代码处理方法、装置、计算设备集群以及存储介质

本申请要求于2024年02月23日提交的申请号为202410204960.3、发明名称为“代码处理方法、装置、计算设备集群以及存储介质”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

技术领域

本申请涉及人工智能技术领域，特别涉及一种代码处理方法、装置、计算设备集群以及存储介质。

背景技术

人工智能技术可以应用于代码处理领域，能够在代码处理过程中，基于用户的原始代码，向用户提出修改建议，该修改建议通常包括代码段和描述性文本。其中，描述性文本用于指示对原始代码的处理方式，该处理方式可以是将修改建议中的代码段插入到原始代码中、从原始代码中删除修改建议中的代码段或将原始代码中的部分代码替换为修改建议中的代码段。用户根据该修改建议对原始代码进行处理。

目前，一种代码处理方法是，为用户提供一键插入功能，该一键插入功能是为用户提供一个按钮，用户点击该按钮即可将修改建议中的代码段插入到光标所在处，或，将选中的原始代码中的代码段替换为修改建议中的代码段。

上述代码处理方法需要人工理解上述修改建议中的描述性文本、识别原始代码中与修改建议中的代码段对应的修改位置，在代码处理过程中的出错率较高，代码的处理效率较低。

发明内容

本申请实施例提供了一种代码处理方法、装置、计算设备集群以及存储介质，能够降低代码处理过程中的出错率，提高代码的处理效率。该技术方案如下。

第一方面，提供了一种代码处理方法，该方法包括：代码开发客户端获取开发得到的第一代码；代码开发客户端响应于代码分析指令，对获取到的第一代码进行分析，得到针对第一代码的修改意见信息，该修改意见信息包括描述性文本和待处理的代码段，描述性文本指示基于代码段对第一代码的修改意图，其中，修改意图包括从第一代码中删除该代码段、在第一代码中增加该代码段以及用该代码段对第一代码中的代码段进行替换；代码开发客户端基于分析得到的所述修改意见信息和第一代码进行分析，得到修改意图和第一代码中与修改意见信息对应的代码修改位置；代码开发客户端基于修改意图和代码修改位置，在第一代码中的代码修改位置，基于代码段执行与修改意图对应的处理。

其中，修改意见信息中的描述性文本为文字生成模型生成且采用自然语言描述的修改建议，可以指示第一代码的错误原因、发生错误的位置以及针对错误的修改方法等。修改意见信息中的代码段与描述性文本相对应，该代码段可能是完整的、修改后的第一代码，也可能是第一代码中代码修改位置处的修改结果。针对第一代码的修改意图包括：插入修改意见信息中的代码段、删除修改意见信息中的代码段以及将第一代码中的部分代码修改为修改意见信息中的代码段等，可以简称为增加、删除和修改，代码修改位置可以用行号或段号表示。

本申请实施例提供的代码处理方法，在代码处理过程中，获取第一代码和修改意见信息，由于该修改意见信息的语义以及第一代码和修改意见信息之间的关联等，与修改意图和代码修改位置之间存在一定规律，因此，根据该第一代码和修改意见信息，能够获取修改意图和代码修改位置，以实现对修改意见信息的自动化理解。基于该修改意图和代码修改位置，对第一代码进行处理。由于待处理的代码量可能巨大，因此，与人工理解上述修改意见信息后再对代码进行处理相比，本申请实施例提供的代码处理方法能够降低代码处理过程中的出错率，提高代码的处理效率。

可选地，上述代码开发客户端基于分析得到的所述修改意见信息和第一代码进行分析，得到修改意图和第一代码中与修改意见信息对应的代码修改位置包括：代码开发客户端响应于针对修改意见信息的触发操作，基于修改意见信息和第一代码进行分析，得到修改意图和代码修改位置。响应于针对修改意见信息

