推荐序一

宇宙斗转星移，时间呼啸向前，信息科技正在深刻改变着这个世界。

自 20 世纪末以来，信息科技一直是引领社会变革、产业变革和科技革命的重要力量。它在重塑知识发现，并不断与其他学科交叉融合，推动了各个学科的进步与发展，正在对人类社会产生重大、广泛的影响。

技术发展对社会变革有着深远的影响。它可以改变人们的生活方式、经济结构、社会关系和文化观念，推动社会的发展和进步。如果说互联网的出现促进了信息交流，推动了全球化和数字化的发展，那么大数据、云计算、机器人等核心技术的出现，则改变了生产和服务的方式。作为引领新一轮科技革命和产业变革的重要驱动力，人工智能则催生了大批新产品、新技术、新业态和新模式。

自 1956 年美国达特茅斯会议上“人工智能(AI)”概念的诞生，一直到 2022 年 11 月 30 日 ChatGPT 发布，人工智能经历了 4 次浪潮，ChatGPT 的出现是又一次标志性事件。ChatGPT 上线两个月，活跃用户即过亿，成为互联网用户增长最快的应用之一，是人工智能发展史上的一个重大转折点，随后国内国际各种语言大模型、多模态大模型等 AI 大模型如雨后春笋般冒出来，一时大有风起云涌之势。

AI 大模型是指具有巨大规模和强大计算能力的人工智能模型，能够模拟人类的认知过程和智能表现。认知大模型在多个领域具有很好的表现，如自然语言处理、图像识别、机器翻译等。这些模型的性能超越了传统方法，取得了重大的突破，引发了人们对其潜力的广泛兴趣。

AI 大模型对教育提出了全新的挑战，因为它有潜力在许多传统的专业性工作领域取代人类，在某些领域可以替代一些重复性和烦琐的任务，从而减少人力需求，可能导致一些传统的职业面临失业风险。

各类 AI 大模型的出现会不会是人类迈向通用人工智能的奇点？由 ChatGPT 和认知大模型掀起的热潮带来了许多积极的影响，推动了人工智能技术的进步和创新，为各种应用场景提供了更强大的解决方案。当然，它也引发了公众对人工智能的关注和思考，推动了有关伦理、隐私和安全等问题的讨论与研究。

比尔·盖茨断言：“ChatGPT 表明人工智能的历史意义不亚于 PC 和互联网。”他认为，人工智能将从根本上改变工作、医疗保健和教育领域，预言人工智能未来将彻底改变我们使用计算机的方式并颠覆软件行业，带来计算机领域最大的革命。

从自动驾驶汽车到智能家居，从语音助手到机器翻译，人工智能正在逐步渗透我们生活的方方面面，人类正在进入智能时代。

- 在职场办公领域，人工智能将成为个人工作的得力助手，辅助人们提高绩效和成功机会。尽管在各种不同工作场所 AI 应用不尽相同，但 AI 工具可以显著提升工作效率是毋庸置疑的。
- 在企业商业领域，人工智能将改变企业的经营方式和市场竞争力。通过大数据分析，企业能够更好地理解消费者需求、优化运营流程，并提供个性化的产品和服务。人工智能也为企业带来了更高效的决策和创新的机会，使其能够更好地应对市场的挑战和变化。
- 在科学研究领域，人工智能正成为重要的工具和助手。通过模拟和分析大量的数据，人工智能能够帮助科学家发现新的规律和解决复杂的问题。
- 在医学领域，人工智能已经开始在疾病诊断、药物研发和基因编辑等方面发挥作用，为人类健康带来了新的希望。
- 在教育教学领域，人工智能将改变目前的教育范式和内容。它可以答疑解惑，提供适合学生的个性化学习内容，充当学生的学习助手，可以协助老师进行教学设计和备课、做演示文稿、自动批改学生作业，为师生提供更加便捷、高效的教与学体验。

本书内容丰富，依赖于微软雄厚的技术实力，对人工智能涉及的技术进行了解剖，揭示它背后的原理、算法和技术框架、代码，从人工智能概念、知识表示与专家系统等基本理论，到深度学习、神经网络、计算机视觉等核心技术，从算法到实例，由浅及深，层层递进，具有实操性和指导性，对有志于进入人工智能领域的爱好者和开发者具有很大的助益，是一本不可多得的人工智能入门书。

我与本书译者冯磊老师相识于一个很偶然的机缘。他是一名对创客教育有着强烈兴趣的资深创客。他和

他的团队工作在深圳非常有名的创客空间——万科云城的柴火创客空间。那是一个充满活力和创新精神的地方，为创客们提供了一个理想的交流和实现想法的平台。在一个充满物质欲望的社会中，仍然怀有对教育的热情和理想，让我深为感动。同时，冯磊老师又是一名航拍爱好者，经常纵情于山水之间，在朋友圈发全国各地的航拍美照，让我看到了他灵性洒脱、无拘无束的另一面，深感敬佩。今为冯磊老师翻译的这本书作序，本人感到由衷的高兴，故欣然命笔，是为序。

傅霖
正高级工程师，
深圳大学信息中心主任助理，
教育信息技术研究所副所长

推荐序二

尊敬的读者，欢迎你踏入人工智能的世界。对于大多数人来说，AI 可能还是一个相对神秘且难以捉摸的概念。然而，你手中的这本《机器学习从入门到入行：24 个项目实践 AI》将把陌生的知识转化为触手可及的实践。本书由冯磊领衔的译者团队翻译，旨在为你揭开 AI 的神秘面纱。

AI 无处不在——无论是每天在手机上与朋友视频的应用，还是新兴的自动驾驶汽车。《机器学习从入门到入行：24 个项目实践 AI》旨在帮助你更好地理解 AI，它将复杂的概念和技术简化，将 AI 的理论知识与现实应用完美结合。这本书不仅讲解理论基础，还提供实战体验。对于初学者而言，书中的程序实例将帮助你深入理解 AI 的运作机理。

特别是在编程实战部分，本书提供了众多具体实例，使你有机会亲手实现 AI 算法，并体验 AI 所带来的无限可能。这本书是理论与实践的完美结合，无论你是 AI 领域的专业人士，还是 AI 初学者，都能从中获得丰富的信息，甚至为你的职业生涯指明新方向。

衷心感谢冯磊团队的努力，他们精心翻译和整合了这本书，使我们有更多机会接触和了解 AI。同时，我对每一位读者表示敬意，你们的好奇心使这个世界更加有趣和多元。

《机器学习从入门到入行：24 个项目实践 AI》是 AI 领域理论与实践结合的佳作。通过本书，你将深入理解 AI，并有机会亲身实践。我相信，这本书将成为每个想要了解甚至掌握 AI 的人的宝贵财富。

记住，你正开启一次科技的旅程，勇敢面对未知。期待你在阅读中有所收获和理解，甚至有所改变。

乘风破浪，未来已来。你，准备好了吗？

江大白

AI 自媒体 Up 主，AIHIA 联盟创始人，CV 技术专家

原著作者序

亲爱的读者，欢迎来到人工智能的精彩世界！我真心认为人工智能是一个激动人心的领域。它之所以让人兴奋，不仅因为它是能够自动发挥人类智能和创造力的最后领域，还因为它能帮助我们更好地理解自己、理解我们的思维方式，以及智能的本质。

20 世纪 90 年代初期，我刚开始教授人工智能课程时，主要是从人类身上提取知识，并将其表示成机器可以使用的形式。如今，有了海量的计算资源和互联网信息，我们的重点已转向那些能够从数据中自动学习的技术，即所谓的机器学习，尤其是深度学习——一个处理多层神经网络的分支。这些网络在某种程度上类似于人类大脑的工作方式。正如你所见，人工智能领域在这些年发生了显著变化，因此本课程只是你学习的起点，绝不是终点。请做好准备，随时迎接新知识的挑战，以跟上人工智能的发展步伐！

