垂直领域大语言模型实战------以古籍信息智能化为例

目录

第一章　大语言模型基础原理与垂直领域应用

章节内容：该节介绍语言模型的发展简史和大语言模型构建的基础原理。并介绍大语言模型在三个垂直领域的应用场景。

教学目标：1.使学生初步了解大语言模型的基础原理以及垂直领域的概念。
2.了解大语言模型在古籍处理领域有哪些具体的应用场景。

1.1 语言模型简介

语言模型（Language Model，简称 LM）是自然语言处理（NLP）领域的核心技术，其主要任务是对自然语言序列进行概率建模。具体来说，语言模型就是用来计算一个句子或者一段文本在某种语言中出现的概率。这个概率可以帮助我们理解和预测自然语言的生成规律，从而在各种 NLP 任务中发挥关键作用。例如在实体识别任务中，为从文本中识别出有特定意义的实体词，如人名、地名等，语言模型会根据上下文为每个字分配一个概率，通过概率信息判断该字是否可能是实体词的一部分。在机器翻译任务上，为将文本从一种语言翻译成另一种语言，语言模型能够结合上下文，帮助预测翻译后句子的合理性，通过概率选择最符合目标语言的翻译内容。在文本生成任务中，要求语言模型能够根据给定的提示生成连贯的文本，那么语言模型可以根据当前文本生成下一个最有可能出现的字或词。

1.2 语言模型发展史

语言模型的发展史大概有三个关键阶段，分别是统计语言模型、神经网络语言模型、预训练语言模型。每个阶段的模型都是对前一阶段模型的改进和优化，使得语言模型的性能一

直在不断提升。

第一个阶段是统计语言模型。统计语言模型主要利用统计学的方法，认为一个字符的出现概率依赖于前面的若干字符，通过计算字符序列的概率分布来预测下一个字符出现的可能性，是早期自然语言处理中的重要工具。此阶段的经典模型有 N-gram（N 元模型）与 HMM（隐马尔可夫模型）等。

N-gram 模型的基本假设为某一个词语出现的概率只有前面的 n-1 个词语所决定，即当 n=3 时，词语出现的概率受前面的两个词语影响，也被称为三元模型。N-gram 模型的优势在于简单高效，模型易于实现，对计算资源要求低；可解释性强，计算字符的概率分布，模型结果相对直观且易于理解。其缺点在于缺乏长距离依赖，仅能考虑到有限范围内的上下文；缺乏语义理解，模型仅基于词频计算概率，无法理解一词多义、多词一义等复杂语义和语法结构。HMM 模型用来描述一个含有隐含未知状态的马尔可夫过程，可以通过观测序列来推测隐藏状态序列及其概率分布，主要用于序列标注任务，例如命名实体识别和词性标注等。HMM 模型的优势在于结构相对简单，可解释性强，可以有效捕捉数据间时序关系；其劣势在于难以处理长依赖关系，难以处理高维度数据。

第二个阶段是神经网络语言模型。随着深度学习技术的兴起，语言模型的发展进入神经网络语言模型阶段。神经网络通过模拟人类大脑神经元之间的连接来理解复杂的语言结构和语义信息，显著提升了语言模型的性能，与上一阶段的统计语言模型相比，神经网络语言模型能更好地处理长距离依赖关系。此阶段的经典模型有 Word2Vec 和 CNN（卷积神经网络）等。

Word2Vec 是一种基于神经网络的词嵌入模型，能够将文本内容映射到向量空间中，并通过词向量捕捉词汇间的语义关系，语义相近的词语在向量空间中的距离更近。其优势有考虑到上下文信息，能够实现有效的语义信息捕捉；训练速度相对较快等。缺点在于忽略了词序信息，不考虑文本中上下文的词语顺序；缺乏解释性，虽然词向量能够准确捕捉词间关系，但向量本身难以解释。CNN 模型的核心思想是通过卷积提取局部特征，使用池化层对卷积层提取的特征进行降维，在全连接层整合特征，用于输出预测结果。其优势在于能够提取局部特征；通过多层卷积和池化操作，能够学习到低级到高级的层次化特征。其缺点包括模型解释性较差；需要相对较大的样本训练以达到较好性能。

第三个阶段是预训练语言模型。这一阶段的模型首先在大规模的文本数据集上进行预训练（Pre-trained），学习通用的特征和知识，再通过微调（Fine-tuning）使模型能够适应不同的下游任务。这一阶段中，Transformer 架构成为主流，并利用自注意力机制捕捉文本中的长距离依赖关系，使其能够更加精准的理解文本语义，显著提升了模型在文本处理任务中的性能。以 BERT 为代表的预训练模型宣告了第三阶段的开始，而 GPT 系列模型的出现将该阶段推向新的高潮。

BERT 模型使用大规模文本进行预训练，利用上下文双向信息来构建每个词的语义表示。在预训练阶段利用两个任务捕捉文本信息：MLM（Masked Language Modeling，掩码语言模型）任务随机遮盖输入文本中的部分字符，训练模型根据上下文预测遮盖内容的能力；NSP（Next Sentence Prediction，下一句预测）任务用于判断两个句子在文本中是否相邻，训练模型学习句间关系的能力。BERT 模型的优势在于双向编码能力能够更全面的理解语义信息；泛化能力强，通过大规模文本的预训练学习到丰富的通用知识；结合微调能够专注于不同类型的下游任务，显著提升了模型在文本分类、命名实体识别等多种自然语言处理任务中的性能。GPT 系列模型是由 OpenAI 开发的一系列生成式预训练语言模型，其通过自回归语言建模进行预训练，根据已经生成的内容预测下一个字符，从而学习文本生成能力。其中最著名的 ChatGPT 模型是基于 GPT 系列模型的对话系统，通过强化学习和人类反馈进行优化，能

够生成连续且符合人类偏好的对话内容，其优势在于交互性强，具有强大的文本生成能力，能够兼顾对话上下文，生成连贯自然的文本。

1.3 大语言模型技术原理

大语言模型（Large Language Models, LLMs）是自然语言处理领域的前沿技术，具有强大的文本理解和生成能力。大语言模型通常基于 Transformer 架构，该架构由 Google 在论文 Attentions is All you need 中提出的模型, 其使用 Self-Attention 结构取代了在 NLP 任务中常用的 RNN 网络结构。Transformer 的核心在于其自注意力机制，能够有效处理长距离依赖关系，从而捕捉语言中的复杂特征。与传统的循环神经网络（RNN）相比，Transformer 可以并行处理数据，大幅提高了训练效率和效果。自注意力机制允许模型在处理每个单词时考虑整个输入序列。这种机制使得模型能够理解上下文信息，从而生成更为连贯和准确的文本。例如，在生成句子时，模型可以利用之前的单词来预测下一个单词。大语言模型在多个领域都有广泛应用，例如在文本生成方面，能够撰写文章、编写代码或创作诗歌，为内容创作者提供便利。在问答系统方面，理解并回答用户问题，通过分析关键词和上下文信息生成准确答案，广泛应用于教育和医疗等领域。大语言模型作为自然语言处理领域的重要技术之一，不仅在理论上具有深厚的基础，其实际应用也正在改变我们的生活方式。通过深入了解其技术原理和应用场景，我们可以更好地利用这一技术为人类服务。

1.3.1 预训练技术

预训练技术（Pretraining）是当前自然语言处理（NLP）领域中语言模型（如 GPT 系列、BERT、T5 等）核心技术之一。其基本思想是利用大量无标签的文本数据，首先进行预训练，让模型从海量数据中学习到语言的通用知识。大语言模型的预训练方法旨在解决语言建模的任务，即如何让机器理解并生成自然语言。传统的机器学习方法通常依赖大量的人工标注数据，但标注数据的获取成本高且限制性强。因此，预训练通过对无标签的庞大语料库进行学习，让模型自主地获取语言中的基础知识和结构信息。预训练模型主要包括语言建模能力和表征学习能力两个方面。语言建模能力是学习语言的统计特征，例如词语的共现规律、上下文关系、语法结构等。表征学习能力是学会如何有效地表示单词、短语、句子等语言单位的特征，以便用于后续任务。

1.3.1.1 自回归模型（Autoregressive Models）

自回归模型是最早的预训练模型之一，代表性模型如 GPT 系列。它通过预测序列中的下一个单词来进行训练。具体而言，给定一个文本序列，模型会基于当前上下文（前面的单词）预测下一个单词。该模型的训练目标是最大化模型生成当前单词的概率，具体训练过程为：

$$P(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} P(w_i | w_1, w_2, \ldots, w_{i-1}) \tag{1-1}$$

