第 3 章　大语言模型预训练数据

训练大语言模型需要数万亿的各类型数据。如何构造海量“高质量”数据对于大语言模型的训练具有至关重要的作用。截至 2023 年 9 月，还没有非常好的大语言模型的理论分析和解释，也缺乏对语言模型训练数据的严格说明和定义。但是，大多数研究人员认为预训练数据是影响大语言模型效果及样本泛化能力的关键因素之一。当前的研究表明，预训练数据需要涵盖各种类型的文本，也需要覆盖尽可能多的领域、语言、文化和视角，从而提高大语言模型的泛化能力和适应性。目前，大语言模型采用的预训练数据通常来自网络、图书、论文、百科和社交媒体。

本章将介绍常见的大语言模型预训练数据的来源、处理方法、预训练数据对大语言模型影响的分析及开源数据集等。

3.1　数据来源

文献 [5] 介绍了 OpenAI 训练 GPT-3 使用的主要数据来源，包含经过过滤的 CommonCrawl 数据集[1]、WebText 2、Books 1、Books 2 及英文 Wikipedia 等数据集。其中 CommonCrawl 的原始数据有 45TB，过滤后仅保留了 570GB 的数据。通过词元方式对上述语料进行切分，大约包含 5000 亿词元。为了保证模型使用更多高质量数据进行训练，在 GPT-3 训练时，根据语料来源的不同，设置不同的采样权重。在完成 3000 亿词元训练时，英文 Wikipedia 的语料平均训练轮数为 3.4 次，而 CommonCrawl 和 Books 2 仅有 0.44 次和 0.43 次。由于 CommonCrawl 数据集的过滤过程烦琐复杂，Meta 公司的研究人员在训练 OPT[31] 模型时采用了混合 RoBERTa[71]、Pile[72] 和 PushShift.io Reddit[73] 数据的方法。由于这些数据集中包含的绝大部分数据都是英文数据，因此 OPT 也从 CommonCrawl 数据集中抽取了部分非英文数据加入训练语料。

大语言模型预训练所需的数据来源大体上分为通用数据和专业数据两大类。**通用数据**（General Data）包括网页、图书、新闻、对话文本等[14, 31, 46]。通用数据具有规模大、多样性和易获取等特点，因此支持大语言模型的语言建模和泛化能力。**专业数据**（Specialized Data）包括多语言数据、科学文本数据、代码及领域特有资料等。通过在预训练阶段引入专业数据可以有效提升大语言模型的任务解决能力。图 3.1 给出了一些典型的大语言模型所使用数据类型的分布情况。可以看到，不同的大语言模型在训练数据类型分布上的差距很大，截至 2023 年 9 月，还没达成广泛的共识。

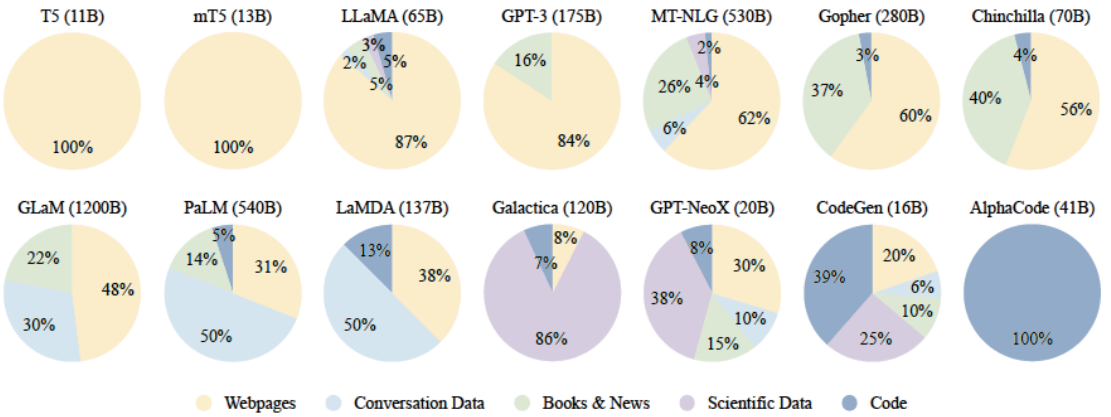

图 3.1　典型的大语言模型所使用数据类型的分布情况[18]