AI for Beginners 课程是我在微软工作期间最自豪的成就之一。这门课程几乎凝聚了我在各大高校 20 年的教学经验。课程在数学方面进行了简化，以便让没有复杂数学背景的读者也能轻松理解。

自课程发布以来已过去了一段时间，因此本课程中没有包含人工智能领域的一些最新成果，如多模态 Transformer。尽管如此，它依然提供了关于深度神经网络如何组织和训练的整体解读，这将是你在人工智能领域继续探索的良好起点！

我希望大家在阅读这本书时都能感受到乐趣，并像我一样欣赏人工智能的魅力！

德米特里 • 索什尼科夫博士
莫斯科航空技术大学、高等经济学院、莫斯科物理技术学院副教授，
MAILabs 创始人，HSE 设计学院生成式人工智能实验室技术主管，
微软前员工，AI for Beginners 课程主要作者和负责人

Dear readers, welcome to the exciting world of Artificial Intelligence! I really think that AI is exciting, not only because it tackles the final frontier of automating human intelligence and creativity, but also because it helps us better understand ourselves, our own reasoning and nature of intelligence.

When I started teaching AI in the early 1990s, it was mostly about extracting knowledge from human beings and representing it in machine-usable form. Nowadays, with huge computing resources and vast amounts of information on the internet, we mostly focus on techniques that learn automatically from data - so-called machine learning, and specifically deep learning—a branch that deals with neural networks with many layers that somehow resemble the work of our brain. As you can see, the area of AI changes significantly over the years, so this course would be a starting point for you, but by no means the final destination. Be prepared to learn new exciting things constantly to keep up with AI development!

AI for Beginners curriculum is one of the things from my work at Microsoft that I am mostly proud of. It represents almost 20 years of teaching experience in various universities, however, the program is a bit simplified in terms of strict mathematics, to make is understandable by a person with no deep knowledge of linear algebra or optimization theory.

Since it has been a couple of years ago that the course was released, it misses some latest achievements in AI, such as multi-modal transformers. However, it gives you overall understanding of how deep neural networks are organized and trained, which will be a good starting point to continue your journey in the field of AI!

I hope you all have great time reading this book, and enjoy the AI as much as I do!

Dmitry Soshnikov, Ph.D.
Associate Professor at Faculty of Computer Science at HSE / MAI
Technical Lead of Generative AI Laboratory at HSE Design School
ex-Microsoft, Primary Author and Lead of AI for Beginners Curriculum

译者序

在我 2023 年带领团队完成微软物联网入门课程——IoT for Beginners 的翻译并出版《深入浅出 IoT：完整项目实战》之后，我开始寻找一门既简单又系统的人工智能入门课程，以丰富自己在这个迅速发展的领域中的知识。经过一番比较，我最终选择了由德米特里・索什尼科夫博士撰写的微软 AI for Beginners 课程。一方面，这个课程延续了我熟悉的、对初学者极为友好的 IoT for Beginners 课程结构；另一方面，课程内容完全从科普和应用的角度设置。这正符合我这样一名缺乏复杂人工智能数学基础和非编程专业背景的初学者的需求。在德米特里・索什尼科夫博士的引导下，我得以系统地了解人工智能技术的体系结构，并通过实际程序练习，深入理解这些技术的运作原理。

另一个让我有信心完成本书翻译的原因是 ChatGPT 的出现和其快速发展。2022 年 11 月，当 ChatGPT 发布时，我正在修订 IoT for Beginners 的译稿（即《深入浅出 IoT：完整项目通关实践》）。我尝试使用这一 AI 工具解释课程中的多个程序，并借助 AI 解决程序运行中出现的各种问题。到了 2023 年 7 月决定翻译 AI for Beginners 时，ChatGPT 已经经历了多次迭代，变得更加强大，我对 AI 工具的日常工作依赖也日益增加。因此，我决定大胆尝试，在 AI 的协助下完成这本 AI 读物的翻译和出版工作。

为此，我设计了一个工作流程。首先请周慧梅女士将 GitHub 上的 AI for Beginners 英文 markdown 文本，使用 ChatGPT 按预设提示词翻译成中文版本，以保留原文的样式和链接。然后，以初译文为基础，我让 ChatGPT 对照译文和英文原文再次进行修订，同时检查两者之间的差异，选择我认为更合适的结果，或进行进一步修改。这一流程的效率和质量，相比我翻译 IoT for Beginners 时有了显著提升。

AI for Beginners 课程提供了大量的实践程序，均以 Notebook 形式呈现。为方便初学者理解，我请 AI 为所有程序添加了中文注释，这在过去是难以想象的。有了中文注释的程序，初学者能更好地理解程序的运作方式，而不再对着看似天书般的程序发愣。在尝试修改程序时，我也能迅速找到关键点（当然，现在更多时候，我会直接告诉 AI 我的修改需求，并请其直接输出修改后的程序）。

在验证这些程序时，我遇到了一些软件环境方面的问题。由于作者编写这些课程已有近两年时间，许多程序的运行环境发生了变化，导致可能无法正常运行。在 ChatGPT 的帮助下，我几乎不需要依赖任何专业程序员即可解决这些问题。我希望读者在学习本书时，如果遇到程序运行问题，也能像我一样，借助 AI 解决。

翻译过程中，我还遇到了一个难题：作者在介绍 AI 背景知识时，提供了一些插图或程序输出的图形结果，但英文原文未提供详尽解释，起初让我有些困惑。幸运的是，随着 ChatGPT 升级至能够解释图形的版本，这一问题终于得到解决。我只需提供插图或程序输出图的上下文并上传图片，便能获得详尽的解释。

在 AI 的协助下，我花费了大约 5 个月的业余时间（主要是晚上和周末）顺利完成了这本 AI 入门书籍的翻译和修订工作。这一过程不仅是一次自我学习的旅程，也让我对人工智能的知识体系和应用有了全面而深入的认识。同时，我对于人工智能技术的快速发展深感钦佩。它让我们勇于尝试那些过去可能望而却步的学习和任务，让这一过程变得更加充满信心和期待。这次翻译的经历，让我深刻体会到，在接近于通用人工智能的技术面前，我们的学习方法和创造过程正在经历一场深刻的变革。

致谢

周慧梅女士，在初期翻译时，帮助我完成了大量“手工”工作，让我能专注于对内容有效性的评估，并在修订阶段提供了许多积极有效的改进建议。姬宇璐女士在项目前期的协助让我们得以顺利启动这个项目。

孟依卉设计师，为这本书设计了优雅的封面和目录。

清华大学出版社的王中英女士和编辑团队，他们的努力让这本书最终能和广大读者见面。

最后，感谢我的可靠 AI 协作者 ChatGPT，是它帮助我们完成了本书翻译的大部分工作。

冯磊

矽递科技技术支持组负责人

前言

本书内容

欢迎来到《机器学习从入门到入行：24 个项目实践 AI》—— 微软 AI for Beginners 课程的中文版！本课程由微软 Azure 云倡导者团队精心设计，旨在为初学者提供一个全面且易于理解的人工智能入门指南。课程为期 12 周，共 24 节课，涵盖从传统符号人工智能到现代深度学习的广泛主题。在本课程中，你将学习：