其中，$w_1, w_2, \ldots, w_n$是一个序列中的单词。通过这种方式，模型不断调整权重以提升语言生成的质量。

1.3.1.2 自编码模型（Autoencoding Models）

自编码模型 (如 BERT) 采用了填空任务的形式进行预训练。它通过随机遮掩 (masking) 输入序列中的一些单词，要求模型根据上下文预测这些被遮掩的词。BERT 的预训练任务包括两个主要部分：（1）Masked Language Modeling（MLM）随机遮掩输入序列中的一些单词，模型学习通过上下文信息来预测这些单词。（2）Next Sentence Prediction（NSP）模型判断给定的两个句子是否在原文中连续出现。这种训练方式让模型能够同时捕捉局部的语义信息（通过填空任务）以及全局的句子级信息（通过下一个句子的预测任务）。

1.3.1.3 预训练的关键步骤

预训练大语言模型需要准备大规模无标签文本数据。（1）数据收集：从互联网、书籍、新闻等多种来源收集文本数据。（2）数据清洗：去除有害内容及冗余信息，以确保数据质量。(3) 分词处理: 将文本转换为 token 形式, 常用的方法包括 WordPiece 和 Byte Pair Encoding (BPE)。（4）批量切分：将处理后的数据分成小批次，以便于高效训练。（5）模型训练：在预训练阶段，模型通过大量的无标签文本进行自监督学习，优化模型参数。具体训练方法包括最大化似然估计（MLE）、最小化交叉熵损失等。训练过程通常需要大量的计算资源和时间。（6）模型评估与微调：预训练阶段结束后，模型通常会在多个任务上进行评估，确保其具备较强的语言理解能力。然后，通过微调阶段，模型会针对特定任务 (如文本分类、问答系统等) 进行调整，进一步提升在特定任务上的表现。

预训练技术是当今大语言模型的核心之一, 它通过在海量无标签文本上进行自监督学习, 让模型能够在多种自然语言处理任务中表现出色。自回归模型（如 GPT）、自编码模型（如 BERT）、序列到序列模型（如 T5）等不同架构在预训练中各有特点，能够适应不同的应用需求。大语言模型的预训练主要通过多种自监督学习任务来实现，包括语言建模、掩码语言模型、下一句预测、去噪自编码等。这些方法使得模型能够从海量无标签文本中学习到丰富的语言知识和结构，为后续微调和特定任务提供了坚实基础。

1.3.2 指令微调技术

指令微调（Instruction Tuning）是一种通过为大语言模型提供指令（或任务描述）来指导模型行为的微调技术。这一技术已经成为当前大规模语言模型优化的重要手段之一，尤其是在像 GPT-3 和 GPT-4 等模型中，指令微调被用来提升模型在多任务、复杂任务和 zero-shot 任务中的表现。指令微调主要解决的问题是使得模型能够根据用户给出的自然语言指令（任务描述）进行灵活的响应，而无需特定任务的标签数据。在这一过程中，模型学习如何根据给定的指令生成相应的输出。这种方法弥补了传统语言模型在处理特定任务时的不足，使得模型能够更好地理解和遵循人类的指令。

在传统的微调（Fine-Tuning）中，大语言模型通常会根据特定任务的标注数据来进行训练，以便调整模型参数并提高其在该任务上的表现。然而，这种方法依赖大量任务特定的数

据，并且训练时间较长。指令微调通过引入自然语言指令，试图让模型能够“理解”任务意图并自主适应不同的任务。与传统微调的目标不同，指令微调的目的是训练模型如何基于简单的语言指令完成多种任务。指令微调的核心思想是通过优化模型在不同任务上的反应，使得大语言模型能根据自然语言指令灵活响应各种任务。

1.3.2.1 构建数据集

指令微调的数据集由人类编写的指令和期望输出组成，强调模型对指令的理解能力。指令微调的关键步骤之一是构建一个包含自然语言指令和相应任务结果的数据集。数据集中的每个例子通常包括两部分：（1）任务的自然语言描述。例如：“请为以下文本生成摘要”。（2）模型基于指令应该产生的输出。例如：“这段文本的摘要是...”。这些指令可以是针对各种任务的描述，例如文本分类、问答、情感分析、翻译等。数据集可以通过人工创建、从现有任务中提取或者利用生成模型生成。

1.3.2.2 微调过程

在指令微调中，原始的大语言模型首先在标准的预训练语料上进行训练，学习通用语言知识和语法结构。然后，模型在指令-响应数据集上进行微调，目标是使模型学习如何根据给定的指令进行任务响应。优化的目标是最大化模型生成正确响应的概率，通过这样的训练，模型逐渐能够理解不同任务的自然语言指令，并生成合适的输出。

1.3.2.3 任务描述的多样性

指令微调的另一个关键点是任务描述的多样性。不同于传统的单一任务训练，指令微调通过引入多样化的任务指令让模型学习如何处理各种不同类型的任务。例如：回答问题、文本生成、翻译、情感分析等任务。通过多种任务指令的训练，模型能够更好地适应不同的应用场景，提升泛化能力。

1.3.2.4 指令微调的技术细节

（1）为了确保模型能够正确理解并执行任务，指令的定义和规范化非常重要。指令通常是简短的自然语言句子，表达任务的核心需求。例如：请根据以下内容生成摘要。指令应当简洁、明确，以减少理解偏差，规范化有助于统一模型的输出格式，从而提高模型在不同任务中的一致性和准确性。（2）对话式任务和推理任务：一些指令微调不仅限于单一的任务，还扩展到对话式任务或推理任务。例如，模型可能需要根据复杂的指令做出推理或逐步解答。这样的任务要求模型能够理解上下文，并按照指令进行详细的推理过程。（3）强化学习与指令微调结合：在某些高级应用中，指令微调与强化学习（Reinforcement Learning, RL）结合，形成所谓的 RLHF（Reinforcement Learning from Human Feedback） 模型。这种方法通过利用人类反馈来进一步微调模型，优化模型的行为。例如，GPT-3 在指令微调的基础上加入了强化学习，以便模型不仅生成符合要求的答案，还能够根据用户的反馈进行调整。

指令微调是一种有效的大语言模型训练方法，通过让模型理解和遵循人类指令来增强其能力和可控性。该技术利用结构化的数据集，使得预训练模型能够更好地适应多样化的下游任务。

指令微调技术通过自然语言指令引导大语言模型的行为，解决了传统微调在处理多任务时的局限性。它使得模型能够根据不同任务的指令自适应地进行处理，显著提高了大语言模型的泛化能力和 zero-shot 学习能力。指令微调不仅推动了大规模预训练语言模型的发展，也为多任务学习和人工智能的应用提供了更加灵活和有效的解决方案。然而，指令设计、数据偏差以及长期推理等挑战依然需要进一步探索和解决。

1.3.3 人类对齐技术

人类对齐技术（Human Alignment）是近年来在大语言模型（LLM）研究中越来越重要的一个领域，它主要关注如何确保大语言模型的行为与人类的价值观、意图和期望一致。简单来说，人类对齐技术旨在通过调整模型的行为，使其输出符合人类的价值观、文化和社会规范。这一过程通常涉及对模型进行微调，以便它能够理解并响应人类指令，同时避免生成有害或不适宜的内容。大语言模型虽然在语言生成、推理和各种任务中表现出色，但它们的行为并不总是符合人类的道德或实用标准，因此人类对齐技术的提出就是为了弥补这一缺陷。随着大型语言模型在各个领域的应用，确保其输出的安全性和道德性变得至关重要。无论是在教育、医疗还是社交媒体等场景中，模型生成的内容都可能影响用户的决策和情感，因此需要确保这些内容是无害、有用且诚实的。因此，确保大语言模型的行为与人类的价值观对齐，成为了人工智能安全、伦理学和技术发展的一个重要方向。具体来说，人类对齐技术旨在从伦理对齐、任务对齐、公平性和透明性等方面使模型应该理解和正确执行人类意图，避免错误或不相关的行为以及确保模型的行为不会对特定群体造成不公平的影响，且其决策过程应具有可解释性。

奖励模型（Reward Model）用于评估生成文本的质量，并为优化过程提供反馈。使用人类反馈数据来训练奖励模型，使其能够根据生成文本与人类偏好的匹配程度给予评分。强化学习（Reinforcement Learning）强化学习是实现人类对齐的重要技术之一，RLHF（Reinforcement Learning from Human Feedback）是目前最常见的对齐方法之一。通过在预训练模型的基础上进行强化学习训练，利用人类反馈来微调模型的行为，以保证其在特定任务中的表现符合人类期望。在 RLHF 中，人类评估员会评定模型生成的输出，给出奖励信号，或者明确指出哪些输出符合预期，哪些不符合。RLHF 是一个迭代过程，模型通过持续的人类反馈不断改进。每次迭代都会使模型的行为更加对齐人类的预期。通过微调预训练的大语言模型，可以进一步增强其与人类价值观的一致性，包括参数微调和对抗训练。其旨在使用标注的数据集对模型进行再训练，以使其在特定任务上表现更好以及引入对抗样本以提高模型的鲁棒性，确保其能够抵御潜在攻击并保持输出的一致性。