3.1.1　通用数据

通用数据在大语言模型训练数据中占比非常高，主要包括网页、对话文本、书籍等不同类型的数据，为大语言模型提供了大规模且多样的训练数据。

网页（Webpage）是通用数据中数量最多的一类。随着互联网的大规模普及，人们通过网站、论坛、博客、App 创造了海量的数据。根据 2016 年 Google 公开的数据，其搜索引擎索引处理了超过 130 万亿网页数据。网页数据所包含的海量内容，使语言模型能够获得多样化的语言知识并增强其泛化能力[4, 19]。爬取和处理海量网页内容并不是一件容易的事情，因此一些研究人员构建了 ClueWeb09[74]、ClueWeb12[75]、SogouT-16[76]、CommonCrawl 等开源网页数据集。虽然这些爬取的网络数据包含大量高质量的文本（如维基百科），但也包含非常多低质量的文本（如垃圾邮件等）。因此，过滤并处理网页数据以提高数据质量对大语言模型训练非常重要。

对话文本（Conversation Text）是指有两个或更多参与者交流的文本内容。对话文本包含书面形式的对话、聊天记录、论坛帖子、社交媒体评论等。当前的一些研究表明，对话文本可以有效增强大语言模型的对话能力[31]，并潜在地提高大语言模型在多种问答任务上的表现[14]。对话文本可以通过收集、清洗、归并等过程从社会媒体、论坛、邮件组等处构建。相较于网页数据，对话文本数据的收集和处理更加困难，数据量也少很多。常见的对话文本数据集包括 PushShift.io Reddit[73, 77]、Ubuntu Dialogue Corpus[78]、Douban Conversation Corpus、Chromium Conversations Corpus 等。此外，文献 [79] 也提出了使用大语言模型自动生成对话文本数据的 UltraChat 方法。

书籍（Book）是人类知识的主要积累方式之一，从古代经典著作到现代学术著述，承载了丰富多样的人类思想。书籍通常包含广泛的词汇，包括专业术语、文学表达及各种主题词汇。利用

书籍数据进行训练，大语言模型可以接触多样化的词汇，从而提高其对不同领域和主题的理解能力。相较于其他语料库，书籍也是最重要的，甚至是唯一的长文本书面语的数据来源。书籍提供了完整的句子和段落，使大语言模型可以学习到上下文之间的联系。这对于模型理解句子中的复杂结构、逻辑关系和语义连贯性非常重要。书籍涵盖了各种文体和风格，包括小说、科学著作、历史记录，等等。用书籍数据训练大语言模型，可以使模型学习到不同的写作风格和表达方式，提高大语言模型在各种文本类型上的能力。受限于版权因素，开源书籍数据集很少，现有的开源大语言模型研究通常采用 Pile 数据集[72] 中提供的 Books 3 和 BookCorpus 2 数据集。

3.1.2　专业数据

虽然专业数据在通用大语言模型中所占比例通常较低，但是其对改进大语言模型在下游任务上的特定能力有着非常重要的作用。专业数据有非常多的种类，文献 [18] 总结了当前大语言模型使用的三类专业数据。

多语言数据（Multilingual Text）对于增强大语言模型的语言理解和生成多语言能力具有至关重要的作用。当前的大语言模型训练除了需要目标语言中的文本，通常还要整合多语言语料库。例如，BLOOM[33] 的预训练语料中包含 46 种语言的数据，PaLM[14] 的预训练语料中甚至包含高达 122 种语言的数据。此前的研究发现，通过多语言混合训练，预训练模型可以在一定程度上自动构建多语言之间的语义关联[89]。因此，多语言数据混合训练，可以有效提升翻译、多语言摘要和多语言问答等任务能力。此外，由于不同语言中不同类型的知识获取难度不同，多语言数据还可以有效地增加数据的多样性和知识的丰富性。