的触发操作，才能执行后续代码处理过程，能够确保后续代码处理过程所使用的修改意见信息的准确性，进而提高代码处理的准确性。

可选地，上述代码开发客户端基于修改意图和代码修改位置，在第一代码中的代码修改位置，基于代码段执行与修改意图对应的处理包括：代码开发客户端基于修改意图和代码修改位置，在第一代码中的代码修改位置，基于代码段执行与修改意图对应的处理，得到第二代码；代码开发客户端在预览窗口中显示第二代码；若接收到同意指令，则代码开发客户端将第一代码替换为第二代码。得到修改意图和代码修改位置后，不对第一代码自动实施修改，而是将修改结果提供给用户预览，用户同意后，再用修改后的代码替换原始代码，能够避免修改意图或代码修改位置识别错误导致代码处理出错的情况。

可选地，上述代码开发客户端基于修改意图和代码修改位置，在第一代码中的代码修改位置，基于代码段执行与修改意图对应的处理，得到第二代码包括：若修改意图为增加，则代码开发客户端在第一代码中代码修改位置处的代码后，插入修改意见信息中的代码段，得到第二代码；或，若修改意图为删除，则代码开发客户端从第一代码中删除代码修改位置处的代码，得到第二代码；或，若修改意图为修改，则代码开发客户端从第一代码中删除代码修改位置处的代码，在代码修改位置处插入修改意见信息中的代码段，得到第二代码。

可选地，上述修改意图和代码修改位置通过代码处理模型得到，代码处理模型的训练过程包括：获取第一训练数据集，第一训练数据集包括多组第一样本数据和第一标签，第一样本数据包括一组第一代码和修改意见信息，第一标签包括一组与第一样本数据对应的多组修改意图和代码修改位置；基于第一训练数据集，对代码处理模型进行训练，该代码处理模型具有强大的自然语言处理能力，能够同时生成修改意图和代码修改位置，效率较高。

可选地，上述修改意图通过意图识别模型得到，意图识别模型的训练过程包括：获取第二训练数据集，第二训练数据集包括多组第二样本数据和第二标签，第二样本数据包括一组第一代码和修改意见信息，第二标签包括一个与第二样本数据对应的修改意图；基于第二训练数据集，对意图识别模型进行训练。

可选地，上述代码修改位置通过定位模型得到，定位模型的训练过程包括：获取第三训练数据集，第三训练数据集包括多组第三样本数据和第三标签，第三样本数据包括一组第一代码、修改意见信息和修改意图，第三标签包括一个与第三样本数据对应的代码修改位置；基于第三训练数据集，对定位模型进行训练。

上述意图识别模型和定位模型为可用于文本生成或分类的自然语言处理（natural language processing，NLP）模型，如传统机器学习模型（朴素贝叶斯分类等）或深度学习模型（卷积神经网络模型、循环神经网络模型以及转换器模型（Transformers）等）。

第二方面，提供了一种代码处理装置，用于执行上述代码处理方法。具体地，该代码处理装置包括用于执行上述第一方面或上述第一方面的任一种可选方式提供的代码处理方法的功能模块。

第三方面，提供一种计算设备集群，包括至少一个计算设备，每个计算设备包括处理器和存储器；至少一个计算设备的处理器用于执行至少一个计算设备的存储器中存储的指令，以使得计算设备集群执行以实现如上述第一方面或上述第一方面的任一种可选方式提供的代码处理方法所执行的操作。

第四方面，提供一种计算机可读存储介质，包括计算机程序指令，当计算机程序指令由计算设备集群执行时，计算设备集群执行如上述第一方面或上述第一方面的任一种可选方式提供的代码处理方法所执行的操作。

第五方面，提供了一种包含指令的计算机程序产品，当指令被计算设备集群运行时，使得计算设备集群执行上述第一方面或上述任一方面的任一种可选实现方式中提供的代码处理方法。

本申请在上述各方面提供的实现方式的基础上，还可以进行进一步组合以提供更多实现方式。

附图说明

图1是本申请实施例提供的一种代码处理方法的实施环境示意图；

图2是本申请实施例提供的一种代码处理系统的结构示意图；

图3是本申请实施例提供的一种计算设备的结构示意图；

图4是本申请实施例提供的一种计算设备集群的结构示意图；

图5是本申请实施例提供的另一种计算设备集群的结构示意图；

图6是本申请实施例提供的一种代码处理方法的简要流程图；
图7是本申请实施例提供的一种代码处理方法的流程图；
图8是本申请实施例提供的一种修改意见信息的示意图；
图9是本申请实施例提供的一种代码处理模型训练方法的流程图；
图10是本申请实施例提供的另一种代码处理方法的简要流程图；
图11是本申请实施例提供的另一种代码处理方法的流程图；
图12是本申请实施例提供的一种意图识别模型训练方法的流程图；
图13是本申请实施例提供的一种定位模型训练方法的流程图；
图14是本申请实施例提供的一种代码处理装置的结构示意图。