人类对齐技术是确保大型语言模型输出符合人类价值观的重要手段，通过数据收集、奖励模型构建、强化学习和参数微调等方法，使得模型能够更好地理解并执行人类指令。然而，由于伦理、价值观的多样性和技术实施的复杂性，如何完美地实现人类对齐仍然是一个挑战。随着技术的发展，未来大语言模型将能更好地理解并遵循人类的期望，在实际应用中展现出更强的社会适应能力。

1.3.4 工具使用与智能体

在过去的几年里，随着大语言模型（如 GPT-3 和 GPT-4）的发展，研究者发现，通过使模型能够与工具互动，它们在处理一些复杂、需要外部数据和功能的任务时，能够显著提高其性能。这种能力不仅推动了机器学习和自然语言处理技术的进步，也为应用场景提供了

更多的可能性，如自动化工作流、交互式对话系统、机器人等。工具使用与智能体（Tool Use and Agents）是近年来在大语言模型（LLM）和人工智能（AI）领域中日益重要的研究方向之一。这一技术旨在使大语言模型不仅能够理解和生成自然语言，还能与外部世界进行交互，执行具体任务，利用工具来增强其能力，模拟智能体的行为。工具使用是指将外部 API 或功能模块集成到大语言模型中，使其能够执行特定的任务。这些工具可以是天气查询、计算、信息检索等功能，允许模型在生成文本的同时，调用外部资源来增强其能力。智能体是指能够自主执行任务并根据环境反馈进行调整的系统。在大语言模型中，智能体可以理解用户输入、选择合适的工具并调用它们，以完成更复杂的任务。

在大语言模型的框架中，工具使用通常是指模型在推理过程中调用外部工具、系统或数据库，以补充模型本身的功能。这些工具可以是程序化的、计算性的，也可以是数据性的。常见的工具包括：（1）API 调用：大语言模型可以被设计成能够调用外部 API 的智能体，通过 HTTP 请求与外部系统进行交互。（2）数据库查询：当模型需要访问大量结构化数据时，可以通过数据库查询工具与数据库进行交互。（3）计算工具：大语言模型可以通过调用数学计算库或服务来执行复杂的计算任务。例如，模型可以请求调用一个数值分析工具来解答数学问题，或者调用一个图形绘制工具生成图表。（4）文档与文件处理：模型可以使用文件处理工具（如 OCR、PDF 处理、文本提取工具等）来分析和理解文档内容，进行结构化数据提取或内容摘要。

大语言模型中的智能体需要处理的任务通常是多步骤的。模型不仅需要理解任务指令，还需要进行多步推理，决定在每一步中是否调用工具、调用哪种工具以及如何将工具输出与下一步的决策结合。例如，在做一个复杂的数学问题时，智能体可能需要依赖计算工具，而在进行文本分析时，可能需要调用自然语言处理 API。包括：（1）智能体需要通过各种输入获取外部环境的信息。这些输入可能来自传感器、用户指令、API 响应、文本输入等。（2）基于感知到的环境信息，智能体通过推理来做出决策。决策过程可能基于目标导向的规划，或者基于模仿学习（Imitation Learning）、强化学习（Reinforcement Learning）等技术来调整行为策略。（3）在决策过程中，智能体可能需要调用外部工具来执行某些任务。例如，查询数据库、进行计算、发送邮件、调用图像处理工具等。（4）最终，智能体会执行相应的操作。这可能是对外界做出的物理操作或者是输出信息。 在与用户或外部系统的交互过程中，智能体需要具备处理反馈的能力。通过实时反馈，智能体可以进行调整、学习，并优化其行为。

大语言模型中的工具使用与智能体结合，为其提供了强大的扩展能力，使得模型不仅能够生成文本，还能执行复杂任务。通过有效地解析用户意图、选择合适的工具以及整合结果，这一技术显著提升了 LLMs 在实际应用中的实用性和灵活性。大语言模型中的工具使用与智能体技术使得模型能够超越传统的语言生成任务，进入到更为广泛和复杂的应用领域。通过与外部工具的互动，模型不仅可以执行计算、查询数据，还能够进行实时推理、环境感知和决策优化。工具使用和智能体结合的技术为自动化系统、个性化服务和智能机器人等应用提供了强大的支持，推动了人工智能在实际世界中的广泛应用。

1.3.5 大语言模型强化推理

DeepSeek-R1 推理模型采用冷启动监督微调、面向推理强化学习、拒绝采样与多领域监督微调、全场景强化学习四个阶段流程进行训练，使模型会通过一步步分析问题、反复验证来提高准确率。在垂直领域大语言模型的古籍信息智能化应用中，可以通过强化推理的方式，提升对古籍内容的理解、分析和应用。现有研究通过知识融合、多模态处理和技术优化等手段，显著提升了模型对古籍文本的深度解析与逻辑推导能力。（1）知识图谱与推理结合。

强化推理可以引入更强的上下文记忆，增强模型对跨段落和章节信息的处理能力。大语言模型可以帮助从古籍中提取结构化信息，生成知识图谱，包括人物、事件、地点、时间等节点，以及它们之间的关系。通过在知识图谱上执行推理，模型能够发现潜在的隐含关系。（2）推理训练方法。对于古籍文本，尤其是未完全标注的部分，可以使用自监督学习方法，通过预测缺失的部分或识别与上下文不一致的部分来进行推理训练。模型不仅学习古籍中的词汇和句法，还能通过上下文推理出更深层次的历史背景和人物关系。通过将大规模预训练模型应用到特定古籍领域，可以借助现有的语言模型强化对古文文本的理解，从而在推理过程中对复杂的历史背景进行有效建模。采用强化学习框架，让模型通过奖励机制优化对历史事件因果关系的推断。（3）跨模态推理。在古籍中，不仅有文本信息，还有可能包括地图、插图等。跨模态推理不仅是对文本的推理，还可以结合图像信息，例如从古代地图或插画中提取空间关系，再结合文本推理历史事件的发生地点和范围。通过结合古籍的碑帖拓片、刻本图像等多模态数据，采用跨模态推理的方法提升模型对古文异体字、避讳字等特殊文本特征的识别准确率。大语言模型可以同时处理文本和其他类型的数据（如音频、图片等），并进行跨模态推理，这对于古籍的研究具有重要意义，因为古籍不仅仅是纯文本，也包含了许多历史背景和文化语境。（4）推理与上下文理解。古籍文本常常涉及长时间跨度的历史事件和人物。强化推理能够帮助大语言模型在长篇文献中进行全局上下文理解，识别出跨章节、跨事件的联系。例如，推理某一历史人物的家族关系、官职，以及这些关系如何影响他在历史事件中的选择和行动。古籍中的历史事件往往不是孤立发生的。大语言模型可以通过因果推理机制，分析事件的前因后果，揭示潜在的历史动因。（5）历史信息智能化的应用。大语言模型强化推理能够在大规模历史文献中自动发现不同历史事件之间的联系，进行智能分类和关联，为研究人员提供高效的文献检索和事件分析工具。模型能够通过推理揭示历史人物的社会关系、影响力等深层信息，帮助构建更精准的人物画像。通过这些强化推理的应用，大语言模型能够为古籍信息智能化提供更为精准的知识挖掘与推理支持，推动传统文化和历史研究的数字化与智能化发展。

1.4 大语言模型的应用

1.4.1 通用领域

通用大语言模型是指经过丰富的大规模数据集训练后，具有强泛化能力，能够广泛应用于各种场景和任务的大模型，经典的大模型包括 OpenAI 开发的 ChatGPT，阿里云研发的通义千问，百度研发的文心一言等。此类大模型在通用领域的应用主要包括：

a 自然语言理解与生成：利用大模型对自然语言的理解能力，能将其用于文本阅读、文本摘要、内容创作等多种自然语言处理任务。如可使用 ChatGPT 帮助用户理解文章内容、提供建议、翻译外文等，适用于多种场景与行业。

b 信息检索与知识问答：大模型能够高效快速的从大量文本中提取信息，将其整合为人类便于阅读和理解的内容。例如可以询问豆包大模型养生知识、旅游攻略等内容，也可以让他帮助回答生活中遇到的问题。

c 辅助办公任务：大模型能够辅助完成文档编写、数据分析及会议记录等办公任务，如可以利用 Kimi 大模型结合文本内容自动生成 PPT，提升办公效率。

通用大模型的优势在于其泛化能力和跨领域适应性，在各种场景中具有灵活性和高效性，能够快速适应新任务，降低开发成本。

1.4.2 垂直领域

相较于通用大模型，将大模型应用于垂直领域能进一步强化模型专业化程度，提升模型在回答专业知识方面的性能。垂直领域的大语言模型需要结合专业知识与数据，以便精准响应需求。目前大语言模型在众多垂直领域都有所应用，如医学、法律、金融、古籍等。