科学文本（Scientific Text）数据包括教材、论文、百科及其他相关资源。这些数据对于提升大语言模型在理解科学知识方面的能力具有重要作用[34]。科学文本数据的来源主要包括 arXiv 论文[81]、PubMed 论文[82]、教材、课件和教学网页等。由于科学领域涉及众多专业领域且数据形式复杂，通常还需要对公式、化学式、蛋白质序列等采用特定的符号标记并进行预处理。例如，公式可以用 LaTeX 语法表示，化学结构可以用 SMILES（Simplified Molecular Input Line Entry System）表示，蛋白质序列可以用单字母代码或三字母代码表示。这样可以将不同格式的数据转换为统一的形式，使大语言模型更好地处理和分析科学文本数据。

代码（Code）是进行程序生成任务所必需的训练数据。近期的研究和 ChatGPT 的结果表明，通过在大量代码上进行预训练，大语言模型可以有效提升代码生成的效果[83-84]。代码不仅包含程序代码本身，还包含大量的注释信息。与自然语言文本相比，代码具有显著的不同。代码是一种格式化语言，它对应着长程依赖和准确的执行逻辑[85]。代码的语法结构、关键字和特定的编程范式都对其含义和功能起着重要的作用。代码的主要来源是编程问答社区（如 Stack Exchange[86-87]）和公共软件仓库（如 GitHub[29, 83, 88]）。编程问答社区中的数据包含了开发者提出的问题、其他开发者的回答及相关代码示例。这些数据提供了丰富的语境和真实世界中的代码使用场景。公共

软件仓库中的数据包含了大量的开源代码，涵盖多种编程语言和不同领域。这些代码库中的很多代码经过了严格的代码评审和实际的使用测试，因此具有一定的可靠性。

3.2　数据处理

大语言模型的相关研究表明，数据质量对于模型的影响非常大。因此，在收集了各种类型的数据之后，需要对数据进行处理，去除低质量数据、重复数据、有害信息、个人隐私等内容[14, 89]。典型的数据处理流程如图 3.2 所示，主要包括质量过滤、冗余去除、隐私消除、词元切分这几个步骤。本节将依次介绍上述内容。

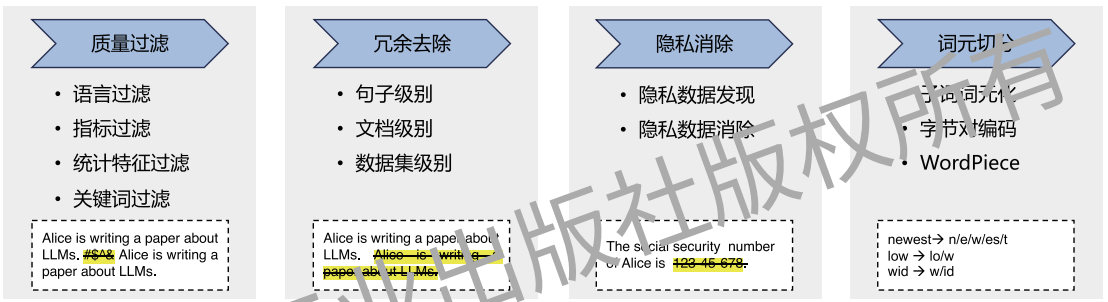

图 3.2　典型的数据处理流程图[18]

3.2.1　质量过滤

互联网上的数据质量参差不齐，无论是 OpenAI 联合创始人 Andrej Karpathy 在微软 Build 2023 的报告，还是当前的一些研究都表明，训练数据的质量对于大语言模型效果具有重大影响。因此，从收集到的数据中删除低质量数据成为大语言模型训练中的重要步骤。大语言模型训练中所使用的低质量数据过滤方法可以大致分为两类：**基于分类器的方法**和**基于启发式的方法**。