具体实施方式

为使本申请的目的、技术方案和优点更加清楚，下面将结合附图对本申请实施方式作进一步地详细描述。

图1是根据本申请实施例提供的一种代码处理方法的实施环境示意图。如图1所示，该实施环境包括终端101和服务器102，终端101与服务器102之间通过有线网络或无线网络方式进行通信连接。图1所示的终端和服务器的数量仅为举例，本申请实施例不对该实施环境中终端和服务器的数量进行限定。

终端101用于作为代码开发客户端来获取第一代码，向服务器102发送第一代码。终端101包括但不限于：扫描仪、智能手机、智能电视、电视盒子、键盘、鼠标、游戏手柄、智能穿戴设备、平板电脑、便携式个人计算机、移动互联网设备等，本申请实施例不对终端101的类型进行限定。服务器102用于根据本申请实施例提供的代码处理方法，对第一代码进行处理，得到修改意图和代码修改位置，向终端101发送修改意图和代码修改位置。服务器102可以是独立的物理服务器，也可以是由多个物理服务器构成的服务器集群或者分布式系统，还可以是提供云服务、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（content delivery network，CDN）、大数据和人工智能平台等基础云计算服务的云服务器。

本申请实施例中，由终端101获取第一代码，向服务器102发送第一代码，由服务器102对第一代码进行处理。在一些实施例中，终端101或服务器102也能够独立完成获取第一代码，对第一代码进行处理的过程。

上述代码处理方法可以通过代码处理系统实现，如图2所示，该代码处理系统包括代码处理装置200，该代码处理装置200包括修改意见获取模块201和代码智能处理模块202，用于与服务器交互，以获取修改意图和代码修改位置。在代码处理过程中，修改意见获取模块201与服务器交互，以获取服务器上的意见获取模型针对第一代码所输出的修改意见信息。根据该修改意见信息和第一代码，代码智能处理模块202通过与服务器交互，以获取服务器上的代码处理模型（或意图识别模型和定位模型）针对该修改意见信息所输出的修改意图和代码修改位置，从而，该代码处理装置200可以基于获取到的修改意图和代码修改位置，对第一代码进行进一步处理。需要说明的是，上述模块划分仅为示例，本申请实施例并不限于此。其中，意见获取模型可以为大语言模型（large language model，LLM），也可以为其他自然语言处理模型。LLM是一种规模巨大、参数众多的深度学习模型，能够从大规模的数据集中学习语言结构、语法知识和语义，从而理解和生成人类语言。代码处理模型为LLM等具有强大自然语言处理能力的模型。意图识别模型可以是可用于文本生成或分类的NLP模型，如传统机器学习模型（朴素贝叶斯分类等）或深度学习模型（卷积神经网络模型、循环神经网络模型以及Transformers等），本申请实施例对此不作限定。

例如，代码处理装置200为集成开发环境（integrated development environment，IDE）客户端中的编程插件。代码调试过程中，该编程插件与服务器交互，以获取针对代码的修改意图和修改位置。其中，修改意图为代码的处理方式，例如，在代码中增加几行代码、删除几行代码或替换几行代码。代码修改位置为与修改意图对应的代码的行号。

代码处理装置200可以通过软件实现，或者可以通过硬件实现。示例性的，接下来介绍代码处理装置200的实现方式。

装置作为软件功能单元的一种举例，代码处理装置200可以包括运行在计算实例上的代码。其中，计算实例可以是物理主机（计算设备）、虚拟机、容器等计算设备中的至少一种。进一步地，上述计算设备可以是一台或者多台。例如，代码处理装置200可以包括运行在多个主机/虚拟机/容器上的代码。需要说明的是，