（1）医学领域大模型：包括扁鹊、华佗、神农等。BianQue（扁鹊）大模型是华南理工大学和广东省数字孪生人重点实验室开发的中文医疗大模型，使用 ChatGLM-6B 作为初始模型开发出扁鹊 2.0，新增了药品说明书查询和更多医学百科知识。HuatuoGPT（华佗 GPT）是香港中文大学和深圳市大数据研究院开发的医疗大模型，具备诊断能力，能保持对用户流畅的交互和内容的丰富性，对话丝滑。神农大模型（ShenNong-TCM-LLM）是由华东师范大学开发的中文中医药大模型，能够提升模型在中医药方面的知识与回答医学咨询的能力。

（2）法律领域大模型：包括 ChatLaw、HanFei（韩非）、LaWGPT 等。ChatLaw 是由北京大学团队开发的一款中文法律大模型，包括 ChatLaw-13B 和 ChatLaw-33B，能够提供普惠的法律服务，帮助解决法律专业人士和普通用户在法律问题上的疑问。HanFei（韩非）大模型是国内首个全参数训练的法律大模型，参数量 7B，主要功能包括：法律问答、多轮对话、撰写文章等。LaWGPT 模型在通用中文基座模型的基础上扩充法律领域专有词表、大规模中文法律语料预训练，增强了大模型在法律领域的基础语义理解能力。在此基础上，构造法律领域对话问答数据集、中国司法考试数据集进行指令精调，提升了模型对法律内容的理解和执行能力。

（3）金融领域大模型：包括 Cornucopia (聚宝盆）、XuanYuan (轩辕）等。Cornucopia (聚宝盆）基于中文金融领域问答数据构建指令数据集，对 Llama 模型进行了指令微调，提高了 Llama 在金融领域的问答效果。应用场景包括：金融新闻自动化、市场分析报告、金融文档生成、金融领域问答系统等。XuanYuan (轩辕）是国内首个开源的千亿级中文对话大模型，同时也是首个针对中文金融领域优化的千亿级开源对话大模型。XuanYuan 针对中文通用领域和金融领域进行了针对性的预训练与微调，它不仅可以应对通用领域的问题，也可以解答与金融相关的各类问题，为用户提供准确、全面的金融信息和建议。

（4）古籍领域大模型：包括 Xunzi（荀子）、AI 太炎等。Xunzi（荀子）大模型是全国首个古籍大语言模型，由南京农业大学推出的古籍处理与研究的智能工具，该开源模型包括两个部分：基座模型 XunziALLM 与对话模型 XunziChat。其功能多样，包括智能标引、信息抽取、诗歌生成、阅读理解、句法分析等等，具体应用场景详见 1.4.3。AI 太炎是由北京师范大学开发的古汉语大语言模型，专用于古汉语文本理解，其功能包括字典释义、文白翻译、句读标点和识别用典等。

相较于通用大模型，垂直领域大模型结合行业知识和特定任务，更关注模型的精准性和专业性，以便更好的服务于不同的领域需求。

1.4.3 大语言模型在古籍处理领域的应用

随着人工智能技术的迅猛发展，大语言模型（LLM）在自然语言处理领域取得了显著进展。这些模型不仅在现代文本处理上表现出色，还逐渐被应用于古籍的整理、研究和传播。古籍作为中华文化的重要载体，承载着丰富的历史、哲学、文学和科学知识。然而，由于古文的语言特点和历史背景，传统的古籍研究方法往往面临诸多挑战。大语言模型的引入，为古籍处理带来了新的机遇与可能性。

古籍研究通常涉及大量的文本分析、信息提取和知识整合工作。传统方法依赖于人工校对和解读，效率低下且容易出现错误。古文中使用的词汇、句式和语法结构与现代汉语存在较大差异，这使得研究者在理解和翻译古文时常常感到困难。此外，古籍数量庞大，分散在各个图书馆和档案馆中，缺乏系统化的数字化管理。南京农业大学于 2023 年 12 月发布了“荀子”古籍大语言模型，这是国内首个专门针对古籍处理与研究的开源智能工具。该模型基于超过 40 亿字的混合语料数据，涵盖了《四库全书》等大量古籍文献。其主要功能包括智能标引、信息抽取、诗歌生成、高质量翻译等。此外，用户可以根据需求微调模型，以适应特定的研究任务。北京师范大学于 2024 年发布了“AI 太炎”大语言模型，该模型专注于提高古汉语文本的理解能力。它支持词义注释、文白翻译、句读标点等多种功能，为教育和研究提供了强有力的支持。2024 年 9 月，由农业农村部及多家机构联合开发的“齐民”大语言模型专注于农业古籍文本。该模型通过挖掘古代农业知识，为农业相关问题提供精准解答，并支持文本自动处理和语义检索等功能。这一应用展示了大语言模型在特定领域（如农业）中的潜力。

大语言模型通过深度学习技术，可以从海量文本数据中学习语言规律，具备强大的文本理解与生成能力。以下是大语言模型在古籍处理中的几个主要应用：

1.4.3.1 信息抽取

大语言模型（LLM）在古籍处理领域的应用，尤其是在信息抽取方面，展现了其强大的潜力和广泛的应用前景。信息抽取是指从大量文本中自动识别和提取有用信息的过程，这对于古籍研究尤为重要，因为古籍中的信息往往分散且难以获取。古籍文献通常包含丰富的历史、文化和社会信息，但由于其语言结构复杂、词汇古老，传统的信息提取方法效率低下且容易出错。大语言模型通过深度学习技术，能够有效地从这些文本中提取关键信息，如人物、事件、地点等，从而大幅提升研究效率。基于大语言模型通过多级预训练任务来替代传统的 BERT 模型。通过设置多个预训练任务，使模型能够在不同层面上理解文本，从而提高信息抽取的准确性。并添加卷积层和句子级聚合结构，以优化生成的词表示，使得模型在处理古文时更加有效，从而更好地捕获古文中的语义信息。例如，南京农业大学研发的“荀子”古籍大语言模型采用了这种方法，能够自动从古籍文本中识别并提取实体（如历史人物、地名）和关系（如事件发生的时间和地点）。

大语言模型在古籍处理领域的信息抽取应用，不仅提升了研究效率，也为中华优秀传统文化的传承与传播提供了新的动力。大语言模型能够快速处理大量文本数据，显著降低人工整理和分析的工作量。通过深度学习算法，模型在理解和提取古文信息时表现出更高的准确性。信息抽取技术使得研究者可以更专注于分析和解读提取出的信息，从而推动古籍研究的新发展。随着技术的不断进步，这种智能化的信息处理方式将会越来越普及，为更多研究者提供便利，使得古籍中的丰富知识得以更好地被发掘和利用。

1.4.3.2 语法标注

大语言模型（LLM）在古籍处理中的语法标注应用，主要体现在词法分析、自动分词和词性标注等方面。这些技术的引入，不仅提升了古籍文本的可读性，还为研究者提供了更为高效的工具，以便于理解和分析古文。古籍文本通常使用复杂的语言结构和古老的词汇，传统的文本处理方法在对这些文献进行分析时面临诸多挑战。语法标注是自然语言处理中的一项基本任务，旨在为文本中的每个词汇分配相应的词性标签，从而帮助计算机理解句子的结构和意义。在古籍研究中，准确的语法标注对于后续的信息提取、翻译和文本分析至关重

要。

大语言模型能够对古籍文本进行自动分词，将连续的汉字序列切分为有意义的词汇。由于古汉语中缺乏明确的词边界，传统分词方法往往难以适用，而基于深度学习的模型能够通过上下文信息来判断词汇边界，从而提高分词的准确性。在完成分词后，大语言模型还能够对每个词汇进行词性标注。这一过程涉及到识别名词、动词、形容词等不同词性的功能。通过训练，模型能够理解古文中的语法规则，并为每个词汇赋予正确的标签。例如，在“孔子曰”这样的句子中，模型可以识别出“孔子”为名词，“曰”为动词，从而帮助研究者理解句子的结构。通过准确的语法标注，用户在阅读古籍时能够更清晰地理解句子的结构和含义，这对于非专业人士尤其重要，使得古籍更加易于接触和学习。随着大语言模型技术的发展，未来在古籍处理领域，语法标注将会变得更加智能化和自动化。研究者可以利用这些工具进行更深入的文本分析，如句法树构建、语义角色标注等，从而推动古籍研究的新进展。此外，结合更多领域知识与数据，大语言模型将在提升古籍处理效率、促进文化传承方面发挥更大的作用。