（1）**人工智能简史：**介绍人工智能的发展历程。

（2）**符号人工智能：**探讨知识表示与专家系统。

（3）**神经网络简介：**从感知机到多层感知机，再到神经网络框架。

（4）**计算机视觉：**包括卷积神经网络、预训练网络、生成对抗网络（GAN）等。

（5）**自然语言处理（NLP）：**涵盖文本表示、嵌入、语言模型、循环神经网络（RNN）等。

（6）**其他人工智能技术：**如遗传算法、深度强化学习和多智能体系统。

（7）**人工智能的伦理与责任：**讨论人工智能的社会影响和伦理问题。

课程链接编号

英文版课程包含大量较长的链接，不便使用，中文版将绝大部分链接通过链接编号提供，读者可以通过此书的链接列表页面（扫描下面二维码）访问，依据索引编号访问对应的链接。

如何使用本书

存储库

本书配套有存储库，其中提供了中文版的 Jupyter Notebook 文件。这些 Notebook 文件包含了课程中的代码示例、实践练习和理论讲解，帮助读者更好地理解和应用人工智能技术。存储库的地址为 https://gitee.com/mouseart2023/AI-For-Beginners-notebook-ch

运行 Jupyter Notebook 的两种方法

本书包含大量可执行的示例和实践内容，你需要在 Jupyter Notebook 中运行 Python 程序。为了简化操作流程，以下是为中文用户推荐的两种主要方法。

方法一：在本地计算机上运行

（1）安装 Miniconda。Miniconda 是一个轻量级的 Python 发行版，支持创建和管理不同的虚拟环境。

① 下载 Miniconda 安装包：在 Miniconda 的官网选择适合你的操作系统的版本，并下载。

② 根据提示完成安装。

（2）获取中文版课程存储库。使用如下代码

```
git clone https://gitee.com/
mouseart2023/AI-For-Beginners-
notebook-ch.git
```

（3）创建并激活虚拟环境。打开终端或命令提示符，导航到复制的存储库目录，然后创建并激活虚拟环境，代码如下：

```
cd AI-For-Beginners-notebook-ch
conda env create --name ai4beg --file
environment.yml
conda activate ai4beg
```

（4）安装 Visual Studio Code 和 Python 扩展。

① 下载并安装 Visual Studio Code。

② 启动 VS Code，安装官方的 Python 扩展（可以在扩展市场中搜索“Python”并安装由 Microsoft 提供的扩展）。

（5）运行 Jupyter Notebook。

① 在VS Code 中打开 AI-For-Beginners-notebook-ch 文件夹。

② 打开任意一个 .ipynb 文件，VS Code 会自动提示安装所需的依赖项，请按照提示完成安装。

③ 选择刚刚创建的 ai4beg 虚拟环境作为 Python 解释器。

④ 现在，你可以在 VS Code 中直接运行和编辑 Notebook 了。

方法二：使用本地 Jupyter 环境

（1）安装 Miniconda。同方法一中的步骤（1）。

（2）获取中文版课程存储库。同方法一中的步骤（2）。

（3）创建并激活虚拟环境。同方法一中的步骤（3）。

（4）安装 Jupyter Notebook。在激活的虚拟环境中安装 Jupyter Notebook，代码如下：

```
conda install jupyter
```

（5）启动 Jupyter Notebook。在终端或命令提示符中，导航到存储库目录。运行以下命令：

```
jupyter notebook
```

浏览器会自动打开 Jupyter 的界面，你可以在其中打开并运行任意 .ipynb 文件。

推荐使用方法

对于大多数用户，我们推荐方法一：在本地计算机上运行，因为它提供了一个集成的开发环境，便于编写和调试代码。同时，使用 Visual Studio Code 可以获得更好的代码提示和版本控制支持。

自学建议

阅读本书需要一些 Python 编程和线性代数、统计学的基础，本书不展开讲解，网上可以找到丰富的学习资源，有需要的读者可以自行学习。下面是几点学习建议：

- 从课前小测验开始，激发学习兴趣。
- 阅读课程内容，理解理论知识。
- 运行并修改 Notebook 中的代码，进行实践操作。
- 完成课后测验，巩固所学知识。
- 如果课程包含实践内容，尽量完成以加深理解。

注意事项

- 网络访问：确保你的网络能够访问 Gitee 和 GitHub（如果选择从 GitHub 复制）。
- 依赖安装：创建虚拟环境时，environment.yml 文件会自动安装所需的依赖项，请确保你的网络连接稳定。
- 资源需求：某些课程内容可能需要较高的计算资源，建议使用性能较好的计算机。

如果在安装或运行过程中遇到问题，请参考以下资源：

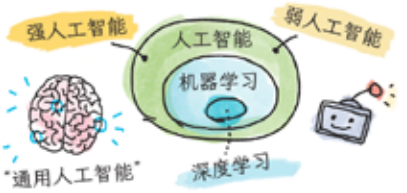
- 课程链接索引：见上面“课程链接编号”部分的二维码。
- 中文社区支持：加入相关技术社区或论坛，寻求更多帮助。

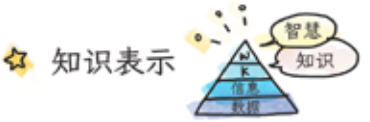
我们希望这些简化的步骤能帮助你顺利开始学习人工智能。祝学习愉快！

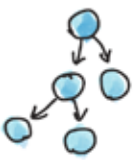
荣誉与贡献

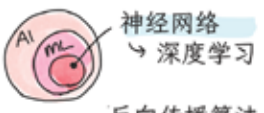
- 主要作者：Dmitry Soshnikov 博士
- 编辑：Jen Looper 博士
- 插画家：Tomomi Imura
- 中文翻译团队：冯磊、周慧梅
- 封面设计：孟依卉
- 中文版式设计：冯磊

