任务处理方法、代码补全方法、代码处理模型训练方法、基于代码处理模型的信息处理方法及模型训练平台

本公开要求于 2024 年 03 月 21 日提交中国专利局、申请号为 202410331805.8、申请名称为“任务处理方法、代码补全方法、代码处理模型训练方法、基于代码处理模型的信息处理方法及模型训练平台”的中国专利申请的优先权，其全部内容通过引用结合在本公开中。

技术领域

本公开实施例涉及计算机技术领域，特别涉及任务处理方法、代码补全方法、代码处理模型训练方法、基于代码处理模型的信息处理方法及模型训练平台。

背景技术

随着计算机技术的发展，大模型开始大放异彩，在语言理解、生成、交互和推理方面表现出的非凡能力，广泛应用于对话、翻译、代码处理等自然语言处理领域。以代码处理领域为例，利用大模型可以为开发者提供代码补全、代码注释等智能化能力，因此逐渐成为研究重点。

目前，通常直接利用大模型对目标任务的代码文本进行处理，获得代码处理结果。然而，上述方法中，大模型一次需要预测很长的内容，导致预测时间过长且准确性较低，因此，亟需一种高效且精准的任务处理方案。

发明内容

有鉴于此，本公开实施例提供了一种任务处理方法。本公开一个或者多个实施例同时涉及一种代码补全方法，一种代码处理模型训练方法，一种基于代码处理模型的信息处理方法，一种任务处理装置，一种代码补全装置，一种代码处理模型训练装置，一种基于代码处理模型的信息处理装置，一种模型训练平台，一种计算设备，一种计算机可读存储介质以及一种计算机程序产品，以解决现有技术中存在的“大模型一次需要预测很长的内容，导致预测时间过长且准确性较低”的技术缺陷。

根据本公开实施例的第一方面，提供了一种任务处理方法，包括：

获取目标任务的待处理代码数据；

将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到。

根据本公开实施例的第二方面，提供了一种代码补全方法，包括：

获取代码补全任务的待补全代码数据；

将待补全代码数据输入代码补全模型中，获得代码补全任务的代码补全结果，其中，代码补全模型基于多个样本补全结果和多个样本补全结果分别对应的样本代码块数据训练得到，样本补全结果和样本代码块数据基于对样本代码数据进行层次采样得到。

根据本公开实施例的第三方面，提供了一种代码处理模型训练方法，包括：

获取样本集，其中，样本集包括多个样本代码数据；

对多个样本代码数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据；

将多个样本代码块数据输入代码处理模型，获得多个样本代码块数据分别对应的预测处理结果；

根据样本处理结果和预测处理结果，对代码处理模型进行训练，获得训练完成的代码处理模型。

根据本公开实施例的第四方面，提供了一种基于代码处理模型的信息处理方法，包括：

接收任务生成请求，其中，任务生成请求包括请求信息；

基于请求信息，获取代码处理模型，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到；

基于代码处理模型，生成任务信息，其中，任务信息用于执行目标任务。

根据本公开实施例的第五方面，提供了一种任务处理装置，包括：

第一获取模块，被配置为获取目标任务的待处理代码数据；

第一输入模块，被配置为将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到。

根据本公开实施例的第六方面，提供了一种代码补全装置，包括：

第二获取模块，被配置为获取代码补全任务的待补全代码数据；

第二输入模块，被配置为将待补全代码数据输入代码补全模型中，获得代码补全任务的代码补全结果，其中，代码补全模型基于多个样本补全结果和多个样本补全结果分别对应的样本代码块数据训练得到，样本补全结果和样本代码块数据基于对样本代码数据进行层次采样得到。

根据本公开实施例的第七方面，提供了一种代码处理模型训练装置，包括：

第三获取模块，被配置为获取样本集，其中，样本集包括多个样本代码数据；

采样模块，被配置为对多个样本代码数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据；

第三输入模块，被配置为将多个样本代码块数据输入代码处理模型，获得多个样本代码块数据分别对应的预测处理结果；

训练模块，被配置为根据样本处理结果和预测处理结果，对代码处理模型进行训练，获得训练完成的代码处理模型。

根据本公开实施例的第八方面，提供了一种基于代码处理模型的信息处理装置，包括：

第一接收模块，被配置为接收任务生成请求，其中，任务生成请求包括请求信息；

第四获取模块，被配置为基于请求信息，获取代码处理模型，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到；

生成模块，被配置为基于代码处理模型，生成任务信息，其中，任务信息用于执行目标任务。

根据本公开实施例的第九方面，提供了一种模型训练平台，包括请求接口单元和模型单元；

请求接口单元，用于接收模型训练请求，其中，模型训练请求包括请求信息；

模型单元，用于获取代码处理模型；基于请求信息，对代码处理模型进行训练，获得训练完成的代码处理模型，其中，请求信息包括多个样本处理结果和多个样本处理结果分别对应的样本代码块数据，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到。

根据本公开实施例的第十方面，提供了一种计算设备，包括：

存储器和处理器；

所述存储器用于存储计算机程序/指令，所述处理器用于执行所述计算机程序/指令，该计算机程序/指令被处理器执行时实现上述第一方面或者第二方面或者第三方面或者第四方面所提供方法的步骤。

根据本公开实施例的第十一方面，提供了一种计算机可读存储介质，其存储有计算机程序/指令，该计算机程序/指令被处理器执行时实现上述第一方面或者第二方面或者第三方面或者第四方面所提供方法的步骤。

根据本公开实施例的第十二方面，提供了一种计算机程序产品，包括计算机程序/指令，该计算机程序/指令被处理器执行时实现上述第一方面或者第二方面或者第三方面或者第四方面所提供方法的步骤。

本公开一个实施例提供的任务处理方法，包括：获取目标任务的待处理代码数据；将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到。由于代码处理模型基于样本代码块数据训练得到，实现了以代码块为单位进行预测，缩短了预测内容，并且，通过层次采样，可以使得代码处理模型适应各种代码处理场景，从而精准地预测各种类型的代码块，提高任务处理效率以及精准性。

附图说明

图 1 是本公开一个实施例提供的一种任务处理系统的架构图；

图 2 是本公开一个实施例提供的另一种任务处理系统的架构图；

图 3 是本公开一个实施例提供的一种任务处理方法的流程图；

图 4 是本公开一个实施例提供的一种代码补全方法的流程图；

图 5 是本公开一个实施例提供的一种代码处理模型训练方法的流程图；

图 6 是本公开一个实施例提供的一种代码处理模型训练方法的处理过程流程图；

图 7 是本公开一个实施例提供的一种基于代码处理模型的信息处理方法的流程图；

图 8 是本公开一个实施例提供的一种模型训练平台的结构示意图；

图 9 是本公开一个实施例提供的一种任务处理装置的结构示意图；

图 10 是本公开一个实施例提供的一种代码补全装置的结构示意图；

图 11 是本公开一个实施例提供的一种代码处理模型训练装置的结构示意图；

图 12 是本公开一个实施例提供的一种基于代码处理模型的信息处理装置的结构示意图；

图 13 是本公开一个实施例提供的一种计算设备的结构框图。

具体实施方式

在下面的描述中阐述了很多具体细节以便于充分理解本公开。但是本公开能够以很多不同于在此描述的其它方式来实施，本领域技术人员可以在不违背本公开内涵的情况下做类似推广，因此本公开不受下面公开的具体实施的限制。

在本公开一个或多个实施例中使用的术语是仅仅出于描述特定实施例的目的，而非旨在限制本公开一个或多个实施例。在本公开一个或多个实施例和所附权利要求书中所使用的单数形式的“一种”、“所述”和“该”也旨在包括多数形式，除非上下文清楚地表示其他含义。还应当理解，本公开一个或多个实施例中使用的术语“和/或”是指并包含一个或多个相关联的列出项目的任何或所有可能组合。

应当理解，尽管在本公开一个或多个实施例中可能采用术语第一、第二等来描述各种信息，但这些信息不应限于这些术语。这些术语仅用来将同一类型的信息彼此区分开。例

如，在不脱离本公开一个或多个实施例范围的情况下，第一也可以被称为第二，类似地，第二也可以被称为第一。取决于语境，如在此所使用的词语“如果”可以被解释成为“在……时”或“当……时”或“响应于确定”。

此外，需要说明的是，本公开一个或多个实施例所涉及的用户信息（包括但不限于用户设备信息、用户个人信息等）和数据（包括但不限于用于分析的数据、存储的数据、展示的数据等），均为经用户授权或者经过各方充分授权的信息和数据，并且相关数据的收集、使用和处理需要遵守相关国家和地区的相关法律法规和标准，并提供有相应的操作入口，供用户选择授权或者拒绝。

本公开一个或多个实施例中，大模型是指具有大规模模型参数的深度学习模型，通常包含上亿、上百亿、上千亿、上万亿甚至十万亿以上的模型参数。大模型又可以称为基石模型/基础模型（Foundation Model），通过大规模无标注的语料进行大模型的预训练，产出亿级以上参数的预训练模型，这种模型能适应广泛的下游任务，模型具有较好的泛化能力，例如大规模语言模型(LLM,Large Language Model)、多模态预训练模型(multi-modal pre-training model）等。

大模型在实际应用时，仅需少量样本对预训练模型进行微调即可应用于不同的任务中，大模型可以广泛应用于自然语言处理（NLP, Natural Language Processing）、计算机视觉等领域，具体可以应用于如视觉问答（VQA, Visual Question Answering）、图像描述（IC, Image Caption）、图像生成等计算机视觉领域任务，以及基于文本的情感分类、文本摘要生成、机器翻译等自然语言处理领域任务，大模型主要的应用场景包括数字助理、智能机器人、搜索、在线教育、办公软件、电子商务、智能设计等。

首先，对本公开一个或多个实施例涉及的名词术语进行解释。

智能编码助手：通过利用大模型技术，增强软件开发流程中编程任务，提升开发者的编码效率，智能编码助手主要的能力包括代码自动补全、代码问答、自动生成单元测试、自动代码优化等等。

代码自动补全：也可称为代码补全，是指在开发者编写代码的过程中，实时预测并建议可能的代码完成项，提高开发者编码速度和准确性。

抽象语法树（AST, Abstract Syntax Tree）：是源代码的一种中间表示形式，用于将源代码的结构以树状结构进行抽象和解析。

中间填充（Fill-In-Middle）：一种训练代码大模型的数据模式，为了使得模型能够理解代码上下文的情况下，预测待补全的代码内容，所构造出来的数据组织形式。

代码元数据（Code Metadata）：是代码本身的结构化数据，它包含了描述代码组件（如类、方法、属性、接口、文件名等）的附加信息，这些信息并不直接构成代码的执行逻辑，但对理解、管理和使用代码至关重要。