1.4.3.3 语内翻译与语际翻译

大语言模型（LLM）在古籍处理中的语内翻译（即同一语言内部的翻译）与语际翻译（不同语言之间的翻译）方面，展示了其强大的文本理解和生成能力。这些技术不仅提升了古籍文献的可读性，还为研究者提供了更为高效的工具，以便于理解和传播古代文化。大语言模型能够将古文文本自动翻译成现代汉语。这一过程涉及对古文中复杂句式和词汇的理解，模型通过训练能够识别古汉语的词义及其在特定上下文中的用法，从而生成准确且流畅的现代汉语版本。在进行语内翻译时，大语言模型还可以分析句子的结构，识别主谓宾等成分，确保翻译后的文本保留原意。这种能力对于古籍中的诗词、散文等文学作品尤为重要，因为这些作品常常蕴含深厚的文化背景和情感表达。大语言模型不仅限于将古文翻译为现代汉语，还可以实现古文与其他语言之间的翻译。例如，通过构建跨语言知识库，模型能够将古代经典文本翻译成英语、法语等多种语言，使得国际学者能够接触和研究中国古籍。随着技术不断进步，大语言模型在古籍处理领域的语内与语际翻译能力将继续提升。未来，这些技术可能会结合更多人工智能手段，如情感分析和上下文理解，以进一步增强翻译质量。

1.4.3.4 文献总结与标引

大语言模型（LLM）在古籍处理中的文献总结与标引应用，极大地提升了古籍文献的可读性和检索效率。这些技术不仅帮助研究者快速获取关键信息，还促进了古籍的数字化和智能化管理。古籍文献通常内容庞杂，信息密集，研究者在查阅时常常面临大量文本的挑战。文献总结技术可以帮助用户快速了解文献的核心内容、主题和重要观点，从而节省时间并提高研究效率。通过自动化的方式，研究者可以获得简洁明了的摘要，方便进行后续深入研究。字节跳动研发的“识典古籍”数字化平台上线了智能助手功能，通过大语言模型技术实现了对古籍内容的自动总结与标引。该平台支持用户以“问 AI”的方式获取全文内容总结，并提供相关参考资料。这一创新应用受到了广泛好评，并被视为提升古籍查阅效率的重要工具。大语言模型能够通过自然语言处理技术对古籍文本进行自动摘要生成。模型首先分析输入的古籍文本，识别出重要的信息和主题。通过算法提取关键信息，如主要观点、事件和人物等。基于提取的信息，生成简洁的文本摘要，确保保留原文的重要内容和结构。例如，在“识典古籍”平台上，用户可以通过智能助手请求特定文献的摘要，系统会自动生成相关内容，帮助用户快速了解文献要点。

文献标引是指为文献提供相关标签或关键词，以便于检索和分类。有效的标引能够提升古籍数据库的可用性，使得用户能够更轻松地找到所需资料。大语言模型可以自动为古籍文献生成关键词和主题标签。主要包括语义分析、关键词抽取和标签生成等。这种智能标引功能使得用户在查找相关资料时，可以通过关键词快速定位到感兴趣的文献，提高了检索效率。例如，“荀子”古籍大语言模型具备此功能，可以为相关古籍提供准确的主题标引。

大语言模型在古籍处理中的文献总结与标引应用，不仅提高了研究效率，还为用户提供了更为便捷的信息获取方式。通过智能化的总结与标引，这些工具使得古籍中的丰富知识得以更好地被发掘和传播。

1.4.3.5 智能问答

大语言模型（LLM）在古籍处理中的智能问答应用，极大地提升了用户对古籍文献的访问和理解能力。这一技术通过自然语言处理和语义检索，能够为用户提供即时、准确的回答，从而改善古籍研究和学习的体验。古籍文献通常内容复杂、信息量大，研究者和普通读者在查阅时常常面临困难。传统的检索方式往往无法满足用户对具体信息的快速获取需求。智能问答系统通过自然语言交互，使得用户能够以更自然的方式提出问题，并获得精准的回答，这对于推动古籍知识的传播与理解具有重要意义。字节跳动开发的“识典古籍”数字化平台上线了智能助手功能，该助手利用大语言模型技术支持用户进行自然对话。例如，用户可以通过简单提问获取古文翻译、全文总结或相关研究资料。该平台目前已开放 2900 多部古籍，为普通人提供了便捷的阅读体验。

大语言模型能够理解用户提出的问题，并根据上下文进行意图判断。根据问题类型，从古籍数据库中提取相关信息，确保回答的准确性。例如，当用户询问“以铜为镜，可以正衣冠；以古为镜，可以知兴替；以人为镜，可以明得失。”时，系统会检索相关古籍段落并进行分析，以提供详尽的回答。为了提高回答的相关性，智能问答系统采用了语义检索技术。这种方法不仅依赖于关键词匹配，还能够理解不同表述之间的语义关系，从而找到更为相关的信息。例如，用户询问与“北京”相关的古籍时，系统可以识别出不同词汇但含义相近的段落。大语言模型在古籍处理中的智能问答应用，不仅提高了查阅效率，还为用户提供了更为直观的信息获取方式。通过自然语言理解和语义检索技术，这些工具使得古籍中的丰富知识得以更好地被发掘和传播，为中华优秀传统文化的传承与弘扬开辟了新的路径。

1.4.3.6 多模态任务

大语言模型（LLM）在古籍处理领域的多模态任务应用，主要体现在结合文本、图像和音频等多种信息形式，以实现更全面的古籍理解和处理。古籍文献不仅包含文本信息，还常常伴随有图像和音频。传统的文本处理方法往往无法充分利用这些多样化的信息资源。南京农业大学开发的“荀子”古籍大语言模型不仅支持传统的文本处理功能，还具备多模态任务能力。该模型整合了大量古籍文献及其相关图像和音频资料，使得用户能够通过智能助手进行更为全面的查询和学习。多模态任务通过整合不同类型的数据，能够为用户提供更为丰富和直观的文化体验，促进对古籍内容的全面理解。大语言模型可以通过分析古籍文本与其附带图像之间的关系，进行有效的信息提取。例如，模型能够识别书法作品中的文字，并将其与相关的历史背景、文化内涵进行关联。利用计算机视觉技术，模型能够识别古籍中的插图、书法等，并将其与文本内容进行关联。在古籍朗读和解说方面，大语言模型可以生成与文本内容相匹配的音频输出。对于已有音频资料，模型可以进行语音识别，将其转化为可编辑文本，并进行分析。这一功能不仅适用于古文的朗读，还可以用于对古籍内容的解说，使

得用户能够通过听觉体验更深入地理解古籍。

综上所述，大语言模型正在逐步改变传统古籍处理的方法，为这一领域带来了全新的视角和工具。尽管大语言模型在古籍处理方面展现出巨大的潜力，但仍然面临一些挑战。例如，数据质量直接影响到模型的表现，而许多古籍由于年代久远，其文本质量参差不齐。通过智能化的数据处理和分析能力，这些技术不仅提升了研究效率，也为中华优秀传统文化的传承与传播提供了新的动力。随着技术不断进步，大语言模型将在未来继续发挥重要作用，使更多人能够接触并理解这些珍贵的文化遗产。

1.4.3.7 古籍推理

在古籍处理领域，术语、词汇、典故等可能与现代语言有很大差异，因此模型需要能够理解和运用特定领域的语言，并且强化对古代文献的语境和文化背景的理解。（1）古籍内容的深度理解与推理。通过大语言模型对古籍中人物信息的解析，模型可以推理出人物之间的关系，包括亲缘关系、政治联盟、敌对关系等。比如，从《史记》《资治通鉴》等历史文献中，模型不仅能提取人物的基本信息，还能推理出人物的社会关系和他们在不同历史事件中的角色及作用。古籍中的事件通常具有复杂的因果关系。大语言模型能够通过对历史文本的推理，揭示事件之间的关联。（2）知识图谱推理。大语言模型能够从古籍文本中提取出人物、事件、地点、时间等实体，并通过强化推理，构建一个具有因果关系和时序顺序的历史知识图谱。通过对图谱的推理，模型可以推断出缺失的历史信息，例如某个事件发生的前因后果，或者某个历史人物的潜在社会角色。古籍中的事件和人物常常跨越不同的篇章和章节，模型通过推理技术可以跨章节地进行上下文理解，推测人物的动机、行为以及事件发展的脉络。这有助于发现文献中隐含的知识或信息。（3）隐性信息挖掘与推理。古籍的语言结构复杂，且存在较多模糊、隐含的语义信息。大语言模型可以通过推理技术，识别文中的隐性信息。例如，古籍中可能有对一个人物的单一记载，而另一些文本可能涉及该人物的其他身份或事件的介绍，模型需要通过多维度的关联推理将这些分散的信息汇聚，形成完整的知识图谱。（4）多模态推理与图文结合。许多古籍包含插图、地图等辅助信息。大语言模型在进行古籍推理时，不仅限于文本内容，还可以将图像信息与文本内容相结合，进行跨模态推理。通过将文本与图像结合，模型不仅能够更准确地理解古籍中的复杂历史事件、人物和文化背景，还能进行更加精细的推理，揭示出那些图文互补的隐性信息。（5）大语言模型在古籍推理方面的应用。一是古籍校勘辅助：利用大语言模型自动识别不同版本古籍中的异体字演变规律，提高校勘效率。二是历史事件推演：结合时间推理模块，重构古籍中历史事件的时空逻辑关系，生成可视化推演图谱。

