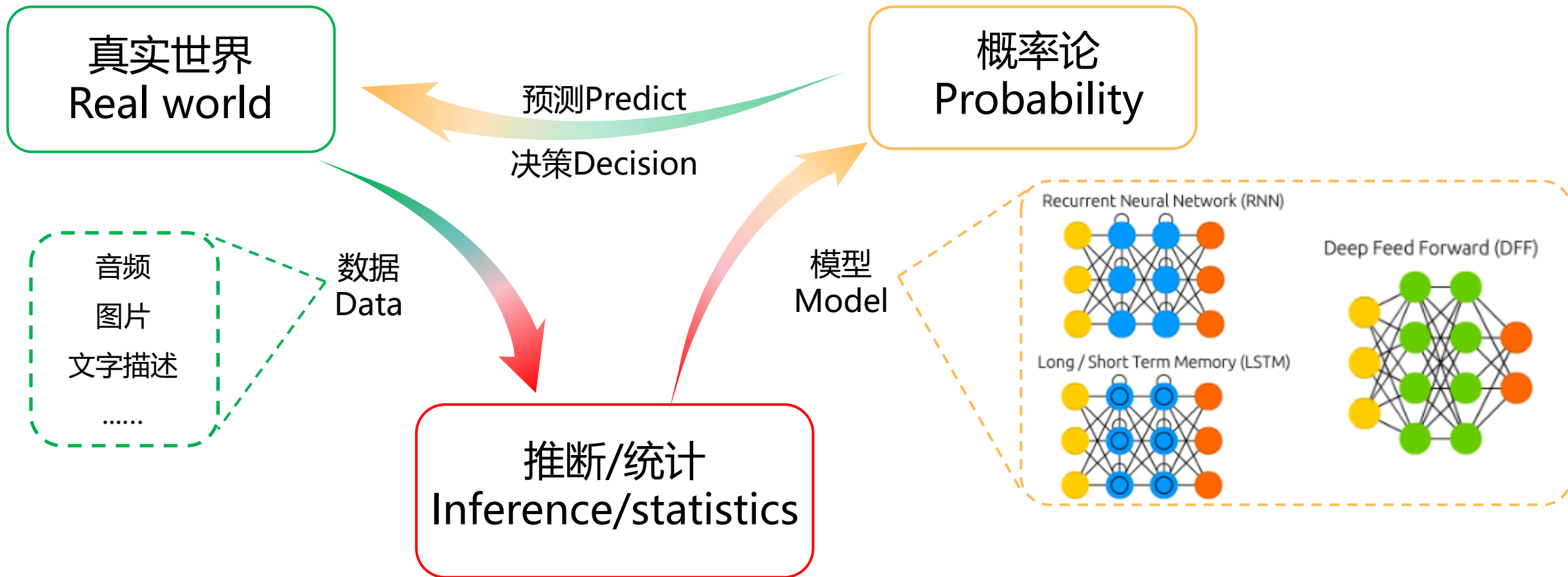
## 第三讲

# 人工智能实践基础

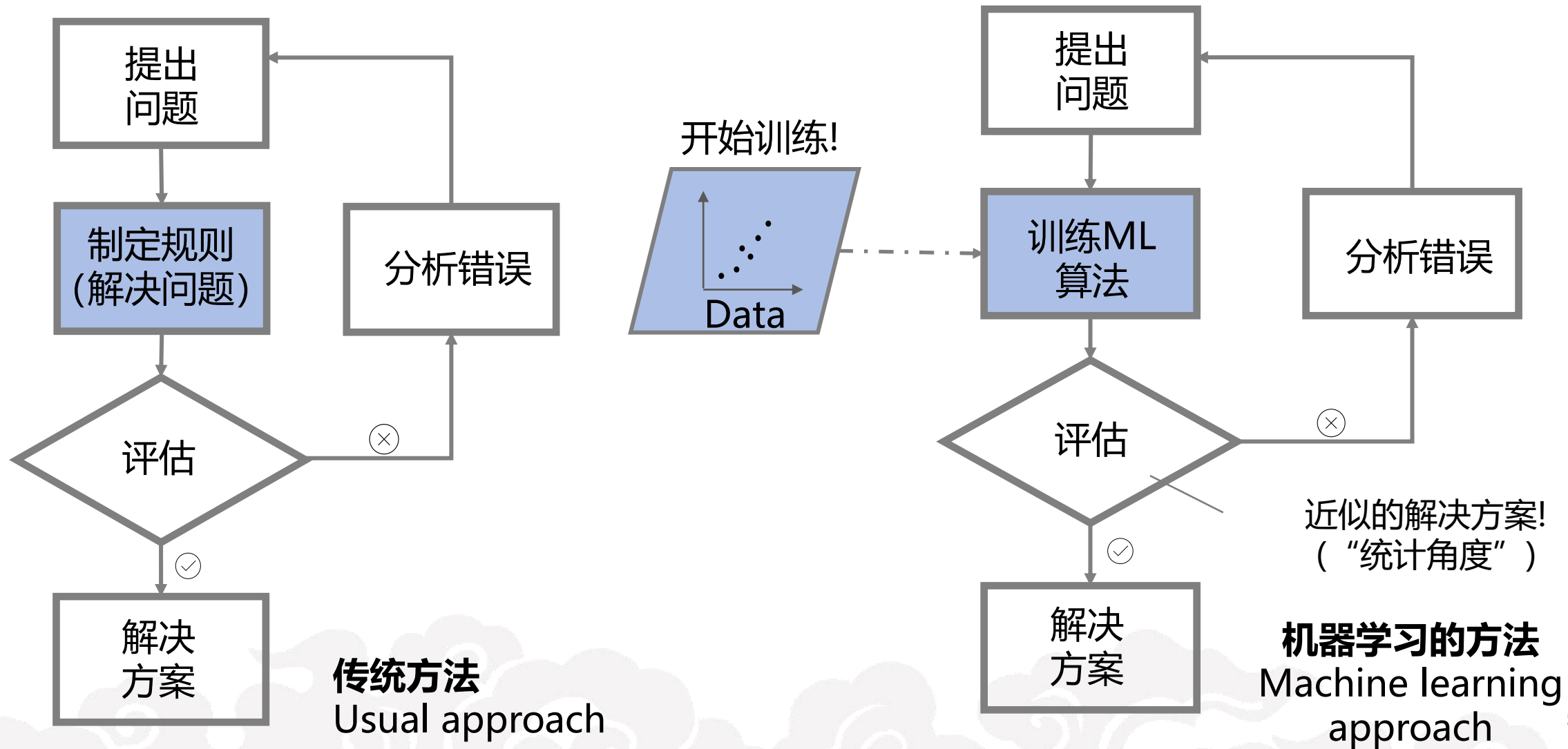


## 未来淘汰的不仅是工作，更是技能。





## 传统方法与“机器学习”方法





# 主 讲 内 容

## 3.1

### 人工智能算法开发流 程

## 3.2

### 人工智能的开发语言

## 3.3

### 数据标注过程中可使 用的工具

## 3.4

### 深度学习的开源框架

## 3.5

### 如何搜索人工智能 算法的开源资源

## 3.6

### 本讲小结





# 主 讲 内 容

## 3.1

### 人工智能算法开发流 程

## 3.2

### 人工智能的开发语言

## 3.3

### 数据标注过程中可使 用的工具

## 3.4

### 深度学习的开源框架

## 3.5

### 如何搜索人工智能 算法的开源资源

## 3.6

### 本讲小结



# 3.1 人工智能算法开发流程

人工智能算法开发流程通常包括以下几个关键步骤：

## 1. 问题定义

**明确目标：**首先，明确要解决的问题，并定义算法的目标。例如，是要进行分类、回归、聚类，还是其他任务？

**业务需求分析：**与业务领域专家讨论，了解问题的业务背景和要求。

## 2. 数据收集

**数据来源确定：**确定数据的来源，可能包括数据库、API、传感器数据、文本数据等。

**数据采集：**通过脚本、API调用或手动方式收集所需数据。

## 3. 数据预处理

**数据清洗：**处理缺失值、异常值、重复数据等问题，确保数据的质量。

**数据转换：**将原始数据转换为算法可处理的格式，包括标准化、归一化、等操作。

**特征选择：**从数据中选择与问题最相关的特征，以提高模型的性能和效率。

## 4. 模型选择

**选择算法：**根据问题的性质选择合适的算，如线性回归、决策树、随机森林、神经网络等。

**模型架构设计：**如果使用的是复杂模型，如深度学习模型，需要设计网络架构。



# 3.1 人工智能算法开发流程

人工智能算法开发流程通常包括以下几个关键步骤：

## 5. 模型训练

**划分数据集：**通常将数据集划分为训练集、验证集和测试集。

**训练模型：**使用训练数据来优化模型的参数。

**调整超参数：**通过网格搜索、随机搜索或贝叶斯优化等方法，调整模型的超参数，以提高性能。

## 6. 模型评估

**性能评估：**使用验证集来评估模型的性能，常用的指标包括精度、召回率、F1分数、ROC-AUC等。

**交叉验证：**通过交叉验证技术，进一步确认模型的稳健性。

## 7. 模型优化

**过拟合与欠拟合：**通过调整模型复杂度、增加正则化或获取更多数据来避免过拟合或欠拟合。

**进一步调参：**根据评估结果进一步调整超参数或模型结构。

## 8. 模型部署

**集成到系统中：**将经过验证的模型集成到实际业务系统中，可能涉及API开发、嵌入式部署等。





**实时推理：**在生产环境中进行实时或批量数据推理。



# 3.1 人工智能算法开发流程

## 2. 数据集收集

MNIST数据集为开源的。MNIST数据集来自美国国家标准与技术研究所, National Institute of Standards and Technology (NIST)。训练集 (training\_set) 由来自250个不同人手写的数字构成, 其中50%是高中学生, 50%来自人口普查局 (the Census Bureau) 的工作人员。测试集 (test\_set) 也是同样比例的手写数字数据, 但保证了测试集和训练集的作者集不相交。MNIST数据集一共有7万张图片, 其中6万张是训练集, 1万张是测试集。

测试数据集	 t10k-images-idx3-ubyte	2024/8/19 10:49	文件	7,657 KB
测试数据集标签	 t10k-labels-idx1-ubyte	2024/8/19 10:50	文件	10 KB
训练数据集	 train-images-idx3-ubyte	2024/8/19 10:49	文件	45,938 KB
训练数据集标签	 train-labels-idx1-ubyte	2024/8/19 10:49	文件	59 KB

数据集下载地址: <https://yann.lecun.com/exdb/mnist/>

```
train-images-idx3-ubyte.gz: training set images (9912422 bytes)
train-labels-idx1-ubyte.gz: training set labels (28881 bytes)
t10k-images-idx3-ubyte.gz: test set images (1648877 bytes)
t10k-labels-idx1-ubyte.gz: test set labels (4542 bytes)
```

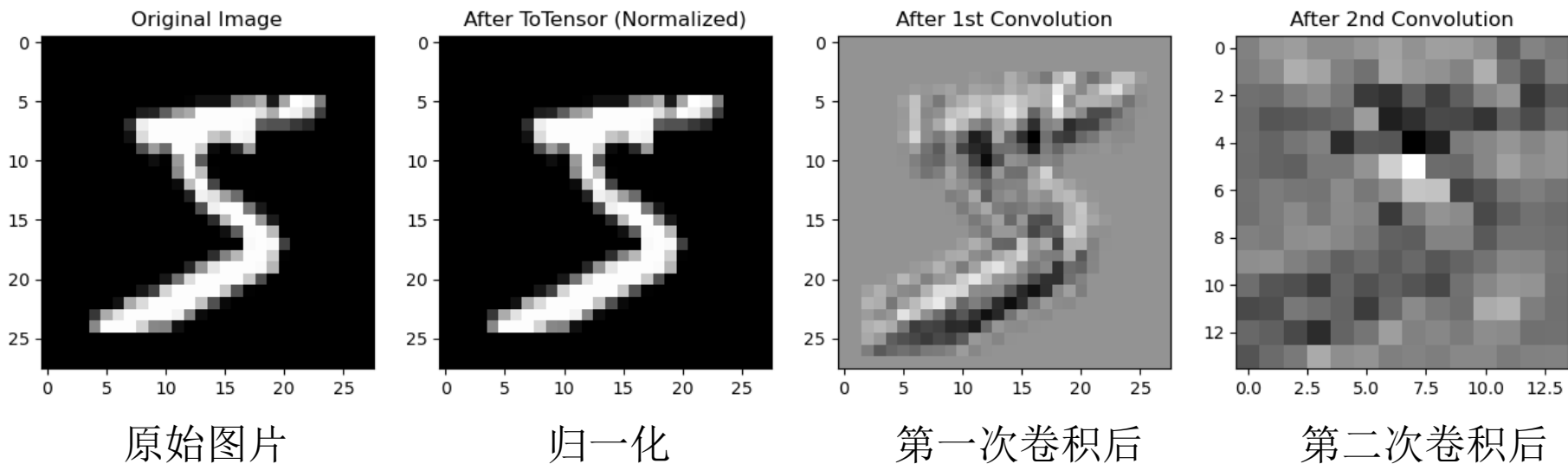
# 3.1 人工智能算法开发流程

## 3. 数据预处理

数据预处理是提高模型性能的重要步骤。在本案例中，数据预处理的主要任务包括：

**标准化：**将像素值缩放到 $[0, 1]$ 的范围。由于MNIST数据集中的像素值在0到255之间，通过除以255，可以将数据标准化，确保每个特征在相似的尺度上，这有助于加快模型收敛。

**数据增强：**为了提高模型的泛化能力，可以对训练数据进行随机旋转、平移、缩放等操作，以扩展数据集，模拟真实场景中可能遇到的各种手写体变化。



# 3.1 人工智能算法开发流程

## 4-5. 模型选择与训练

在解决图像分类问题时，卷积神经网络（CNN）是最常用的模型之一。CNN擅长处理图像数据，因为它能够自动学习到图像中的空间特征，如边缘、角点等。在本案例中，我们选择一个典型的CNN架构。

训练过程中，我们通过反向传播算法来调整模型的权重，使其能够准确分类手写数字。

主要步骤包括：

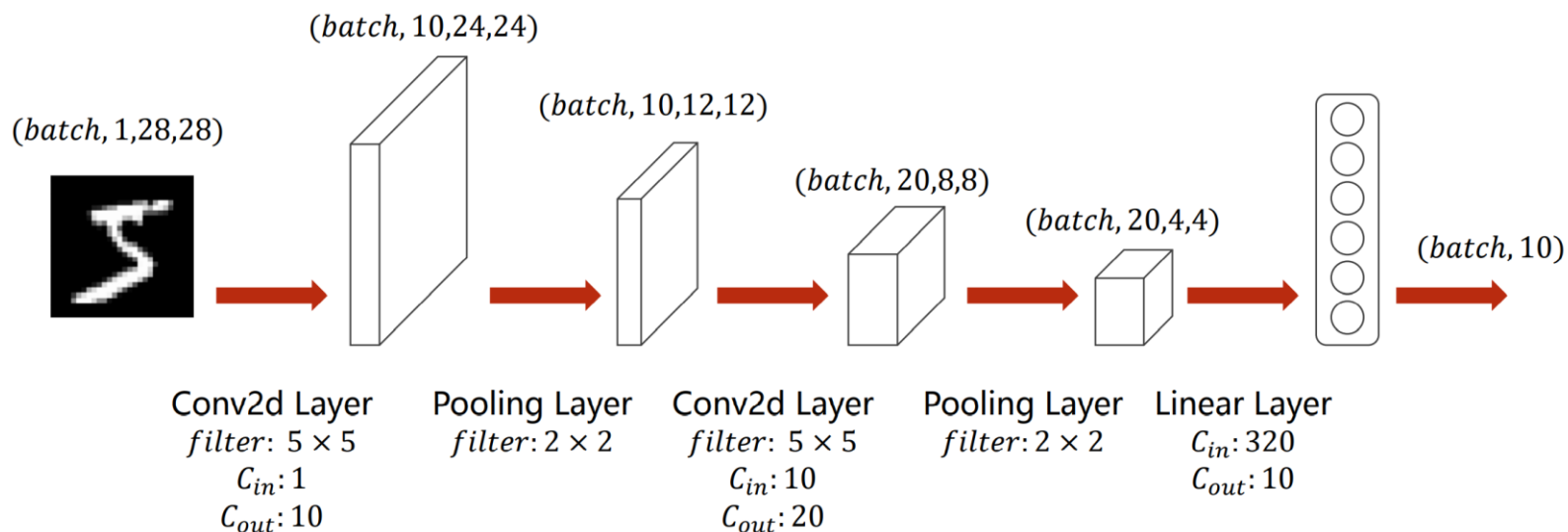
**定义损失函数---选择优化器（Adam优化器）---设置超参数（学习率、批大小、训练轮数等）---训练**