孩子节点（Child Node）：在抽象语法树（AST, Abstract Syntax Tree）中，一个代码块节点下的所有直接包含的子节点被称为该代码块的孩子节点。例如，在 Java 代码中，一个代码块内的每一条语句或表达式都是该代码块节点的孩子节点。

兄弟节点（Sibling Node）：在 AST 中，具有相同父节点的任意两个节点被称为兄弟节点。

在软件开发领域，一方面，编写和维护代码是一项耗时且需要高度集中的工作。智能编码助手通过代码自动补全、代码语法检查、代码生成等功能，可以显著减少开发者的手动输入时间，加快开发速度。另一方面，智能编码助手可以基于大量训练数据学习编程规范和较优实践，帮助开发者避免常见错误，并提供符合标准的高质量代码建议。

以代码自动补全为例，目前，通常可以通过以下两种方式进行代码自动补全：第一种，可以利用 Fill-In-Middle 的方法训练大模型的自动补全能力，但是在模型训练过程中，通过简单的随机采样方式在给定的源代码中随机出代码上文（prefix）的结束位置、代码

下文（suffix）的开始位置，然后将两个位置中间（middle）的所有代码作为预测结果，构造<prefix,suffix,middle>的数据进行训练。然而，上述方案中，模型对预测的middle部分的代码没有任何处理，使得模型一次性预测较多的内容，例如预测5个函数，这不仅导致预测的时间较长，而且会干扰用户的编码节奏，严重影响智能编码助手的使用体验。第二种，模型大多以单行补全的形式进行预测。然而，在某些情况下，例如开发者编写函数注释之后想完成整个函数的开发时，代码自动补全需要多次触发，这会大大降低用户的效率、体验度。

为了解决上述问题，本公开实施例提出了一种训练智能编码助手的代码处理模型的方法，通过对原始的样本代码数据进行语法解析以及层次采样，并通过Fill-In-Middle模式构造成可以训练模型的训练数据，使得模型能够根据用户编写待处理代码数据进行代码任务处理。具体地，可以获取目标任务的待处理代码数据；将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到。

值得说明的是，应用本公开实施例提出了的方案，智能编码助手可以基于先进的人工智能技术，通过深入理解开发者的编程意图和上下文信息，实现代码自动补全、代码片段推荐、错误检测与修复等功能，显著提升开发者的编码效率。由于代码处理模型基于样本代码块数据训练得到，实现了以代码块为单位进行预测，缩短了预测内容，并且，通过层次采样，可以使得代码处理模型适应各种代码处理场景（如代码补全场景、代码推荐场景、代码注释生成场景等等），从而精准地预测各种类型的代码块（如类、函数、逻辑块、注释、语句等等），提高任务处理效率以及精准性。

在本公开中，提供了一种任务处理方法，本公开同时涉及一种代码补全方法，一种代码处理模型训练方法，一种基于代码处理模型的信息处理方法，一种任务处理装置，一种代码补全装置，一种代码处理模型训练装置，一种基于代码处理模型的信息处理装置，一种模型训练平台，一种计算设备，一种计算机可读存储介质以及一种计算机程序产品，在下面的实施例中逐一进行详细说明。

参见图1，图1示出了本公开一个实施例提供的一种任务处理系统的架构图，任务处理系统可以包括客户端100和服务端200；

客户端100，用于向服务端200发送目标任务的待处理代码数据；

服务端200，用于将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到；向客户端100发送代码处理结果；

客户端100，还用于接收服务端200发送的代码处理结果。

应用本公开实施例的方案，由于代码处理模型基于样本代码块数据训练得到，实现了以代码块为单位进行预测，缩短了预测内容，并且，通过层次采样，可以使得代码处理模型适应各种代码处理场景，从而精准地预测各种类型的代码块，提高任务处理效率以及精准性。

参见图2，图2示出了本公开一个实施例提供的另一种任务处理系统的架构图，任务处理系统可以包括多个客户端100以及服务端200，其中，客户端100可以包括端侧设备，服务端200可以包括云侧设备。多个客户端100之间通过服务端200可以建立通信连接，在任务处理场景中，服务端200即用来在多个客户端100之间提供代码处理服务，多个客户端100可以分别作为发送端或接收端，通过服务端200实现通信。

用户通过客户端100可与服务端200进行交互以接收其它客户端100发送的数据，或将数据发送至其它客户端100等。在任务处理场景中，可以是用户通过客户端100向服务

端200发布数据流，服务端200根据该数据流生成代码处理结果，并将代码处理结果推送至其他建立通信的客户端中。

其中，客户端100与服务端200之间通过网络建立连接。网络为客户端100与服务端200之间提供了通信链路的介质。网络可以包括各种连接类型，例如有线、无线通信链路或者光纤电缆等等。客户端100所传输的数据可能需要经过编码、转码、压缩等处理之后才发布至服务端200。

客户端100可以为浏览器或者APP（Application，应用程序）或者网页应用如H5（HyperText Markup Language5，超文本标记语言第5版）应用或者轻应用（也被称为小程序，一种轻量级应用程序）或者云应用等，客户端100可以基于服务端200提供的相应服务的软件开发工具包（SDK，Software Development Kit），如基于实时通信（RTC，Real Time Communication）SDK开发获得等。客户端100可以部署在电子设备中，需要依赖设备运行或者设备中的某些APP而运行等。电子设备例如可以具有显示屏并支持信息浏览等，如可以是个人移动终端如手机、平板电脑、个人计算机等。在电子设备中通常还可以配置各种其它类应用，例如人机对话类应用、模型训练类应用、文本处理类应用、网页浏览器应用、购物类应用、搜索类应用、即时通信工具、邮箱客户端、社交平台软件等。

服务端200可以包括提供各种服务的服务器，例如为多个客户端提供通信服务的服务器，又如为客户端上使用的模型提供支持的用于后台训练的服务器，又如对客户端发送的数据进行处理的服务器等。需要说明的是，服务端200可以实现成多个服务器组成的分布式服务器集群，也可以实现成单个服务器。服务器也可以为分布式系统的服务器，或者是结合了区块链的服务器。服务器也可以是云服务、云数据库、云计算、云函数、云存储、网络服务、云通信、中间件服务、域名服务、安全服务、内容分发网络（CDN，Content Delivery Network）以及大数据和人工智能平台等基础云计算服务的云服务器，或者是带人工智能技术的智能云计算服务器或智能云主机。

值得说明的是，本公开实施例中提供的任务处理方法一般由服务端执行，但是，在本公开的其它实施例中，客户端也可以与服务端具有相似的功能，从而执行本公开实施例所提供的任务处理方法。在其它实施例中，本公开实施例所提供的任务处理方法还可以是由客户端与服务端共同执行。

参见图3，图3示出了本公开一个实施例提供的一种任务处理方法的流程图，具体包括以下步骤：

步骤302：获取目标任务的待处理代码数据。

本公开一个或多个实施例中，可以获取目标任务的待处理代码数据，从而基于目标任务对待处理代码数据基于语法解析和层次采样的任务处理方法进行处理，获得目标任务的代码处理结果。

具体地，目标任务是指代码处理任务，目标任务包括但不限于代码补全任务、代码推荐任务、代码注释生成任务。待处理代码数据可以是多语言的代码数据，如Python、Java等等。待处理代码数据可以是代码文本数据，也可以是代码块数据，具体根据实际情况进行选择，本公开实施例对此不做任何限定。代码块数据可以称为代码区块（Code Block）数据。代码区块是指较为完整的代码语义逻辑区块，代码区块可以是一行，也可以是多行。有些代码区块会有区块头，如if区块的条件语句，方法定义区块的方法定义。代码区块包括坐标范围，方便从原始代码中截取区块代码，部分代码区块定义有唯一的名称，方便在引用还原或测试用例关联时，通过唯一名称将引用方和定义相关联。

实际应用中，获取目标任务的待处理代码数据的方式有多种，具体根据实际情况进行选择，本公开实施例对此不做任何限定。本公开一种可能的实现方式中，可以接收用户发送的目标任务的待处理代码数据。本公开另一种可能的实现方式中，可以从其他数据获取设备或数据库中读取目标任务的待处理代码数据。

步骤 304：将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到。

本公开一个或多个实施例中，获取目标任务的待处理代码数据之后，进一步地，可以将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果。

具体地，代码处理模型可以是预训练代码大模型，也可以是基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据对大模型训练得到的机器学习模型，代码处理模型如双向编码表示的深度自注意力模型（BERT，Bidirectional Encoder Representations from Transformers）模型、文本对文本的深度自注意力模型（T5，Text-to-Text Transfer Transformer）模型。任务处理结果与目标任务对应，若目标任务为代码补全任务，则任务处理结果为代码补全结果，若目标任务为代码注释生成任务，则任务处理结果为代码注释结果。

需要说明的是，代码处理模型包括编码单元和解码单元，将待处理代码数据输入代码处理模型，首先，编码单元通过自注意力机制捕获待处理代码数据内部各元素间的复杂依赖关系，并逐层将待处理代码数据转换为高级别的语义嵌入表示，得到编码向量。接着，解码单元通过自注意力机制和掩码机制对编码单元提供的编码向量逐步解码，得到代码处理结果。

本公开一种可选的实施例中，将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果之前，还可以对待处理代码数据进行语法解析，获得多个待处理代码块数据，进一步地，可以将多个待处理代码块数据输入代码处理模型中，获得目标任务的代码处理结果。

应用本公开实施例的方案，由于代码处理模型基于样本代码块数据训练得到，实现了以代码块为单位进行预测，缩短了预测内容，并且，通过层次采样，可以使得代码处理模型适应各种代码处理场景，从而精准地预测各种类型的代码块，提高任务处理效率以及精准性。

实际应用中，将待处理代码数据输入代码处理模型时，可以直接将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果。进一步地，为了保证代码处理模型可以精准地对待处理代码数据进行处理，还可以获取代码处理模型的处理提示信息，利用处理提示信息规范待处理代码数据。

本公开一种可选的实施例中，上述将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果，可以包括以下步骤：

获取代码处理模型的处理提示信息；

将处理提示信息和待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果。

具体地，处理提示信息用于提示引导代码处理模型对待处理代码数据进行处理。处理提示信息可以包括代码元数据，如代码文件名，处理提示信息具体根据实际情况进行设置，本公开实施例对此不做任何限定。通过代码文件名，可以使得代码处理模型了解待处理代码数据的语言类型，从而基于语言类型对应的语法信息对待处理代码数据进行精准处理。