目录

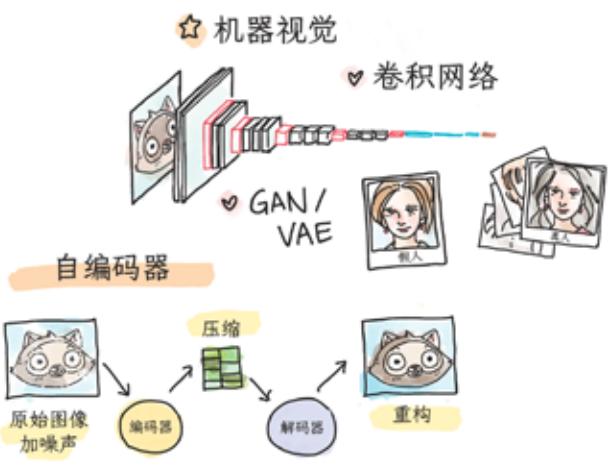


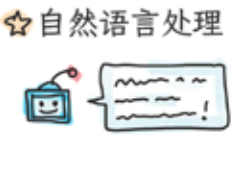


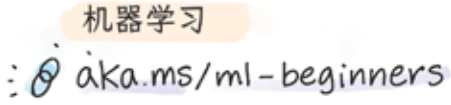




感知
= 单层神经网络 ↔ 多层感知器





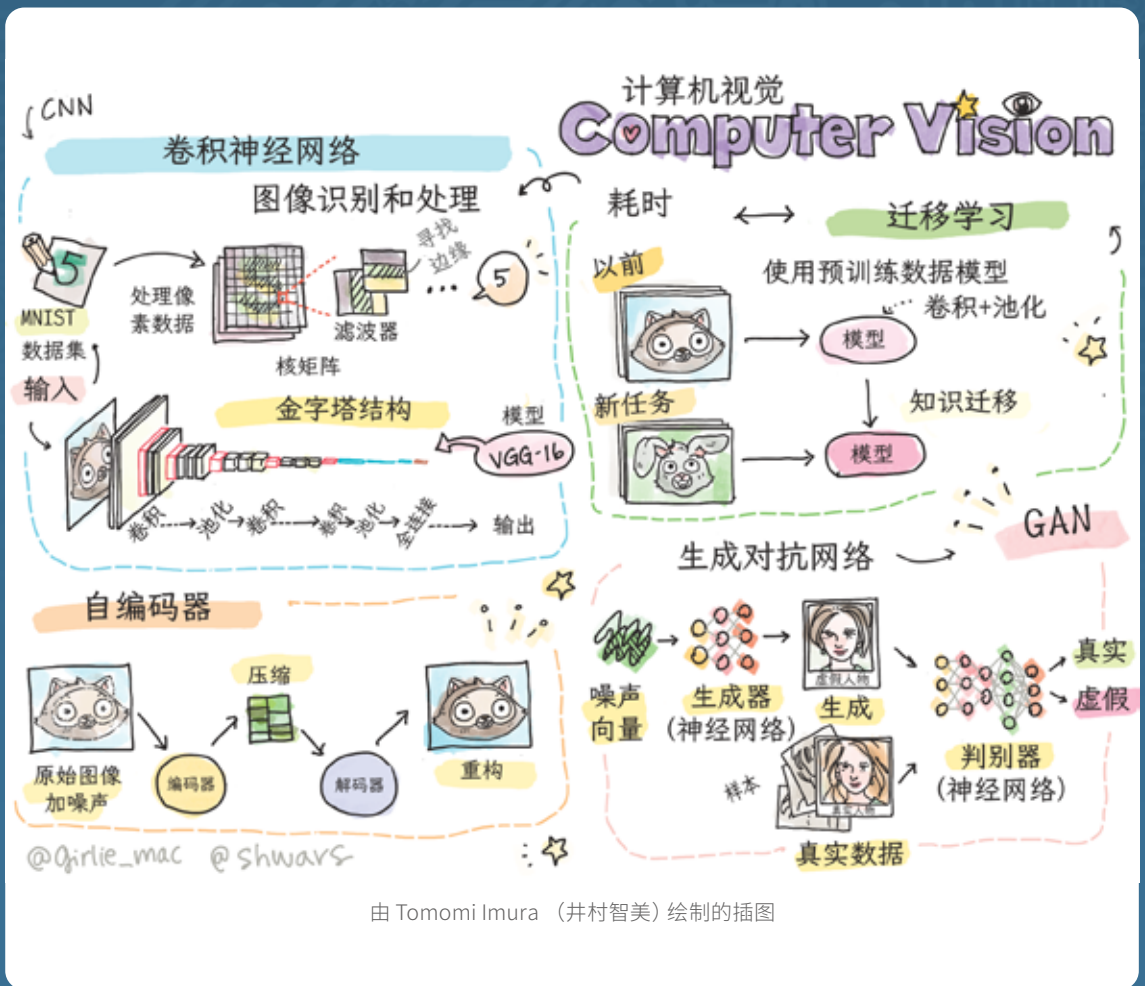


扫码看详细目录

第 3 篇　计算机视觉

在本篇中，将学习：

- 计算机视觉与 OpenCV
- 卷积神经网络
- 预训练网络与迁移学习
- 自编码器
- 生成对抗网络
- 目标检测
- 图像分割



由 Tomomi Imura （井村智美）绘制的插图

第 6 课
计算机视觉与 OpenCV

课前准备

计算机视觉（Computer Vision，CV）[L6–1] 是一个学科，其目标是使计算机能够对数字图像获得高层次的理解。这个定义相当宽泛，因为“理解”一词的含义多样，它可以是在图片中找到一个对象（对象检测）、理解正在发生的事件（事件检测）、用文本描述一张图片或者重建一个三维场景等。此外，还有一些专注于人类图像的特殊任务，如年龄和情感估计、人脸检测和识别及 3D 姿态估计等。

简介

本课将介绍如下内容：

6.1 计算机视觉的基本任务之一：图像分类

6.2 练习——计算机视觉和 OpenCV：OpenCV.zh.ipynb

6.3 结论

6.4 挑战

6.5 复习与自学

6.6 作业——使用光流检测手掌移动：MovementDetection.zh.ipynb

课前小测验

（1）计算机视觉旨在使计算机获得（ ）的高层次理解。

a. 图像
b. 文本
c. 计算机

（2）用于图像处理的 Python 库包括（ ）。

a. OpenCV
b. Pillow
c. a 和 b

（3）在 Python 中,图像不能被表示为 NumPy 数组，这一说法（ ）。

a. 正确
b. 错误

6.1 图像分类：计算机视觉的基础任务

图像分类是计算机视觉中最基本的任务之一。如今，大多数计算机视觉任务都是使用一种名为卷积神经网络的人工智能技术来解决的。在深入学习卷积神经网络之前，需要掌握一些基础知识和工具，这些工具将帮助我们进行图像处理和预处理，为后续的神经网络模型训练打下基础。

6.1.1 图像处理技术

在将图像传递给神经网络之前，通常需要使用一些算法和技术来增强图像。以下是一些常用的 Python 图像处理库：

- **OpenCV** [L6–2]：功能强大的图像处理库，用 C++ 编写，是图像处理领域的业界标准。它提供了方便的 Python 接口。
- **imageio** [L6–3]：用于读取和写入不同图像格式的库，它还支持 ffmpeg[①]，这是一个将视频帧转换为图像的有用工具。

① ffmpeg 是一个免费的、开源的多媒体处理工具，可以用于处理音频、视频和其他多媒体文件。它可以进行格式转换、视频剪辑、音频提取等操作，是一个功能强大的命令行工具，被广泛应用于视频编辑、流媒体等领域。

- **Pillow**（又名 PIL）[L6-4]：功能全面的图像处理库，支持变形、色彩调整等多种操作。
- **dlib** [L6-5]：包含多种机器学习算法的 C++ 库，可用于人脸检测、面部特征点检测等任务。它也提供了 Python 接口。

6.1.2 OpenCV

OpenCV 是公认的图像处理标准库。它包含许多用 C++ 实现的实用算法，也可以在 Python 中调用。

有个学习 OpenCV 的好地方是这个 OpenCV 的入门课程[L6-6]。在课程中，目标不是系统学习 OpenCV，而是通过一些示例让读者初步感受如何使用它。

6.1.3 加载和处理图像

在 Python 中，一般用 NumPy 数组来表示图像。例如，一个 200 像素 ×320 像素的灰度图像可以存储在一个 200×320 的数组中；而彩色图像则是 200×320×3 的数组（这里的“×3”表示 3 个颜色通道：红、绿、蓝）。用下面的程序可以加载一个图像：

```
import cv2  # 导入 OpenCV 库
import matplotlib.pyplot as plt  # 导入 matplotlib 的 pyplot 模块

im = cv2.imread('image.jpeg')  # 使用 OpenCV 读取名为 'image.jpeg' 的图像文件
plt.imshow(im)  # 使用 matplotlib 的 pyplot 模块显示图像
```

在 Python 中处理彩色图像时，需要注意一个重要的区别：OpenCV 默认使用 BGR（蓝绿红）格式来表示彩色图像。而其他大多数 Python 库则使用更常见的 RGB（红绿蓝）格式。

为了让图像在不同的库之间正常显示，需要进行色彩空间的转换。这可以通过 NumPy 数组的维度交换来实现，也可以直接用 OpenCV 提供的 `cv2.cvtColor()` 函数，其程序如下：

```
im = cv2.cvtColor(im, cv2.COLOR_BGR2RGB)  # 将图像从 BGR 格式转换为 RGB 格式
```

`cvtColor()` 函数还可用于执行其他颜色空间转换，如将图像转换为灰度或转换为 HSV（色相、饱和度、明度）色彩空间。

也可以使用 OpenCV 逐帧加载视频——其示例见本课 6.2 节的 OpenCV 练习 Notebook。

6.1.4 图像预处理技巧

在将图像输入神经网络之前，通常需要做一些预处理。OpenCV 提供了很多实用的图像处理功能比如：

（1）调整图像尺寸参考程序如下：

```
im = cv2.resize(im, (320, 200), interpolation=cv2.INTER_LANCZOS)
```