大语言模型在古籍推理中的应用，主要体现在通过强化推理技术挖掘古籍中隐含的历史信息、建立历史知识图谱、进行因果推理等方面。这不仅可以提升对古籍内容的理解，还能帮助历史学者在大量文献中挖掘出更深层次的历史知识，为古籍信息智能化应用提供强有力的技术支持。

1.5 代码获取与代码运行环境配置

1.5.1 高性能计算平台

科学计算作为与理论研究和科学实验并列的第三种科学研究方法，已经在南京农业大学

的相关学科中得到了深入的应用。随着高性能计算（HPC）和人工智能（AI）训练领域的介入，学校的科研能力得到了显著提升。为了进一步推升高性能计算水平，更好地服务于全校科研工作，2024 年 6 月，学校设立了“高性能计算中心”，该中心挂靠于信息化建设中心，旨在为全校提供强大的计算资源和技术支持。校级高性能计算公共平台现拥有 129 个 CPU 计算节点（8288 核心），22 个 GPU 计算节点（128 张 A800 卡），总算力超过 3000TFLops；平台拥有 44 个存储节点，裸存储容量达 30.24PB。通过 200G Infiniband 网络实现了节点之间的高速通信，为科学计算和 AI 训练提供了高性能网络基础。

在垂直领域大语言模型实战方面，以古籍信息智能化为例，高性能计算中心可以发挥重要作用。古籍信息智能化指的是利用现代信息技术对古代文献进行数字化、整理、分析和解读的过程。这一过程涉及到大量的文本处理、数据分析以及机器学习任务，这些都是高性能计算的理想应用场景。例如，（1）大规模数据处理。古籍数字化通常会产生庞大的数据集，包括图像、文本和其他多媒体资料。高性能计算中心能够提供必要的存储空间和快速的数据处理能力，加速古籍的数字化进程。（2）深度学习模型训练：为了实现古籍内容的理解和翻译，需要构建专门针对古文的语言模型。高性能计算中心提供的 GPU 集群和分布式计算环境非常适合进行这种复杂模型的训练。南京农业大学设立的高性能计算中心将为古籍信息智能化提供强有力的技术支撑，推动这一领域取得更多成果。

1.5.2 常用深度学习 python 库

在训练及微调大语言模型时，需要依赖特定版本的 Python 库，本节主要介绍六种常用的 Python 库及其在模型应用中的作用。

（1） Transformers：是 HuggingFace 开发的基于 transformer 模型结构的开源 Python 库，提供了大量预训练语言模型（如 BERT，GPT2，RoBERTa 等），支持 Pytorch，Tensorflow 等深度学习框架，能够完成文本分类、命名实体识别、机器翻译等多种自然语言处理任务。

（2） Pytorch：是 Torch 的 Python 版本，由 Facebook 开源的神经网络框架，主要用于训练和微调深度学习模型。Pytorch 的特点是动态计算图，与 Python 的数据处理库深度集成且提供了自动求导机制，是深度学习领域的主流框架之一。

（3） Datasets：是一种数据处理工具，能够从本地或 HuggingFace 加载数据集，适用于大规模文本数据，常与 Transformers 结合使用。

（4） Accelerate：是一种用于简化深度学习训练过程的 Python 库，支持单机、多 GPU 或分布式训练等各种环境。通过提供一种通用 API 将原本在单设备上运行的代码扩展到多设备和分布式环境。

（5） PEFT：是一个用于参数高效微调的库，其作用是在微调大模型时无需调整所有参数，只通过调整少量参数来实现与完全微调相当的性能。其集成了多种先进的微调方法，如 LoRA、Prompt Tuning 等，

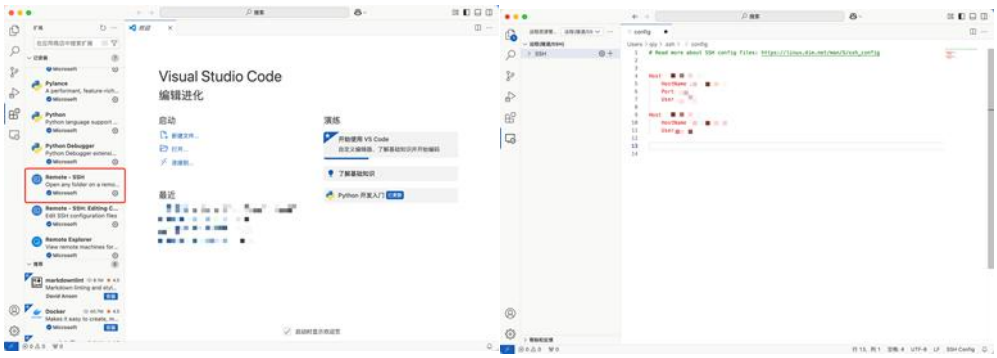
（6） TRL：是一种用于对模型进行优化调整的 Python 库，其结合了 SFT（监督微调）、RL（强化学习）和 PPO（近端策略优化）技术，通过高效的训练流程对模型进行精细化调整，增强模型在特定任务上的性能。

1.5.3 本书代码运行环境

本书代码可以使用 Visual Studio Code（以下简称 Vscode）或 PyCharm 两个软件运行。本节分别描述两种软件下运行模型的前置准备。

1.5.3.1 Visual Studio Code

（1） 远程连接服务器：通过远程开发插件（Remote-SSH）连接服务器。安装插件后通过设置选中配置文件，在配置文件中填写服务器信息。Vscode 在连接服务器后，可以访问并操作远程的 Linux 系统。



图：下载 Remote-SSH 插件、远程连接服务器

（2） 配置环境：安装 Python 插件用于支持代码运行。随后可在集成终端中使用 conda 创建虚拟环境,其中 yourname 为用户自行设定的环境名称，此处选择 python 版本为 3.10，可根据不同项目调整 python 版本。创建虚拟环境后激活并查看环境列表，随后可在环境中安装项目所需的具体 python 库。

创建环境：conda create -n yourname python=3.10

激活环境：conda activate yourname

查看环境列表：conda env list

安装所需 Python 库：pip install transformers/conda install transformers

（3） 运行代码：进入虚拟环境后，在该环境中运行项目代码。在 Vscode 中可使用内置的运行按钮运行简单的 python 代码文件，也可使用集成终端运行，在终端中进入代码所在目录，通过命令行运行。

Python：python filename.py

Bash 脚本：bash filename.sh

1.5.3.2 PyCharm

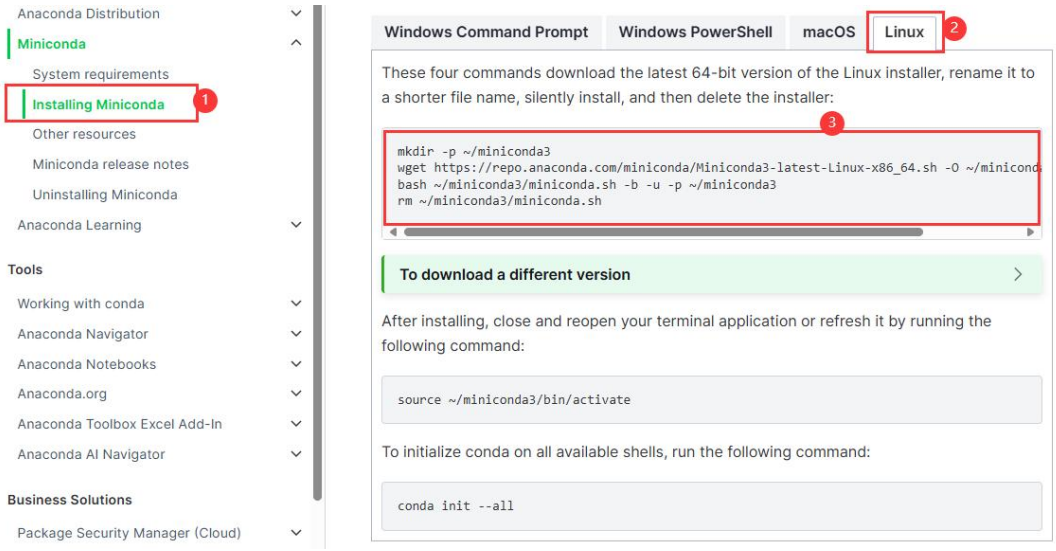
(1) 下载 Miniconda

Anaconda 和 Miniconda 都是由 Continuum Analytics 开发的开源项目，旨在管理 Python 的环境和包。它们都使用 Conda 包管理器来创建和管理虚拟环境，使得在不同项目之间的切换和隔离变得更加简单。然而，两者之间存在一些关键区别。

