

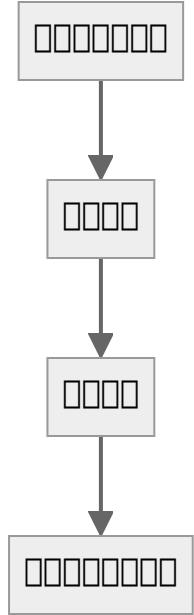


From the basic to the frontier | SPA

Reporter: Jiakai Gu  
Date: Nov 13th, 2023

# Table of contents

1.
2.
3.
4.
5.
6.



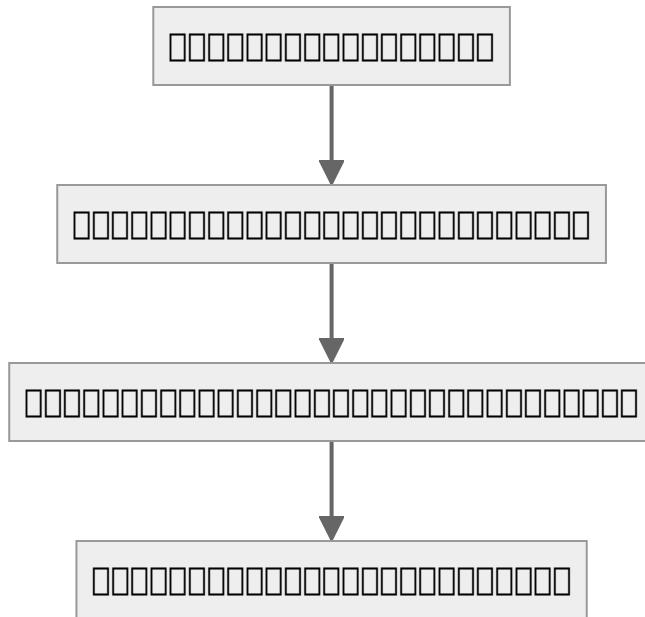
Supervised learning



Learns from being given "right answers"

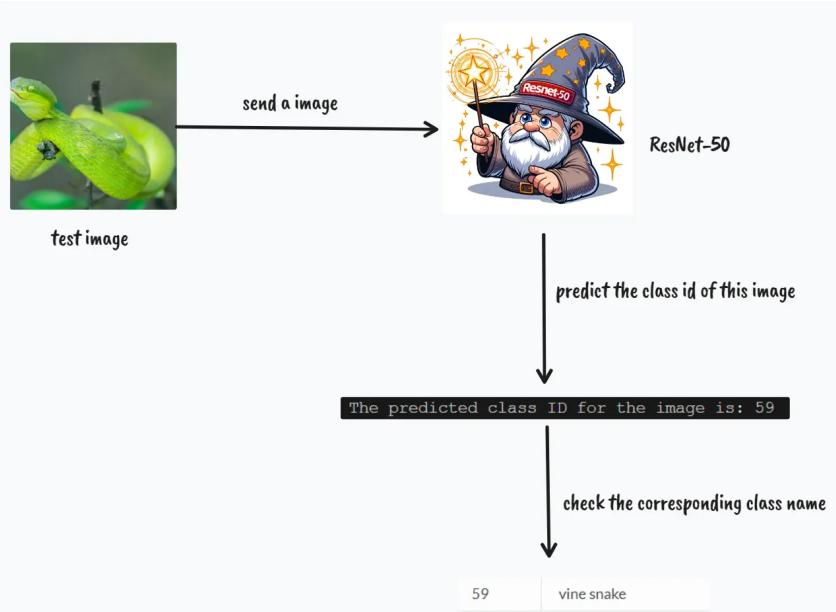
Andrew NG: What is supervised learning.