（2）图像模糊参考程序如下：

```
im = cv2.medianBlur(im, 3)  # 中值滤波模糊
im = cv2.GaussianBlur(im, (3,3), 0)  # 高斯模糊
```

（3）调整图像亮度和对比度：可以通过 NumPy 数组操作改变图像的亮度和对比度，具体方法可参考这篇 Stackoverflow 的问答[L6-7]。

（4）图像二值化[L6-8]：用 `cv2.threshold()` 或 `cv2.adaptiveThreshold()` 函数，通常比调亮度和对比度更好用。

（5）图像变换[L6-9]：OpenCV 提供了多种图像变换功能，可以对图像进行各种几何变换。其中最常用的两种变换是仿射变换和透视变换。

① **仿射变换（Affine Transformations）**[L6-10] 是一种保持直线和平行性的线性变换，可以通过一系列的线性变换（如平移、缩放、旋转、翻转、剪切等）的组合来实现。在仿射变换中，原图中的任意 3 个点可以映射到目标图像中的任意位置，同时保持它们之间的直线关系。仿射变换通常用于以下场景：校正图像的几何失真，如摄像头镜头导致的失真；对图像进行旋转、缩放、平移等几何变换；对图像进行剪切变换，改变图像的视角。

② **透视变换（Perspective Transformations）**[L6-11] 也称为投影映射（Projective

Mapping），是一种更常用的二维变换。与仿射变换类似，透视变换也可以通过矩阵乘法来实现。但不同的是，透视变换不要求保持直线和平行性。在透视变换中，原图中的任意四个点可以映射到目标图像中的任意位置。透视变换常用于以下场景：校正图像的透视失真，如将倾斜拍摄的文档调整为正视图；创建图像的透视效果，如将 2D 图像转换为 3D 效果；实现图像的空间变换，如将广告牌嵌入到视频场景中。

(6) 光流分析🔗 **[L6–12]**：用于分析图像中的运动信息。

6.1.5 计算机视觉应用示例

在本课 6.2 节的 OpenCV 练习 Notebook🔗 **[L6–12]** 中，有一些计算机视觉的实际应用案例。

(1) **盲文书籍照片预处理：** 展示了如何用图像二值化、特征检测、透视变换等手段，从书籍照片中分割出每个盲文字符，为后续的字符识别做准备，如图 6-1 所示。

图 6-1 对盲文书籍照片进行预处理，以进行盲文符号分离提取的过程。来自 OpenCV.zh.ipynb 文件中的图像

(2) **帧差法检测视频运动：** 对于固定摄像头拍摄的视频，如果场景中无运动，那么相邻帧之间的差异应该很小。可以用帧差（像素差的绝对值）来检测运动的出现，效果如图 6-2 所示。

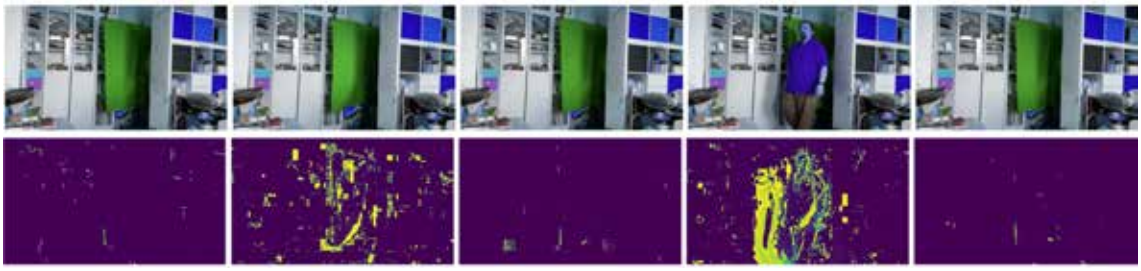


图 6-2 静态场景帧和出现大量运动的动态场景帧，以及对应的帧像素差值图像。来自 OpenCV.zh.ipynb 文件中的图像

(3) **光流法检测运动：** 光流（Optical Flow）可以估计图像中每个像素的运动速度。光流分为如下两种：

- **稠密光流（Dense Optical Flow）：** 为每个像素都计算速度，效果如图 6-3 所示。
- **稀疏光流（Sparse Optical Flow）：** 只为图像中的一些特征点（如角点）计算速度。

图 6-3 视频帧的稠密光流效果图，绿色对应于向左移动，而蓝色对应于向右移动。来自 OpenCV.zh.ipynb 文件中的图像

6.2 ✍️ 练习——计算机视觉与 OpenCV：OpenCV.zh.ipynb

通过探索 OpenCV Notebook🔗 **[L6–12]** 进行一些 OpenCV 实验。

OpenCV 被认为是图像处理的事实标准。OpenCV 包含许多有用的算法，是用 C++ 实现的。也可以从 Python 中调用 OpenCV。

在这些 Notebook 中，将提供使用 OpenCV 的一些示例。有关更多详细信息，可以访问 OpenCV 的入门课程🔗 [L6-6]。

在开始之前，需要在当前环境中安装 OpenCV 的 Python 接口 cv2。

安装 cv2 模块，使用 pip 命令安装，代码如下：

```
pip install opencv-python
```

或使用 conda 命令安装，代码如下：

```
conda install -c conda-forge opencv
```

安装完成后，重新启动 Notebook，再次运行下面的程序即可。

首先，导入 cv2，以及一些其他有用的库，程序如下：

```
# 导入所需库
import cv2
import matplotlib.pyplot as plt
import numpy as np

# 定义一个函数，用于在一行中显示多张图像，并根据需要添加标题
def display_images(l, titles=None, fontsize=12):
    n = len(l)  # 获取图像列表的长度
    fig, ax = plt.subplots(1, n)  # 创建 1 行 n 列的子图
    for i, im in enumerate(l):  # 遍历图像列表
        ax[i].imshow(im)
# 在子图中显示图像
        ax[i].axis('off')  # 关闭坐标轴
        if titles is not None:  # 如果有标题，则添加标题
            ax[i].set_title(titles[i], fontsize=fontsize)
    fig.set_size_inches(fig.get_size_inches() * n)  # 设置图像大小
    plt.tight_layout()
    plt.show() # 显示图像
```

6.2.1 加载图像

在 Python 中，一般用 NumPy 数组来表示图像。例如，一个 200 像素 ×320 像素的灰度图像可以存储在一个 200×320 的数组中；而彩色图像则是 200×320×3 的数组（这里的“×3”表示 3 个颜色通道：红、绿、蓝）。下面的程序从文件中读取图像，打印图像的形状并显示图像：

```
# 从文件中读取图像
im = cv2.imread('data/braille.jpeg')
print(im.shape) # 打印图像的形状
plt.imshow(im) # 显示图像
```

程序输出如下，图像部分如图 6-4 所示。

```
(242, 531, 3)
```

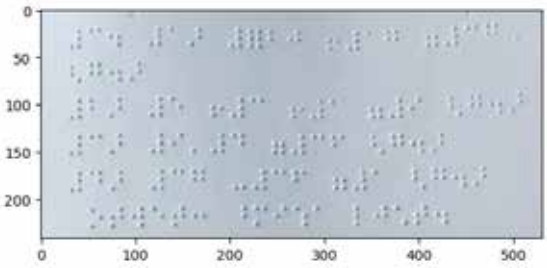


图 6-4　由程序加载的一张盲文图像

由于图像的颜色对我们来说无关紧要，所以，可将其转换为灰度图像，下面是将图像从 BGR 转为灰度并以默认的伪彩色显示的程序：

```
# 将图像从 BGR 转换为灰度
bw_im = cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
print(bw_im.shape) # 打印灰度图像的形状
plt.imshow(bw_im) # 以默认的伪彩色显示灰度图像
```