实际应用中，获取代码处理模型的处理提示信息的方式有多种，具体根据实际情况进行选择，本公开实施例对此不做任何限定。本公开一种可能的实现方式中，可以接收用户发送的代码处理模型的处理提示信息。本公开另一种可能的实现方式中，可以从其他数据获取设备或数据库中读取代码处理模型的处理提示信息。

应用本公开实施例的方案，获取代码处理模型的处理提示信息；将处理提示信息和待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果。通过将处理提示信息输入代码处理模型，使得代码处理模型更加了解待处理代码数据，保证了代码处理结果的

精准性。

本公开一种可选的实施例中，上述将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果之后，还可以包括以下步骤：

将代码处理结果发送至前端用户；

接收前端用户发送的结果反馈信息，其中，结果反馈信息为基于目标任务的任务信息对代码处理结果进行反馈的信息；

根据结果反馈信息，构建模型优化数据；

利用模型优化数据，对代码处理模型进行参数调整。

具体地，结果反馈信息可以是对代码处理结果的内容、质量及完成度进行反馈的信息，反映了前端用户对代码处理结果的真实感受和期望，结果反馈信息包括但不限于结果质量评价信息、修正后的准确的代码处理结果、模型的优化领域。模型优化数据是指用于优化代码处理模型的准确的优化样本数据。

实际应用中，根据结果反馈信息，构建模型优化数据的方式有多种，具体根据实际情况进行选择，本公开实施例对此不做任何限定。本公开一种可能的实现方式中，可以直接根据结果反馈信息，构建模型优化数据。本公开另一种可能的实现方式中，可以基于结果反馈信息生成优化提示信息，接收前端用户基于优化提示信息发送的模型优化数据。

需要说明的是，以直接根据结果反馈信息，构建模型优化数据为例，若结果反馈信息为准确的修正后的准确的代码处理结果，则可以根据目标任务的待处理代码数据和修正后的准确的代码处理结果构建模型优化数据。若结果反馈信息为模型的优化领域，如 XXX 领域，则可以获取 XXX 领域的样本代码数据，将 XXX 领域的样本代码数据确定为模型优化数据。其中，利用模型优化数据，对代码处理模型进行参数调整的过程与代码处理模型的训练过程相同，本公开实施例便不再进行赘述。

应用本公开实施例的方案，将代码处理结果发送至前端用户；接收前端用户发送的结果反馈信息，其中，结果反馈信息为基于目标任务的任务信息对代码处理结果进行反馈的信息；根据结果反馈信息，构建模型优化数据；利用模型优化数据，对代码处理模型进行参数调整。通过收集和利用结果反馈信息，从而不断优化代码处理模型的表现，以更准确地满足前端用户的实际需求，并提升最终代码处理结果的质量和准确性。

本公开一种可选的实施例中，上述根据结果反馈信息，构建模型优化数据，可以包括以下步骤：

根据结果反馈信息，生成优化提示信息，其中，优化提示信息用于引导前端用户发送用于优化代码处理模型的模型优化数据；

将优化提示信息发送至前端用户，并接收前端用户基于优化提示信息发送的模型优化数据。

需要说明的是，根据结果反馈信息，生成优化提示信息的方式有多种，具体根据实际情况进行选择，本公开实施例对此不做任何限定。

本公开一种可能的实现方式中，可以直接获取预设提示信息，并将结果反馈信息添加预设提示信息中，得到优化提示信息，如预设提示信息为“非常抱歉给您带来了不准确的信息。请您指出具体哪里不够准确或者提供相关问题的正确答案，我将尽快修正并优化我的回答，以便更好地为您服务”。结果反馈信息为“结果不准”，则优化提示信息为“针对您反馈的结果不准的问题，非常抱歉给您带来了不准确的信息。请您指出具体哪里不够准确或者提供相关问题的正确答案，我将尽快修正并优化我的回答，以便更好地为您服务”。

本公开另一种可能的实现方式中，可以对结果反馈信息进行类型识别，确定结果反馈信息的信息类型，进一步将信息类型与提示信息库中各提示信息的提示类型进行匹配，并将提示类型与信息类型相同的提示信息确定为优化提示信息。

应用本公开实施例的方案，根据结果反馈信息，生成优化提示信息，其中，优化提示信息用于引导前端用户发送用于优化代码处理模型的模型优化数据；将优化提示信息发送至前端用户，并接收前端用户基于优化提示信息发送的模型优化数据。通过交互式的引导方式获取模型优化数据，提高了与用户之间的交互性，提高了用户满意度。

本公开一种可选的实施例中，上述将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果之后，还可以包括以下步骤：

对代码处理结果中的关键信息进行标注，获得更新后的代码处理结果；

将更新后的代码处理结果发送至前端用户。

具体地，关键信息是指代码处理结果中有助于用户理解、调试、维护代码的信息，如函数、类、对象等等。

实际应用中，对代码处理结果中的关键信息进行标注，获得更新后的代码处理结果之前，可以对代码处理结果进行关键信息识别，确定代码处理结果中的关键信息。其中，对代码处理结果进行关键信息识别的方式有多种，具体根据实际情况进行选择，本公开实施例对此不做任何限定。本公开一种可能的实现方式中，可以将代码处理结果与预设关键信息进行匹配，将代码处理结果中出现的与预设关键信息相同的信息确定为关键信息。本公开另一种可能的实现方式中，可以将代码处理结果输入关键信息识别模型，获得关键信息，其中，关键信息识别模型基于多个训练代码数据和各训练代码数据的关键信息标签训练得到。

需要说明的是，对代码处理结果中的关键信息进行标注时，可以对关键文本信息进行加粗、高亮、倾斜、添加边框等操作，关键信息的标注方式具体根据实际情况进行选择，本公开实施例对此不做任何限定。

应用本公开实施例的方案，对代码处理结果中的关键信息进行标注，获得更新后的代码处理结果；将更新后的代码处理结果发送至前端用户，以便于前端用户进行结果查看，提高了用户体验度。

本公开一种可选的实施例中，上述将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果之后，还可以包括以下步骤：

将代码处理结果发送至前端用户；

接收前端用户发送的修改信息，其中，修改信息用于对代码处理结果进行修改；

根据修改信息，对代码处理结果进行修改，获得修改后的代码处理结果。

具体地，修改信息用于描述前端用户对代码处理结果的修改需求。修改需求包括但不限于添加注释需求、生成测试信息需求。进一步地，修改信息还可以是针对代码处理结果中的关键信息的修改需求信息。

实际应用中，根据修改信息，对代码处理结果进行修改的方式有多种，具体根据实际情况进行选择，本公开实施例对此不做任何限定。本公开一种可能的实现方式中，可以调用代码修改工具（如文本编辑器）基于修改信息对代码处理结果进行修改。本公开另一种可能的实现方式中，可以将任务处理结果和修改信息输入修改内容生成模型，获得修改信息对应的修改内容，并利用修改内容更新代码处理结果，获得修改后的代码处理结果，其中，修改内容生成模型基于多个样本代码和各样本代码的样本修改信息和真实修改结果训练得到。

应用本公开实施例的方案，将代码处理结果发送至前端用户；接收前端用户发送的修改信息，其中，修改信息用于对代码处理结果进行修改；根据修改信息，对代码处理结果进行修改，获得修改后的代码处理结果。通过根据前端用户发送的修改信息对代码处理结果进行修改，增加了人机交互性，也提升了任务处理的适应性和灵活性。

本公开一种可选的实施例中，对代码处理模型的训练方式进行说明，也即，上述将待

处理代码数据输入代码处理模型中，获得目标任务的代码处理结果之前，还可以包括以下步骤：

获取样本集，其中，样本集包括多个样本代码数据；

对多个样本代码数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据；

将多个样本代码块数据输入代码处理模型，获得多个样本代码块数据分别对应的预测处理结果；

根据样本处理结果和预测处理结果，对代码处理模型进行训练，获得训练完成的代码处理模型。

具体地，代码处理模型的训练方式为有监督训练，也即，样本代码块数据是携带真实处理标签（样本处理结果）的，样本处理结果为代码处理模型的生成目标，用于指导代码处理模型的训练过程。样本代码块数据可以包括样本处理结果的代码块上文和代码块下文，通过样本代码块数据，模型可以预测样本处理结果。代码处理模型可以是预先训练的大模型，也可以是未经训练的自然语言的神经网络模型。样本代码数据可以称为源代码数据，样本代码数据涉及各种各样的编程语言、领域，这样可以使得模型在丰富的源代码数据中进行学习，使得训练过程中见到的数据与开发者的真实开发场景接近，同时提升模型的知识密度和泛化能力，在任何开发者的任何开发场景中都能提供有价值的代码处理建议。

值得说明的是，在真实的代码处理场景中，由于代码块之间有相互嵌套的关系，例如SortAlgorithm 类里面包含两个函数，第一个函数里面又包含 2 条语句和 1 个逻辑块等内容。相互嵌套的代码块之间的数据特征属于不同层级，不同层级的数据特征可以看作一种变量，这些变量的每种取值情况可能都需要各种组合，采样的时候，可以尽可能地考虑到这些变量，以及这些变量的组合情况，从而模拟出更丰富的上下文场景，提升采样后的模型训练数据的泛化性。

因此，为了让模型在各种开发场景下提供有价值的代码处理建议，本公开实施例提出了一种层次采样的方式，也即，基于代码不同层级的数据特征进行层次采样，构造出各个语言、各种上下文场景下的模型训练数据（样本代码块数据和样本处理结果）。层次采样时，针对不同层级的数据特征，可以随机采样，也可以根据用户实际需求进行有目的的采样。采样后得到的模型训练数据可以以<prefix+suffix,middle>的形式输出，以供模型进行持续训练。例如，不同层级的数据特征包括处理主题和上下文场景，对多个样本代码数据进行层次采样时，可以先随机采样代码处理主题，接着，无论在哪一种代码处理主题下，需要找到一个代码块数据作为模型生成目标（样本处理结果），也即，在采样得到的代码处理主题下，可以随机采样上下文场景确定模型生成目标。