**卷积层（Conv）**：用于提取图像的局部特征，通常包括多个卷积层叠加使用。

**池化层（Pooling）**：通常使用最大池化（Max Pooling），用于下采样，减少特征图的维度，同时保持重要特征。

**全连接层（Linear）**：将卷积层提取的特征映射到最终的分类标签上。

具体的模型架构可以包括多个卷积层和池化层。



## 6. 模型评估

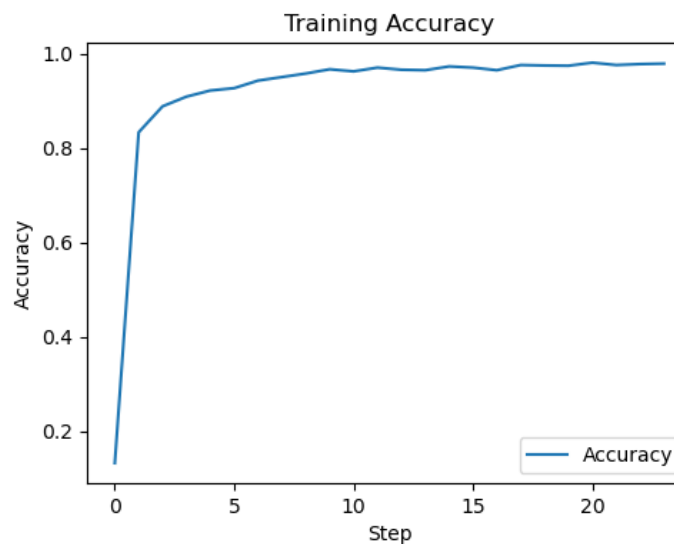
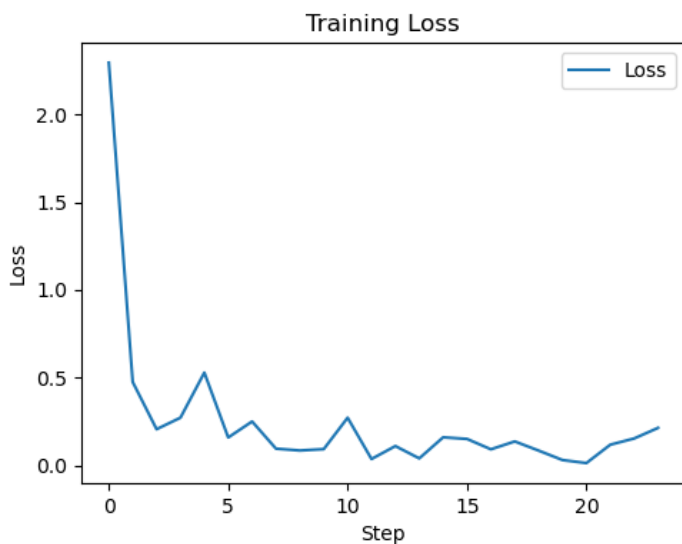
模型评估用于衡量模型在未见过的数据上的表现。常用的评估指标包括：

**准确率**：预测正确的比例，是最直观的衡量标准。

**混淆矩阵**：显示每个类别的真实标签与预测结果的对应关系，可以更细致地分析模型的分类效果。

**精确率、召回率和F1分数**：在不平衡数据集上，单纯的准确率可能不足以评估模型性能，此时可以使用这些指标。

在本案例中，我们使用测试集对模型进行评估，通过准确率和混淆矩阵来分析模型的表现。



对训练完模型的数据制作成曲线，方便后续对模型优化

### 7. 模型优化

为了提升模型性能，我们可以从以下几个方面进行优化：

**调整超参数：**通过网格搜索或随机搜索找到最优的学习率、批大小、网络深度等超参数组合。

**正则化：**如使用Dropout或L2正则化，防止模型过拟合。

**更复杂的模型架构：**如引入残差网络（ResNet）等先进的CNN架构来提高模型性能。

### 8. 模型部署

当模型的性能达到满意的水平后，我们可以将其部署到实际应用中。部署的步骤包括：

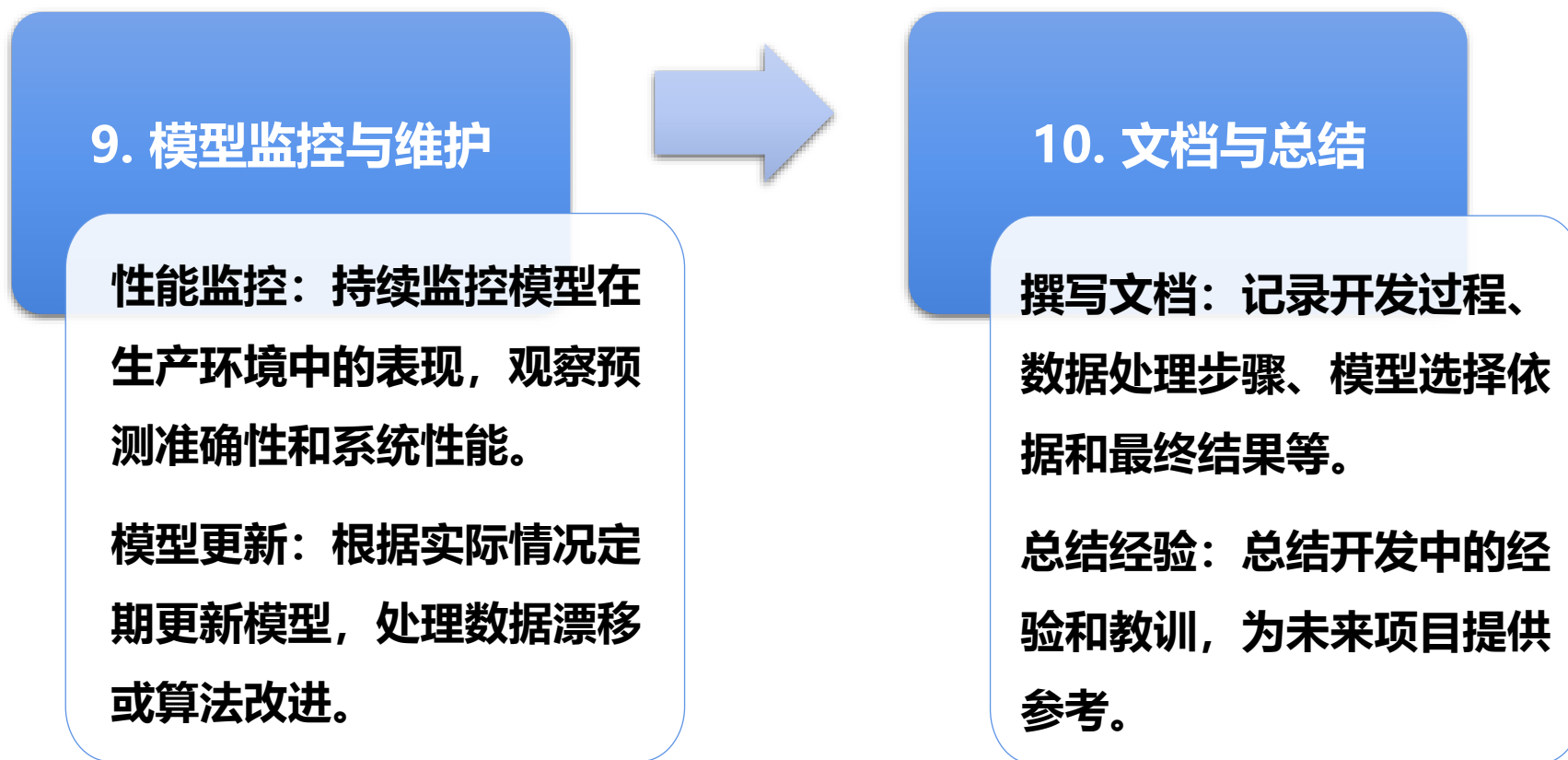
**模型保存：**将训练好的模型保存为标准格式（如HDF5或ONNX），便于加载和使用。

**接口开发：**开发API接口，使得外部应用可以方便地调用模型进行预测。

**推广应用：**教育平台或应用程序可以使用手写数字识别技术来帮助学生学习数学或练习书写数字，实现作业自动批改等；自动化处理手写文档或表单，减少人工输入和数据录入的工作量；金融机构可以使用手写数字识别技术来自动化支票处理和识别工作。

# 3.1 人工智能算法开发流程

人工智能算法开发流程通常包括以下几个关键步骤：

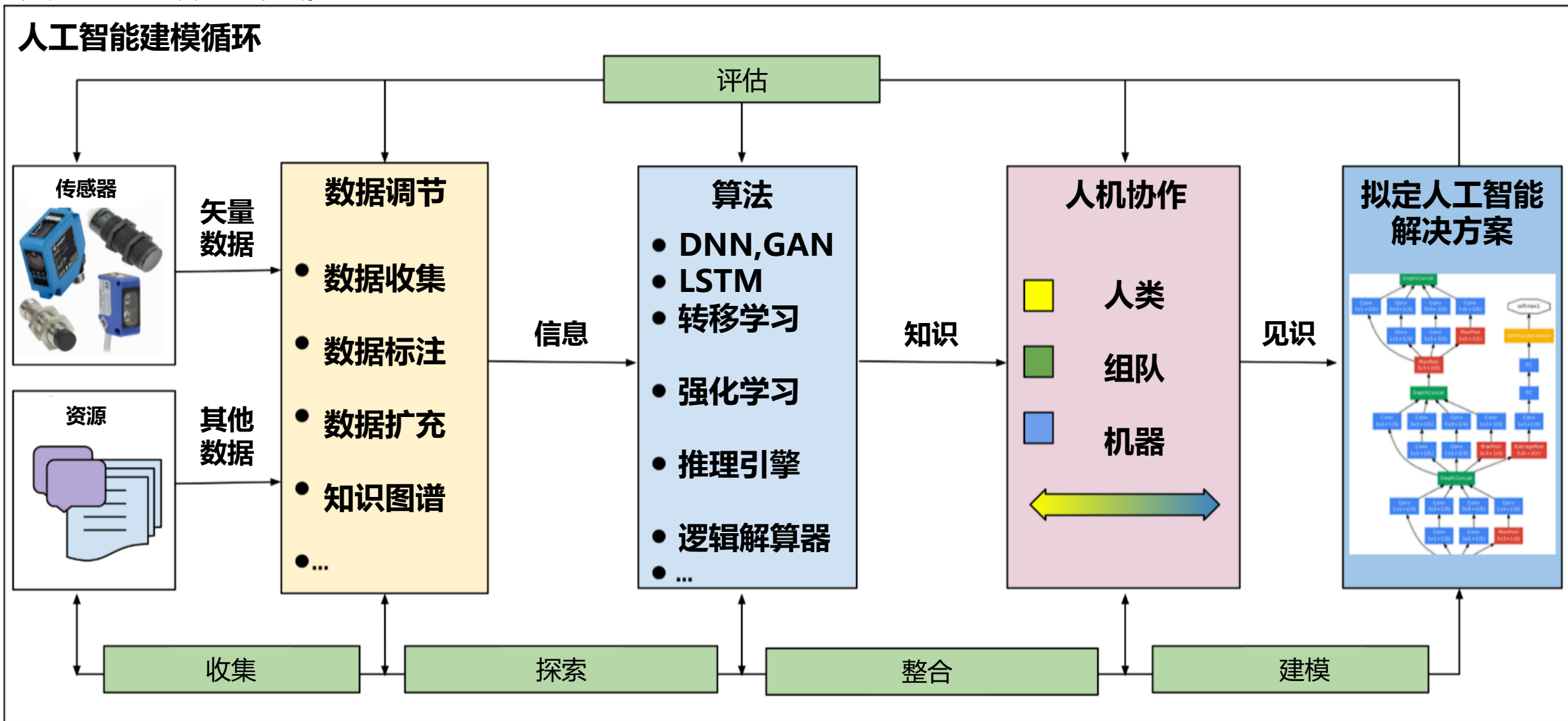


这个流程可以根据项目的需求和复杂度进行调整，有时步骤之间也可能会有迭代和循环。



# 3.1 人工智能算法开发流程

## 开发人工智能模型的步骤





# 主 讲 内 容

3.1

人工智能算法开发流  
程

3.2

人工智能的开发语言

3.3

数据标注过程中可使  
用的工具

3.4

深度学习的开源框架

3.5

如何搜索人工智能  
算法的开源资源

3.6

本讲小结



### 人工智能开发语言的选择标准

#### 性能要求

- 高效率的计算能力
- 低延迟的数据处理
- 优化的算法执行速度

#### 生态系统与库支持

- 完善的工具链
- 丰富的第三方库
- 高度集成的开发环境

#### 社区与文档支持

- 活跃的开发社区
- 详细的官方文档
- 丰富的学习资源

#### 开发效率

- 快速的开发周期
- 易于维护的代码
- 高度模块化的设计

### 常见的人工智能开发语言



Python

简单易学的语法  
强大的库支持  
广泛的应用领域



R

专注于统计分析  
强大的可视化库  
适用于数据科学领域



Java

良好的跨平台性能  
强大的企业级应用支持  
丰富的开发工具



C++

高性能计算  
系统级编程  
复杂算法实现

# Python在人工智能领域的优势

01

### 简洁的语法

易于理解和学习  
提高开发效率  
降低出错率

02

### 丰富的库和框架

TensorFlow、PyTorch等深度学习框架  
NumPy、Pandas等数据处理库  
Matplotlib、Seaborn等可视化工具



03

### 强大的社区支持

丰富的教程和案例  
实时的技术支持  
持续的库更新

04

### 跨平台性

支持Windows、Linux、macOS等操作系统  
适用于Web、桌面、移动等多种平台  
无需修改代码即可迁移

### 开发工具选择



#### 常见Python开发工具

PyCharm  
Visual Studio Code  
Jupyter Notebook



#### 选择合适的开发工具

根据项目需求选择  
考虑个人习惯  
注意工具的兼容性



#### 安装与配置IDE

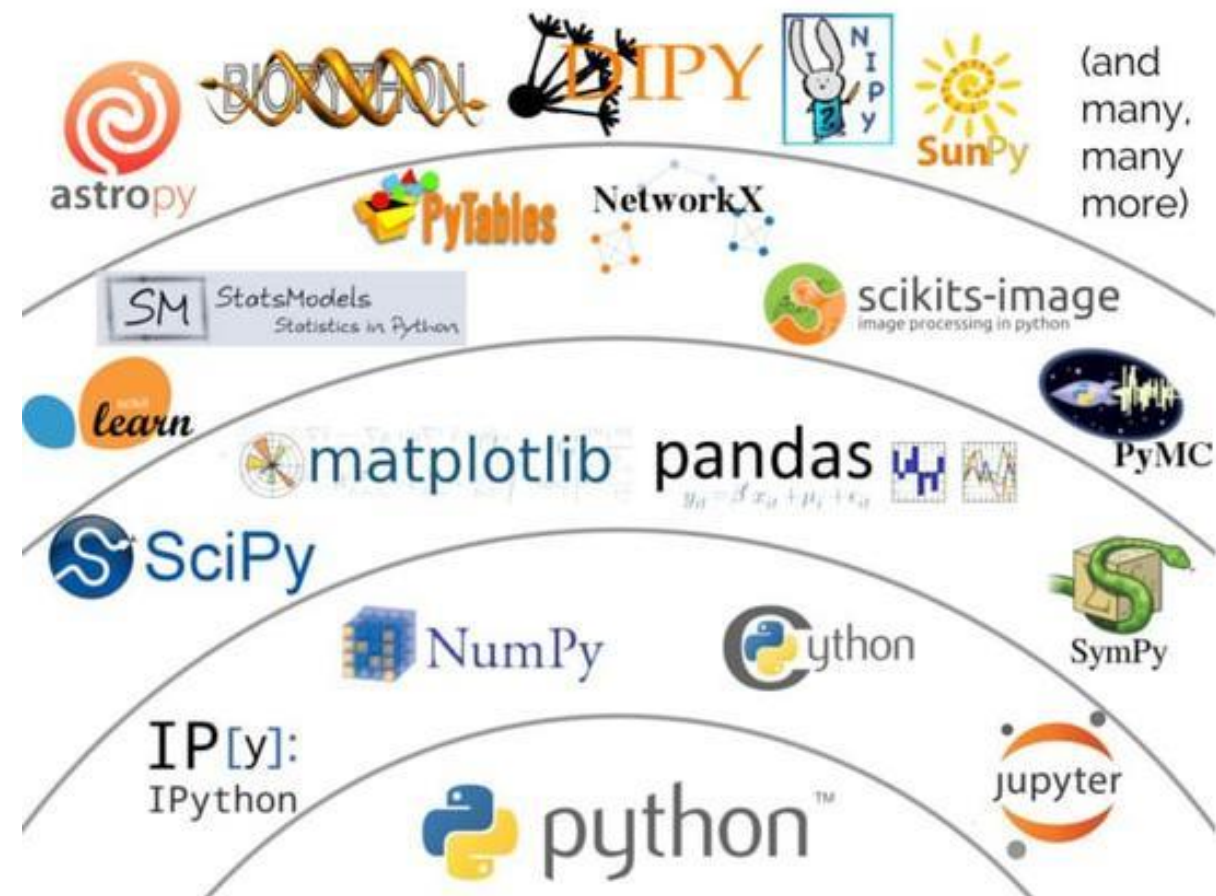
下载IDE安装包  
运行安装程序  
配置项目和工作区



#### 配置代码编辑器

安装代码编辑器  
设置代码字体和主题  
安装插件和扩展

## 常用Python人工智能库



### NumPy

用于进行高性能的数学计算  
提供多维数组对象和一系列数学函数库  
是许多Python科学计算库的基础



### Pandas

提供数据结构和数据分析工具  
用于数据清洗和准备  
支持数据框 (DataFrame) 对象, 易于处理表格数据



### Matplotlib

用于绘制2D图形和图表  
提供丰富的绘图接口  
支持多种图表类型和自定义图形样式



### Scikit-learn

提供简单有效的数据挖掘和数据分析工具  
用于数据预处理、模型选择、模型训练和模型评估  
包含许多监督和非监督学习算法

### 深度学习框架

#### TensorFlow

由Google开发的开源深度学习框架  
支持广泛的机器学习和深度学习模型  
具有强大的分布式计算能力

#### PyTorch

由Facebook开发的开源深度学习库  
提供灵活的动态计算图易于调试和实验

#### 飞桨 Paddle Paddle

由百度自主开发的开源深度学习框架  
同时支持动态图和静态图  
能方便的调试模型和部署非常适合业务应用的落地实现



#### Keras

提供快速实验能力的深度学习库  
可以作为TensorFlow、CNTK或Theano的顶层接口  
支持快速构建和迭代深度学习模型

#### MXNet

支持灵活的编程模式和高效的计算  
由亚马逊支持，适合云计算和边缘设备  
提供多种语言的接口，包括Python、R和Scala

#### 昇思 MindSpore

由华为推出的新一代全场景AI框架  
提升易用性并降低AI开发者的开发门槛  
高效执行包括计算效率、数据预处理效率和分布式训练效率



### Python IDE安装

进入Pycharm官网地址下载后安装：  
<https://www.jetbrains.com/pycharm/>

进入官网地址：  
<https://www.python.org/downloads/>下载python解释器

### CONDA 环境安装

conda : data science package & environment manager

创建环境:

```
conda create --name python3 python=3
```

切换环境:

```
windows :activate python3
```

```
linux/macOS : source activate python3
```