程序输出如下，图像部分如图 6-5 所示。

```
(242, 531)
```

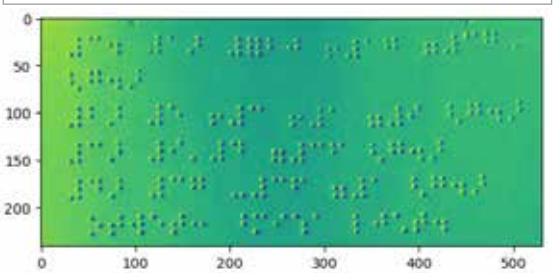


图 6-5　由 BGR 转换为以伪彩色显示的灰度图像

6.2.2 盲文图像处理

如果想应用图像分类来识别文本，就需要切割出单个符号，使其与之前见过的 MNIST （手写数字图像数据集）图像相似。可以使用本书第 11 课将讨论的目标检测技术来完成，也可以尝试使用纯粹的计算机视觉技术来实现。这篇博客文章🔗 [L6–14] 对计算机视觉如何用于字符分割做了很好的阐述，在此仅重点介绍一些用得着的计算机视觉技术。

首先，使用阈值化处理（在这篇 OpenCV 的文章🔗 [L6–8] 中有很好的阐述）稍微增强一下图像，其程序如下：

```
# 使用 3x3 的核对灰度图像进行均值模糊
im = cv2.blur(bw_im,(3,3))

# 应用自适应阈值处理，使用邻域均值计算阈值，并反转二值化的结果
im = cv2.adaptiveThreshold(im, 255,
cv2.ADAPTIVE_THRESH_MEAN_C,
                           cv2.THRESH_
BINARY_INV, 5, 4)

# 对图像进行中值模糊，使用 3x3 的核
im = cv2.medianBlur(im, 3)

# 使用 Otsu 方法确定阈值并应用全局阈值处理
_,im = cv2.threshold(im, 0, 255, cv2.
THRESH_OTSU)

# 对图像进行高斯模糊，使用 3x3 的核
im = cv2.GaussianBlur(im, (3,3), 0)

# 再次使用 Otsu 方法应用全局阈值处理
_,im = cv2.threshold(im, 0, 255, cv2.
THRESH_OTSU)

# 显示处理后的图像
plt.imshow(im)
```

程序输出如图 6-6 所示。

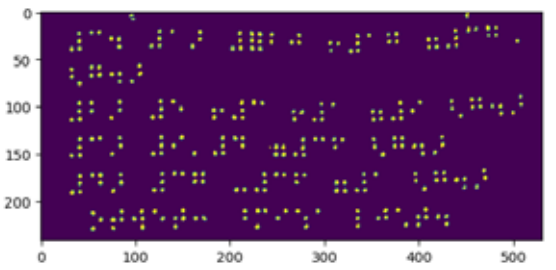


图 6-6　经过阈值化处理的图像

要处理图像，需要“提取”单个点，即将图像转换为单个点的坐标集。可以使用特征提取技术来实现，如 SIFT（Scale-Invariant Feature Transform，尺度不变特征变换）、SURF （Speeded-Up Robust Features，加速稳健特征）或 ORB🔗 [L6–15]（Oriented FAST and Rotated BRIEF，方向性快速和旋转二进制鲁棒独立元素特征）。下面的程序将检测和计算图像中的特征点和描述符，并打印前 5 个特征点的坐标：

```
# 创建一个 ORB 对象，用于检测和计算图像中的特征点和描述符
orb = cv2.ORB_create(5000)
# 检测并计算图像中的特征点和描述符
f,d = orb.detectAndCompute(im,None)

# 打印前 5 个特征点的坐标
print(f"First 5 points: { [f[i].pt for
i in range(5)]}")
```

程序输出如下：

```
First 5 points:
[(307.20001220703125,
40.80000305175781),
(297.6000061035156,
114.00000762939453),
(423.6000061035156,
133.20001220703125),
(242.40000915527344,
144.0), (103.68000793457031,
57.60000228881836)]
```

绘制所有点以验证是否做得正确，其程序如下：

```
# 定义一个函数，用于绘制特征点
def plot_dots(dots):
    img = np.zeros((250,500)) # 创建一张空白图像
    for x in dots:
        cv2.circle(img,(int(x[0]),int
(x[1])),3,(255,0,0)) # 在图像上绘制特征点
    plt.imshow(img)

# 获取特征点的坐标
pts = [x.pt for x in f]

# 绘制特征点
plot_dots(pts)
```

程序输出如图 6-7 所示。

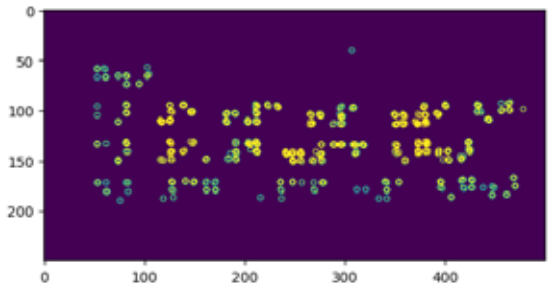


图 6-7　根据提取的特征点重绘后的盲文图像

要分离单个字符，需要知道整个盲文文本的边界框。要找出它，可以只计算最小和最大坐标，其程序如下：

```
# 获取特征点的最小和最大 x,y 坐标值
min_x, min_y, max_x, max_y =
[int(f([z[i] for z in pts])) for f in
(min, max) for i in (0,1)]
min_y += 13 # 微调最小 y 坐标

# 显示裁剪区域
plt.imshow(im[min_y:max_y,min_x:max_
x])
```

程序输出如图 6-8 所示。

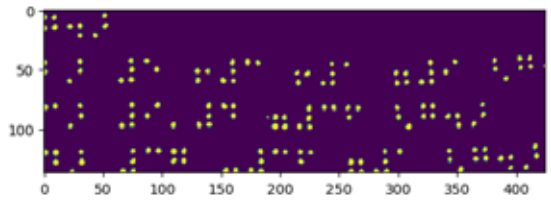


图 6-8　经过裁剪后的重绘图像

此外，此盲文文本可能有轻微的旋转，为了使其完美对齐，需要进行所谓的透视变换。取由点 (x_{min}, y_{min})，(x_{min}, y_{max})，(x_{max}, y_{min})，(x_{max}, y_{max}) 定义的矩形，并将其与具有成比例尺寸的新图像对齐，其程序如下：

```
# 定义源坐标点的偏移量
off = 5

# 定义源坐标点和目标坐标点
src_pts = np.array([(min_x-off,min_
y-off),(min_x-off,max_y+off),
                    (max_x+off,min_
y-off),(max_x+off,max_y+off)])
w = int(max_x-min_x+off*2)
h = int(max_y-min_y+off*2)
dst_pts = np.array([(0,0),(0,h),(w,0),
(w,h)])
# 使用 findHomography 函数找到单应性矩阵
ho, m = cv2.findHomography(src_pts,
dst_pts)
# 使用 warpPerspective 函数进行透视变换，
裁剪图像
trim = cv2.warpPerspective(im, ho, (w,
h))
plt.imshow(trim)
```