进一步地，对多个样本代码数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据之后，可以直接将多个样本代码块数据输入代码处理模型，获得多个样本代码块数据分别对应的预测处理结果。还可以根据预设格式将样本处理结果和样本处理结果对应的样本代码块数据组织成模型训练数据，并利用这些模型训练数据持续训练代码处理模型，接着将训练完成的代码处理模型部署至智能编码助手。其中，预设格式如“<filename>代码文件名<fim_prefix>代码块上文<fim_suffix>代码块下文<fim_middle>目标代码块”，代码块上文和代码块下文即为样本代码块数据，目标代码块即为样本处理结果。

需要说明的是，获取样本集的方式有多种，具体根据实际情况进行选择，本公开实施例对此不做任何限定。本公开一种可能的实现方式中，可以从其他数据获取设备或者数据库(如包含各种语言的项目的源代码数据的 The Stack 数据集)中读取大量样本代码数据。本公开另一种可能的实现方式中，可以从网络上爬取大量样本代码数据。

实际应用中，根据样本处理结果和预测处理结果，对代码处理模型进行训练时，可以

根据样本处理结果和预测处理结果计算损失值，根据损失值调整代码处理模型的模型参数，直至训练过程满足预设停止条件，得到训练完成的代码处理模型。其中，计算损失值的函数有很多，如交叉熵损失函数、L1 范数损失函数、最大损失函数、均方误差损失函数、对数损失函数等，具体根据实际情况进行选择，本公开实施例对此不做任何限定。预设停止条件包括但不限于损失值小于或等于预设阈值、迭代次数达到预设迭代次数，其中，预设阈值和预设迭代次数具体根据实际情况进行选择，本公开实施例对此不做任何限定。

本公开一种可能的实现方式中，计算损失值之后，将损失值与预设阈值进行比较。具体地，若损失值大于预设阈值，则说明样本处理结果和预测处理结果之间的差异较大，代码处理模型对于样本代码块数据的预测能力较差，此时可以调整代码处理模型的模型参数，并返回执行将多个样本代码块数据输入代码处理模型，获得多个样本代码块数据分别对应的预测处理结果的步骤，继续对代码处理模型进行训练，直至损失值小于或等于预设阈值，说明样本处理结果和预测处理结果之间的差异较小，达到预设停止条件，获得完成训练的代码处理模型。

本公开另一种可能的实现方式中，除了比较损失值和预设阈值的大小关系之外，还可以结合迭代次数，确定当前的代码处理模型是否训练完成。具体地，若损失值大于预设阈值，则调整代码处理模型的模型参数，并返回执行将多个样本代码块数据输入代码处理模型，获得多个样本代码块数据分别对应的预测处理结果的步骤，继续对代码处理模型进行训练，直至达到预设迭代次数的情况下，停止迭代，得到完成训练的代码处理模型。

应用本公开实施例的方案，根据样本处理结果和预测处理结果，对代码处理模型进行训练，在不满足预设停止条件的情况下继续训练代码处理模型，直至达到预设停止条件，完成训练获得代码处理模型。通过不断对代码处理模型的模型参数进行调整，能使最终得到的代码处理模型更加精准。

实际应用中，对多个样本代码数据进行层次采样的方式有多种，具体根据实际情况进行选择，本公开实施例对此不做任何限定。本公开一种可能的实现方式中，可以直接对样本代码数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据。本公开另一种可能的实现方式中，为了保证可以以代码块为单位进行层次采样，可以对多个候选代码块数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据，其中，多个候选代码块数据基于对样本代码数据进行语法解析得到。

本公开一种可选的实施例中，对多个样本代码数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据，可以包括以下步骤：

针对第一样本代码数据，对第一样本代码数据进行语法解析，获得多个候选代码块数据，其中，第一样本代码数据为多个样本代码数据中的任一个；

对多个候选代码块数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据。

具体地，语法解析用于确定代码数据的多个代码块。通过代码块可以使得模型理解代码数据中哪块代码是一个类、哪块代码是一个函数、哪块代码是一段逻辑块等等，还可以使模型对生成的结果有语法树感知，明确生成的内容应该是什么粒度。

实际应用中，对第一样本代码数据进行语法解析，获得多个候选代码块数据的方式有多种，具体根据实际情况进行选择，本公开实施例对此不做任何限定。本公开一种可能的实现方式中，可以将第一样本代码数据与预设代码块数据进行匹配，将第一样本代码数据中出现的与预设代码块数据相同的数据确定为候选代码块数据。本公开另一种可能的实现方式中，可以利用 AST 解析工具对第一样本代码数据进行语法解析，获得多个候选代码块数据，其中，抽象语法树是编程语言源代码的一种结构化表示形式。

需要说明的是，对多个候选代码块数据进行层次采样时，可以通过上下文感知的层次

化采样方法，从涉及不同主题、上下文场景、完整度以及补全粒度的多个候选代码块数据中，采样得到样本处理结果和各样本处理结果对应的样本代码块数据。

应用本公开实施例的方案，针对第一样本代码数据，对第一样本代码数据进行语法解析，获得多个候选代码块数据，其中，第一样本代码数据为多个样本代码数据中的任一个；对多个候选代码块数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据，通过基于语法解析和丰富场景层次化采样的方法，构造出各个语言、各种上下文场景下的模型训练数据，使得模型训练数据更加符合真实开发场景。

本公开一种可选的实施例中，以分为代码处理主题和代码采样策略的层次采样方式为例，对层次采样的具体实现过程进行说明，也即，上述对多个候选代码块数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据，可以包括以下步骤：

从代码处理主题中采样得到目标代码处理主题，其中，代码处理主题包括代码块处理主题、代码块内容处理主题、代码块部分内容处理主题、代码块空内容处理主题中的至少一种；

在目标代码处理主题下，根据代码采样策略，从多个候选代码块数据中采样得到多个样本处理结果和多个样本处理结果分别对应的样本代码块数据。

具体地，代码采样策略是指基于代码上下文场景进行采样的策略。代码上下文场景是指由代码上下文内容所构成的场景。代码处理主题是指每次采样的代码数据的处理主题。代码处理主题可以用来描述模型对代码数据的目标处理范围。以代码补全主题为例，代码补全主题包括代码块补全主题、代码块内容补全主题、代码块部分内容补全主题、代码块空内容补全主题中的至少一种。

代码块处理主题表征了在该主题下，模型的目标处理范围是单个代码块。以代码块补全主题为例，在代码块补全主题下，模型的目标是补全一个完整的代码块。例如补全一个完整的函数，或者补全一个完整的逻辑块。

代码块内容处理主题表征了在该主题下，模型的目标处理范围是单个代码块内部的完整内容。以代码块内容补全主题为例，在代码块内容补全主题下，模型的目标是在一个代码块内部进行内容的完全补全，也即补全全部内容。例如对去除掉中间一部分的语句进行补全，最终补全出这个代码块的完整内容。

代码块部分内容处理主题表征了在该主题下，模型的目标处理范围是单个代码块内部的部分内容。以代码块部分内容补全主题为例，在代码块部分内容补全主题下，模型的目标是在一个代码块内部补全部分内容。例如，当前一条不完整的语句是“arr[]”，仅补全[]括号内的内容，[]括号外的内容不需要补全。

代码块空内容处理主题表征了在该主题下，模型的目标处理范围是空内容，也即不处理任何内容。以代码块空内容补全主题为例，在代码块空内容补全主题下，模型的目标是在一个完整的上下文中不补全任何内容。例如，当前语句是“result = sort.bubble_sort(arr)”，当光标位于“arr”后以及“)”前时，模型预测补全空内容。

应用本公开实施例的方案，从代码处理主题中采样得到目标代码处理主题；在目标代码处理主题下，根据代码采样策略，从多个候选代码块数据中采样得到多个样本处理结果和多个样本处理结果分别对应的样本代码块数据。通过丰富的场景层次化采样的方法，构造出包括不同语言（如 Python 语言、Java 语言等等）、不同上下文场景下的模型训练数据，使得模型训练数据更加符合真实开发场景。

实际应用中，在目标代码处理主题下，根据代码采样策略，从多个候选代码块数据中采样得到多个样本处理结果和多个样本处理结果分别对应的样本代码块数据时，可以从多个代码采样策略中随机选择至少一种策略，利用选择得到的策略从多个候选代码块数据中采样得到多个样本处理结果和多个样本处理结果分别对应的样本代码块数据。进一步地，

为了保证采样得到的样本处理结果和样本代码块数据的准确性，可以将代码采样策略划分为用于采样得到样本处理结果的结果采样策略和用于采样得到样本代码块数据的代码块采样策略，从而分别从结果采样策略和代码块采样策略中随机选择至少一个策略，分别利用选择得到的策略采样得到样本处理结果和样本代码块数据。

本公开一种可选的实施例中，代码采样策略包括结果采样策略和代码块采样策略，结果采样策略包括类别采样策略、上文类别采样策略、下文类别采样策略、孩子节点采样策略和完整度采样策略中的至少一种，代码块采样策略包括上文内容采样策略和下文内容采样策略中的至少一种；上述在目标代码处理主题下，根据代码采样策略，从多个候选代码块数据中采样得到多个样本处理结果和多个样本处理结果分别对应的样本代码块数据，可以包括以下步骤：

在目标代码处理主题下，根据结果采样策略，从多个候选代码块数据中采样得到多个样本处理结果，并根据代码块采样策略，从多个候选代码块数据中采样得到多个样本处理结果分别对应的样本代码块数据。

类别采样策略是指基于类别变量进行样本处理结果采样的策略。类别包括类、函数、逻辑块、注释、语句等等。通过类别采样策略，可以从多个候选代码块数据中选择某一种类别的代码块作为样本处理结果。

上文类别采样策略是指基于前一个代码块的类别变量进行样本处理结果采样的策略。通过上文类别采样策略，可以从多个候选代码块数据中选择前一个代码块是某一种类别的代码块作为样本处理结果。例如，当前一个代码块是注释，样本处理结果是函数的时候，代表用户的编码场景是：用户写了一个函数注释，接下来需要写对应的完整函数。

下文类别采样策略是指基于后一个代码块的类别变量进行样本处理结果采样的策略。通过下文类别采样策略，可以从多个候选代码块数据中选择后一个代码块是某一种类别的代码块作为样本处理结果。例如，样本处理结果是注释，后一个代码块是类的时候，代表用户的编码场景是：用户写好了类，想为这个类添加 documentation 注释。