Anaconda 是一个包含 Conda、Python 以及超过 150 个科学计算和数据分析包的完整发行版。它预装了许多常用库，如 NumPy、Pandas、SciPy 和 Matplotlib，并提供了可视化图形用户界面（Anaconda Navigator），方便新手用户快速上手。相较之下，Miniconda 则是一个更轻量级的选择。它仅包含 Python 和 Conda，没有预装其他库。用户需要手动安装所需的包，这使得 Miniconda 的环境更加简洁，能够根据实际需求灵活安装必要的库，从而避免不必要的存储占用。

国内服务器需要选择 conda 镜像，通常有清华源、中科大、北京外国语镜像。miniconda

版本选择（https://mirrors.tuna.tsinghua.edu.cn/anaconda/miniconda/?C=M&O=A）或者官网（https://docs.anaconda.com/miniconda/install/）

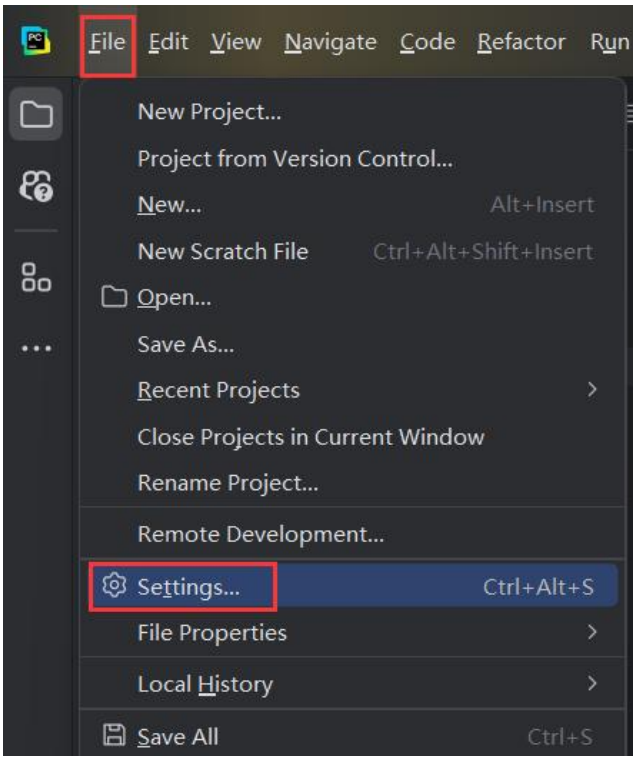


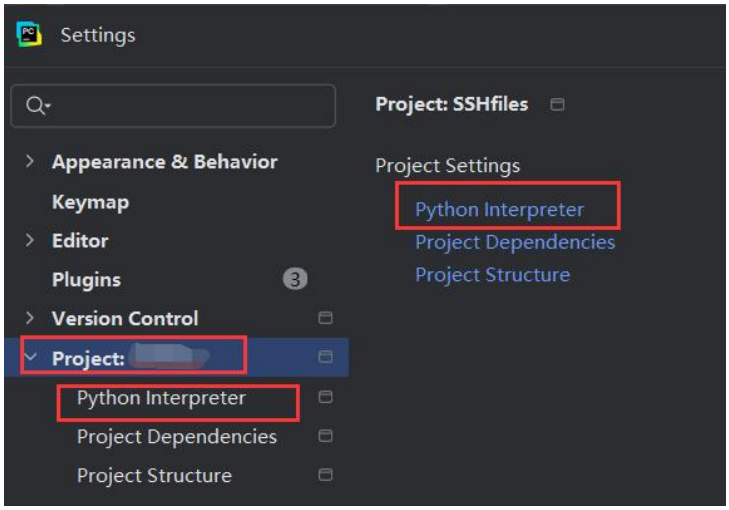
(2) 配置 python 环境

具体内容请查看 Visual Studio Code 章节内容。

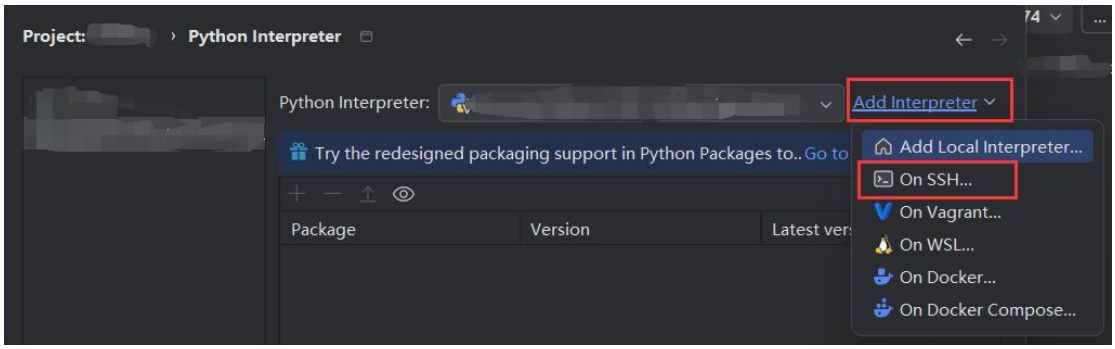
(3) Pycharm 远程连接服务器

（a）打开 pycharm，新建项目之后，连接上服务器的 Python 解释器。打开解释器的过程如下：File → Settings → Project:XXX →Python Interpreter