用于运行该应用程序的多个主机/虚拟机/容器可以分布在相同的区域（region）中，也可以分布在不同的region中。用于运行该代码的多个主机/虚拟机/容器可以分布在相同的可用区（availability zone，AZ）中，也可以分布在不同的AZ中，每个AZ包括一个数据中心或多个地理位置相近的数据中心。其中，通常一个region可以包括多个AZ。

同样，用于运行该代码的多个主机/虚拟机/容器可以分布在同一个虚拟私有云（virtual private cloud，VPC）中，也可以分布在多个VPC中。其中，通常一个VPC设置在一个region内，同一region内两个VPC之间，以及不同region的VPC之间跨区通信需在每个VPC内设置通信网关，经通信网关实现VPC之间的互连。

装置作为硬件功能单元的一种举例，代码处理装置200可以包括至少一个计算设备，如服务器等。或者，代码处理装置200也可以是利用专用集成电路（application-specific integrated circuit，ASIC）实现、或可编程逻辑器件（programmable logic device，PLD）实现的设备等。其中，上述PLD可以是复杂程序逻辑器件（complex programmable logical device，CPLD）、现场可编程门阵列（field-programmable gate array，FPGA）、通用阵列逻辑（generic array logic，GAL）或其任意组合实现。

代码处理装置200包括的多个计算设备可以分布在相同的region中，也可以分布在不同的region中。代码处理装置200包括的多个计算设备可以分布在相同的AZ中，也可以分布在不同的AZ中。同样，代码处理装置200包括的多个计算设备可以分布在同一个VPC中，也可以分布在多个VPC中。其中，所述多个计算设备可以是服务器、ASIC、PLD、CPLD、FPGA和GAL等计算设备的任意组合。

图3是本申请实施例提供的一种计算设备的结构示意图，该计算设备300可以为上述实施环境中的终端101或服务器102，本申请实施例对此不做限定。应理解，以下介绍的计算设备可以实现下述任一方法中的任意功能。

如图3所示，计算设备300包括：总线302、处理器304、存储器306和通信接口308。处理器304、存储器306和通信接口308之间通过总线302通信。应理解，本申请不限定计算设备300中的处理器、存储器的个数。

总线302可以是外设部件互连标准（peripheral component interconnect，PCI）总线或扩展工业标准结构（extended industry standard architecture，EISA）总线等。总线可以分为地址总线、数据总线、控制总线等。为便于表示，图3中仅用一条线表示，但并不表示仅有一根总线或一种类型的总线。总线302可包括在计算设备300各个部件（例如，存储器306、处理器304、通信接口308）之间传送信息的通路。

处理器304可以包括中央处理器（central processing unit，CPU）、图形处理器（graphics processing unit，GPU）、微处理器（micro processor，MP）或者数字信号处理器（digital signal processor，DSP）等处理器中的任意一种或多种。

存储器306可以包括易失性存储器(volatile memory)，例如随机存取存储器（random access memory，RAM）。处理器304还可以包括非易失性存储器(non-volatile memory)，例如只读存储器(read-only memory，ROM)，快闪存储器，机械硬盘（hard disk drive，HDD）或固态硬盘（solid state drive，SSD）。

存储器306中存储有可执行的程序代码，处理器304执行该可执行的程序代码以实现前述代码处理装置200的功能，从而实现代码处理方法。也即，存储器306上存有用于执行代码处理方法的指令。

通信接口308使用例如但不限于网络接口卡、收发器一类的收发模块，来实现计算设备300与其他设备或通信网络之间的通信。

本申请实施例还提供了一种计算设备集群。该计算设备集群包括至少一台计算设备。该计算设备可以是服务器，例如是中心服务器、边缘服务器，或者是本地数据中心中的本地服务器。在一些实施例中，计算设备也可以是台式机、笔记本电脑或者智能手机等终端设备。

如图4所示，所述计算设备集群包括至少一个计算设备300。计算设备集群中的一个或多个计算设备300中的存储器306中可以存有相同的用于执行代码处理方法的指令。

在一些可能的实现方式中，该计算设备集群中的一个或多个计算设备300的存储器306中也可以分别存有用于执行代码处理方法的部分指令。换言之，一个或多个计算设备300的组合可以共同执行用于执行代码处理方法的指令。