孩子节点采样策略是指基于是否包含孩子节点或者包含孩子节点的数目进行样本处理结果采样的策略。通过孩子节点采样策略，可以从多个候选代码块数据中选择包含孩子节点或者不包含孩子节点或者孩子节点数目大于预设数目的代码块作为样本处理结果。例如，有的代码块包含孩子节点，有的代码块不包含孩子节点，那么采样的时候可以对这两种情况进行区分，使得每种情况都有模型训练数据。进一步地，代码块包含的孩子节点的数目可能不同，包含孩子节点越多的代码块难度越大，包含孩子节点越少的代码块难度越小。例如，某些函数可能只包含一条 return 语句，某些函数可能包含各种逻辑块、语句、注释等几十个孩子节点，他们的难度是不一样的，采样的时候可以既采样简单的代码块，也采样复杂的代码块。

完整度采样策略是指基于代码块的完整度进行样本处理结果采样的策略。通过完整度采样策略，可以从多个候选代码块数据中选择不同完整度的代码块作为样本处理结果。例如，当补全的样本处理结果是一个函数的时候，可能会遇到下面三种情况：第一种，用户没有输入待补全的函数的任何内容；第二种，用户输入了待补全的函数的一个缩进(tab)；第三种，用户输入了待补全的函数的部分内容。当开发者编码时，可能会遇到任意情况，因此，可以采样不同完整度的代码块作为样本处理结果。

上文内容采样策略是指基于样本处理结果的上文内容进行样本代码块数据采样的策略。样本处理结果的上文内容可以包括以下几种：第一种，样本代码数据的原始上文内容；第二种，样本处理结果的后 K 个孩子节点的上文内容；第三种，样本处理结果的后 K 个兄弟节点的上文内容；第四种，上文内容为空。通过不同的上文内容，可以采样得到同一样本处理结果的不同的样本代码块数据。其中，K 为正整数。

下文内容采样策略：是指基于样本处理结果的下文内容进行样本代码块数据采样的策

略。样本处理结果的下文内容可以包括以下几种：第一种，样本代码数据的原始下文内容；第二种，样本处理结果的后K个孩子节点的下文内容；第三种，样本处理结果的后K个兄弟节点的下文内容；第四种，下文内容为空。通过不同的下文内容，可以采样得到同一样本处理结果的不同的样本代码块数据。

应用本公开实施例的方案，在目标代码处理主题下，根据结果采样策略，从多个候选代码块数据中采样得到多个样本处理结果，并根据代码块采样策略，从多个候选代码块数据中采样得到多个样本处理结果分别对应的样本代码块数据。类别采样策略是指基于类别变量进行样本处理结果采样的策略。类别包括类、函数、逻辑块、注释、语句等等。通过类别采样策略，可以从多个候选代码块数据中选择某一种类别的代码块作为样本处理结果。通过目标代码处理主题，可以模拟代码处理场景下不同的处理范围，通过结果采样策略，可以采样得到更加丰富、更加符合真实开发场景的样本处理结果，通过代码块采样策略可以采样得到更加丰富、更加符合真实开发场景的样本代码块数据，从而使得模型训练数据更加符合真实开发场景。

下述结合附图4，以本公开提供的任务处理方法在代码自动补全场景的应用为例，对所述任务处理方法进行进一步说明。其中，图4示出了本公开一个实施例提供的一种代码补全方法的流程图，具体包括以下步骤：

步骤402：获取代码补全任务的待补全代码数据。

步骤404：将待补全代码数据输入代码补全模型中，获得代码补全任务的代码补全结果，其中，代码补全模型基于多个样本补全结果和多个样本补全结果分别对应的样本代码块数据训练得到，样本补全结果和样本代码块数据基于对样本代码数据进行层次采样得到。

需要说明的是，步骤402至步骤404的实现方式与步骤302至步骤304的实现方式相同，待补全代码数据、代码补全模型的相关描述均可参考上述待处理代码数据和代码处理模型的相关描述，本公开实施例便不再进行赘述。

应用本公开实施例的方案，针对代码补全模型所需要满足的能力，通过基于代码语法分析和丰富场景的层次采样方法，构造出模型训练数据，并利用模型训练数据训练得到代码补全模型，使得代码补全模型在真实开发的各种场景下，精准地根据用户编写的代码上文、代码下文以及一些代码元数据，准确预测用户下一个想要输入的各种类型的代码块（可能是一个类、函数、逻辑块、注释、语句等），从而提升用户编码效率和体验。

本公开一种可选的实施例中，上述获取代码补全任务的待补全代码数据之前，还可以包括以下步骤：

向前端用户发送补全提示信息，其中，补全提示信息用于引导前端用户发送用于代码补全任务处理的待补全代码数据；

接收前端用户基于补全提示信息发送的代码补全任务的待补全代码数据。

需要说明的是，代码补全任务开始时，可以先向前端用户发送补全提示信息，前端用户可以基于补全提示信息了解代码补全任务的处理流程或处理要求，从而输入与实际需求更加贴切的待补全代码数据。例如，代码补全信息可以是“你好呀，我是你的智能编码助手，已为您准备好补全各类代码，您可以直接输入想要补全的待补全代码数据”。

进一步地，向前端用户发送补全提示信息之后，可以接收前端用户基于补全提示信息发送的代码补全任务的待补全代码数据，将待补全代码数据输入代码补全模型，获得代码补全任务的代码补全结果。

应用本公开实施例的方案，向前端用户发送补全提示信息，其中，补全提示信息用于引导前端用户发送用于代码补全任务处理的待补全代码数据；接收前端用户基于补全提示信息发送的代码补全任务的待补全代码数据。通过补全提示信息引导前端用户输入所要补全的待补全代码数据，从而增加了人机交互，使得待补全代码数据能够准确反映用户的实际需求。

本公开一种可选的实施例中，上述将待补全代码数据输入代码补全模型中，获得代码补全任务的代码补全结果之后，还可以包括以下步骤：

将代码补全结果发送至前端用户；

接收前端用户发送的结果反馈信息，其中，结果反馈信息为基于代码补全任务的任务信息对代码补全结果进行反馈的信息；

将结果反馈信息发送至模型训练平台，其中，模型训练平台用于利用结果反馈信息对代码补全模型进行参数调整。

具体地，模型训练平台可以部署在终端设备，也可以部署在云端设备，具体根据实际情况进行设置，本公开实施例对此不做任何限定。模型训练平台提供了大规模数据处理和高性能计算资源，用于训练、优化和部署各类机器学习模型，特别是深度学习模型。在模型训练平台上，用户可以上传数据、选择或定制算法模型，并通过分布式计算能力高效地进行模型训练与验证。模型训练平台可以接收来自终端设备的任务生成请求，根据请求信息获取相应的代码处理模型，并基于代码处理模型生成任务信息。模型训练平台还可以快速响应不同任务的需求，调用适当的代码处理模型进行任务处理，最终生成高质量的代码处理结果。

实际应用中，接收前端用户发送的结果反馈信息之后，可以将结果反馈信息发送至模型训练平台，模型训练平台可以基于结果反馈信息对代码补全模型进行参数调整。模型训练平台基于结果反馈信息对代码补全模型进行参数调整的方式可以参考上述“根据结果反馈信息，构建模型优化数据；利用模型优化数据，对代码处理模型进行参数调整”的实现方式，本公开实施例便不再进行赘述。

应用本公开实施例的方案，通过模型训练平台基于结果反馈信息对代码补全模型进行参数调整，能够在保证代码补全模型精度和训练效率的同时，降低模型训练成本，为用户提供便捷高效的模型训练服务。

参见图5，图5示出了本公开一个实施例提供的一种代码处理模型训练方法的流程图，具体包括以下步骤：

步骤502：获取样本集，其中，样本集包括多个样本代码数据。

步骤504：对多个样本代码数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据。

步骤506：将多个样本代码块数据输入代码处理模型，获得多个样本代码块数据分别对应的预测处理结果。

步骤508：根据样本处理结果和预测处理结果，对代码处理模型进行训练，获得训练完成的代码处理模型。

需要说明的是，步骤502至步骤508的实现方式可以参考上述代码处理模型的训练方式，本公开实施例便不再进行赘述。

实际应用中，获得训练完成的代码处理模型之后，可以将训练完成的代码处理模型的模型参数发送至终端设备，前端用户可以基于模型参数在本地构建代码处理模型，利用代码处理模型进行代码补全等任务。

应用本公开实施例的方案，根据样本处理结果和预测处理结果，对代码处理模型进行训练，在不满足预设停止条件的情况下继续训练代码处理模型，直至达到预设停止条件，完成训练获得代码处理模型。通过不断对代码处理模型的模型参数进行调整，能使最终得到的代码处理模型更加精准。

参见图6，图6示出了本公开一个实施例提供的一种代码处理模型训练方法的处理过程流程图，具体包括：

收集源代码数据：收集涉及各种各样的编程语言、领域的源代码数据作为样本代码数

据，可以使得模型在丰富的源代码数据中进行学习，保证模型训练过程中见到的数据与用户真实开发场景接近，同时提升模型的知识密度和泛化能力，在任何开发者的任何开发场景中都能提供有价值的补全建议；

对源代码数据进行 AST 语法解析：针对第一样本代码数据，对第一样本代码数据进行语法解析，获得多个候选代码块数据，其中，第一样本代码数据为多个样本代码数据中的任一个。通过对源代码数据进行语法解析，可以让模型理解代码数据中哪块代码是一个类、哪块代码是一个函数、哪块代码是一段逻辑块，并且对生成的结果有语法树感知，明确生成的内容应该是什么粒度，从而只预测下一个代码块内容（一个函数或逻辑块或一条语句），使得预测的内容大幅缩短并且精准满足用户的需求，在不干扰用户编码节奏的同时，给予用户更有价值的代码处理建议；

多场景层次化采样构造训练数据：从代码处理主题中采样得到目标代码处理主题，其中，代码处理主题包括代码块处理主题、代码块内容处理主题、代码块部分内容处理主题、代码块空内容处理主题中的至少一种；在目标代码处理主题下，根据结果采样策略，从多个候选代码块数据中采样得到多个样本处理结果，并根据代码块采样策略，从多个候选代码块数据中采样得到多个样本处理结果分别对应的样本代码块数据，其中，结果采样策略包括类别采样策略、上文类别采样策略、下文类别采样策略、孩子节点采样策略和完整度采样策略中的至少一种，代码块采样策略包括上文内容采样策略和下文内容采样策略中的至少一种；通过层次化采样方法，对进行语法解析后得到的多个候选代码块数据进行采样，构造出各个语言、各种上下文场景下的模型训练数据；

训练模型：将多个样本代码块数据输入代码处理模型，获得多个样本代码块数据分别对应的预测处理结果；根据样本处理结果和预测处理结果，对代码处理模型进行训练，获得训练完成的代码处理模型。