官方地址: <https://www.anaconda.com/download/>

## Python第三方库安装

在使用Python的过程中，势必会涉及到许多第三方库，例如上述案例中的matplotlib，安装过程：

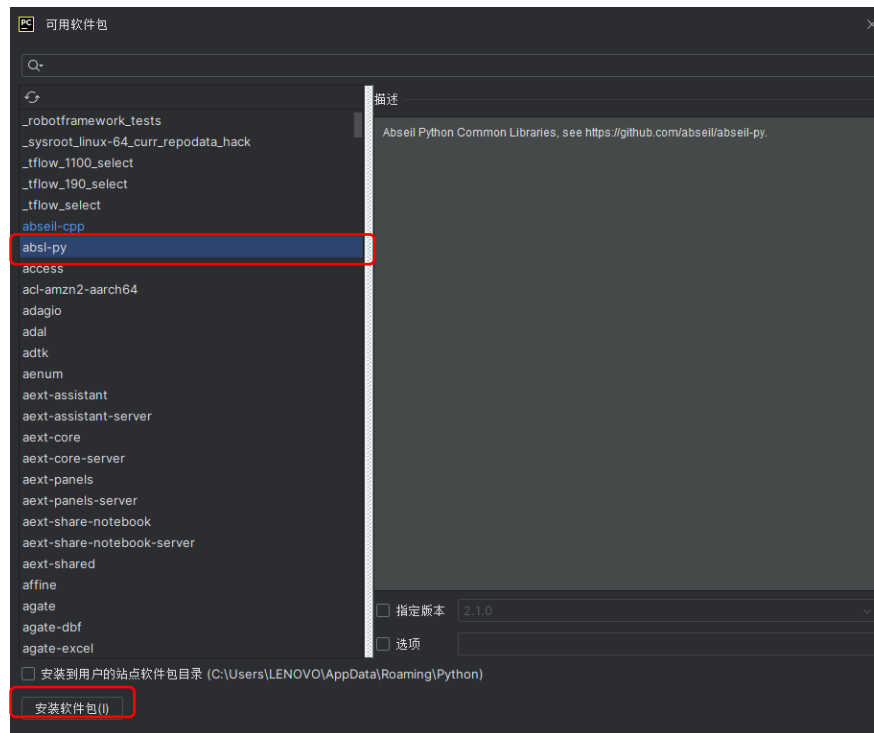
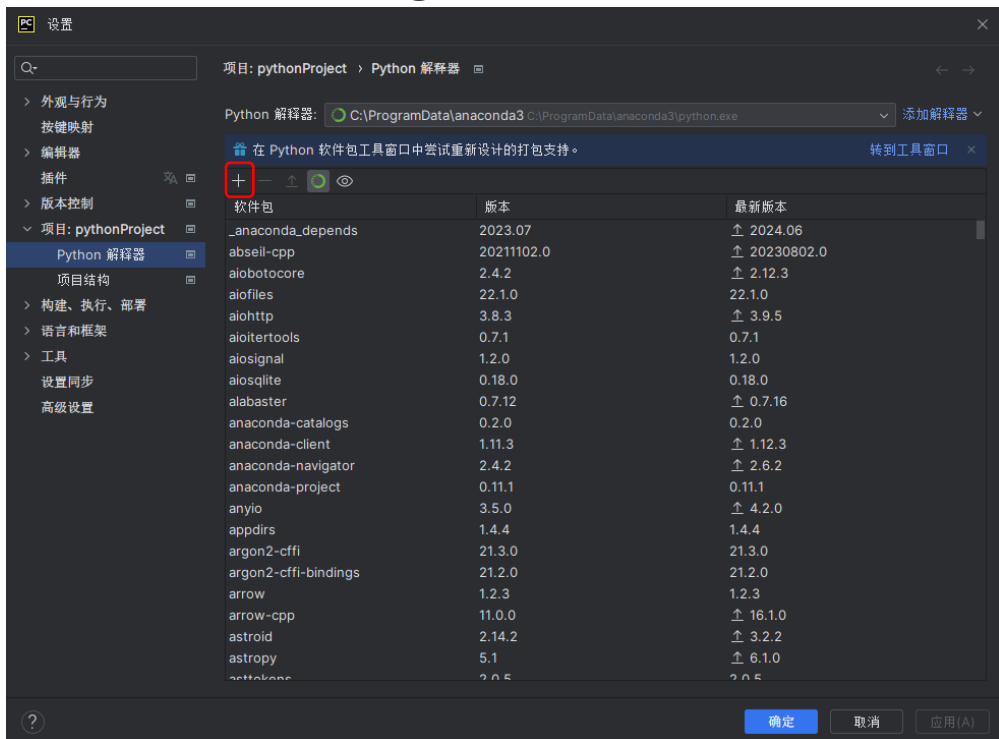
1、先检查一下你的pip版本、如果pip版本过低(现在最高是20.0.2)、需更新pip

2、win+R在“运行”中输入cmd调出终端，在终端中输入 **pip install + 需要安装的库名 (如图)**

```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.19043.928]
(c) Microsoft Corporation. 保留所有权利。
C:\Users\LENOVO>pip install torch
```

第三方库也可在Pycharm编辑器中安装：

**File —— Setting —— Project —— Project Interpreter**



**选中目标安装包  
后安装即可**

## Python的常用语句

### 导入模块库 (一般在最开始编译)

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

### 输入字面量 (打印)

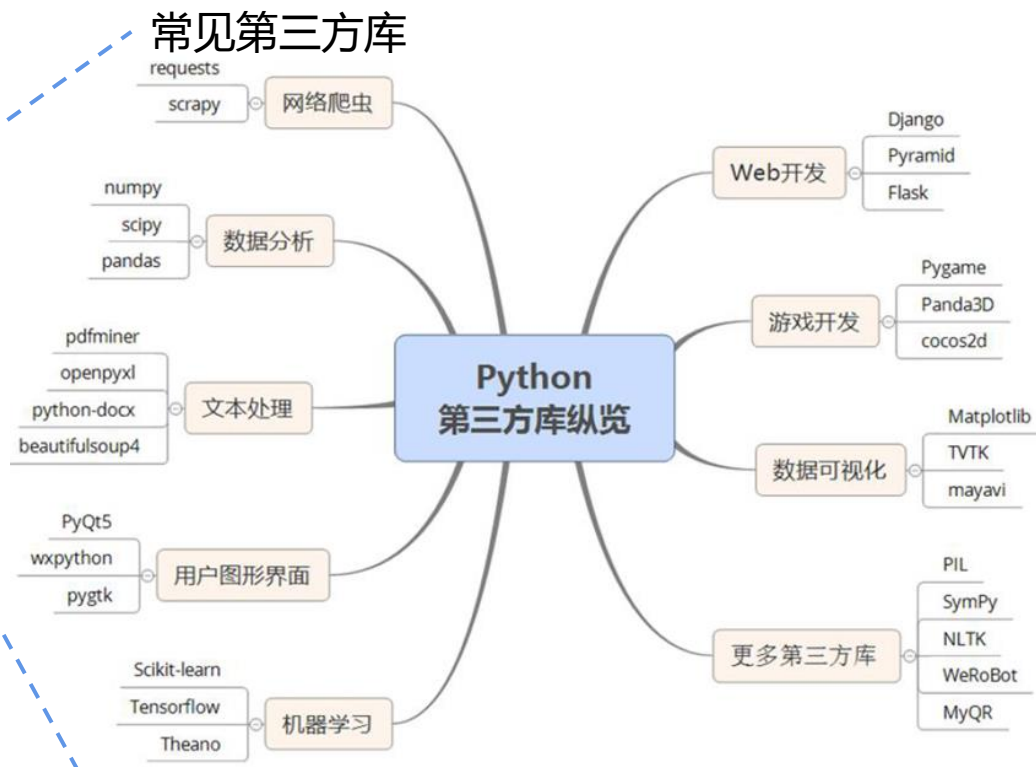
```
1 > print(666) # 整数
2 > print(13.14) # 浮点数
3 > print("HeyStar") # 字符串
```

### 定义变量并输出

```
1 > ##定义一个变量, 用来记录钱包余额
2 > money = 50
3 > ##输出变量的内容*
4 > print("钱包还有: ", money) ##
```

常见的转换语句 (数据类型之间, 在特定的场景下, 可以相互转换, 如字符串转数字等。)

语句(函数)	说明
int(x)	将x转换为一个整数
float(x)	将x转换为一个浮点数
str(x)	将对象x转换为字符串



算数符	
+	加
-	减
*	乘
/	除
//	取整除
%	取余
**	指数

复合赋值算数符	
+=	加法赋值运算符
-=	减法赋值运算符
*=	乘法赋值运算符
/=	除法赋值运算符
//=	取整除赋值运算符
%=	取模赋值运算符
**=	幂赋值运算符

# 3.2 人工智能的开发语言

## 运算符使用

```

1 > *#加法*
2 > print("1 + 1 = ", 1 + 1)
3 > *#减法*
4 > print("2 - 1 = ", 2 - 1)

5 > *#复合运算符*
6 > num += 1 *#相当于num = num + 1*
7 > print("num + 1 = ", num)
8 > num -= 1 *#相当于num = num - 1*
9 > print("num - 1 = ", num)

```

## 注释:

单行注释:以#开头, # (空格) 右边的所有文字当作说明, 一般用来对一行或一小部分的代码进行解释。

多行注释:以一对三个双引号引起来(""""注释内容""")来解释说明, 一般用来对Python文件、类或方法进行解释。

```

# 单行注释

"""
多
行
注
释
"""

```

## 字符串拼接

如果由两个字符串(文本)字面量, 可以将其拼接成一个字符串, 通过+号即可完成。

输入:

```

1 name = "HeyStar"
2 title = "Python | 基础语法"
3 print("这里是 "+name+", 这篇文章为: "+title)

```

输出:

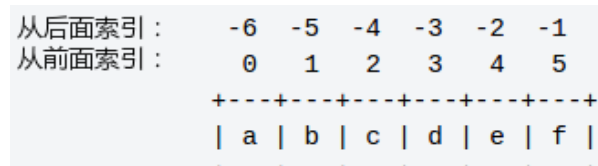
```

1 这里是 HeyStar, 这篇文章为: Python | 基础语法

```

## 字符串分隔

要实现从字符串中获取一段子字符串, 可使用 [头下标:尾下标] 来截取相应的字符串。



```

1 str = 'Hello World!'
2
3 print(str) # 输出完整字符串
4 print(str[0]) # 输出字符串中的第一个字符
5 print(str[2:5]) # 输出字符串中第三个至第六个之间的字符串
6 print(str[2:]) # 输出从第三个字符开始的字符串
7 print(str * 2) # 输出字符串两次
8 print(str + "TEST") # 输出连接的字符串

```

输出:

```

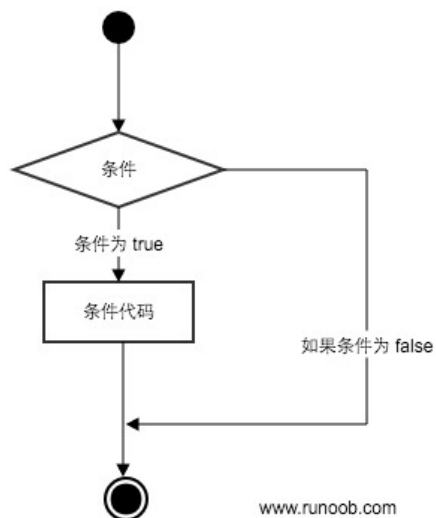
Hello World!
H
llo
llo World!
Hello World!Hello World!
Hello World!TEST

```

# 3.2 人工智能的开发语言

## 条件语句

通过一条或多条语句的执行结果 (True或者False) 来决定执行的代码块。



<pre>1 a = 1 2 while a &lt; 7 : 3     if(a % 2 == 0): 4         print(a, "is even") 5     else: 6         print(a, "is odd") 7     a += 1</pre>	<p>code</p>	<p>output</p>
<p>variables</p>		

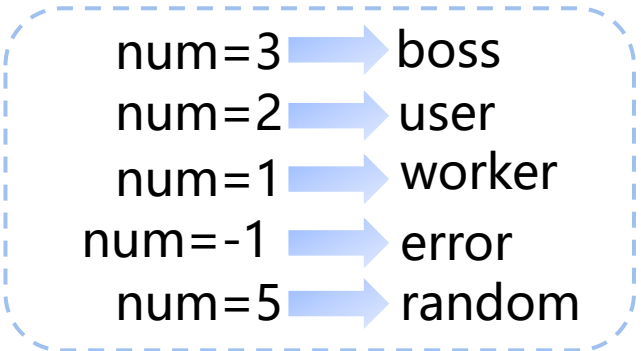
www.penjee.com

当判断条件为多个值时，可以使用以下形式：

```
10 num = 3
11 if num == 3: # 判断num的值
12     print("boss")
13 elif num == 2:
14     print("user")
15 elif num == 1:
16     print("worker")
17 elif num < 0: # 值小于零时输出
18     print("error")
19 else:
20     print("roadman") # 条件均不成立时输出
```

```
if 判断条件1:
    执行语句1.....
elif 判断条件2:
    执行语句2.....
elif 判断条件3:
    执行语句3.....
else:
    执行语句4.....
```

输出:



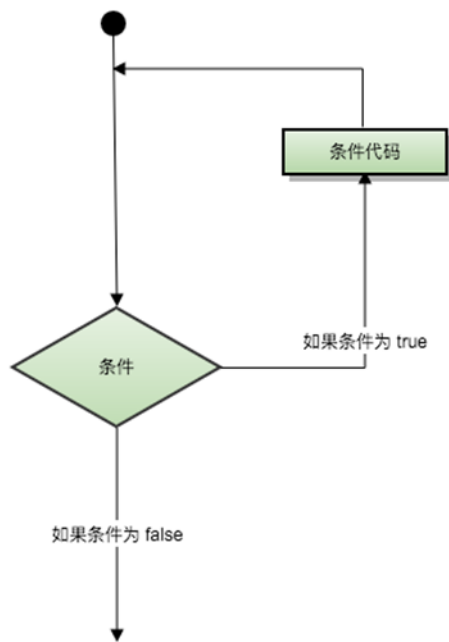
其中"判断条件"成立时 (非零)，则执行后面的语句，而执行内容可以多行，以缩进来区分表示同一范围。  
else 为可选语句，当需要在条件不成立时执行内容则可以执行相关语句。

# 3.2 人工智能的开发语言



## 循环语句

循环语句可执行一个语句或语句组多次



循环类型	描述
while 循环	在给定的判断条件为 true 时执行循环体，否则退出循环体。
for 循环	重复执行语句
嵌套循环	你可以在while循环体中嵌套for循环

控制语句	描述
break 语句	在语句块执行过程中终止循环，并且跳出整个循环
continue 语句	在语句块执行过程中终止当前循环，跳出该次循环，执行下一次循环。
pass 语句	pass是空语句，是为了保持程序结构的完整性。

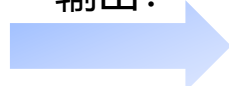
## for循环

可以遍历任何序列的项目，  
如一个列表或者一个字符串

```
for letter in 'Python': # 第一个实例
    print("当前字母: %s" % letter)

fruits = ['banana', 'apple', 'mango']
for fruit in fruits: # 第二个实例
    print('当前水果: %s' % fruit)

print("Good bye!")
```

输出: 

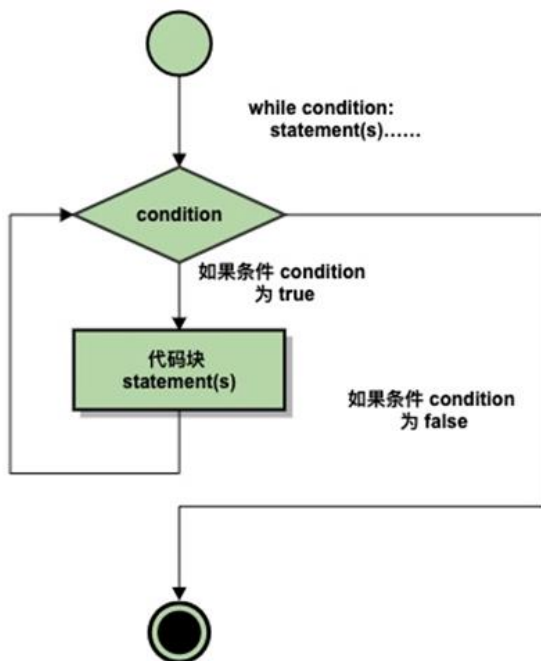
```
当前字母: P
当前字母: y
当前字母: t
当前字母: h
当前字母: o
当前字母: n
当前水果: banana
当前水果: apple
当前水果: mango
Good bye!
```

## 3.2 人工智能的开发语言

### While循环

while 语句用于循环执行程序，即在某条件下，循环执行某段程序，以处理需要重复处理的相同任务。

```
while 判断条件(condition):  
    执行语句(statements).....
```



code

```
1 a = 1  
2 while a < 10:  
3     print (a)  
4     a += 2
```

output

variables

复杂案例：

```
1 numbers = [12, 37, 5, 42, 8, 3]  
2 even = []  
3 odd = []  
4 while len(numbers) > 0 :  
5     number = numbers.pop()  
6     if(number % 2 == 0):  
7         even.append(number)  
8     else:  
9         odd.append(number)
```