需要说明的是，计算设备集群中的不同的计算设备300中的存储器306可以存储不同的指令，分别用于执行代码处理装置200的部分功能。也即，不同的计算设备300中的存储器306存储的指令可以实现修改意见获取模块201和代码智能处理模块202中的一个或多个模块的功能。

在一些可能的实现方式中，计算设备集群中的一个或多个计算设备可以通过网络连接。其中，所述网络可以是广域网或局域网等等。图5示出了一种可能的实现方式。如图5所示，两个计算设备300A和300B之间通过网络进行连接。具体地，通过各个计算设备中的通信接口与所述网络进行连接。在这一类可能的实现方式中，计算设备300A中的存储器306中存有执行修改意见获取模块201的功能的指令。同时，计算设备300B中的存储器306中存有执行代码智能处理模块202的功能的指令。

图5所示的计算设备集群之间的连接方式可以是考虑到本申请提供的代码处理方法需要大量地存储数据，因此考虑将代码智能处理模块202实现的功能交由计算设备300B执行。

应理解，图5中示出的计算设备300A的功能也可以由多个计算设备300完成。同样，计算设备300B的功能也可以由多个计算设备300完成。

在相关技术中，描述性文本不是基于统一的描述模板得到的，修改意见信息中的代码段也不是基于统一的修改方法得到的，因此，给出的修改意见信息具有随机性和多样性，较为模糊，无法明确指示修改意图是增加、删除还是修改中的哪一类，也无法明确指示修改具体是针对代码的哪个位置，导致服务器无法通过固定的识别模板，来对修改意见信息进行识别，以得到修改意图或代码修改位置。目前，通常是人工理解修改意见信息，得到修改意图和代码修改位置，再指示计算设备对第一代码进行处理。在处理大量第一代码时，人工理解耗时较长且容易出错，导致代码处理的出错率较高，代码的处理效率较低。

为了解决上述问题，本申请实施例提供一种代码处理方法，该方法调用预先训练的模型来处理第一代码和修改意见信息，得到修改意图和代码修改位置，以实现理解修改意见信息的目的，该模型通过训练，学习第一代码和修改意见信息的结构、语义以及二者之间的关联等，从而能够模拟人类的思维，来较为准确地理解修改意见信息。使用该模型代替人工，能够降低代码处理的出错率，提高代码处理的效率。

本申请实施例提供了两种使用模型理解修改意见信息的实现方式。第一种：代码处理过程使用一个模型，即代码处理模型，该模型能够基于第一代码和修改意见信息，同时输出修改意图和代码修改位置。第二种，代码处理过程使用两个模型，即意图识别模型和定位模型。意图识别模型能够基于第一代码的修改意见信息，输出修改意图，定位模型能够基于第一代码、修改意见信息和修改意图，输出代码修改位置。下面结合图6至图13，分别对上述两种实现方式进行说明。

图6是基于代码处理模型来进行代码处理的流程示意图。图6中包括两个过程，一个是获取第一代码和修改意见信息，再将第一代码和修改意见信息输入到代码处理模型中，以输出修改意图和代码修改位置，再根据该修改意图和代码修改位置，对第一代码进行处理的过程。另一个是基于训练数据集来对代码处理模型进行训练的过程。下面，基于图7至图9来对这两个过程进行进一步说明。

图7是根据本申请实施例提供的一种代码处理方法的流程图，下面结合图7，以代码开发客户端获取第一代码，服务器获取修改意图和代码修改位置为例，对基于代码处理模型实现的代码处理方法进行更加详细的说明，该方法包括以下步骤：