应用本公开实施例的方案，通过 AST 语法解析和层次化采样方法，实现了在不同主题、不同上下文场景下，精准地预测用户想要处理的逻辑块类型，使得模型可以预测一个完整的类、函数、逻辑块、注释、语句等各个粒度的代码内容。

参见图 7，图 7 示出了本公开一个实施例提供的一种基于代码处理模型的信息处理方法的流程图，具体包括以下步骤：

步骤 702：接收任务生成请求，其中，任务生成请求包括请求信息。

具体地，基于代码处理模型的信息处理方法可以应用于终端设备，也可以应用于模型训练平台。任务生成请求用于请求生成目标任务的任务信息。任务生成请求通常包含任务类型、期望输出格式以及请求信息。例如，当用户在模型训练平台的前端界面上选择“代码补全”功能并上传代码数据时，可以构造一个任务生成请求，任务生成请求中包含了代码数据、任务类型（即代码补全）、代码数据的语言类型等信息。请求信息是指任务生成请求中携带的与目标任务相关的参数或描述信息。请求信息包括但不限于目标任务的任务场景标识、任务模型标识或者目标任务的模型训练数据。

步骤 704：基于请求信息，获取代码处理模型，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到。

本公开一种可选的实施例中，请求信息包括目标任务的任务场景标识，或者，任务模型标识；上述基于请求信息，获取代码处理模型，可以包括以下步骤：

基于任务场景标识，从多个预设场景模板中确定目标场景模板，并基于目标场景模板，从模型库中查找代码处理模型，其中，模型库存储有多个处理模型；

或者，

基于任务模型标识，从模型库中查找代码处理模型。

具体地，任务场景标识是指用于区分不同任务应用场景的唯一或特定标签。本公开实

施例中，任务场景标识是请求信息的一部分。通过任务场景标识可以从一系列的预设场景模板中选择与请求信息匹配的目标场景模板来生成任务信息。例如，任务场景标识是“代码补全”，意味着前端用户希望对上传的代码数据进行代码补全，则可以根据任务场景标识从多个预设场景模板中选择代码补全场景模板。

预设场景模板为针对不同任务应用场景预先定义的标准配置场景模板，每个模板包含与该任务应用场景相匹配的模型信息以及任务处理流程等信息。通过一系列的预设场景模板，可以快速响应不同场景的任务生成请求。不同的预设场景模板对应于不同的任务类型、模型信息和处理流程。例如，预设场景模板中可能有一个专门针对代码补全的场景模板，其中包含了已训练好的代码补全模型的模型信息和处理流程。

目标场景模板是指与任务场景标识相匹配的场景模板。在解析任务生成请求时，可以基于任务场景标识定位到相应的目标场景模板，并依据目标场景模板中包括的模型信息从模型库中选取对应的代码处理模型及其它相关配置信息。例如，当任务场景标识为“代码补全”时，目标场景模板即为包含代码补全模型的模型信息及相关配置参数的模板。

模型库是集中存储深度学习模型的资源库，这些模型经过训练并优化以解决不同的处理任务。模型库存储的处理模型，包括但不限于代码补全模型、代码注释生成模型、代码推荐模型。并且，模型库中的处理模型可以根据适用场景的不同分为不同版本，例如，模型库中可能存有多种版本的代码补全模型，如针对 Python 语言的代码补全模型、针对 Java 语言的代码补全模型。

任务模型标识是指用于区分不同任务所适用的模型的唯一或特定标签。例如任务模型标识可以是“代码补全”，基于任务模型标识，可以从模型库中查找适用于处理代码补全任务的代码补全模型。

应用本公开实施例的方案，基于任务场景标识，从多个预设场景模板中确定目标场景模板，并基于目标场景模板，从模型库中查找代码处理模型，其中，模型库存储有多个处理模型；或者，基于任务模型标识，从模型库中查找代码处理模型。通过预先定义的任务场景模板、任务模型标识与模型库资源，使得代码处理模型的获取过程更加灵活、高效且标准。

本公开另一种可选的实施例中，除了可以从模型库中选择预先训练的代码处理模型，还可以根据请求信息中的模型训练数据，针对性地训练得到代码处理模型，也即，请求信息包括目标任务的多个样本代码数据；上述基于请求信息，获取代码处理模型，可以包括以下步骤：

基于多个样本代码数据，对目标任务对应的代码处理模型进行训练，获得训练完成的代码处理模型。

需要说明的是，“基于多个样本代码数据，对目标任务对应的代码处理模型进行训练，获得训练完成的代码处理模型”的实现方式与上述代码处理模型的训练方式相同，本公开实施例便不再进行赘述。

应用本公开实施例的方案，基于多个样本代码数据，对目标任务对应的代码处理模型进行训练，获得训练完成的代码处理模型，保证代码处理模型更加符合用户需求以及代码处理模型的精度。

步骤 706：基于代码处理模型，生成任务信息，其中，任务信息用于执行目标任务。

具体地，任务信息包含了执行目标任务所需的模型配置和处理流程。终端设备或其他服务端组件可以基于任务信息正确地使用代码处理模型对目标任务进行处理。

需要说明的是，基于代码处理模型，生成任务信息时，可以直接将代码处理模型打包得到任务信息。还可以获取代码处理模型的模型信息，基于模型信息构建任务信息，其中，模型信息如模型参数配置、输入数据的处理方式、预期输出结果的规格，以及可能涉及的中间步骤和其他辅助信息。

示例性地，在代码补全任务中，任务信息可能包括选定的代码补全模型的地址信息、输入代码数据的存储位置、输出补全结果的目标路径，以及代码补全模型运行所需的其他环境配置等参数，这些信息使得落在本地或远程服务器上正确地加载代码补全模型并执行代码补全任务。

应用本公开实施例的方案，通过生成目标任务的任务信息，能够在保证目标任务处理质量和效率的同时，降低系统部署和运维成本，为用户提供便捷高效的任务处理服务。

参见图 8，图 8 示出了本公开一个实施例提供的一种模型训练平台的结构示意图，模型训练平台包括请求接口单元 802 和模型单元 804；

请求接口单元 802，用于接收模型训练请求，其中，模型训练请求包括请求信息；

模型单元 804，用于获取代码处理模型；基于请求信息，对代码处理模型进行训练，获得训练完成的代码处理模型，其中，请求信息包括多个样本处理结果和多个样本处理结果分别对应的样本代码块数据，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到。

需要说明的是，请求接口单元和模型单元的处理方式与上述代码处理模型的训练方式相同，本公开实施例便不再进行赘述。

本公开一种可选的实施例中，模型训练平台，还包括模型库；

模型单元，还用于将训练完成的代码处理模型存储至模型库。

需要说明的是，模型单元基于请求信息，对代码处理模型进行训练，获得训练完成的代码处理模型之后，还可以将训练完成的代码处理模型存储至模型库，从而后续可以直接从模型库中获取训练完成的代码处理模型，无需重复训练。其中，从模型库中获取训练完成的代码处理模型的方式，可以参考上述“基于任务场景标识，从多个预设场景模板中确定目标场景模板，并基于目标场景模板，从模型库中查找代码处理模型，其中，模型库存储有多个处理模型；或者，基于任务模型标识，从模型库中查找代码处理模型”的实现方式，本公开实施例便不再进行赘述。

与上述任务处理方法实施例相对应，本公开还提供了任务处理装置实施例，图 9 示出了本公开一个实施例提供的一种任务处理装置的结构示意图。如图 9 所示，该装置包括：

第一获取模块 902，被配置为获取目标任务的待处理代码数据；

第一输入模块 904，被配置为将待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到。

可选地，第一输入模块 904，进一步被配置为获取代码处理模型的处理提示信息；将处理提示信息和待处理代码数据输入代码处理模型中，获得目标任务的代码处理结果。

可选地，该装置还包括：第一发送模块，被配置为将代码处理结果发送至前端用户；接收前端用户发送的结果反馈信息，其中，结果反馈信息为基于目标任务的任务信息对代码处理结果进行反馈的信息；根据结果反馈信息，构建模型优化数据；利用模型优化数据，对代码处理模型进行参数调整。

可选地，第一发送模块，进一步被配置为根据结果反馈信息，生成优化提示信息，其中，优化提示信息用于引导前端用户发送用于优化代码处理模型的模型优化数据；将优化提示信息发送至前端用户，并接收前端用户基于优化提示信息发送的模型优化数据。

可选地，该装置还包括：标注模块，被配置为对代码处理结果中的关键信息进行标注，获得更新后的代码处理结果；将更新后的代码处理结果发送至前端用户。

可选地，该装置还包括：第二发送模块，被配置为将代码处理结果发送至前端用户；接收前端用户发送的修改信息，其中，修改信息用于对代码处理结果进行修改；根据修改

信息，对代码处理结果进行修改，获得修改后的代码处理结果。

可选地，该装置还包括：第五获取模块，被配置为获取样本集，其中，样本集包括多个样本代码数据；对多个样本代码数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据；将多个样本代码块数据输入代码处理模型，获得多个样本代码块数据分别对应的预测处理结果；根据样本处理结果和预测处理结果，对代码处理模型进行训练，获得训练完成的代码处理模型。

可选地，第五获取模块，进一步被配置为针对第一样本代码数据，对第一样本代码数据进行语法解析，获得多个候选代码块数据，其中，第一样本代码数据为多个样本代码数据中的任一个；对多个候选代码块数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据。

可选地，第五获取模块，进一步被配置为从代码处理主题中采样得到目标代码处理主题，其中，代码处理主题包括代码块处理主题、代码块内容处理主题、代码块部分内容处理主题、代码块空内容处理主题中的至少一种；在目标代码处理主题下，根据代码采样策略，从多个候选代码块数据中采样得到多个样本处理结果和多个样本处理结果分别对应的样本代码块数据。

可选地，代码采样策略包括结果采样策略和代码块采样策略，结果采样策略包括类别采样策略、上文类别采样策略、下文类别采样策略、孩子节点采样策略和完整度采样策略中的至少一种，代码块采样策略包括上文内容采样策略和下文内容采样策略中的至少一种；第五获取模块，进一步被配置为在目标代码处理主题下，根据结果采样策略，从多个候选代码块数据中采样得到多个样本处理结果，并根据代码块采样策略，从多个候选代码块数据中采样得到多个样本处理结果分别对应的样本代码块数据。