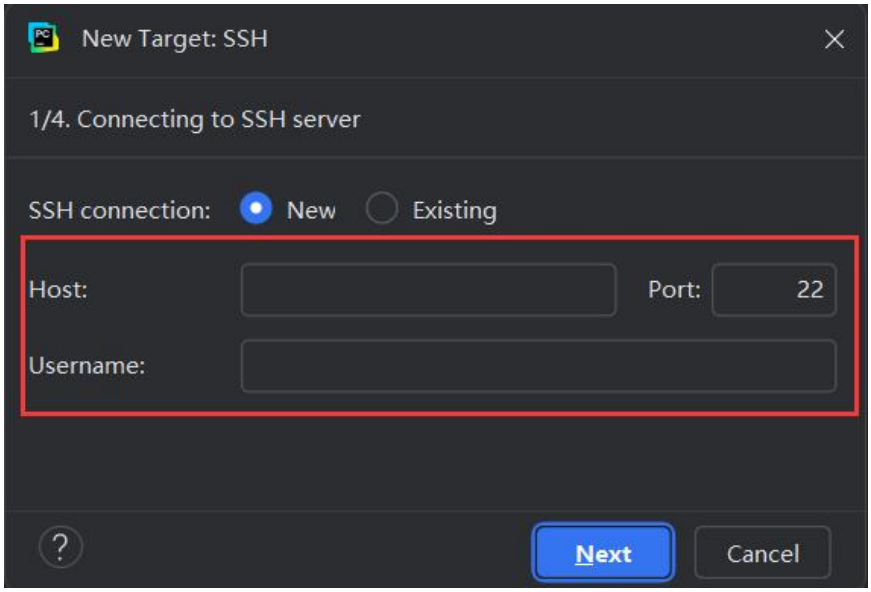




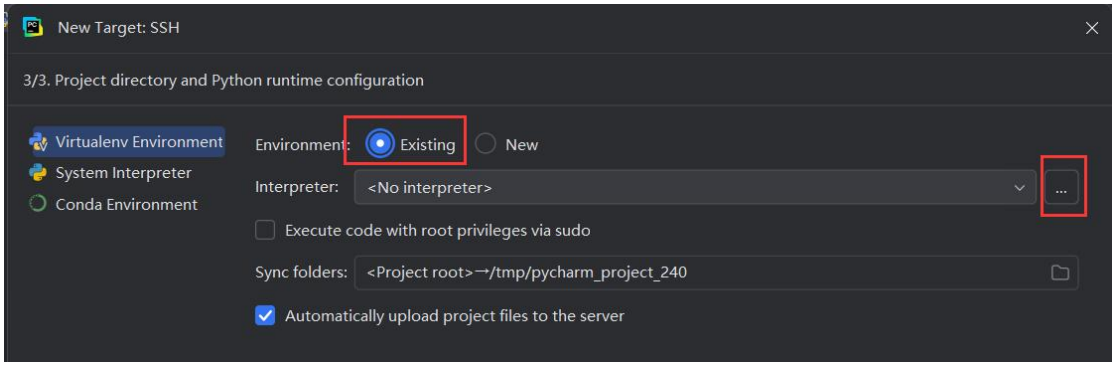
(b) 打开之后选择 Add interpreter，选择 On SSH。



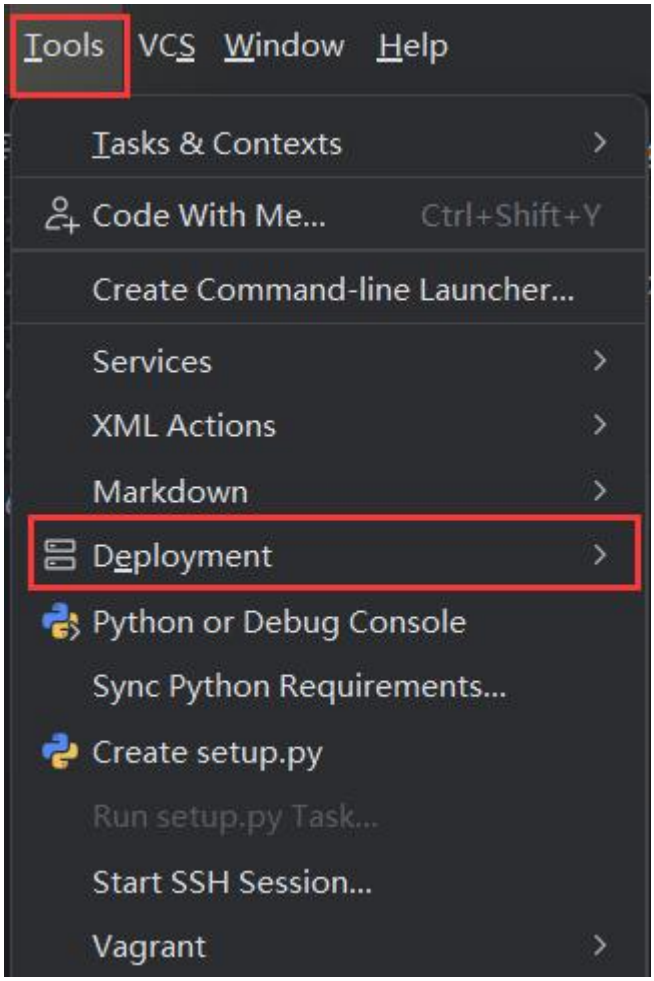
(c) 输入服务器地址 IP、端口号、用户名、登录密码。



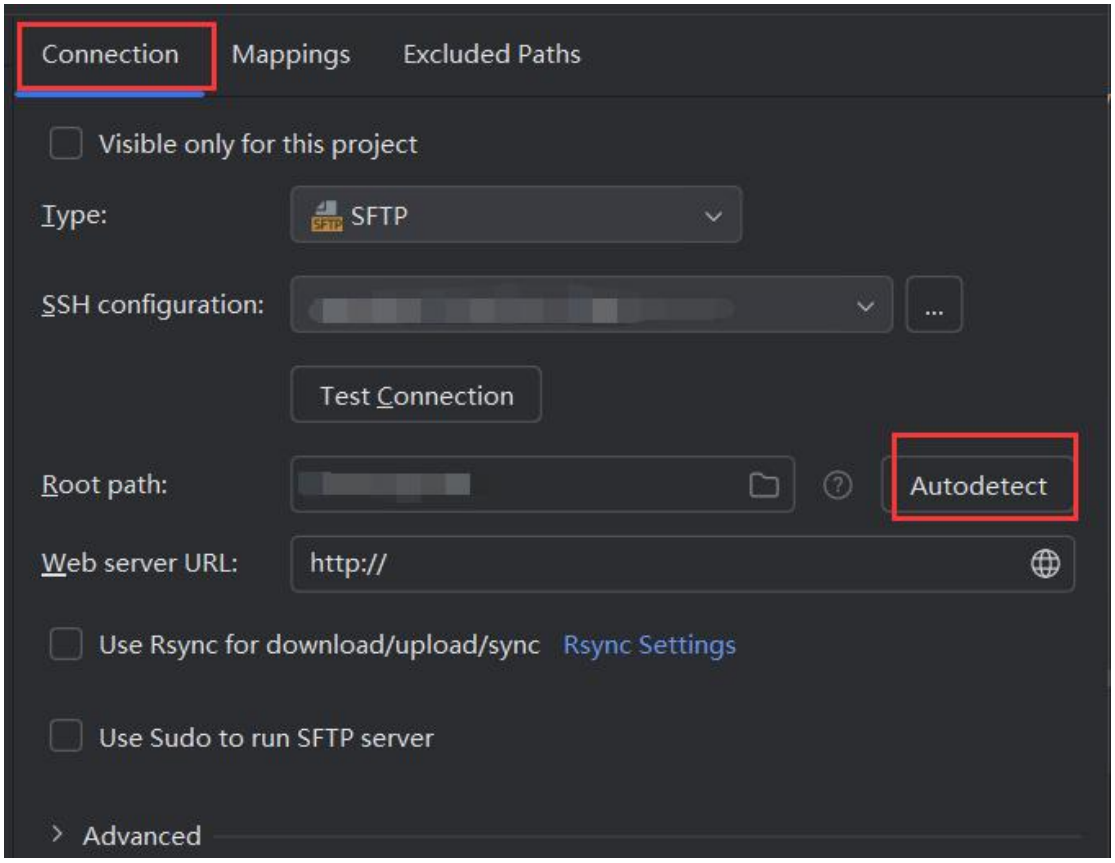
(d) 如果前面已经完成创建 python 环境项目，选择 Existing。如果没有则先完成创建 python 环境。Interpreter 这里是解释器的地址，找到你虚拟环境里的 Python。Sync folder 是你本地和服务器上存代码的地址映射，不用在这里设置。

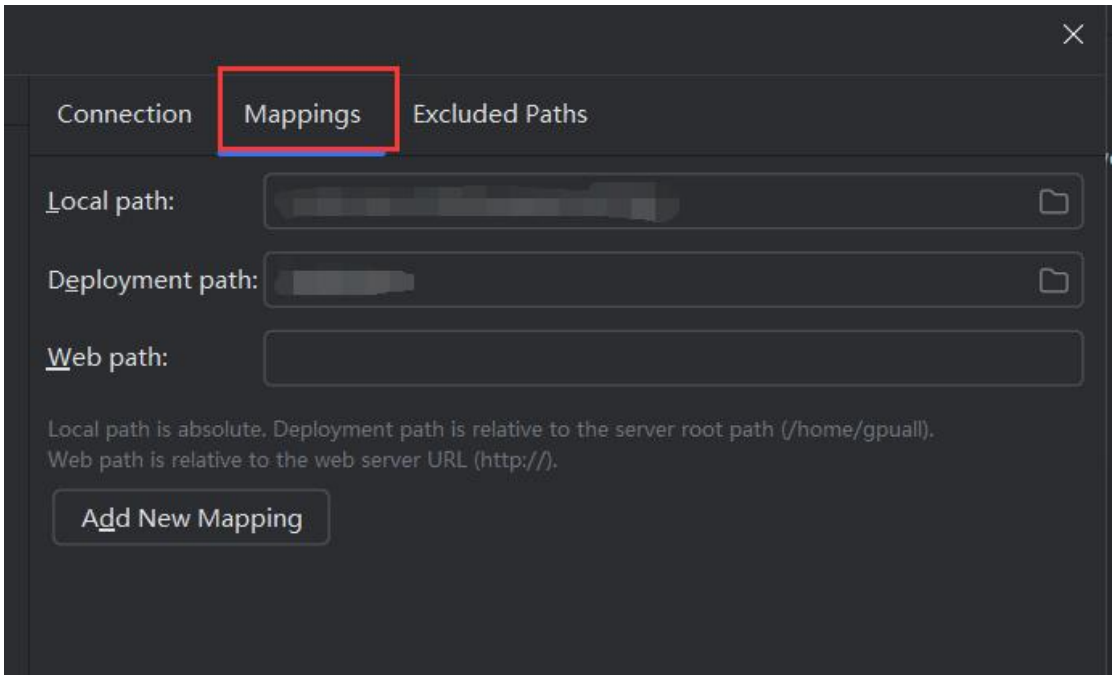


（e）同步。选择 Tool→Deployment→Configuration。



（f）在 connection 选项卡，点 Autodetect 自动给你定位到你账户下的 home 目录，然后再去 mapping 选项卡设置到你想映射的地址。





（g）设置完毕以后在项目名称上右键，选择 Deployment，可以根据自己的需要选择向服务器上传代码还是从服务器下载代码到本地了。

1.6 课后作业

本章节作业要求根据书中的代码和相关流程，在本地机器或服务器上配置相关代码运行环境。