700、代码开发客户端显示代码处理界面，该代码处理界面用于提供代码处理功能。

其中，代码处理界面是代码开发客户端提供的一种功能界面，其中包括多个功能按钮，每个功能按钮对应不同的代码处理功能，能够辅助进行代码处理。

例如，该代码处理界面为针对代码调试的功能界面，能够实现为IDE客户端的编程插件，从而使用户能够在IDE客户端中打开该代码处理界面，对代码进行处理。

701、代码开发客户端响应于在该代码处理界面上对该代码处理功能的操作，获取开发得到的第一代码。

其中，代码处理功能可以提供多种输入方式，以获取第一代码。例如，该代码处理功能可以提供一个输入界面，该输入界面中显示不同的代码输入方式，例如，输入第一代码或输入第一代码的相关信息等。若代码输入方式为输入第一代码，则代码开发客户端直接通过该输入界面，获取输入的第一代码。若代码输入方式为输入第一代码的相关信息，则先通过该输入界面，获取第一代码的相关信息，再根据该第一代码的相关信息获取第一代码，该第一代码的相关信息包括第一代码的地址和第一代码的标识信息等，本申

请实施例对此不作限定。

例如，在IDE客户端中打开代码处理界面，从该代码处理界面提供的输入界面中输入代码，IDE客户端从该输入界面中获取输入的代码。或者，在IDE客户端中打开代码处理界面，从该代码处理界面提供的输入界面中输入代码的地址，IDE客户端从该输入界面中获取代码的地址，再从该地址中获取代码。

702、代码开发客户端响应于对该第一代码的提交操作，向服务器发送代码处理指令，该代码处理指令携带第一代码。

本申请实施例中，上述代码处理界面为用户提供一个提交按钮，用户点击该按钮以发起提交操作。响应于该提交操作，代码开发客户端向服务器发送携带第一代码的代码处理指令。

上述步骤700至702是以代码处理指令携带第一代码为例进行说明的，在一些实施例中，代码处理指令携带第一代码的相关信息。用户在输入界面中输入第一代码的相关信息后，代码开发客户端不根据该相关信息去获取第一代码，而是响应于对第一代码的提交操作，向服务器发送携带第一代码的相关信息的代码处理指令。

703、服务器接收代码处理指令。

本申请实施例是以服务器接收代码开发客户端发送的代码处理指令为例进行说明的，在一些实施例中，服务器周期性触发代码处理指令，以对服务器自身存储的第一代码进行处理。

704、服务器基于代码处理指令，获取第一代码。

本申请实施例中，代码处理指令携带第一代码，服务器直接获取代码处理指令携带的第一代码。在一些实施例中，代码处理指令携带第一代码的相关信息，服务器通过该相关信息获取第一代码。

705、服务器基于第一代码，通过意见获取模型，获取针对第一代码的修改意见信息，向代码开发客户端返回第一代码的修改意见信息，该修改意见信息包括描述性文本和待处理的代码段，描述性文本指示基于代码段对第一代码的修改意图，其中，修改意图包括从第一代码中删除该代码段、在第一代码中增加该代码段以及用该代码段对第一代码中的代码段进行替换。

本申请实施例中，服务器调用意见获取模型，对第一代码进行处理，得到针对第一代码的修改意见信息。在向代码开发客户端返回修改意见信息时，该修改意见信息可以采用图8的形式。参见图8，区域A为修改意见信息的显示区域，区域B1和B2中为描述性文本，区域C中为修改后的代码段，本申请实施例对修改意见信息所采用的形式不作限定。

例如，意见获取模型能够基于学习到的代码结构和编程语法等，检查代码中的变量设置、数据结构、系统和标准库调用、冗余代码和循环条件等，找出代码中的问题，针对该问题生成修改意见信息，向代码开发客户端返回该修改意见信息，该修改意见信息包括修改的代码段和针对该修改的描述性文本，该意见获取模型存储在服务器中，代码开发客户端通过与服务器交互，能够使服务器中的意见获取模型对代码进行处理，输出修改意见信息。

本申请实施例是以服务器通过意见获取模型，获取修改意见信息后，向代码开发客户端返回该修改意见信息为例进行说明的。通过向代码开发客户端提供修改意见信息，以向用户进行展示，进而提高代码处理的准确性。在一些实施例中，服务器获取修改意见信息后，直接基于该修改意见信息，获取修改意图和代码修改位置，也即是，服务器能够自动执行后续代码处理过程，从而避免人工检查造成的时延等问题，效率较高。