应用本公开实施例的方案，由于代码处理模型基于样本代码块数据训练得到，实现了以代码块为单位进行预测，缩短了预测内容，并且，通过层次采样，可以使得代码处理模型适应各种代码处理场景，从而精准地预测各种类型的代码块，提高任务处理效率以及精准性。

上述为本实施例的一种任务处理装置的示意性方案。需要说明的是，该任务处理装置的技术方案与上述的任务处理方法的技术方案属于同一构思，任务处理装置的技术方案未详细描述的细节内容，均可以参见上述任务处理方法的技术方案的描述。

与上述代码补全方法实施例相对应，本公开还提供了代码补全装置实施例，图 10 示出了本公开一个实施例提供的一种代码补全装置的结构示意图。如图 10 所示，该装置包括：

第二获取模块 1002，被配置为获取代码补全任务的待补全代码数据；

第二输入模块 1004，被配置为将待补全代码数据输入代码补全模型中，获得代码补全任务的代码补全结果，其中，代码补全模型基于多个样本补全结果和多个样本补全结果分别对应的样本代码块数据训练得到，样本补全结果和样本代码块数据基于对样本代码数据进行层次采样得到。

可选地，该装置还包括：第二接收模块，被配置为向前端用户发送补全提示信息，其中，补全提示信息用于引导前端用户发送用于代码补全任务处理的待补全代码数据；接收前端用户基于补全提示信息发送的代码补全任务的待补全代码数据。

可选地，该装置还包括：第三接收模块，被配置为将代码补全结果发送至前端用户；接收前端用户发送的结果反馈信息，其中，结果反馈信息为基于代码补全任务的任务信息对代码补全结果进行反馈的信息；将结果反馈信息发送至模型训练平台，其中，模型训练平台用于利用结果反馈信息对代码补全模型进行参数调整。

应用本公开实施例的方案，针对代码补全模型所需要满足的能力，通过基于代码语法分析和丰富场景的层次采样方法，构造出模型训练数据，并利用模型训练数据训练得到代

码补全模型，使得代码补全模型在真实开发的各种场景下，精准地根据用户编写的代码上文、代码下文以及一些代码元数据，准确预测用户下一个想要输入的各种类型的代码块（可能是一个类、函数、逻辑块、注释、语句等），从而提升用户编码效率和体验。

上述为本实施例的一种代码补全装置的示意性方案。需要说明的是，该代码补全装置的技术方案与上述的代码补全方法的技术方案属于同一构思，代码补全装置的技术方案未详细描述的细节内容，均可以参见上述代码补全方法的技术方案的描述。

与上述代码处理模型训练方法实施例相对应，本公开还提供了代码处理模型训练装置实施例，图 11 示出了本公开一个实施例提供的一种代码处理模型训练装置的结构示意图。如图 11 所示，该装置包括：

第三获取模块 1102，被配置为获取样本集，其中，样本集包括多个样本代码数据；

采样模块 1104，被配置为对多个样本代码数据进行层次采样，获得多个样本处理结果和多个样本处理结果分别对应的样本代码块数据；

第三输入模块 1106，被配置为将多个样本代码块数据输入代码处理模型，获得多个样本代码块数据分别对应的预测处理结果；

训练模块 1108，被配置为根据样本处理结果和预测处理结果，对代码处理模型进行训练，获得训练完成的代码处理模型。

应用本公开实施例的方案，根据样本处理结果和预测处理结果，对代码处理模型进行训练，在不满足预设停止条件的情况下继续训练代码处理模型，直至达到预设停止条件，完成训练获得代码处理模型。通过不断对代码处理模型的模型参数进行调整，能使最终得到的代码处理模型更加精准。

上述为本实施例的一种代码处理模型训练装置的示意性方案。需要说明的是，该代码处理模型训练装置的技术方案与上述的代码处理模型训练方法的技术方案属于同一构思，代码处理模型训练装置的技术方案未详细描述的细节内容，均可以参见上述代码处理模型训练方法的技术方案的描述。

与上述方法实施例相对应，本公开还提供了基于代码处理模型的信息处理装置实施例，图 12 示出了本公开一个实施例提供的一种基于代码处理模型的信息处理装置的结构示意图。如图 12 所示，该装置包括：

第一接收模块 1202，被配置为接收任务生成请求，其中，任务生成请求包括请求信息；

第四获取模块 1204，被配置为基于请求信息，获取代码处理模型，其中，代码处理模型基于多个样本处理结果和多个样本处理结果分别对应的样本代码块数据训练得到，样本处理结果和样本代码块数据基于对样本代码数据进行层次采样得到；

生成模块 1206，被配置为基于代码处理模型，生成任务信息，其中，任务信息用于执行目标任务。

可选地，请求信息包括目标任务的任务场景标识，或者，任务模型标识；第四获取模块 1204，进一步被配置为基于任务场景标识，从多个预设场景模板中确定目标场景模板，并基于目标场景模板，从模型库中查找代码处理模型，其中，模型库存储有多个处理模型；或者，基于任务模型标识，从模型库中查找代码处理模型。

可选地，请求信息包括目标任务的多个样本代码数据；第四获取模块 1204，进一步被配置为基于多个样本代码数据，对目标任务对应的代码处理模型进行训练，获得训练完成的代码处理模型。

应用本公开实施例的方案，通过生成目标任务的任务信息，能够在保证目标任务处理质量和效率的同时，降低系统部署和运维成本，为用户提供便捷高效的任务处理服务。

上述为本实施例的一种基于代码处理模型的信息处理装置的示意性方案。需要说明的是，该基于代码处理模型的信息处理装置的技术方案与上述的基于代码处理模型的信息处

理方法的技术方案属于同一构思，基于代码处理模型的信息处理装置的技术方案未详细描述的细节内容，均可以参见上述基于代码处理模型的信息处理方法的技术方案的描述。

图 13 示出了本公开一个实施例提供的一种计算设备的结构框图。该计算设备 1300 的部件包括但不限于存储器 1310 和处理器 1320。处理器 1320 与存储器 1310 通过总线 1330 相连接，数据库 1350 用于保存数据。

计算设备 1300 还包括接入设备 1340，接入设备 1340 使得计算设备 1300 能够经由一个或多个网络 1360 通信。这些网络的示例包括公用交换电话网（PSTN，Public Switched Telephone Network）、局域网（LAN，Local Area Network）、广域网（WAN，Wide Area Network）、个域网（PAN，Personal Area Network）或诸如因特网的通信网络的组合。接入设备 1340 可以包括有线或无线的任何类型的网络接口（例如，网络接口卡（NIC，Network Interface Card））中的一个或多个，诸如 IEEE802.11 无线局域网（WLAN，Wireless Local Area Networks）无线接口、全球微波互联接入（Wi-MAX，World Interoperability for Microwave Access）接口、以太网接口、通用串行总线（USB，Universal Serial Bus）接口、蜂窝网络接口、蓝牙接口、近场通信（NFC，Near Field Communication）接口，等等。

在本公开的一个实施例中，计算设备 1300 的上述部件以及图 13 中未示出的其他部件也可以彼此相连接，例如通过总线。应当理解，图 13 所示的计算设备结构框图仅仅是出于示例的目的，而不是对本公开范围的限制。本领域技术人员可以根据需要，增添或替换其他部件。

计算设备 1300 可以是任何类型的静止或移动计算设备，包括移动计算机或移动计算设备（例如，平板计算机、个人数字助理、膝上型计算机、笔记本计算机、上网本等）、移动电话（例如，智能手机）、可佩戴的计算设备（例如，智能手表、智能眼镜等）或其他类型的移动设备，或者诸如台式计算机或个人计算机（PC，Personal Computer）的静止计算设备。计算设备 1300 还可以是移动式或静止式的服务器。

其中，处理器 1320 用于执行计算机程序/指令，该计算机程序/指令被处理器执行时实现上述任务处理方法或者代码补全方法或者代码处理模型训练方法或者基于代码处理模型的信息处理方法的步骤。

上述为本实施例的一种计算设备的示意性方案。需要说明的是，该计算设备的技术方案与上述的任务处理方法、代码补全方法、代码处理模型训练方法以及基于代码处理模型的信息处理方法的技术方案属于同一构思，计算设备的技术方案未详细描述的细节内容，均可以参见上述任务处理方法或者代码补全方法或者代码处理模型训练方法或者基于代码处理模型的信息处理方法的技术方案的描述。

本公开一实施例还提供一种计算机可读存储介质，其存储有计算机程序/指令，该计算机程序/指令被处理器执行时实现上述任务处理方法或者代码补全方法或者代码处理模型训练方法或者基于代码处理模型的信息处理方法的步骤。

上述为本实施例的一种计算机可读存储介质的示意性方案。需要说明的是，该存储介质的技术方案与上述的任务处理方法、代码补全方法、代码处理模型训练方法以及基于代码处理模型的信息处理方法的技术方案属于同一构思，存储介质的技术方案未详细描述的细节内容，均可以参见上述任务处理方法或者代码补全方法或者代码处理模型训练方法或者基于代码处理模型的信息处理方法的技术方案的描述。

本公开一实施例还提供一种计算机程序产品，包括计算机程序/指令，该计算机程序/指令被处理器执行时实现上述任务处理方法或者代码补全方法或者代码处理模型训练方法或者基于代码处理模型的信息处理方法的步骤。

上述为本实施例的一种计算机程序产品的示意性方案。需要说明的是，该计算机程序产品的技术方案与上述的任务处理方法、代码补全方法、代码处理模型训练方法以及基于

代码处理模型的信息处理方法的技术方案属于同一构思，计算机程序产品的技术方案未详细描述的细节内容，均可以参见上述任务处理方法或者代码补全方法或者代码处理模型训练方法或者基于代码处理模型的信息处理方法的技术方案的描述。

上述对本公开特定实施例进行了描述。其它实施例在所附权利要求书的范围内。在一些情况下，在权利要求书中记载的动作或步骤可以按照不同于实施例中的顺序来执行并且仍然可以实现期望的结果。另外，在附图中描绘的过程不一定要求示出的特定顺序或者连续顺序才能实现期望的结果。在某些实施方式中，多任务处理和并行处理也是可以的或者可能是有利的。

所述计算机指令包括计算机程序代码，所述计算机程序代码可以为源代码形式、对象代码形式、可执行文件或某些中间形式等。所述计算机可读介质可以包括：能够携带所述计算机程序代码的任何实体或装置、记录介质、U 盘、移动硬盘、磁碟、光盘、计算机存储器、只读存储器(ROM,Read-Only Memory)、随机存取存储器(RAM,Random Access Memory)、电载波信号、电信信号以及软件分发介质等。需要说明的是，所述计算机可读介质包含的内容可以根据专利实践的要求进行适当的增减，例如在某些地区，根据专利实践，计算机可读介质不包括电载波信号和电信信号。