程序输出如图 6-9 所示。

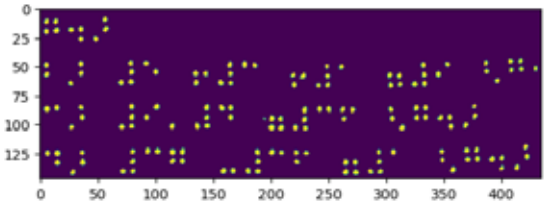


图 6-9　经过透视变换并裁剪后的重绘图像

得到这个对齐良好的图像之后，将其切成片段就相对容易了，切片程序如下：

```
# 定义字符的高度和宽度
char_h = 36 # 设置字符高度为 36 像素
char_w = 24 # 设置字符宽度为 24 像素

def slice(img): # 定义一个名为 slice 的
函数，用于将图像切割成字符
    dy, dx = img.shape # 获取图像的高度
和宽度
    y = 0 # 初始化垂直坐标 y
    while y + char_h < dy: # 当 y 加上字
符的高度小于图像的高度时，继续循环
        x = 0 # 初始化水平坐标 x
        while x + char_w < dx: # 当 x
加上字符的宽度小于图像的宽度时，继续循环
            # 判断当前区域是否为空白
            if np.max(img[y:y+char_h,
x:x+char_w]) > 0:
                # 使用 yield 关键字返回
当前字符的图像区域
                yield img[y:y+char_h,
x:x+char_w]
            x += char_w # 更新水平坐标，
使其向右移动一个字符的宽度
        y += char_h # 更新垂直坐标，使
其向下移动一个字符的高度

# 对图像进行切片
sliced = list(slice(trim))
```

```
# 使用之前定义的 display_images 函数显示切片
display_images(sliced)
```

程序输出如图 6-10 所示。

图 6-10　最终经过切片后盲文图像
（这里只展示了部分输出结果）

可以看到，许多任务可以通过纯粹的图像处理来完成，而无须依赖人工智能。实际上，如果能用计算机视觉的方法简化神经网络的任务，就应该这么做。这样做的好处是，可以用更少的训练数据来解决问题。

6.2.3　使用帧差异进行运动检测

在视频流中检测运动是一项非常常见的任务。例如，当监控摄像头捕捉到某些事件时，我们希望收到警报。如果想了解摄像头上发生了什么，可以使用神经网络来分析视频内容，但仅在确认摄像头确实捕捉到某些动作时才启用神经网络，这样可以降低成本。

运动检测的基本原理如下：如果摄像头是固定的，那么从摄像头获得的连续帧应该彼此非常相似。因为视频帧是以数组的形式存在的，所以，只需将两个连续的帧相减，就能计算出像素间的差异。对于没有运动的静态场景，这种差异通常很小；如果画面中出现了显著的运动，这种差异就会增大。

接下来将学习如何打开视频文件，并把它转换为一连串的帧。打开视频文件并读取帧的程序如下：

```
vid = cv2.VideoCapture('data/motionvideo.mp4') # 打开视频文件

c = 0 # 初始化帧计数器
frames = [] # 初始化帧列表
while vid.isOpened(): # 当视频文件打开时
    ret, frame = vid.read() # 读取一帧
    if not ret: # 如果读取失败
        break # 退出循环
    frames.append(frame) # 将帧添加到帧列表
    c += 1 # 增加帧计数器
vid.release() # 释放视频资源
print(f"Total frames: {c}")
display_images(frames[::150]) # 显示每隔 150 帧的图像
```

程序输出如下，图像部分如图 6-11 所示。

```
Total frames: 876
```

图 6-11　输出视频文件中每隔 150 帧的图像

由于颜色对于运动检测并不重要，所以，可以将所有帧转换为灰度图像，然后计算帧差异，并绘制它们的范数（范数是用于衡量矩阵或向量大小的一种数学工具）以直观地看到正在发生的活动量，其程序如下：

```
bwframes = [cv2.cvtColor(x,cv2.COLOR_BGR2GRAY) for x in frames] # 将帧转换为灰度图像
diffs = [(p2-p1) for p1,p2 in zip(bwframes[:-1],bwframes[1:])] # 计算相邻帧之间的差异
diff_amps = np.array([np.linalg.norm(x) for x in diffs]) # 计算差异的幅度
plt.plot(diff_amps) # 绘制差异幅度图
display_images(diffs[::150],titles=diff_amps[::150]) # 显示差异图像和对应的幅度
```

程序输出如图 6-12 所示。

假设想创建一个报告，当有运动事情发生时就显示合适的图像来展示摄像头前发生了什么。为了实现这一点，就需要找出“事件”的开始和结束帧，并显示中间帧。为了消除一些瞬时出现的非事件干扰动作，可以使用移动平均函数平滑图 6-12 所示的曲线，其程序如下：

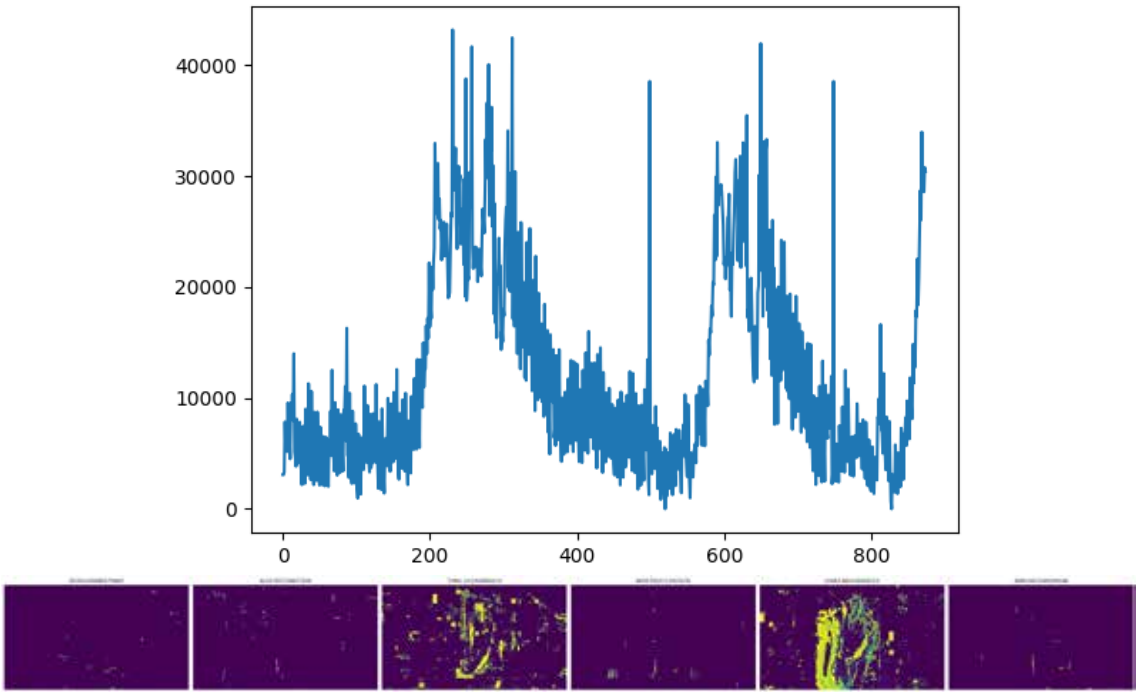


图 6-12　程序输出的差异幅度图和每隔 150 帧的差异图像

```
def moving_average(x, w):
    return np.convolve(x, np.ones(w),
'valid') / w # 定义移动平均函数
threshold = 13000 # 设置阈值

plt.plot(moving_average(diff_amps,10))
# 绘制移动平均图
plt.axhline(y=threshold, color='r',
linestyle='-') # 在图上画出阈值线
```