执行语句可以是单个语句或语句块。判断条件可以是任何表达式，任何非零、或非空 (null) 的值均为true。

当判断条件假 false 时，循环结束。

如果条件判断语句永远为 true，循环将会无限的执行下去

### 代码调试技巧



#### 使用断点

在代码中设置断点以暂停执行  
观察代码执行到断点时的状态  
使用IDE或调试工具管理断点



#### 单步执行与观察变量

逐行执行代码以观察程序流程  
观察和监视变量值的变化  
使用调试工具的变量窗口查看数据



#### 调试器的高级功能

使用条件断点以在特定条件下暂停  
观察调用栈以理解函数调用顺序  
利用日志点记录运行时的关键信息



#### 调试第三方库代码

调试第三方库中的内部逻辑  
使用源码查看或反汇编来分析库代码  
利用调试器的远程调试功能





# 主 讲 内 容

## 3.1

### 人工智能算法开发流 程

## 3.2

### 人工智能的开发语言

## 3.3

### 数据标注过程中可使 用的工具

## 3.4

### 深度学习的开源框架

## 3.5

### 如何搜索人工智能 算法的开源资源

## 3.6

### 本讲小结



### 数据集标注概述

数据标注是机器学习领域的基石，解决了人工智能的一个基本挑战：**将原始数据转换为机器可以理解的格式。**

从本质上讲，数据注释解决了非结构化信息带来的问题：机器难以理解现实世界的复杂性，因为它们缺乏类似人类的认知。在数据和智能之间的这种相互作用中，数据标记承担了编排者的角色，为原始信息注入了上下文和意义。

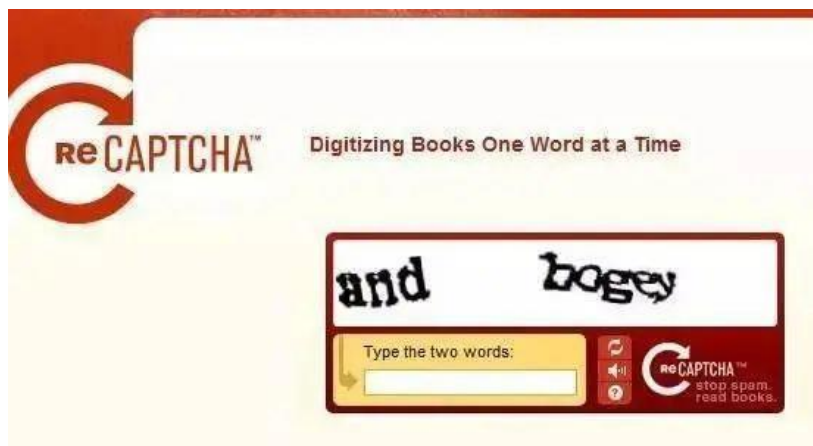
#### 数据集标注的重要性

**提升模型精度：**标注数据能够显著提高机器学习模型的准确性。

**数据质量的保障：**精确的标注能够确保数据的质量，为模型提供可靠的训练数据。

**多样化训练数据：**通过标注，可以处理不同类型的数据和标签，增加数据的多样性。

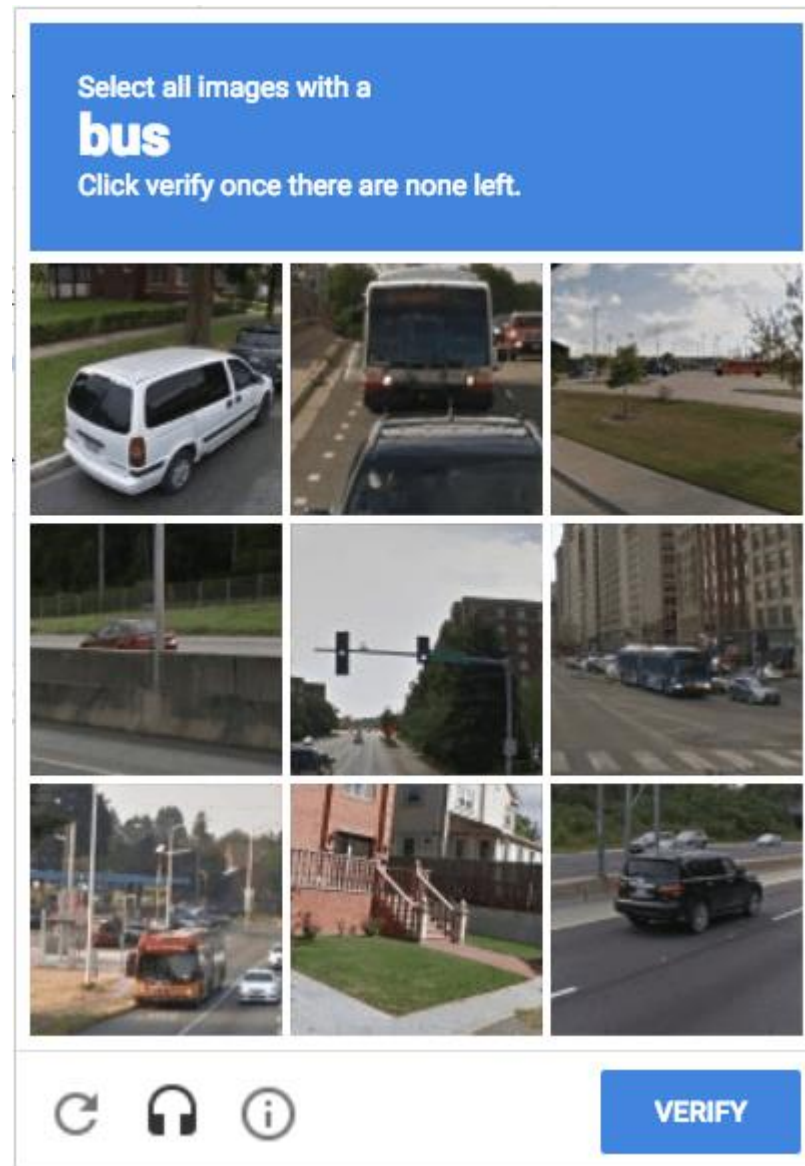
# 3.3 数据标注: 一个有趣的例子 验证码



E码:

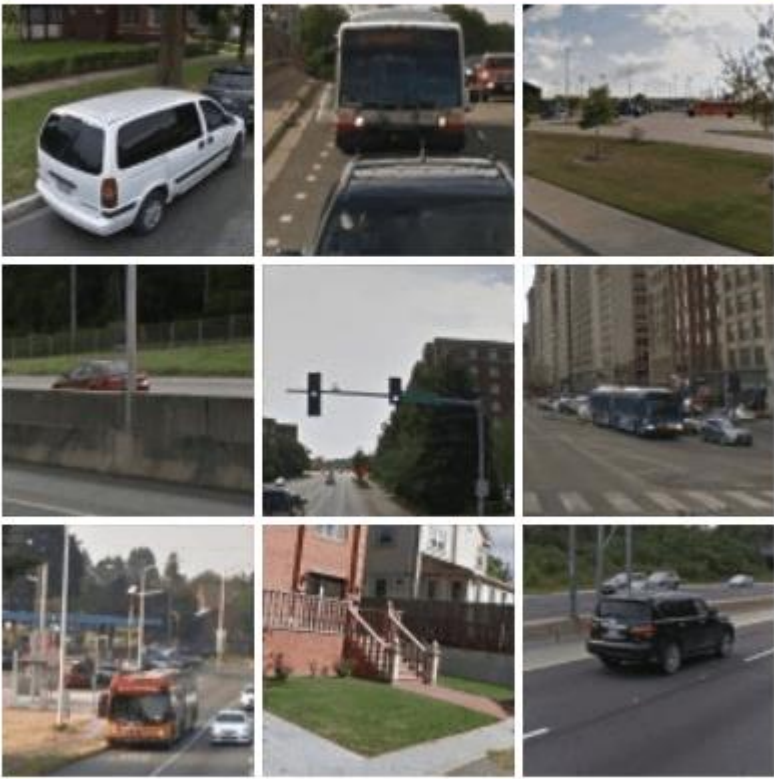


2015年12306验证码



# 3.3 数据标注: 一个有趣的例子 验证码的进化

Select all images with a **bus**  
Click verify once there are none left.



🔄 🎧 ⓘ

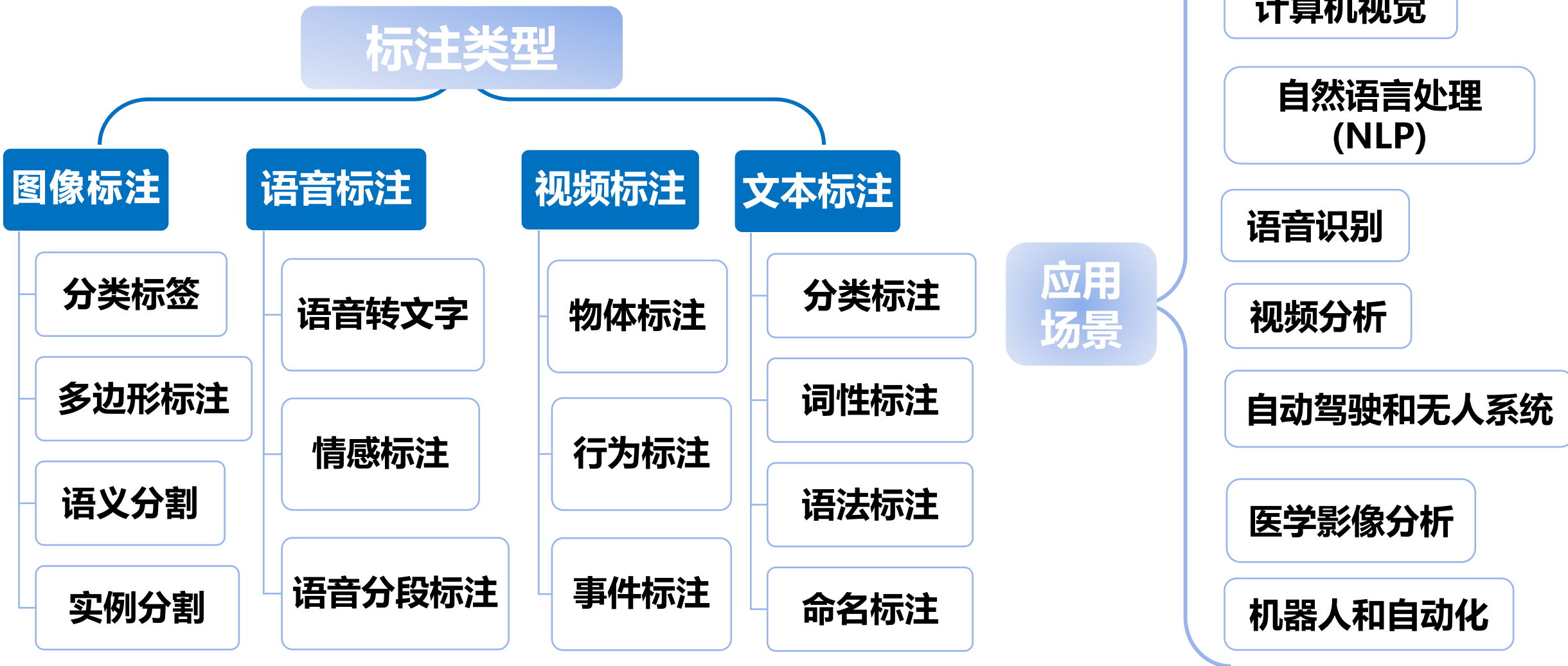
VERIFY

选择最符合描述的图片：“一簇在悬挂花盆中生长的多肉”



# 3.3 数据标注过程中可使用的工具

## 数据集标注的应用



### 数据集标注工具—LabelMe概述

LabelMe 是一个开源的图像标注工具，主要用于手动标注图像数据集中的对象和区域。它广泛应用于计算机视觉任务，如物体检测、图像分割和实例分割等。

#### 主要功能

**多边形标注：**LabelMe 允许用户使用多边形工具精确地标注图像中的任意形状。这对于需要高精度分割的任务非常有用，如实例分割或语义分割。

**标签管理：**用户可以为每个标注的对象添加标签，方便对不同类别的物体进行分类。这些标签可以在标注过程中动态添加或删除。

**多格式支持：**LabelMe 支持将标注结果导出为多种格式，如 JSON 和 XML 等，方便与不同的机器学习框架和工具兼容。

**图像预处理：**工具支持对图像进行简单的预处理操作，如缩放、旋转和翻转，帮助用户更方便地进行标注。

**批量处理：**允许用户批量加载图像进行标注，提高工作效率。完成标注后，可以将所有标注数据一并导出。

## 3.3 数据标注过程中可使用的工具

### 数据集标注工具—LabelMe安装使用

#### 安装

- 1、安装Anaconda环境: <https://www.anaconda.com/download> (地址)
- 2、打开Anaconda [Prompt](#), 创建anaconda虚拟环境labelme, 输入命令: conda create -n labelme python=3.8, 完成后如图

```
done
#
# To activate this environment, use
#
#     $ conda activate labelme1
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

- 3、激活刚刚创建的labelme虚拟环境, 输入命令: conda activate labelme
- 4、在虚拟环境中安装Labelme所需的依赖项。依次执行以下命令:  
①conda install pyqt      ②conda install pillow      ③pip install labelme

# 3.3 数据标注过程中可使用的工具

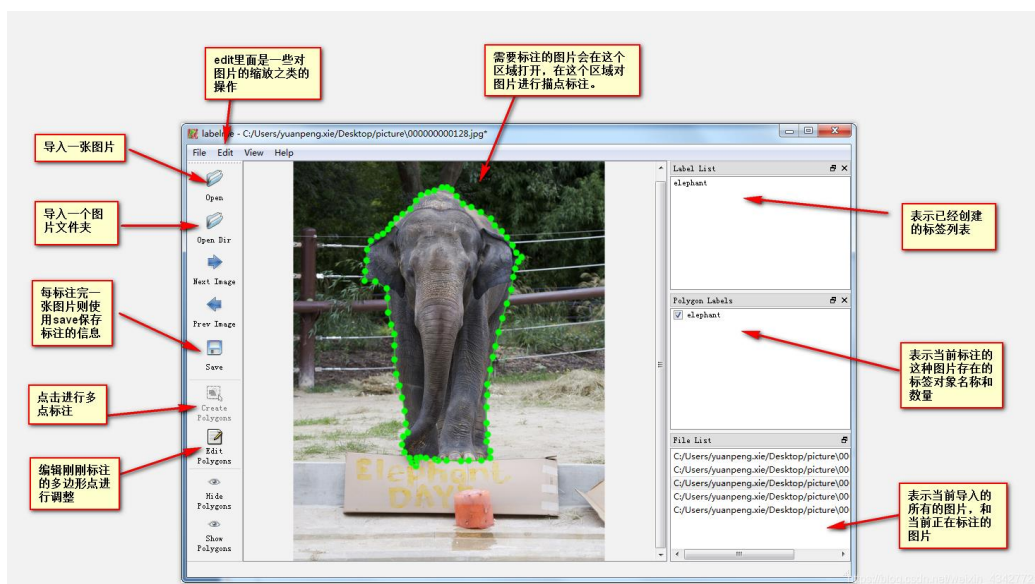
## 数据集标注工具—LabelMe安装使用

### 使用

1、打开Anaconda [Prompt](#), 激活labelme虚拟环境, 输入指令: activate labelme; 后输入环境名 labelme即打开