XXXXXX|XXXXXX|XXXXXX|XXXXXX|XXXXXX/XXXXXX|XXXXXX





## ResNet-50在ImageNet上的应用



结束

```
# 导入必要的库
import torch
import torchvision.models as models
import torchvision.transforms as transforms
from PIL import Image

# 加载预训练模型
# 初始化ResNet-50模型。'DEFAULT'值使模型使用预训练的权重。
model = models.resnet50(weights='DEFAULT')

# 将模型设置为评估模式。在评估模式下，模型的行为（例如，关闭批处理规范化和丢弃层的训练行为）会有所不同。
# 以确保模型在推理时的准确性和可靠性。
model.eval()

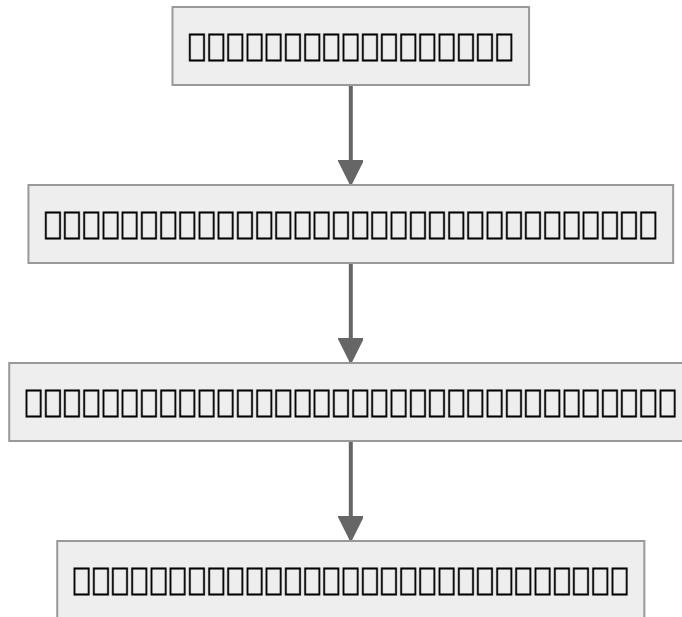
# 定义图像预处理流程
# 这个流程包括：调整图像大小、中心裁剪、将图像转换为张量、以及正则化。
transform = transforms.Compose([
    transforms.Resize(256),           # 调整图像大小为256x256
    transforms.CenterCrop(224),        # 中心裁剪图像为224x224
    transforms.ToTensor(),            # 将图像转换为张量
    transforms.Normalize(             # 正则化张量
        mean=[0.485, 0.456, 0.406],   # 使用Imagenet数据集的均值
        std=[0.229, 0.224, 0.225]     # 使用Imagenet数据集的标准差
    ),
])

# 定义图像类别预测函数
def predict_image_class(image_path):
    # 打开图像
    image = Image.open(image_path)
    # 对图像应用预处理流程
    image = transform(image).unsqueeze(0)
    # 不追踪梯度，以减少内存使用
    with torch.no_grad():
        # 获取模型输出
        outputs = model(image)
        # 获取预测类别的索引
        _, predicted = outputs.max(1)
        # 返回预测类别的索引
    return predicted.item()

# 测试图像类别预测函数
image_path = 'assets/test.jpg' # 替换为您的图片路径
class_id = predict_image_class(image_path)
print(f"The predicted class ID for the image is: {class_id}")
```

ResNet-50

A horizontal row of 20 black 'X' characters, likely representing a password or a key. The characters are evenly spaced and aligned horizontally.



# 词性标注

- NLTK——averaged\_perceptron\_tagger  
Penn Treebank



输出结果

```
import nltk
# 下载需要的数据包
# 下载用于分词的数据包: punkt
nltk.download('punkt')
# 下载用于词性标注的预训练模型: averaged_perceptron_tagger
nltk.download('averaged_perceptron_tagger')

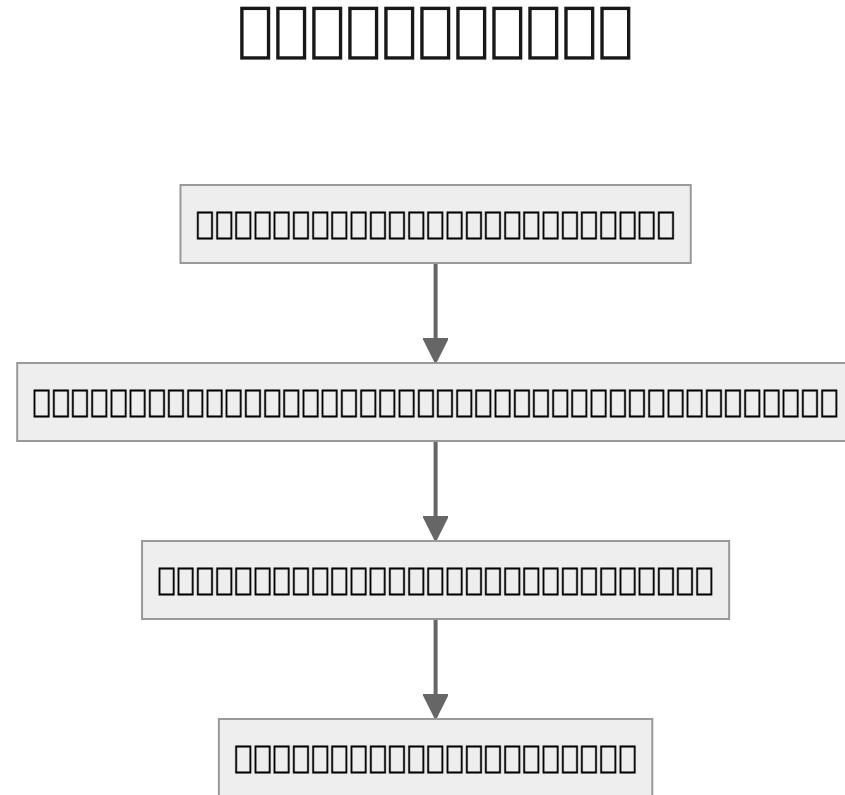
# 输入句子
sentence = "The cat sat on the mat."

# 分词
tokens = nltk.word_tokenize(sentence)

# 词性标注
tagged_tokens = nltk.pos_tag(tokens)

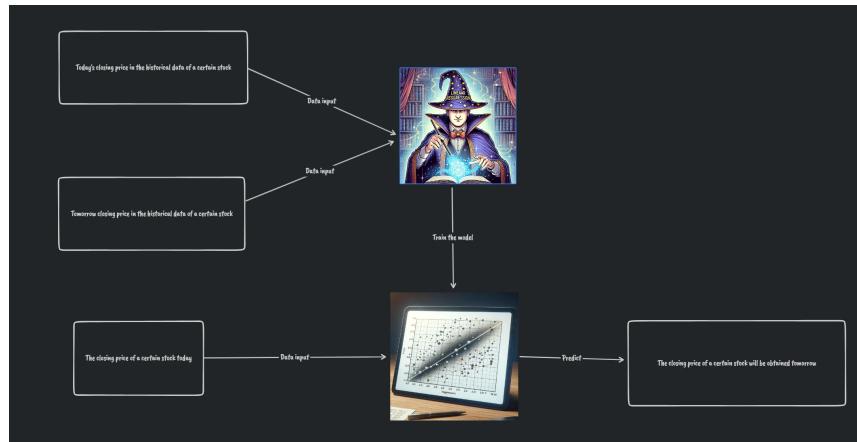
print("Tagged Tokens: ", tagged_tokens)
```