程序输出如图 6-13 所示。

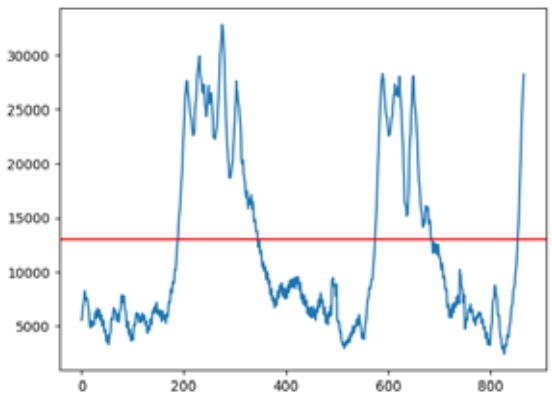


图 6-13　使用移动平均函数平滑后的帧差异幅度曲线图和阈值线（中间的水平线）

现在可以使用 `np.where` 找出更改量高于阈值的帧，并提取连续帧序列，且该序列长于 30 帧，其程序如下：

```
active_frames = np.where(diff_
amps>threshold)[0] # 找到超过阈值的帧
def subsequence(seq,min_length=30):
    ss = [] # 初始化子序列列表
    for i,x in enumerate(seq[:-1]):
        ss.append(x)
        if x+1 != seq[i+1]: # 如果序列
不连续
            if len(ss)>min_length: #
如果子序列长度超过最小长度
                return ss # 返回子序列
            ss.clear() # 清空子序列列表

sub = subsequence(active_frames) # 使
用之前定义的 subsequence 函数从活动帧中
找到第一个长时间的连续运动段
print(sub) # 打印连续运动段的帧索引
```

程序输出如下：

```
[195, 196, 197, 198, 199, 200, 201,
202, 203, 204, 205, 206, 207, 208, 209,
210, 211, 212, 213, 214, 215, 216, 217,
```

```
218, 219, 220, 221, 222, 223, 224, 225,
226, 227, 228, 229, 230, 231, 232, 233,
234, 235, 236, 237, 238, 239, 240, 241,
242, 243, 244, 245, 246, 247, 248, 249,
250, 251, 252, 253, 254, 255, 256, 257,
258, 259, 260, 261, 262, 263, 264, 265,
266, 267, 268, 269, 270, 271, 272, 273,
274, 275, 276, 277, 278, 279, 280, 281,
282, 283, 284, 285, 286, 287, 288, 289,
290, 291, 292, 293, 294, 295, 296, 297,
298, 299, 300, 301, 302, 303, 304, 305,
306, 307, 308, 309, 310, 311, 312, 313,
314, 315, 316, 317, 318, 319, 320, 321,
322]
```

最后，可以显示图像，其程序如下：

```
# 显示连续运动段的中间帧的图像
plt.imshow(frames[(sub[0]+s
ub[-1])//2])
```

程序输出如图 6-14 所示。

图 6-14　程序以 BGR 色彩空间加载的图像色彩看上去很诡异

图 6-14 中的颜色看起来很诡异！这是由于历史原因，OpenCV 以 BGR 色彩空间加载图像，而 matplotlib 使用更传统的 RGB 色彩空间。大多数情况下，在加载图像后需要立即将其转换为 RGB 才能正常显示。将图像从 BGR 色彩空间转换为 RGB 色彩空间的程序如下：

```
# 将同一帧从 BGR 色彩空间转换为 RGB 色彩空
间，并显示
plt.imshow(cv2.
cvtColor(frames[(sub[0]+sub[-
1])//2],cv2.COLOR_BGR2RGB))
```

程序输出如图 6-15 所示。

图 6-15　程序将 BGR 色彩空间的图像转换为 RGB 色彩空间后，图像的颜色正常了

6.2.4　使用光流提取运动

虽然比较两个连续帧可以看到变化的量，但它并未提供关于什么在移动及在哪里移动的信息。为了获得这些信息，可以使用一种称为光流 🔗 [L6–12] 的技术。

- 稠密光流：为每个像素都计算速度。
- 稀疏光流：只为图像中的一些特征点（如角点）计算速度。

在这个很棒的教程 🔗 [L6–16] 可以了解有关光流的更多信息。

在帧之间计算稠密光流，程序如下：

```
# 计算连续帧之间的光流。使用 Farneback 算
法
flows = [cv2.
calcOpticalFlowFarneback(f1, f2, None,
0.5, 3, 15, 3, 5, 1.2, 0)
         for f1,f2 in zip(bwframes[:-
1],bwframes[1:])]
# 打印第一个光流字段的形状
flows[0].shape
```

程序输出如下：

```
(180, 320, 2)
```

对于每一帧来说，光流有着与帧相同的维度，并且包含两个通道，分别对应光流向量的 x 分量和 y 分量。

在二维空间展示光流确实有些难度，但可以采用一个巧妙的方法。如果将光流转换为极坐标形式，那么对于每个像素点，就能得到两个信息：方向和强度。强度可以通过像素的亮度来表示，而方向则

可以用不同的颜色来展示。我们会在 HSV（色调 - 饱和度 - 亮度）色彩空间 🔗 [L6–17] 中创建这样的图像，在这里色调代表方向，亮度代表强度，饱和度固定为 255。

下面这段程序的主要作用是将连续的运动段（光流）转换为易于人眼观察的 HSV 颜色空间图像，并显示这些图像的子集。

```
# 定义一个函数，将光流转换为 HSV 颜色空间的图像，以便于可视化
def flow_to_hsv(flow):
    # 创建一个与输入光流相同形状的零图像，用于存储 HSV 色彩空间的图像
    hsvImg = np.zeros((flow.shape[0],flow.shape[1],3),dtype=np.uint8)

    # 使用 cartToPolar 计算光流的幅值和角度
    mag, ang = cv2.cartToPolar(flow[..., 0], flow[..., 1])

    # 将角度转换为 0 ～ 180 的值，并存储在 HSV 图像的色相通道中
    hsvImg[..., 0] = 0.5 * ang * 180 / np.pi
    # 将饱和度设置为 255
    hsvImg[..., 1] = 255
    # 将幅度归一化到 0 ～ 255，并存储在 HSV 图像的亮度通道中
    hsvImg[..., 2] = cv2.normalize(mag, None, 0, 255, cv2.NORM_MINMAX)

    # 将 HSV 图像转换为 BGR 色彩空间以便显示
    return cv2.cvtColor(hsvImg, cv2.COLOR_HSV2BGR)
# 获取连续运动段的开始和结束帧，并打印
start = sub[0]
stop = sub[-1]
print(start,stop)
# 将选定的连续运动段的光流转换为 HSV 图像
frms = [flow_to_hsv(x) for x in flows[start:stop]]
# 显示转换后的 HSV 图像的子集
display_images(frms[::25])
```

程序输出如下，图像输出如图 6-16 所示。

```
195 322
```

在图 6-16 展示的帧中，绿色代表向左移动，蓝色代表向右移动。

光流是分析运动总体方向的极好工具。如果观察到一帧中的所有像素都大致朝着同一个方向移动，那么可以推断出摄像头正在移动，并且可以尝试进行相应的补偿。

6.3 结论

尽管一些相对复杂的任务（如运动检测或指尖检测）可以仅通过计算机视觉技术解决，但掌握计算机视觉的基本技术和了解诸如 OpenCV 之类的库的功能，对于该领域的深入学习和应用至关重要。

6.4 🚀 挑战

观看这段视频🔗 [L6–18]，了解 Cortic Tigers 项目。了解他们是如何通过机器人构建基于模块化的解决方案，从而使计算机视觉任务更加普及。同时，探索其他类似项目，这些项目有助于帮助新入门的学习者融入这一领域。

6.5 复习与自学

在这个很棒的教程 🔗 [L6–16] 中可以深入了解有关光流的更多信息。

图 6-16　视频帧的稠密光流效果图，绿色对应向左移动，蓝色对应向右移动