基于分类器的方法的目标是训练文本质量判断模型，利用该模型识别并过滤低质量数据。GPT-3[46]、PaLM[14] 和 GLaM[90] 模型在训练数据构造时都使用了基于分类器的方法。文献 [90] 采用了基于特征哈希的线性分类器（Feature Hash Based Linear Classifier），可以非常高效地完成文本质量判断。该分类器使用一组精选文本（维基百科、书籍和一些选定的网站）进行训练，目标是给与训练数据类似的网页较高分数。利用这个分类器可以评估网页的内容质量。在实际应用中，还可以通过使用 Pareto 分布对网页进行采样，根据其得分选择合适的阈值，从而选定合适的数据集。然而，一些研究发现，基于分类器的方法可能会删除包含方言或者口语的高质量文本，从而损失一定的多样性[89-90]。

基于启发式的方法则通过一组精心设计的规则来消除低质量文本，BLOOM[33] 和 Gopher[89] 采用了基于启发式的方法。一些启发式规则如下。

- 语言过滤：如果一个大语言模型仅关注一种或者几种语言，则可以大幅过滤数据中其他语言的文本。
- 指标过滤：利用评测指标也可以过滤低质量文本。例如，可以使用语言模型对给定文本的困惑度进行计算，利用该值可以过滤非自然的句子。
- 统计特征过滤：针对文本内容可以计算包括标点符号分布、符号字比（Symbol-to-Word Ratio）、句子长度在内的统计特征，利用这些特征过滤低质量数据。
- 关键词过滤：根据特定的关键词集，可以识别并删除文本中的噪声或无用元素。例如，HTML 标签、超链接及冒犯性词语等。

在大语言模型出现之前，在自然语言处理领域已经开展了很多**文章质量判断**（Text Quality Evaluation）相关的研究，主要应用于搜索引擎、社会媒体、推荐系统、广告排序及作文评分等任务中。在搜索和推荐系统中，结果的内容质量是影响用户体验的重要因素之一，因此，此前很多工作都是针对用户生成内容（User-Generated Content，UGC）的质量进行判断的。自动作文评分也是文章质量判断领域的一个重要子任务，自 1998 年文献 [91] 提出使用贝叶斯分类器进行作文评分预测以来，基于 SVM[92]、CNN-RNN[93]、BERT[94-95] 等方法的作文评分算法也相继被提出，并取得了较大的进展。这些方法都可以应用于大语言模型预训练数据过滤。由于预训练数据量非常大，并且对质量判断的准确率要求并不非常高，因此一些基于深度学习和预训练的方法还没有应用于低质过滤中。

3.2.2　冗余去除

文献 [96] 指出，大语言模型训练语料库中的重复数据，会降低大语言模型的多样性，并可能导致训练过程不稳定，从而影响模型性能。因此，需要对预训练语料库中的重复数据进行处理，去除其中的冗余部分。**文本冗余发现**（Text Duplicate Detection）也被称为文本重复检测，是自然语言处理和信息检索中的基础任务之一，其目标是发现不同粒度上的文本重复，包括句子、段落、文档等不同级别。冗余去除就是在不同的粒度上去除重复内容，包括句子、文档和数据集等粒度。

在句子级别上，文献 [97] 指出，包含重复单词或短语的句子很可能造成语言建模中引入重复的模式。这对语言模型来说会产生非常严重的影响，使模型在预测时容易陷入**重复循环**（Repetition Loops）。例如，使用 GPT-2 模型，对于给定的上下文："In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English."。如果使用束搜索（Beam Search），当设置 $b = 32$ 时，模型就会产生如下输出，进入重复循环模式。"The

study, published in the Proceedings of the National Academy of Sciences of the United States of America (PNAS), was conducted by researchers from the Universidad Nacional Autónoma de México (UNAM) and the Universidad Nacional Autónoma de México (UNAM/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de ⋯”。由于重复循环对语言模型生成的文本质量有非常大的影响，因此在预训练语料中需要删除这些包含大量重复单词或者短语的句子。