需要说明的是，对于前述的各方法实施例，为了简便描述，故将其都表述为一系列的动作组合，但是本领域技术人员应该知悉，本公开实施例并不受所描述的动作顺序的限制，因为依据本公开实施例，某些步骤可以采用其它顺序或者同时进行。其次，本领域技术人员也应该知悉，说明书中所描述的实施例均属于优选实施例，所涉及的动作和模块并不一定都是本公开实施例所必须的。

在上述实施例中，对各个实施例的描述都各有侧重，某个实施例中没有详述的部分，可以参见其它实施例的相关描述。

以上公开的本公开优选实施例只是用于帮助阐述本公开。可选实施例并没有详尽叙述所有的细节，也不限制该发明仅为所述的具体实施方式。显然，根据本公开实施例的内容，可作很多的修改和变化。本公开选取并具体描述这些实施例，是为了更好地解释本公开实施例的原理和实际应用，从而使所属技术领域技术人员能很好地理解和利用本公开。本公开仅受权利要求书及其全部范围和等效物的限制。

权 利 要 求 书

1、一种任务处理方法，包括：

获取目标任务的待处理代码数据；

将所述待处理代码数据输入代码处理模型中，获得所述目标任务的代码处理结果，其中，所述代码处理模型基于多个样本处理结果和所述多个样本处理结果分别对应的样本代码块数据训练得到，所述样本处理结果和所述样本代码块数据基于对样本代码数据进行层次采样得到。

2、根据权利要求 1 所述的方法，所述将所述待处理代码数据输入代码处理模型中，获得所述目标任务的代码处理结果，包括：

获取所述代码处理模型的处理提示信息；

将所述处理提示信息和所述待处理代码数据输入所述代码处理模型中，获得所述目标任务的代码处理结果。

3、根据权利要求 1 或 2 所述的方法，所述将所述待处理代码数据输入代码处理模型中，获得所述目标任务的代码处理结果之后，还包括：

将所述代码处理结果发送至前端用户；

接收所述前端用户发送的结果反馈信息，其中，所述结果反馈信息为基于所述目标任务的任务信息对所述代码处理结果进行反馈的信息；

根据所述结果反馈信息，构建模型优化数据；

利用所述模型优化数据，对所述代码处理模型进行参数调整。

4、根据权利要求 3 所述的方法，所述根据所述结果反馈信息，构建模型优化数据，包括：

根据所述结果反馈信息，生成优化提示信息，其中，所述优化提示信息用于引导所述前端用户发送用于优化所述代码处理模型的模型优化数据；

将所述优化提示信息发送至所述前端用户，并接收所述前端用户基于所述优化提示信息发送的所述模型优化数据。

5、根据权利要求 1 至 4 任一项所述的方法，所述将所述待处理代码数据输入代码处理模型中，获得所述目标任务的代码处理结果之后，还包括：

对所述代码处理结果中的关键信息进行标注，获得更新后的代码处理结果；

将所述更新后的代码处理结果发送至前端用户。

6、根据权利要求 1 至 5 任一项所述的方法，所述将所述待处理代码数据输入代码处理模型中，获得所述目标任务的代码处理结果之后，还包括：

将所述代码处理结果发送至前端用户；

接收所述前端用户发送的修改信息，其中，所述修改信息用于对所述代码处理结果进行修改；

根据所述修改信息，对所述代码处理结果进行修改，获得修改后的代码处理结果。

7、根据权利要求 1 至 6 任一项所述的方法，所述将所述待处理代码数据输入代码处理模型中，获得所述目标任务的代码处理结果之前，还包括：

获取样本集，其中，所述样本集包括多个样本代码数据；

对所述多个样本代码数据进行层次采样，获得多个样本处理结果和所述多个样本处理结果分别对应的样本代码块数据；

将多个样本代码块数据输入代码处理模型，获得所述多个样本代码块数据分别对应的

预测处理结果；

根据所述样本处理结果和所述预测处理结果，对所述代码处理模型进行训练，获得训练完成的代码处理模型。

8、根据权利要求 7 所述的方法，所述对所述多个样本代码数据进行层次采样，获得多个样本处理结果和所述多个样本处理结果分别对应的样本代码块数据，包括：

针对第一样本代码数据，对所述第一样本代码数据进行语法解析，获得多个候选代码块数据，其中，所述第一样本代码数据为所述多个样本代码数据中的任一个；

对所述多个候选代码块数据进行层次采样，获得多个样本处理结果和所述多个样本处理结果分别对应的样本代码块数据。

9、根据权利要求 8 所述的方法，所述对所述多个候选代码块数据进行层次采样，获得多个样本处理结果和所述多个样本处理结果分别对应的样本代码块数据，包括：

从代码处理主题中采样得到目标代码处理主题，其中，所述代码处理主题包括代码块处理主题、代码块内容处理主题、代码块部分内容处理主题、代码块空内容处理主题中的至少一种；

在所述目标代码处理主题下，根据代码采样策略，从所述多个候选代码块数据中采样得到多个样本处理结果和所述多个样本处理结果分别对应的样本代码块数据。

10、根据权利要求 9 所述的方法，所述代码采样策略包括结果采样策略和代码块采样策略，所述结果采样策略包括类别采样策略、上文类别采样策略、下文类别采样策略、孩子节点采样策略和完整度采样策略中的至少一种，所述代码块采样策略包括上文内容采样策略和下文内容采样策略中的至少一种；

所述在所述目标代码处理主题下，根据代码采样策略，从所述多个候选代码块数据中采样得到多个样本处理结果和所述多个样本处理结果分别对应的样本代码块数据，包括：

在所述目标代码处理主题下，根据所述结果采样策略，从所述多个候选代码块数据中采样得到多个样本处理结果，并根据所述代码块采样策略，从所述多个候选代码块数据中采样得到所述多个样本处理结果分别对应的样本代码块数据。

11、一种代码补全方法，包括：

获取代码补全任务的待补全代码数据；

将所述待补全代码数据输入代码补全模型中，获得所述代码补全任务的代码补全结果，其中，所述代码补全模型基于多个样本补全结果和所述多个样本补全结果分别对应的样本代码块数据训练得到，所述样本补全结果和所述样本代码块数据基于对样本代码数据进行层次采样得到。

12、根据权利要求 11 所述的方法，所述获取代码补全任务的待补全代码数据之前，还包括：

向前端用户发送补全提示信息，其中，所述补全提示信息用于引导所述前端用户发送用于代码补全任务处理的待补全代码数据；

接收所述前端用户基于所述补全提示信息发送的所述代码补全任务的待补全代码数据。

13、根据权利要求 11 或 12 所述的方法，所述将所述待补全代码数据输入代码补全模型中，获得所述代码补全任务的代码补全结果之后，还包括：

将所述代码补全结果发送至前端用户；

接收所述前端用户发送的结果反馈信息，其中，所述结果反馈信息为基于所述代码补全任务的任务信息对所述代码补全结果进行反馈的信息；

将所述结果反馈信息发送至模型训练平台，其中，所述模型训练平台用于利用所述结

果反馈信息对所述代码补全模型进行参数调整。

14、一种代码处理模型训练方法，包括：

获取样本集，其中，所述样本集包括多个样本代码数据；

对所述多个样本代码数据进行层次采样，获得多个样本处理结果和所述多个样本处理结果分别对应的样本代码块数据；

将多个样本代码块数据输入代码处理模型，获得所述多个样本代码块数据分别对应的预测处理结果；

根据所述样本处理结果和所述预测处理结果，对所述代码处理模型进行训练，获得训练完成的代码处理模型。

15、一种基于代码处理模型的信息处理方法，包括：

接收任务生成请求，其中，所述任务生成请求包括请求信息；

基于所述请求信息，获取代码处理模型，其中，所述代码处理模型基于多个样本处理结果和所述多个样本处理结果分别对应的样本代码块数据训练得到，所述样本处理结果和所述样本代码块数据基于对样本代码数据进行层次采样得到；

基于所述代码处理模型，生成任务信息，其中，所述任务信息用于执行目标任务。

16、根据权利要求 15 所述的方法，所述请求信息包括目标任务的任务场景标识，或者，任务模型标识；

所述基于所述请求信息，获取代码处理模型，包括：

基于所述任务场景标识，从多个预设场景模板中确定目标场景模板，并基于所述目标场景模板，从模型库中查找代码处理模型，其中，所述模型库存储有多个处理模型；

或者，

基于所述任务模型标识，从所述模型库中查找代码处理模型。

17、根据权利要求 15 所述的方法，所述请求信息包括目标任务的多个样本代码数据；

所述基于所述请求信息，获取代码处理模型，包括：

基于所述多个样本代码数据，对所述目标任务对应的代码处理模型进行训练，获得训练完成的代码处理模型。

18、一种模型训练平台，包括请求接口单元和模型单元；

所述请求接口单元，用于接收模型训练请求，其中，所述模型训练请求包括请求信息；

所述模型单元，用于获取代码处理模型；基于所述请求信息，对所述代码处理模型进行训练，获得训练完成的代码处理模型，其中，所述请求信息包括多个样本处理结果和所述多个样本处理结果分别对应的样本代码块数据，所述样本处理结果和所述样本代码块数据基于对样本代码数据进行层次采样得到。

19、根据权利要求 18 所述的模型训练平台，还包括模型库；

所述模型单元，还用于将所述训练完成的代码处理模型存储至所述模型库。

20、一种计算设备，包括：

存储器和处理器；

所述存储器用于存储计算机程序/指令，所述处理器用于执行所述计算机程序/指令，该计算机程序/指令被处理器执行时实现权利要求 1 至 10 任意一项或者权利要求 11 至 13 任意一项或者权利要求 14 或者权利要求 15 至 17 任意一项所述方法的步骤。

21、一种计算机可读存储介质，其存储有计算机程序/指令，该计算机程序/指令被处理器执行时实现权利要求 1 至 10 任意一项或者权利要求 11 至 13 任意一项或者权利要求 14 或者权利要求 15 至 17 任意一项所述方法的步骤。

22、一种计算机程序产品，包括计算机程序/指令，该计算机程序/指令被处理器执行时实现权利要求 1 至 10 任意一项或者权利要求 11 至 13 任意一项或者权利要求 14 或者权利要求 15 至 17 任意一项所述方法的步骤。