```
Anaconda Prompt - labelme  
  
(base) C:\Users\LENOVO>activate labelme  
  
(labelme) C:\Users\LENOVO>labelme
```

2、UI界面介绍:



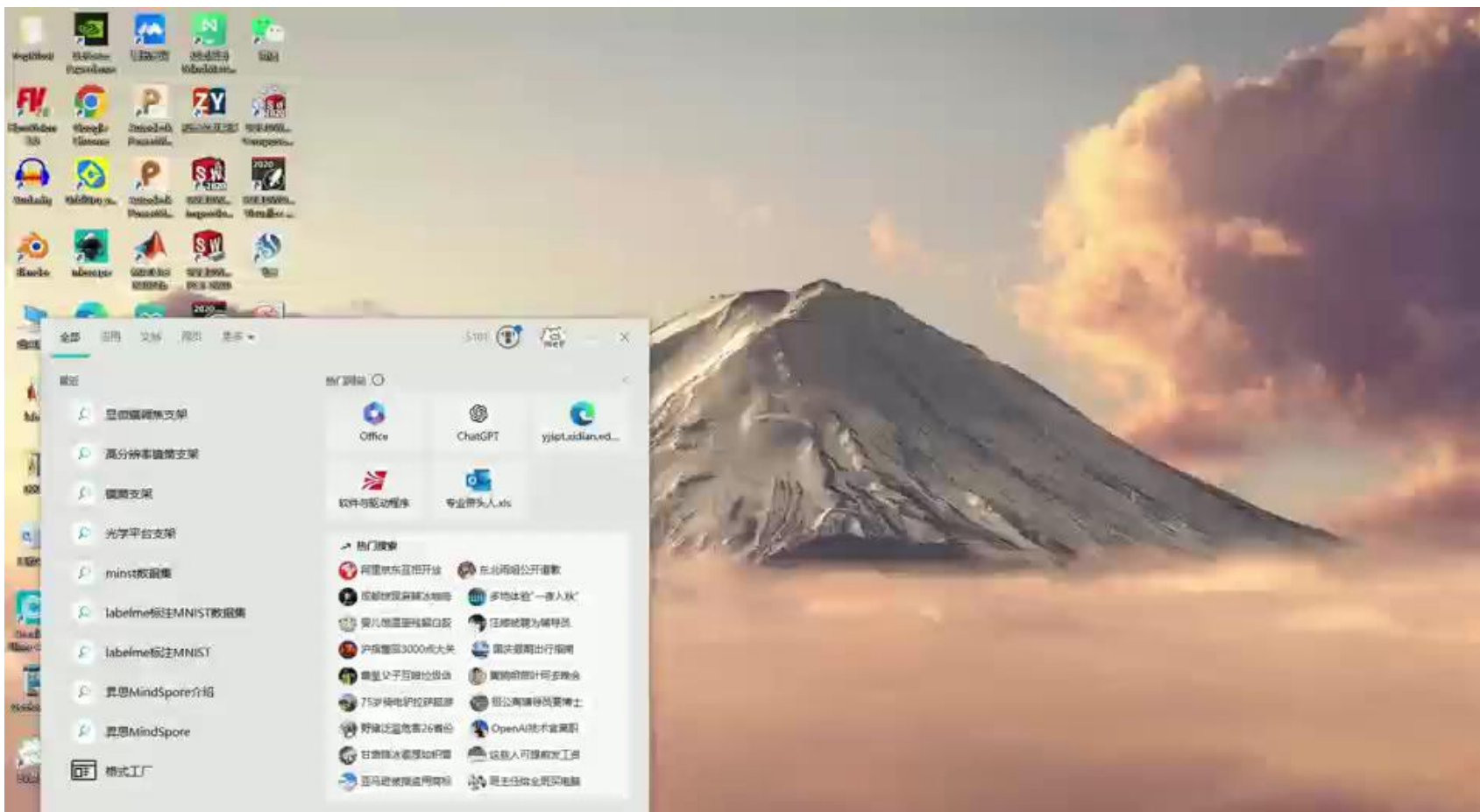
3、标注后保存Json文件, 运行json\_to\_dataset脚本, 将Json文件转换为数据集 (详细步骤见演示)



# 3.3 数据标注过程中可使用的工具

## 数据集标注工具—LabelMe安装使用

### 演示 (MNIST---数字标记)



### 最终数据集



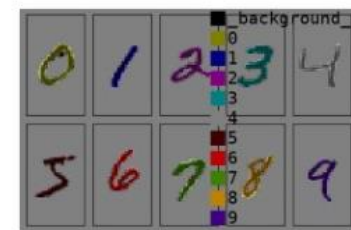
img.png



label.png



label\_names.txt



label\_viz.png



# 主 讲 内 容

## 3.1

### 人工智能算法开发流 程

## 3.2

### 人工智能的开发语言

## 3.3

### 数据标注过程中可使 用的工具

## 3.4

### 深度学习的开源框架

## 3.5

### 如何搜索人工智能 算法的开源资源

## 3.6

### 本讲小结



### PyTorch

PyTorch是一个由Facebook的人工智能研究团队开发的开源深度学习框架。在2016年发布后，PyTorch很快就因其**易用性**、**灵活性**和强大的功能而在科研社区中广受欢迎。

#### 动态图

动态计算图允许你在运行时更改图的行为。这使得PyTorch非常灵活，在处理不确定性或复杂性时具有优势。

#### 易于调试

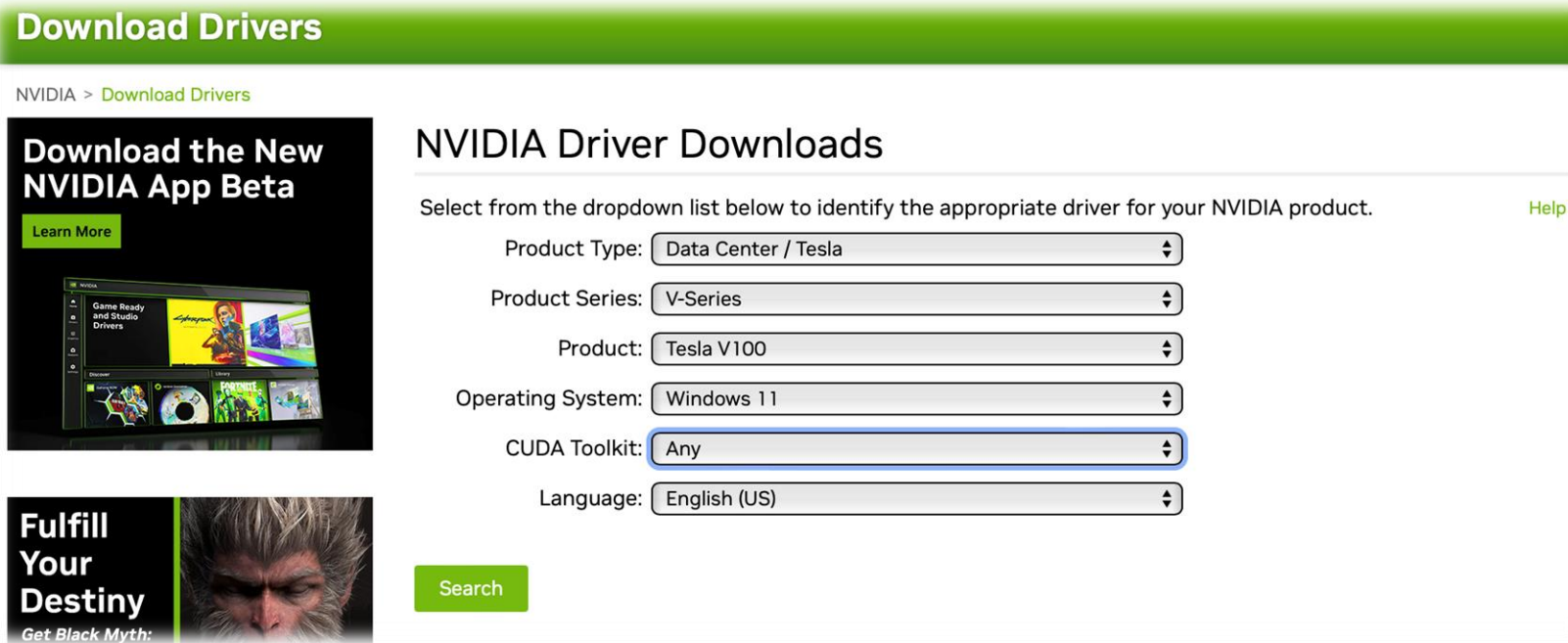
由于PyTorch的动态性和Python性质，可以使用Python的标准调试工具，直接查看每个操作的结果和中间变量的状态。

#### 社区支持

官方论坛、GitHub、Stack Overflow等平台上大量的PyTorch用户和开发者，你可以从中找到大量的资源和帮助。

## 3.4 深度学习的开源框架

### 显卡驱动安装：



Download Drivers

NVIDIA > Download Drivers

### Download the New NVIDIA App Beta

Learn More

Download the New NVIDIA App Beta

### NVIDIA Driver Downloads

Select from the dropdown list below to identify the appropriate driver for your NVIDIA product. [Help](#)

Product Type: Data Center / Tesla

Product Series: V-Series

Product: Tesla V100

Operating System: Windows 11

CUDA Toolkit: Any

Language: English (US)

Search

Fulfill Your Destiny

Get Black Myth:

由于深度学习需要GPU等具有并行计算架构的硬件加速，因此我们在安装PyTorch之前，首先要保证GPU等硬件的驱动程序正确安装。驱动以及后续开发工具的安装将以Nvidia系列显卡为例：

<https://www.nvidia.com/download/index.aspx>打开链接，通过显卡型号、系统信息等检索并下载驱动安装程序。

## 3.4 深度学习的开源框架

### CUDA安装:

CUDA是Nvidia的并行计算开发平台，PyTorch通过调用CUDA编写的GPU算子完成深度学习所需的基础运算。此前我们已经成功安装了Anaconda，为了方便深度学习环境的配置与删除，建议首先创建一个虚拟环境，此处我们以安装CUDA 12.1.1为例，打开命令行，创建并激活虚拟环境。

```
(base) PS C:\Users\bang\Desktop\zuoye> conda create -n torch python==3.9 创建
Retrieving notices: ...working... done
Collecting package metadata (current_repodata.json): done
```

```
(base) PS C:\Users\bang\Desktop\zuoye> conda activate Torch 激活
(Torch) PS C:\Users\bang\Desktop\zuoye>
```

新的环境名称

### ➤ Anaconda安装:

## Anaconda Installers



**Windows**

**Python 3.12**

↓ 64-Bit Graphical Installer (912.3M)



**Mac**

**Python 3.12**

↓ 64-Bit (Apple silicon) Graphical



**Linux**

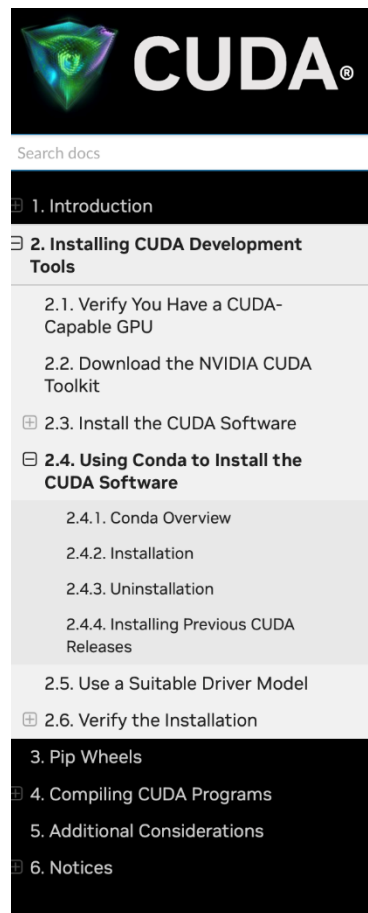
**Python 3.12**

↓ 64-Bit (x86) Installer (1007.9M)

PyTorch是使用Python语言进行开发的深度学习框架，为了方便地使用和管理开发过程中可能需要的环境依赖，我们需要Python的Anaconda发行版本，打开链接

<https://www.anaconda.com/download/success>，选择系统下载安装程序。

## CUDA安装:



### 2.4. Using Conda to Install the CUDA Software

This section describes the installation and configuration of CUDA when using the Conda installer. The Conda packages are available at <https://anaconda.org/nvidia>.

#### 2.4.1. Conda Overview

The Conda installation installs the CUDA Toolkit. The installation steps are listed below.

#### 2.4.2. Installation

To perform a basic install of all CUDA Toolkit components using Conda, run the following command:

```
conda install cuda -c nvidia
```

#### 2.4.3. Uninstallation

To uninstall the CUDA Toolkit using Conda, run the following command:

```
conda remove cuda
```

#### 2.4.4. Installing Previous CUDA Releases

All Conda packages released under a specific CUDA version are labeled with that release version. To install a previous version, include that label in the `install` command such as:

```
conda install cuda -c nvidia/label/cuda-11.3.0
```

<https://docs.nvidia.com/cuda/archive/12.1.1/cuda-installation-guide-microsoft-windows/index.html>,

点击 “Installation CUDA Development Tools” , 再选择 “Using Conda to Install the CUDA Software” , 参照文档步骤, 在**虚拟环境**中安装CUDA 12.1.1 (注意激活虚拟环境)。

## 3.4 深度学习的开源框架



### ➤ PyTorch:

## PyTorch安装指南

我们已经完成了安装PyTorch前的所有前序工作，接下来，打开链接

<https://pytorch.org/get-started/locally/>，此处我们选择Windows、Conda、CUDA

12.1，命令行**激活虚拟环境**，复制并粘贴下方安装指令。

PyTorch Build	Stable (2.4.0)		Preview (Nightly)		
Your OS	Linux	Mac	Windows		
Package	Conda	Pip	LibTorch	Source	
Language	Python		C++ / Java		
Compute Platform	CUDA 11.8	CUDA 12.1	CUDA 12.4	ROCm 6.1	CPU
Run this Command:	<pre>conda install pytorch torchvision torchaudio pytorch-cuda=12.1 -c pytorch -c nvidia</pre>				





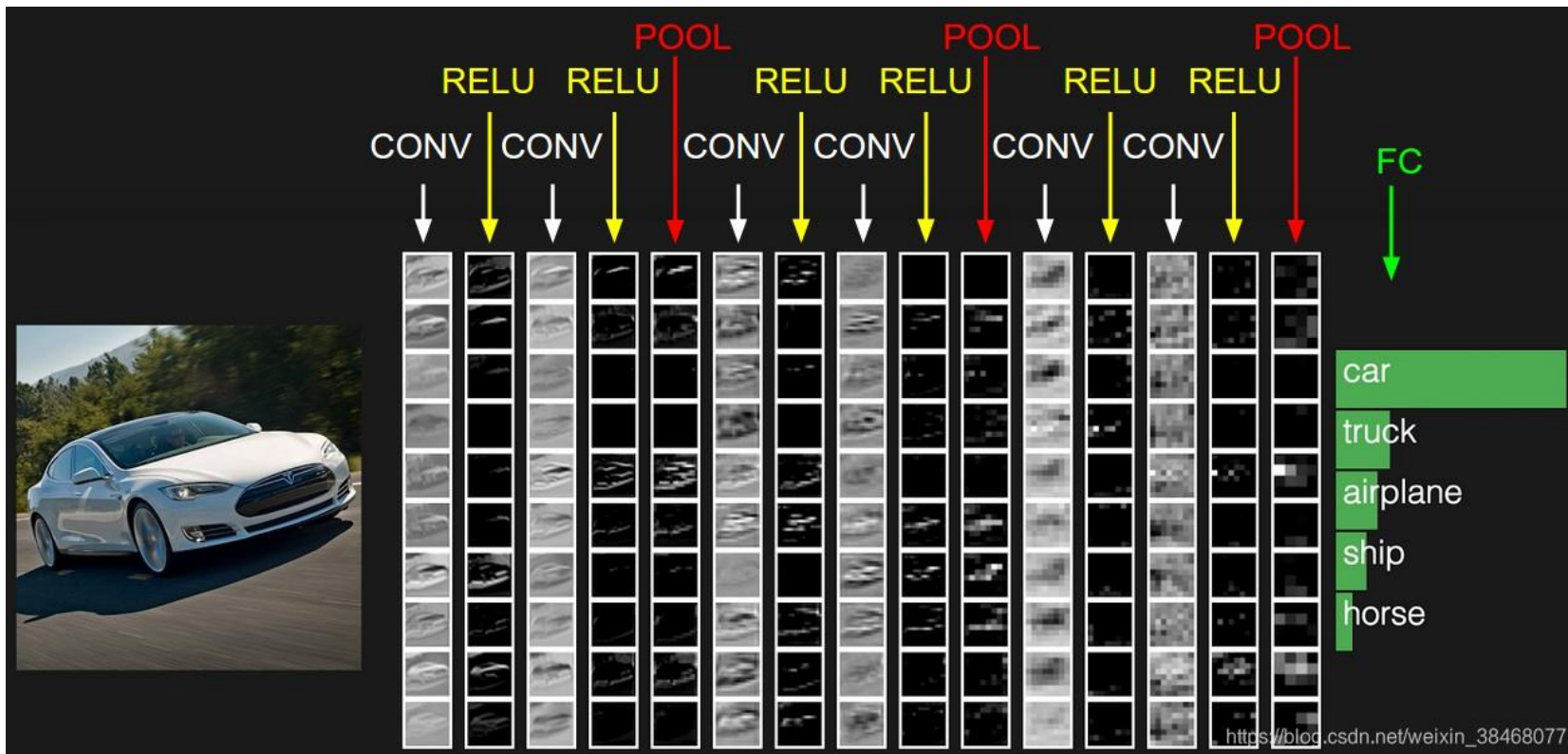
# 神经网络在实践中的应用： 手写数字识别

## 3.4 深度学习的开源框架

### CNN卷积神经网络实例(基于pytorch):手写数字识别

在学习这个基本的实例之前,我们需要对**卷积神经网络** (Convolutional Neural Network,CNN) 有一个基础的认识,卷积神经网络是一种具有局部连接、权重共享等特征的深层前馈神经网络。

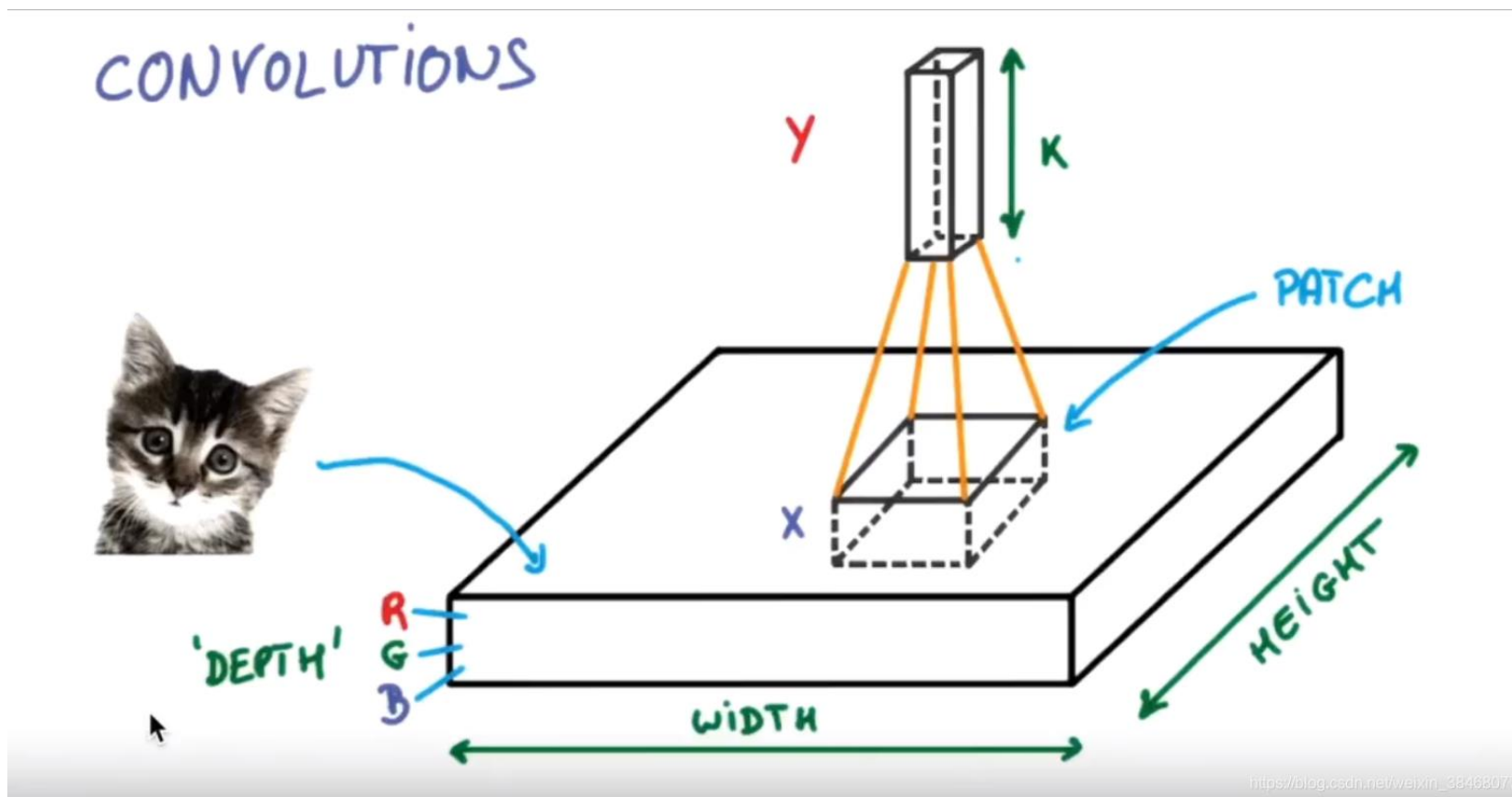
我们先来看一下比较直观的了解一下,对卷积神经网络有个感性的认识:



我们首先对我们输入的图片进行一次卷积,而后加入Relu激活函数,再做一次卷积,再加入Relu激活函数,而后对其处理结果进行汇聚(此处为Pool,也可以称为池化),再后来就是不断地重复前边的步骤,最后会得到一个属于某一类图片的概率。这时概率最大的那个就是我们需要的判别结果了。

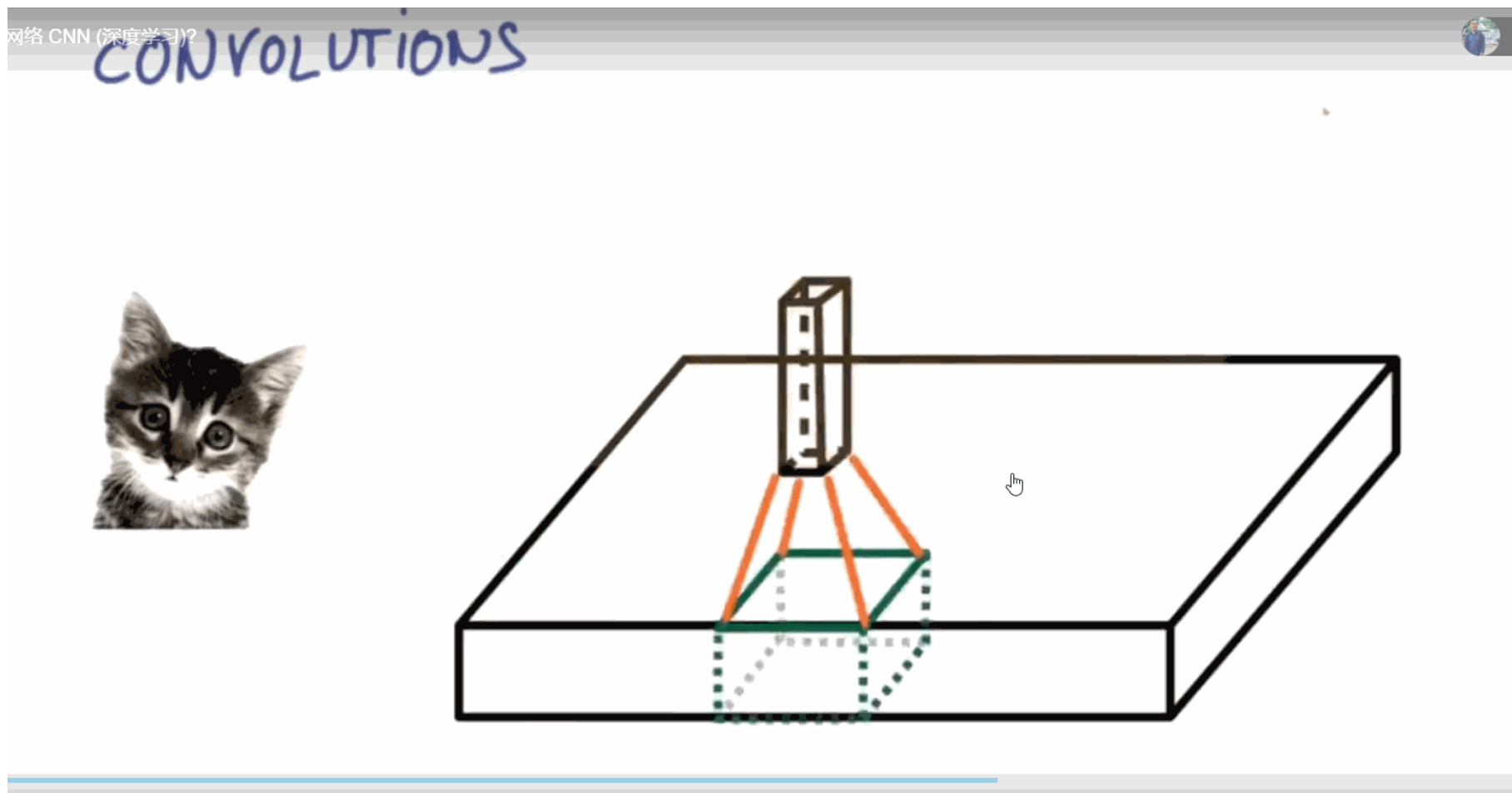
## 3.4 深度学习的开源框架

- 1、以训练这个小猫图片为例，如果输入图像为 $100 * 100 * 3$ （即图像长度为100，宽度为100，3个颜色通道：RGB），如果我们的输入图像为灰色的话，其颜色通道数为1，我们也称为图片的高度。



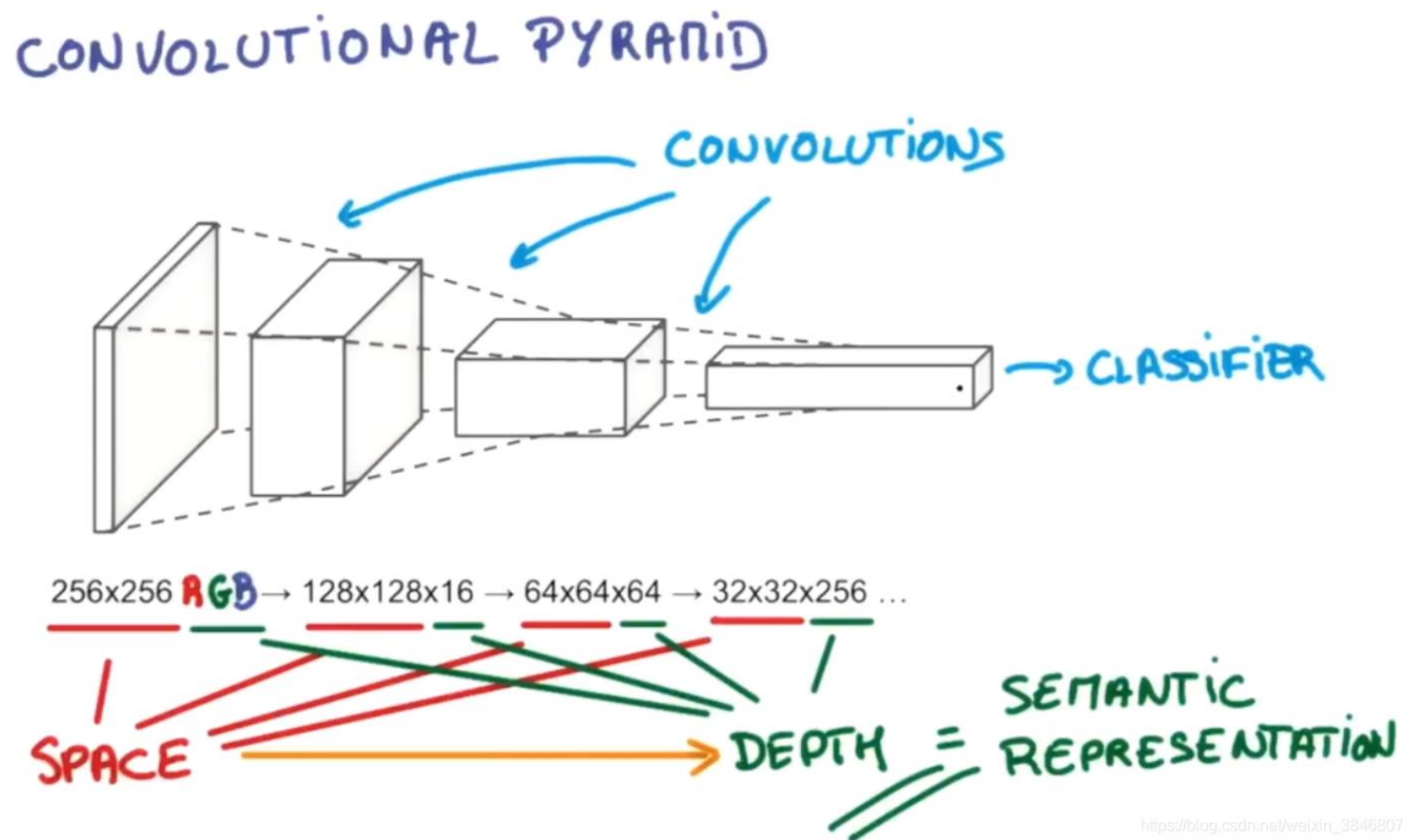
## 3.4 深度学习的开源框架

2、过滤器不断的在图像中收集小批小批的像素块，收集完所有的信息后，输出的值为一个比之前的高度更高，长和宽更小的图片，这个图片中包含了一些边缘信息。



## 3.4 深度学习的开源框架

3、然后再以同样的步骤再进行多次卷积，将图片的长宽再次压缩，高度再次增加，就会有对输入图片更深的理解。



4、然后将压缩，增高的信息嵌入普通的分类神经层上，我们就能对这个图片进行分类了。

# 3.4 深度学习的开源框架

## PyTorch工作流程



## 3.4 深度学习的开源框架

### ➤ 代码解释:

1、导入必要的库，主要包括PyTorch的核心模块、神经网络模块、自动微分模块、数据加载模块以及用于数据增强的Torchvision库。

```
1 import torch
2 import torchvision
3 import torchvision.transforms as transforms
4 import matplotlib.pyplot as plt
5 import numpy as np
```

2、环境设置。

```
9 # device = torch.device("cpu")
10 device = torch.device("cuda")
```

设置设备为cuda，也就是使用GPU进行计算。如果希望使用CPU，只需要将"cuda"改为"cpu"即可。

3、数据加载与预处理。

```
11 # Data loading and transformation
12 transform = transforms.Compose([
13     transforms.ToTensor(), # Convert images to Tensor
14     transforms.Normalize((0.5,),(0.5,)) # Normalize the images
15 ])
```

ToTensor(): 将 PIL 图像或 NumPy 数组转换为 PyTorch张量，并将其值从[0, 255]范围缩放到[0, 1]。

Normalize((0.5,),(0.5,)): 对图像进行归一化，使得张量中的每个值满足标准正态分布（均值为0.5，标准差为0.5），有效地将值范围从[0, 1]转换为[-1, 1]。

## 3.4 深度学习的开源框架

### 3、加载MNIST数据集，并创建训练和测试集的加载器（DataLoader）。

```
17 # Load the training dataset
18 trainset = torchvision.datasets.MNIST(root='./data', train=True, download=True, transform=transform)
19 trainloader = torch.utils.data.DataLoader(trainset, batch_size=4, shuffle=True, num_workers=0)
20
21 # Load the testing dataset
22 testset = torchvision.datasets.MNIST(root='./data', train=False, download=True, transform=transform)
23 testloader = torch.utils.data.DataLoader(testset, batch_size=4, shuffle=False, num_workers=0)
```

### 4、定义网络架构

```
26 class Net(torch.nn.Module):
27     def __init__(self):
28         super(Net, self).__init__()
29         self.conv1 = torch.nn.Conv2d(1, 6, 5) # First convolutional layer
30         self.pool = torch.nn.MaxPool2d(2, 2) # Max pooling layer
31         self.conv2 = torch.nn.Conv2d(6, 16, 5) # Second convolutional layer
32         self.fc1 = torch.nn.Linear(16 * 4 * 4, 120) # First fully connected layer
33         self.fc2 = torch.nn.Linear(120, 84) # Second fully connected layer
34         self.fc3 = torch.nn.Linear(84, 10) # Third fully connected layer
35
36     def forward(self, x):
37         x = self.pool(torch.nn.functional.relu(self.conv1(x))) # Apply ReLU and pooling
38         x = self.pool(torch.nn.functional.relu(self.conv2(x))) # Apply ReLU and pooling
39         x = x.view(-1, 16 * 4 * 4) # Flatten the tensor for the fully connected layer
40         x = torch.nn.functional.relu(self.fc1(x)) # Apply ReLU
41         x = torch.nn.functional.relu(self.fc2(x)) # Apply ReLU
42         x = self.fc3(x) # Output layer
43         return x
44
45 net = Net().to(device)
```

两个卷积层：conv1和conv2，每个卷积层后面跟一个ReLU激活函数和一个最大池化层（pool）。

三个全连接层：fc1，fc2和fc3，其中最后一层fc3输出10个类别，对应MNIST的数字类别（0-9）。

在forward方法中，定义了数据在网络中的前向传播过程。



## 3.4 深度学习的开源框架

### 5、定义损失函数和优化器

```
47 # Define the loss function and optimizer
48 criterion = torch.nn.CrossEntropyLoss() # CrossEntropyLoss for classification
49 optimizer = torch.optim.SGD(net.parameters(), lr=0.001, momentum=0.9) # SGD optimizer
50
```

学习率 动量

### 6、训练网络

```
51 # Train the network
52 for epoch in range(10): # Loop over the dataset multiple times
53     running_loss = 0.0
54     for i, data in enumerate(trainloader, 0):
55         inputs, labels = data
56         inputs, labels = inputs.to(device), labels.to(device)
57
58         optimizer.zero_grad() # Zero the parameter gradients
59
60         outputs = net(inputs)
61         loss = criterion(outputs, labels)
62         loss.backward() # Backpropagation
63         optimizer.step() # Optimize
64
65         running_loss += loss.item()
66         if i % 2000 == 1999: # Print every 2000 mini-batches
67             print(f'Epoch: {epoch + 1}, Batch: {i + 1} Loss: {running_loss / 2000:.3f}')
68             running_loss = 0.0
69
70 print('Training completed')
```

epoch: 每个epoch代表遍历一次整个训练数据集。

inputs, labels = data: 从trainloader中获取输入数据和对应标签, 并将它们转移到GPU。

optimizer.zero\_grad(): 清空优化器的梯度缓存, 以防止前一次计算的梯度对后续计算产生影响。

outputs = net(inputs): 将输入数据传入网络进行前向传播, 得到输出结果。

```
72 # Save the model weights
73 # torch.save(net.state_dict(), 'model_weights_cpu.pth')
74 torch.save(net.state_dict(), 'model_weights_gpu.pth')
75 print('Model weights saved successfully.')
```

保存权重

# PyTorch训练循环

```
51 # Train the network
52 for epoch in range(10): # Loop over the dataset multiple times
53     running_loss = 0.0
54     for i, data in enumerate(trainloader, 0):
55         inputs, labels = data
56         inputs, labels = inputs.to(device), labels.to(device)
57
58         optimizer.zero_grad() # Zero the parameter gradients
59
60         outputs = net(inputs)
61         loss = criterion(outputs, labels)
62         loss.backward() # Backpropagation
63         optimizer.step() # Optimize
64
65     running_loss += loss.item()
66     if i % 2000 == 1999: # Print every 2000 mini-batches
67         print(f'[Epoch: {epoch + 1}, Batch: {i + 1}] Loss: {running_loss / 2000: 3f}')
68         running_loss = 0.0
69
70 print('Training completed')
```

将数据传递给模型若干次

获取输入数据和对应标签，并将它们转移到GPU

将优化器梯度清零（梯度在每个时区都会累积，将其清零后，每个前向过程都将重新开始）

计算损失值（模型预测的错误程度）

对损失函数进行反向传播（计算每个参数的梯度，要求 grad=True）

根据 loss.backward() 计算出的梯度，优化器开始更新模型参数

## 3.4 深度学习的开源框架

### ➤ PyTorch---训练循环

训练循环涉及模型遍历训练数据并学习特征和标签之间的关系。

测试循环涉及检查测试数据并评估模型在训练数据上学习到的模式的优劣（模型在训练期间永远不会看到测试数据）。

	步骤名称	它起什么作用?	代码示例
1	前传	该模型会浏览所有训练数据一次，执行其forward()功能计算。	model(x_train)
2	计算损失	将模型的输出（预测）与基本事实进行比较，并进行评估以查看其错误程度。	loss = loss_fn(y_pred, y_train)
3	零梯度	优化器的梯度设置为零（默认情况下是累积的），因此可以针对特定的训练步骤重新计算它们。	optimizer.zero_grad()
4	对损失进行反向传播	计算每个需要更新的模型参数（每个参数为requires_grad=True）的损失梯度。这被称为 <b>反向传播</b> ，因此是“向后”的。	loss.backward()
5	更新优化器（ <b>梯度下降</b> ）	根据损失梯度更新参数requires_grad=True以改进它们。	optimizer.step()

# 3.4 深度学习的开源框架

## 7、预测结果