在 RefinedWeb[64] 的构造过程中使用了文献 [98] 提出的过滤方法，进行了句子级别的过滤。该方法提取并过滤文档间超过一定长度的相同字符串。给定两个文档 x_i 和 x_j，其中存在长度为 k 的公共子串 $x_i^{a\cdots a+k} = x_j^{b\cdots b+k}$。当 $k \geqslant 50$ 时，就将其中一个子串过滤。公共子串匹配的关键是如何高效完成字符串匹配，文献 [64] 将整个文档 $\mathcal{D}$ 转换为一个超长的字符串序列 $\mathcal{S}$，之后构造序列 $\mathcal{S}$ 的后缀数组（Suffix Array）A。该数组包含该序列中所有后缀按字典顺序排列的列表。具体而言，后缀数组 A 是一个整数数组，其中每个元素表示 $\mathcal{S}$ 中的一个后缀的起始位置。A 中的元素按照后缀的字典顺序排列。例如，序列 “banana” 的后缀包括 “banana”、“anana”、“nana” “ana” “na” “a”，对应的后缀数组 A 为 $[6, 4, 2, 1, 5, 3]$。根据数组 A，可以很容易地找出相同的子串。如果 $\mathcal{S}_{i\cdots i+|s|} = \mathcal{S}_{j\cdots j+|s|}$，那么 i 和 j 在数组 A 中一定在紧邻的位置上。文献 [98] 中设计了并行的后缀数组构造方法，针对 Wiki-40B 训练语料（约包含 4GB 文本内容），使用拥有 96 核 CPU 以及 768GB 内存的服务器，可以在 140 秒内完成计算。对于包含 350GB 文本的 C4 数据集，仅需要 12 小时就可以完成后缀数组构造。

在文档级别上，大部分大语言模型依靠文档之间的表面特征相似度（例如 n-gram 重叠比例）进行检测并删除重复文档[53, 57, 94, 98]。LLaMA[37] 采用 CCNet[99] 的处理模式，先将文档拆分为段落，并把所有字符转换为小写字符、将数字替换为占位符，删除所有 Unicode 标点符号和重音符号对每个段落进行规范化处理。然后，使用 SHA-1 方法为每个段落计算一个哈希码（Hash Code），并使用前 64 位数字作为键。最后，利用每个段落的键进行重复判断。RefinedWeb[64] 先去除页面中的菜单、标题、页脚、广告等内容，仅抽取页面中的主要内容。在此基础上，在文档级别进行过滤，采用与文献 [89] 类似的方法，使用 n-gram 重复程度来衡量句子、段落及文档的相似度。如果重复程度超过预先设定的阈值，则会过滤重复段落或文档。

此外，数据集级别上也可能存在一定数量的重复情况，比如很多大语言模型预训练集合都会包含 GitHub、Wikipedia、C4 等数据集。需要特别注意的是，预训练语料中混入测试语料，造成数据集污染的情况。在实际产生预训练数据时，需要从句子、文档、数据集三个级别去除重复，这对于改善语言模型的训练具有重要的作用[14, 100]。

3.2.3　隐私消除

由于绝大多数预训练数据源于互联网，因此不可避免地会包含涉及敏感或个人信息（Personally Identifiable Information，PII）的用户生成内容，这可能会增加隐私泄露的风险[101]。如图 3.3 所示，输入前缀词“East Stroudsburg Stroudsburg”，语言模型在此基础上补全了姓名、电子邮件地址、电话号码、传真号码及实际地址。这些信息都是模型从预训练语料中学习得到的。因此，非常有必要从预训练语料库中删除包含个人身份信息的内容。

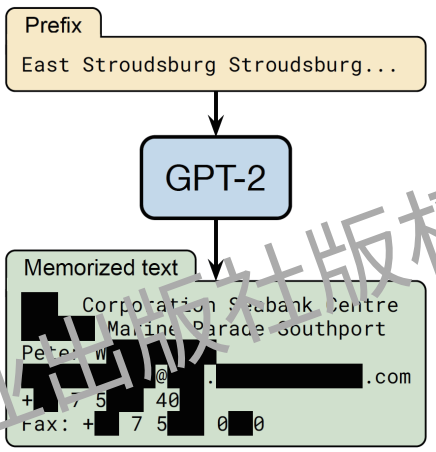

图 3.3　从大语言模型中获得隐私数据的例子[101]