输出结果



# 线性回归

- 线性回归 NumPy
- Python



线性回归

```
# 导入所需的库: sklearn中的线性回归模型和numpy
from sklearn.linear_model import LinearRegression
import numpy as np

# 准备输入数据: 今天的收盘价, 单位是美元。
# reshape(-1, 1) 用于将数组转化为二维矩阵, 以满足scikit-learn模型的输入要求。
X = np.array([120, 125, 130, 135]).reshape(-1, 1)

# 准备标签数据: 即我们要预测的明天的收盘价, 单位也是美元。
y = np.array([125, 130, 135, 140])

# 初始化线性回归模型
model = LinearRegression()

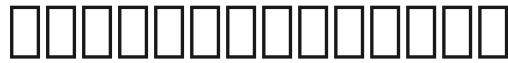
# 使用fit方法对模型进行训练。这一步会找到最适合描述X和y之间关系的直线。
model.fit(X, y)

# 准备一个新的输入数据 (今天的收盘价) 以进行预测。
# 同样使用reshape(-1, 1)确保它是二维矩阵。
new_X = np.array([140]).reshape(-1, 1)

# 使用模型进行预测
prediction = model.predict(new_X)

# 输出预测结果: 预测明天的收盘价, 单位是美元。
print(f"预测明天的收盘价为: {prediction[0]:.2f}美元")
```

线性回归



ChatGPT 通过 (RLHF) via

ChatGPT is based on particular GPT foundation models, namely GPT-3.5 and GPT-4, that were fine-tuned to target conversational usage.<sup>[9]</sup> The fine-tuning process leveraged both supervised learning as well as reinforcement learning in a process called reinforcement learning from human feedback (RLHF).<sup>[10][11]</sup> Both approaches employed human trainers to improve model performance. In the case of supervised learning, the trainers played both sides: the user and the AI assistant. In the reinforcement learning stage, human trainers first ranked responses that the model had created in a previous conversation.<sup>[12]</sup> These rankings were used to create "reward models" that were used to fine-tune the model further by using several iterations of Proximal Policy Optimization (PPO).<sup>[10][13]</sup>

OpenAI GPT-4V OpenAI whisper via

Whisper is an automatic speech recognition (ASR) system trained on 680,000 hours of multilingual and multitask supervised data collected from the web. We show that the use of such a large and diverse dataset leads to improved robustness to accents, background noise and technical language. Moreover, it enables transcription in multiple languages, as well as translation from those languages into English. We are open-sourcing models and inference code to serve as a foundation for building useful applications and for further research on robust speech processing.

- ChatGLM2-6B (train.json dev.json)
- OpenAI Create a GPT GPT

 You  
Emalink, 请向广西民族大学人工智能学院2023级的研究生问好。顺带介绍一下你自己。

 Emalink  
你好, 广西民族大学人工智能学院2023级的研究生们! 我是Emalink, 由Jiakai与OpenAI共同创建的一个定制版聊天机器人。我的设计宗旨是提供情感支持和友好的陪伴。如果你们有什么疑问或需要倾听, 我在这里随时准备帮助大家。希望我们能有愉快的交流!

Fine-Tuning in LLM.



- $\text{Input } X$   $\rightarrow$   $\text{Output } Y$
- $\text{Input } X$   $\rightarrow$   $\text{Output } Y$
- $\text{Input } X$   $\rightarrow$   $\text{Output } Y$

| Input ( $X$ )     | Output ( $Y$ )                       | Application         |
|-------------------|--------------------------------------|---------------------|
| email             | $\rightarrow$ spam? (0/1)            | spam filtering      |
| audio             | $\rightarrow$ text transcripts       | speech recognition  |
| English           | $\rightarrow$ Spanish                | machine translation |
| ad, user info     | $\rightarrow$ click? (0/1)           | online advertising  |
| image, radar info | $\rightarrow$ position of other cars | self-driving car    |
| image of phone    | $\rightarrow$ defect? (0/1)          | visual inspection   |

Thank you for listening!