```
36 # Load the model weights
37 net = Net().to(device)
38 # net.load_state_dict(torch.load('model_weights_cpu.pth'))
39 net.load_state_dict(torch.load('model_weights_gpu.pth'))
40 net.eval() # Set the model to evaluation mode
41
42 # Load the MNIST test dataset
43 transform = transforms.Compose([
44     transforms.ToTensor(),
45     transforms.Normalize((0.5,), (0.5,))
46 ])
47
48 testset = torchvision.datasets.MNIST(root='./data', train=False, download=True, transform=transform)
49 testloader = torch.utils.data.DataLoader(testset, batch_size=4, shuffle=False)
50
```

加载权重

加载测试数据集

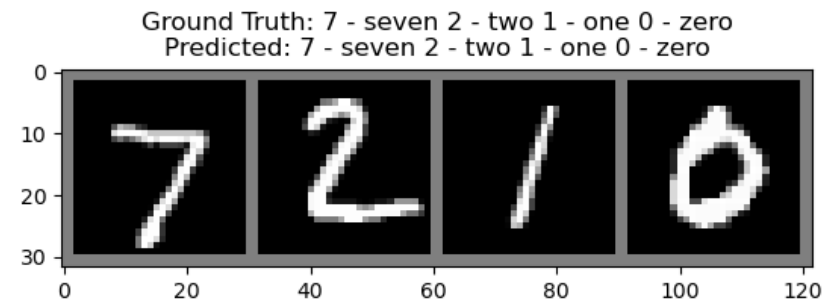
```
52 def imshow_with_labels(img, labels, predicted_labels):
53     img = img / 2 + 0.5 # Unnormalize the images
54     npimg = img.numpy()
55     plt.imshow(np.transpose(npimg, (1, 2, 0)))
56     plt.title('Ground Truth: ' + ' '.join(f'{testset.classes[labels[j]]:5s}' for j in range(len(labels))) +
57             '\nPredicted: ' + ' '.join(f'{testset.classes[predicted_labels[j]]:5s}' for j in range(len(predicted_labels))))
58     plt.show()
59
60 # Get some random testing images
61 dataiter = iter(testloader)
62 images, labels = next(dataiter)
63
64 # Predict images
65 outputs = net(images.to(device))
66 _, predicted = torch.max(outputs, 1)
67
68 # Show images with labels
69 imshow_with_labels(torchvision.utils.make_grid(images), labels, predicted)
70
71
```

定义图像展示函数

获取测试数据并进行预测

输出结果

## 8、输出结果



### ➤ 其他深度学习框架



Caffe



## ICT基础设施战略:拥抱智能时代，打造领先的ICT数字基础设施

政府 教育 医疗 金融 电力 油气 矿山 交通 制造 互联网

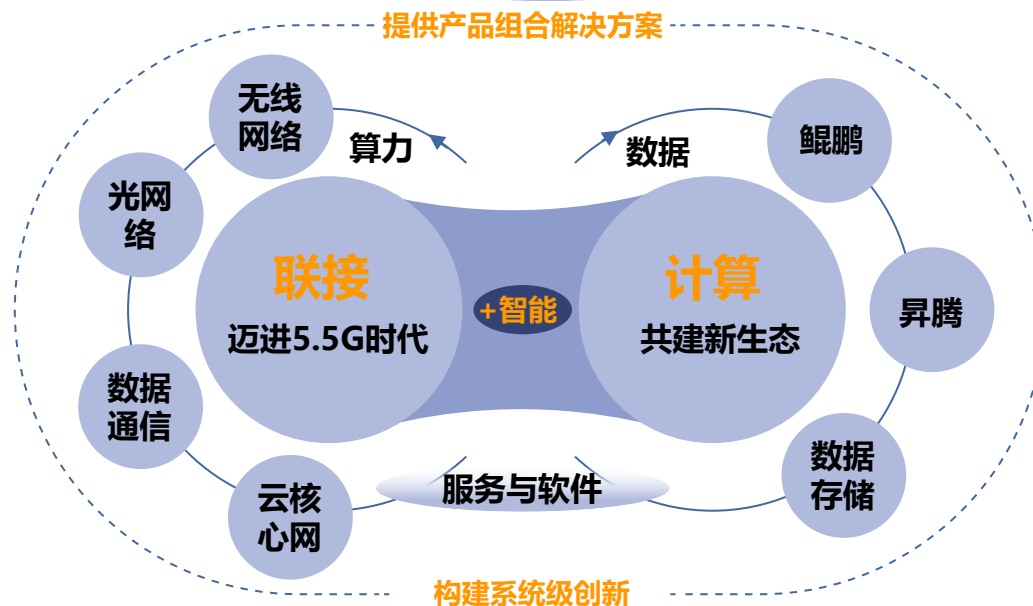
使能百模千态，赋能千行万业

### 联接

- 迈进5.5G时代，十倍网络能力提升
- 下行万兆，上行千兆，千亿互联，内生智能

### 计算

- 共建多样性计算新生态，赋能千行万业
- 架构创新，介质创新，算法创新，构建先进数据存力



### 基础软件生态

鸿蒙

欧拉

鸿蒙

昇思

毕昇

# 3.4 深度学习的开源框架

计算:昇腾AI, 使能百模千态、赋能千行万业



开放创新的昇腾 AI 基础软硬件平台

昇腾AI成为全球唯二支持万亿模型训练的技术路线



已原生孵化和适配业界一半以上的主流大模型

# 3.4 深度学习的开源框架

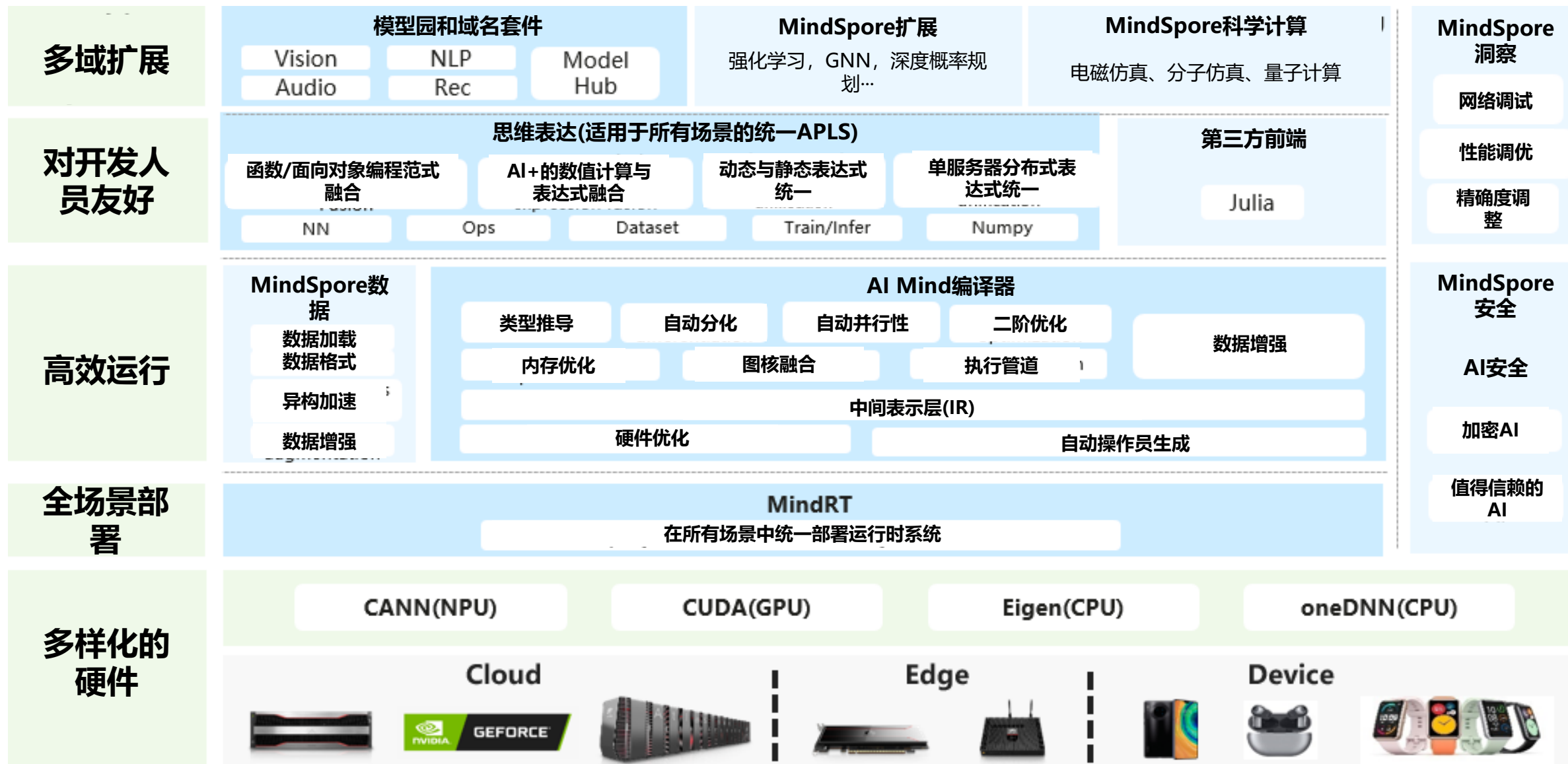
“新质生产力”是产业发展的引擎，而高校“新质生产力”的摇篮

GDP  
提升  
总量





# 3.4 深度学习的开源框架



## 3.4 深度学习的开源框架

# [M]<sup>s</sup> MindSpore



### 本地分布式模型训练

为基础模型培训提供多种并行功能。



### AI4S 融合计算框架

支持 AI+HPC 全过程编程能力和微可编程功能，可满足各种应用需求。



### 充分释放硬件潜力

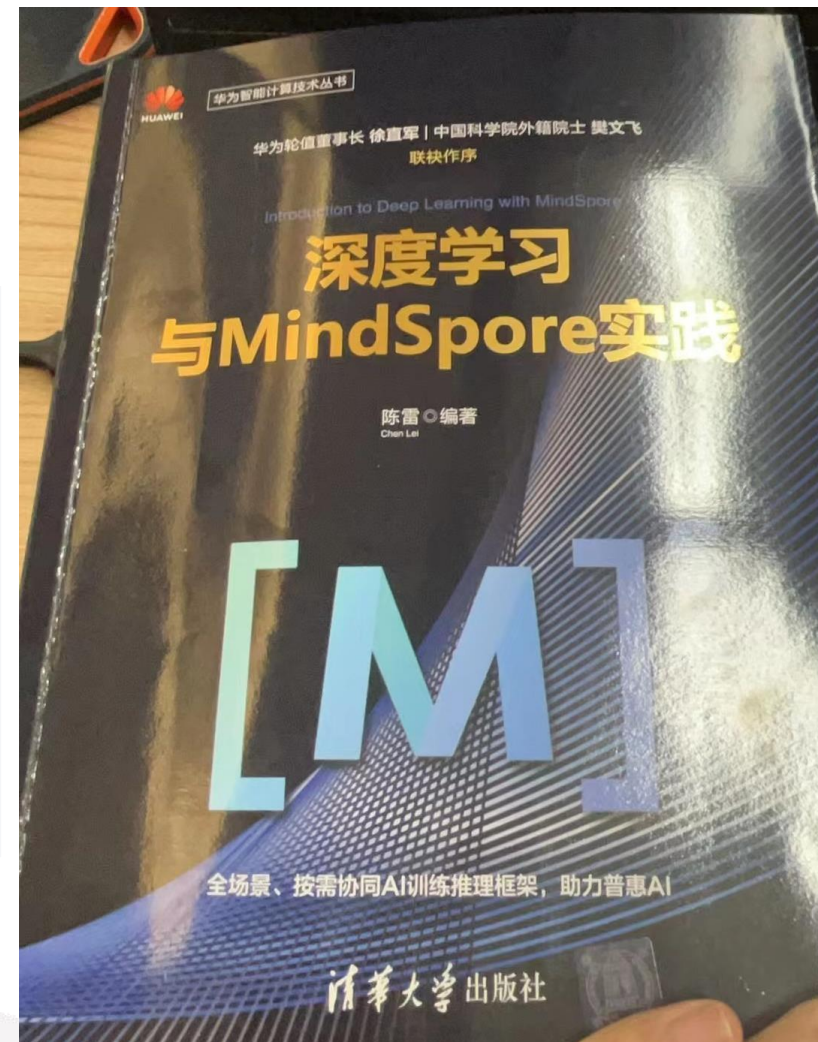
提供统一的动态和静态编程。



### 在所有场景下快速部署

支持在云端、边缘和移动设备上快速部署

<https://www.mindspore.cn/en>



# 3.4 深度学习的开源框架

## 深化产教融合，培养产业创新人才

从单一学科向交叉  
学科扩展

从本科生教学向研究生  
教学扩展

从产业知识融入向激发  
原生创新扩展

农学院

医学院

材料学院

金融学院

文院

...

支持本硕一体  
教学方案开发

支持交叉学科微  
专业课程设计

鲲鹏/昇腾产业  
知识学习成果

鲲鹏/昇腾  
原生开发  
创新贡献成果



计算机学院|人工智能学院|软件学院

支持计算产业难  
题研究

支持根技术  
创新论文发表

鲲鹏/昇腾奖学金



# 主 讲 内 容

## 3.1

### 人工智能算法开发流 程

## 3.2

### 人工智能的开发语言

## 3.3

### 数据标注过程中可使 用的工具

## 3.4

### 深度学习的开源框架

## 3.5

### 如何搜索人工智能 算法的开源资源

## 3.6

### 本讲小结



### 常见的开源平台



#### GitHub

- 最大的代码托管平台
- 丰富的开源项目和库



#### Kaggle

- 数据科学竞赛平台
- 提供数据集和内置的开源代码



#### Papers with Code

- 将研究论文与实现代码关联
- 提供最新的算法实现和基准测试

### 1. Github (<https://github.com/home>)

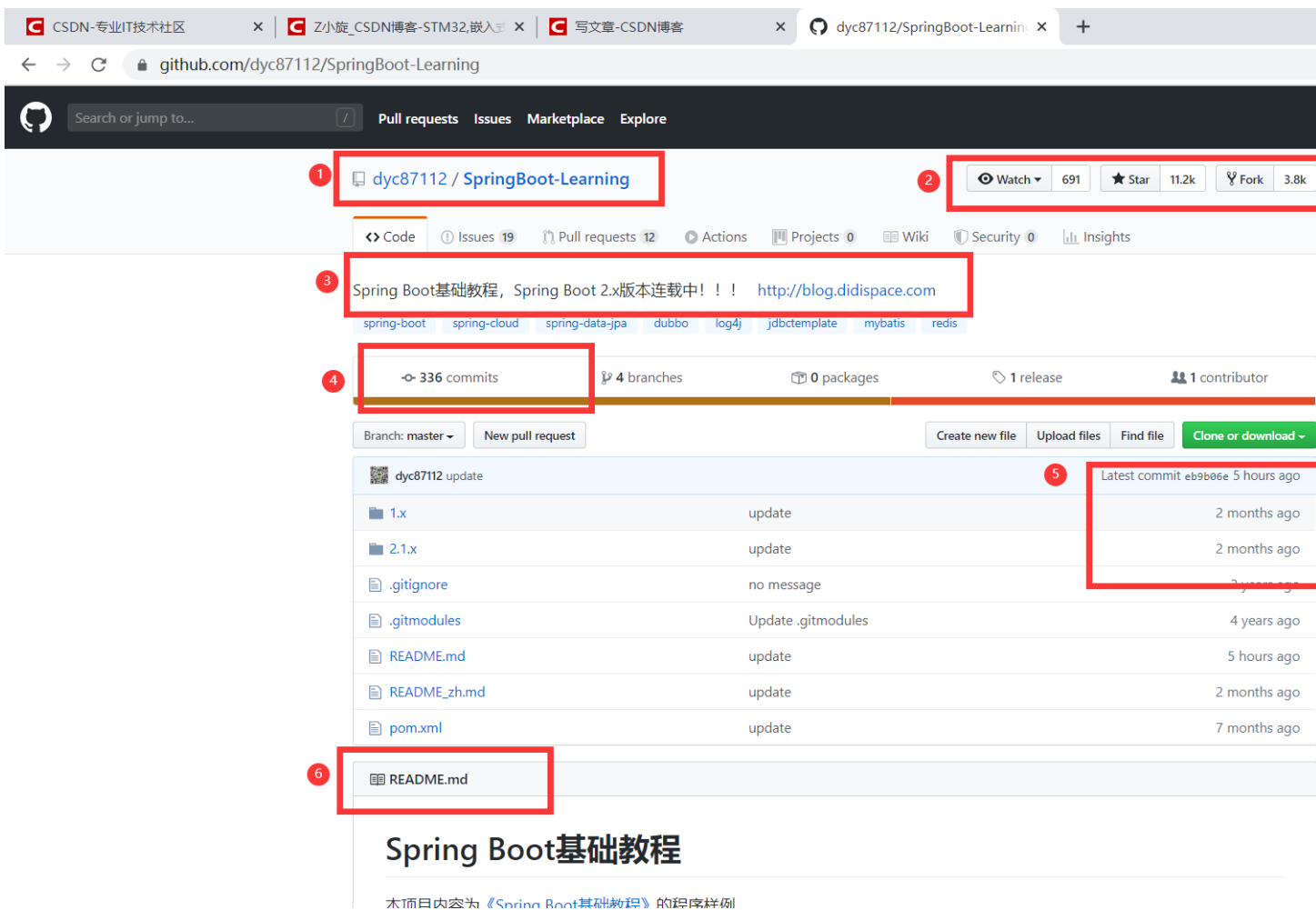


GitHub是一个面向开源及私有软件项目的托管平台，因为只支持Git作为唯一的版本库格式进行托管，故名GitHub。

通过 Github ，可以很方便的下载自己需要的项目，了解实时热点的项目，通过对优秀的开源项目的学习，更好的进行学习与提高

# 3.5 如何搜索人工智能算法的开源资源

## 1. Github (https://github.com/home)



The screenshot shows the GitHub repository page for 'dyc87112 / SpringBoot-Learning'. The page is annotated with red boxes and numbers 1 through 6, corresponding to the numbered list on the right. The repository name and path are highlighted with a red box and number 1. The star and fork counts are highlighted with a red box and number 2. The repository description and tags are highlighted with a red box and number 3. The commit count is highlighted with a red box and number 4. The commit history table is highlighted with a red box and number 5. The README.md file is highlighted with a red box and number 6.

- ① 此处是项目作者名/项目名