删除隐私数据最直接的方法是采用基于规则的算法，BigScience ROOTS Corpus[102] 在构建过程中就采用了基于命名实体识别的方法，利用命名实体识别算法检测姓名、地址、电话号码等个人信息内容并进行删除或者替换。该方法使用了基于 Transformer 的模型，并结合机器翻译技术，可以处理超过 100 种语言的文本，消除其中的隐私信息。该方法被集成在 muliwai 类库中。

3.2.4　词元切分

传统的自然语言处理通常以单词为基本处理单元，模型都依赖预先确定的词表 $\mathbb{V}$，在编码输入词序列时，这些词表示模型只能处理词表中存在的词。因此，使用时，如果遇到不在词表中的未登录词，模型无法为其生成对应的表示，只能给予这些**未登录词**（Out-of-Vocabulary，OOV）一个默认的通用表示。在深度学习模型中，词表示模型会预先在词表中加入一个默认的“[UNK]”（unknown）标识，表示未知词，并在训练的过程中将 [UNK] 的向量作为词表示矩阵的一部分一起训练，通过引入某些相应机制来更新 [UNK] 向量的参数。使用时，对全部未登录词使用 [UNK] 向

量作为表示向量。此外，基于固定词表的词表示模型对词表大小的选择比较敏感。当词表大小过小时，未登录词的比例较高，影响模型性能；当词表大小过大时，大量低频词出现在词表中，这些词的词向量很难得到充分学习。理想模式下，词表示模型应能覆盖绝大部分的输入词，并避免词表过大所造成的数据稀疏问题。

为了缓解未登录词问题，一些工作通过利用亚词级别的信息构造词表示向量。一种直接的解决思路是为输入建立字符级别表示，并通过字符向量的组合获得每个单词的表示，以解决数据稀疏问题。然而，单词中的词根、词缀等构词模式往往跨越多个字符，基于字符表示的方法很难学习跨度较大的模式。为了充分学习这些构词模式，研究人员提出了**子词词元化**（Subword Tokenization）方法，试图缓解上文介绍的未登录词问题。词元表示模型会维护一个词元词表，其中既存在完整的单词，也存在形如“c”“re”“ing”等单词的部分信息，称为**子词**（Subword）。词元表示模型对词表中的每个词元计算一个定长向量表示，供下游模型使用。对于输入的词序列，词元表示模型将每个词拆分为词表内的词元。例如，将单词“reborn”拆分为“re”和“born”，模型随后查询每个词元的表示，将输入重新组成词元表示序列。当下游模型需要计算一个单词或词组的表示时，可以将对应范围内的词元表示合成需要的表示。因此，词元表示模型能够较好地解决自然语言处理系统中未登录词的问题。**词元分析**（Tokenization）是将原始文本分割成词元（Token）序列的过程。词元切分也是数据预处理中至关重要的一步。

字节对编码（Byte Pair Encoding，BPE）[103] 是一种常见的子词词元模型。该模型采用的词表包含最常见的单词及高频出现的子词。使用时，常见词通常位于 BPE 词表中，而罕见词通常能被分解为若干个包含在 BPE 词表中的词元，从而大幅减小未登录词的比例。BPE 算法包括以下两个部分。

（1）词元词表的确定。

（2）全词切分为词元及词元合并为全词的方法。

BPE 模型中词元词表的计算过程如图 3.4 所示。首先，确定语料库中全词的词表和词频，然后将每个单词切分为单个字符的序列，并在序列最后添加符号“</w>”作为单词结尾的标识。例如，单词“low”被切分为序列“l␣o␣w␣</w>”。所切分出的序列元素称为字节，即每个单词都切分为字节的序列。之后，按照每个字节序列的相邻字节对和单词的词频，统计每个相邻字节对的出现频率，合并出现频率最高的字节对，将其作为新的词元加入词表，并将全部单词中的该字节对合并为新的单一字节。在第一次迭代时，出现频率最高的字节对是 (e,s)，故将“es”作为词元加入词表，并将全部序列中相邻的 (e,s) 字节对合并为 es 字节。重复这一步骤，直至 BPE 词元词表的大小达到指定的预设值，或没有可合并的字节对为止。