- ② 此处是项目的点赞数，和fork数，越火的项目点赞和fork就会越多
- ③ 项目的Description 和Website 和tags 也就是项目的说明和标签，通过此处你可以一眼了解该项目的功能和简介
- ④ 项目的commits提交数，一般比较好的项目，维护会比较频繁，更新也会频繁，提交数就会多
- ⑤ 项目提交时间，通过这里你可以看到项目的提交时间，防止自己下载了一些远古项目
- ⑥ README.md README.md文件是一个项目的入门手册，里面介绍了整个项目的使用、功能等等。所以README文件写得好不好，关系到这个项目能不能更容易的被其他人了解和使用。

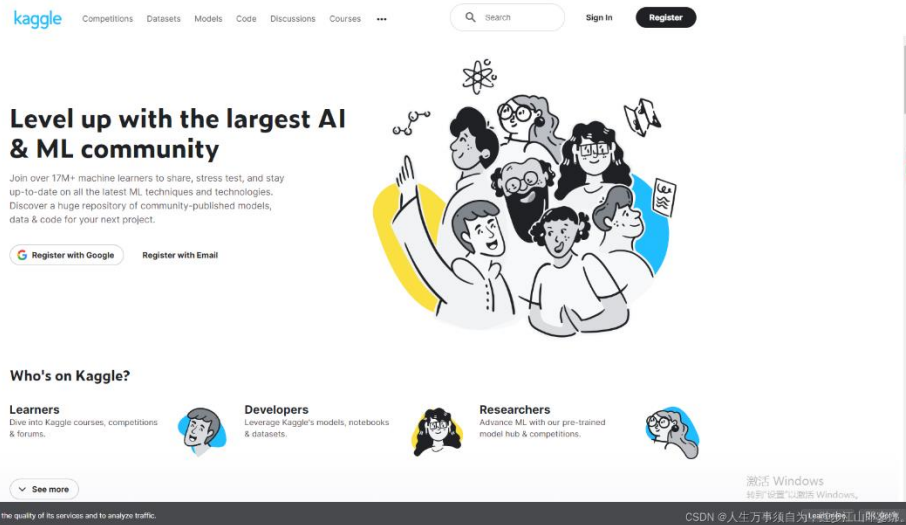
# 2. Kaggle (<https://www.kaggle.com>)



Kaggle是由联合创始人、首席执行官安东尼·高德布卢姆(Anthony Goldbloom) 2010年在墨尔本创立的,主要为开发商和数据科学家提供举办机器学习竞赛、托管数据库、编写和分享代码的平台。

Kaggle 最有特色的功能之一是举办各种数据科学竞赛,参与者需要使用机器学习或其他数据处理技术来解决这些问题。获胜者通常会获得现金奖励或其他形式的奖励。

除了竞赛, Kaggle 还允许用户上传和分享数据集。这些数据集可以是公共数据集,也可以是竞赛数据集。用户可以浏览这些数据集,并使用它们来训练模型或进行数据分析。

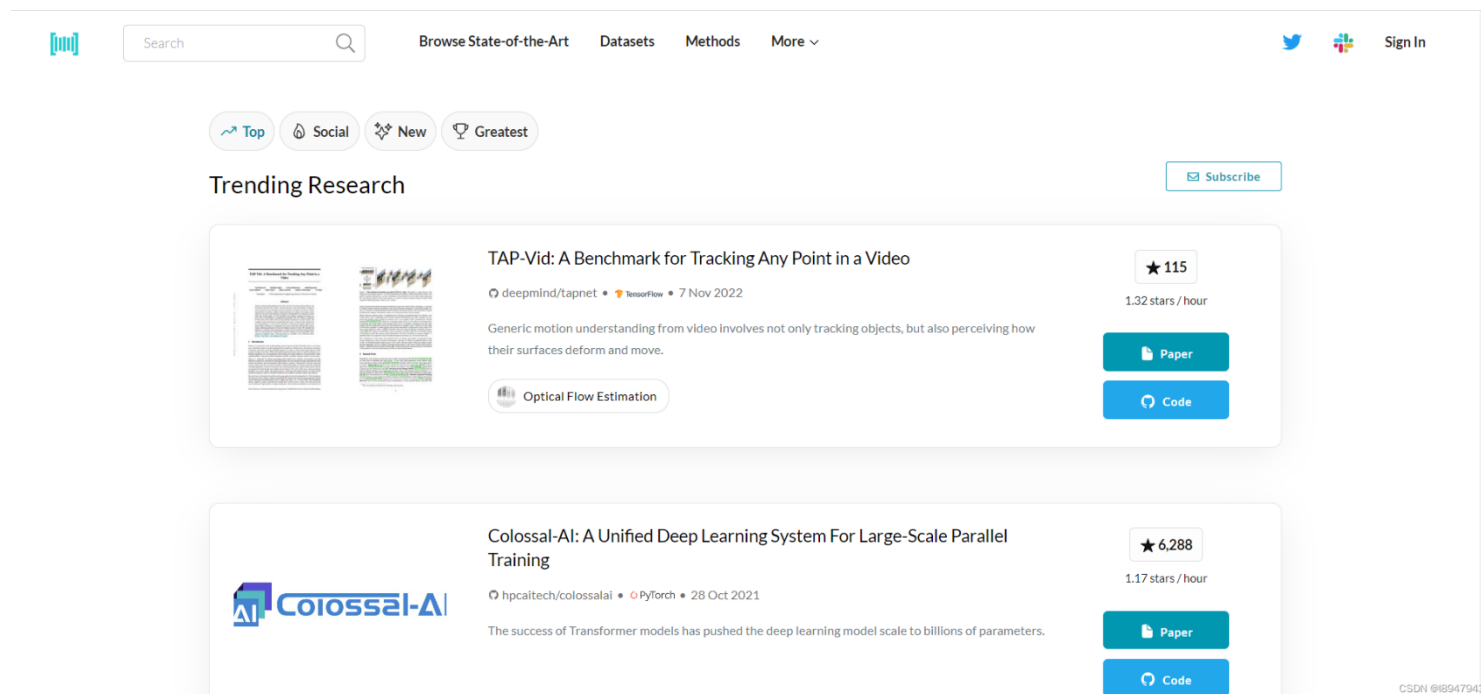




# 3. Papers with Code (<https://paperswithcode.com/>)



Papers with Code 是一个包含机器学习论文及其代码实现的网站。大多数论文都是有GitHub代码的，这个网站厉害地方就是对机器学习方向做了任务分类，检索对应的论文、数据、代码和精度榜单一目了然。



The screenshot shows the Papers with Code website interface. At the top, there is a search bar and navigation links for "Browse State-of-the-Art", "Datasets", "Methods", and "More". Below the navigation, there are filters for "Top", "Social", "New", and "Greatest". The main content area is titled "Trending Research" and features two research entries. The first entry is "TAP-Vid: A Benchmark for Tracking Any Point in a Video" by deepmind/tapnet, published on Nov 7, 2022. It has 115 stars and a rate of 1.32 stars per hour. The second entry is "Colossal-AI: A Unified Deep Learning System For Large-Scale Parallel Training" by hpcaltech/colossalai, published on Oct 28, 2021. It has 6,288 stars and a rate of 1.17 stars per hour. Both entries include buttons for "Paper" and "Code".

# 3. Papers with Code (<https://paperswithcode.com/>)

如果想找一些最近的前沿新文章：点击Browse State-of-the-Art进行查看最近的文章，网页会按照方向分类好



### Computer Vision

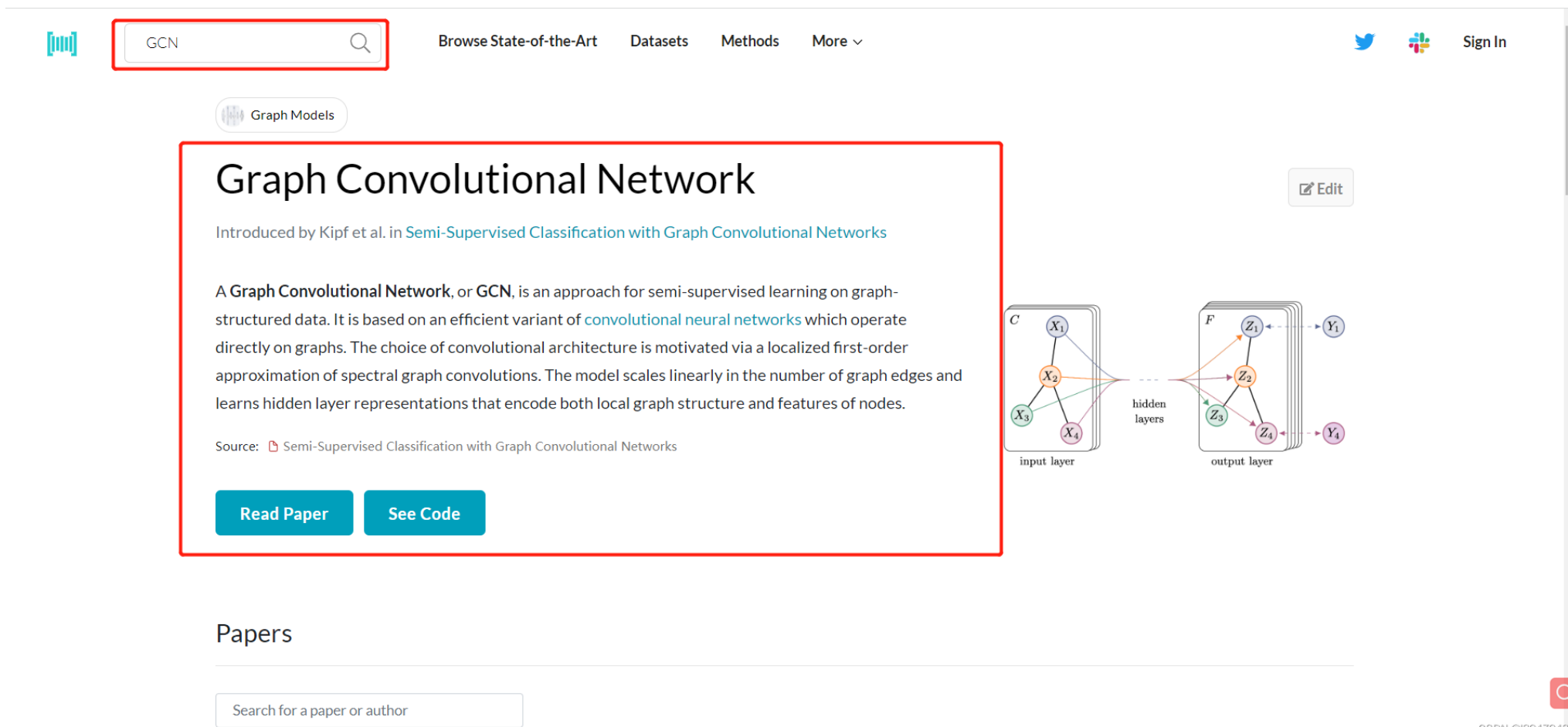
 Semantic Segmentation 175 benchmarks 3321 papers with code	 Image Classification 386 benchmarks 2727 papers with code	 Object Detection 259 benchmarks 2487 papers with code	 Image Generation 198 benchmarks 1060 papers with code	 Contrastive Learning 2 benchmarks 1053 papers with code
--	---	---	---	---

[▶ See all 1384 tasks](#)

### Natural Language Processing

## 3. Papers with Code (<https://paperswithcode.com/>)

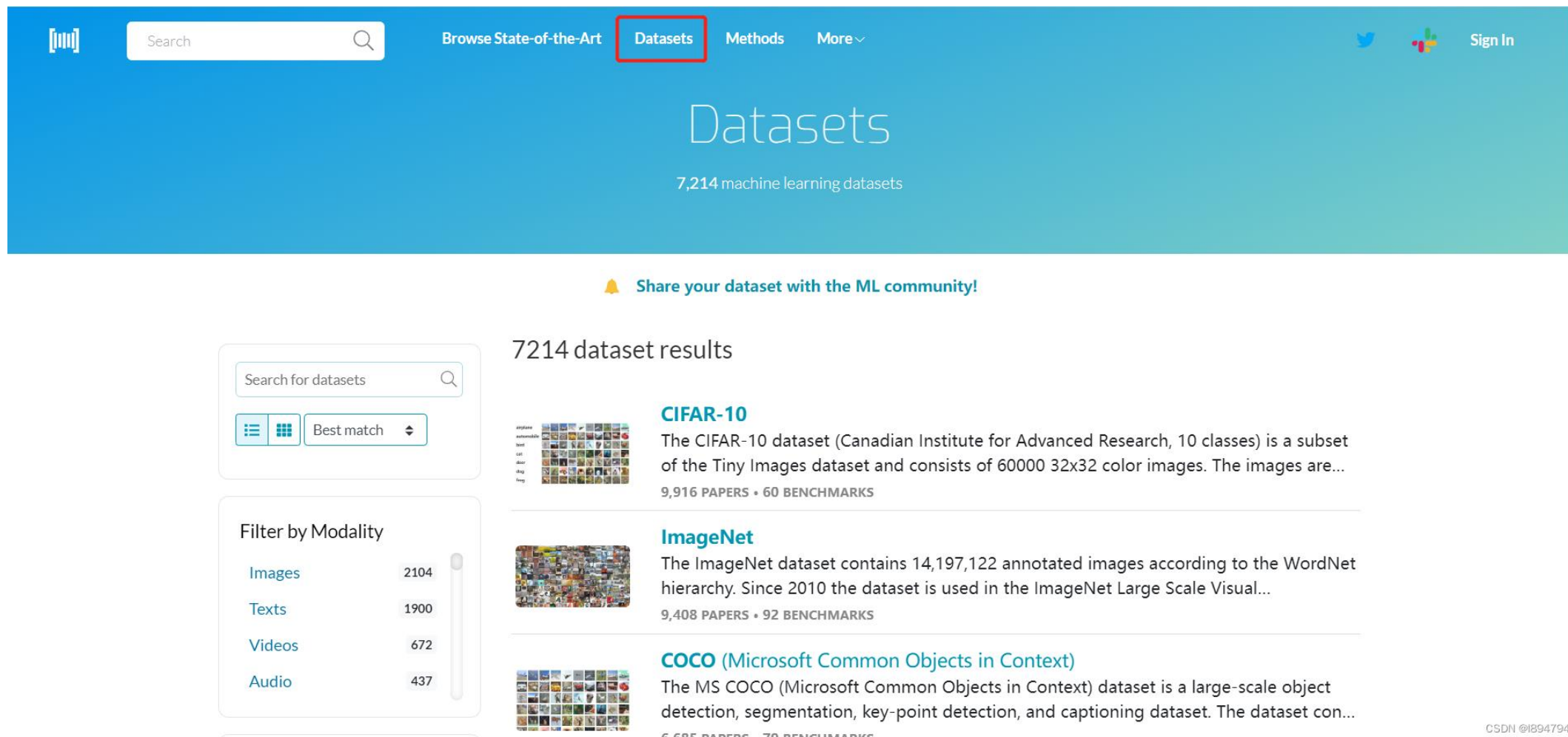
**如果想看指定论文:**比如想查找GCN论文, 搜索简称或者论文名即可检索, 回显内容有简介、论文地址、还有对应的代码, 如图:



The screenshot shows the Papers with Code website interface. At the top, there is a search bar containing the text "GCN". Below the search bar, the results for "Graph Models" are displayed. The main result is for "Graph Convolutional Network", which is highlighted with a red box. The result includes the title "Graph Convolutional Network", the authors "Introduced by Kipf et al. in [Semi-Supervised Classification with Graph Convolutional Networks](#)", a detailed description of the model, the source link, and two buttons: "Read Paper" and "See Code". To the right of the text is a diagram of a Graph Convolutional Network (GCN) architecture. The diagram shows an input layer with nodes  $X_1, X_2, X_3, X_4$  and an output layer with nodes  $Z_1, Z_2, Z_3, Z_4$ . The input layer is labeled "input layer" and the output layer is labeled "output layer". The hidden layers are represented by a dashed line. The diagram also shows the output nodes  $Y_1, Y_4$ .

## 3. Papers with Code (<https://paperswithcode.com/>)

如果想找所需的数据集：如图，点击datasets即可，界面如图：





# 主 讲 内 容

## 3.1

### 人工智能算法开发流 程

## 3.2

### 人工智能的开发语言

## 3.3

### 数据标注过程中可使 用的工具

## 3.4

### 深度学习的开源框架

## 3.5

### 如何搜索人工智能 算法的开源资源

## 3.6

### 本讲小结



- 人工智能算法开发流程 —— **基本8步骤**
- 人工智能的开发语言 —— **Python, C++, matlab .....**
- 数据标注过程中可使用的工具 —— **LabelMe .....**
- 深度学习的开源框架 —— **PyTorch .....**
- 如何搜索人工智能算法的开源资源 —— **Github, Kaggle, Papers with Code .....**