上述过程是以服务器通过意见获取模型获取修改意见信息为例进行说明的，在一些实施例中，服务器可以不通过意见获取模型获取修改意见信息，而是直接接收其他设备已生成的修改意见信息，也即是，服务器接收第一代码和修改意见信息，该第一代码和修改意见信息可以是代码开发客户端发出的代码处理指令携带的，也可以是其他设备发出的代码处理指令携带的。例如，代码开发客户端向其他设备发送第一代码，其他设备向代码开发客户端返回修改意见信息，若代码开发客户端接收到针对该修改意见信息的继续处理指令，则向服务器发送携带第一代码和修改意见信息的代码处理指令。再例如，代码开发客户端向其他设备发送第一代码，其他设备基于该第一代码获取修改意见信息后，直接向服务器发送携带第一代码和修改意见信息的代码处理指令。

上述步骤702至705是响应于代码分析指令，对获取到的第一代码进行分析，得到针对第一代码的修改意见信息的一种可能实现方式，该实现方式是通过代码开发客户端与服务器之间的交互来获取修改意见信息的，能够处理较大规模的代码处理任务。

706、代码开发客户端显示针对第一代码的修改意见信息，响应于针对该修改意见信息的触发操作，向服务器发送继续处理指令，或者，响应于针对该修改意见信息的触发操作，停止代码处理过程。

在一些实施例中，代码开发客户端所显示的针对第一代码的修改意见信息可以采用上述图8的形式。显示时，代码开发客户端不仅显示该修改意见信息，还显示继续处理按钮和停止处理按钮。其中，继续处理按钮用于响应于触发操作，指示代码开发客户端向服务器发送继续处理指令，停止处理按钮用于响应于触发操作，指示代码开发客户端停止代码处理过程。

在一些实施例中，代码开发客户端通过继续处理按钮，响应于针对修改意见信息的触发操作，向服务器发送继续处理指令。或者，代码开发客户端通过停止处理按钮，响应于针对修改意见信息的触发操作，停止代码处理过程。其中，停止代码处理过程可以是代码开发客户端向服务器发送停止处理指令，以使服务器不再执行后续代码处理过程，也可以是代码开发客户端不再执行后续代码处理过程，若服务器在预设时长内没有接收到针对该修改意见信息的继续处理指令，则不再执行后续代码处理过程。上述停止代码处理过程还可以是代码开发客户端打开人工修改界面，在该人工修改界面中展示第一代码，接收针对该第一代码的修改操作，本申请实施例对此不作限定。

707、若接收到继续处理指令，服务器基于第一代码和针对第一代码的修改意见信息，通过代码处理模型，获取针对第一代码的修改意图和第一代码中与修改意见信息对应的代码修改位置。

本申请实施例中，若接收到继续处理指令，服务器按照预设顺序，拼接第一代码和修改意见信息，调用代码处理模型，对拼接后的第一代码和修改意见信息进行处理，得到修改意图和代码修改位置。其中，该预设顺序与代码处理模型的训练数据中第一代码和修改意见信息的拼接顺序一致。代码处理模型具有语言理解和文本生成能力，能够对第一代码和修改意见信息进行分析，以理解针对第一代码的修改意图，并且，找到第一代码中修改意见信息对应的代码修改位置，再基于预先训练的输出格式或学习到的语言知识，按照固定格式输出修改意图和代码修改位置。

上述获取修改意图和代码修改位置的过程可以是，对拼接后的第一代码和修改意见信息进行文本分词，得到多个词语，将这多个词语转换为多个向量，调用代码处理模型，对这多个向量进行计算，得到多个组合及各个组合的概率，从多个组合中选出一个组合，按照固定格式输出该组合。其中，每个组合包括一对修改意图和代码修改位置，该概率指示输入为第一代码的情况下，输出为对应组合的概率。从多个组合中选出一个组合可以是选择其中概率最大的组合，也可以是在大于概率阈值的多个组合中，随机采样一个组合。固定格式包括：修改意图和代码修改位置的顺序、修改意图的形式以及代码修改位置的形式等。修改意图的形式可以是修改意图本身，也可以是修改意图对应的标识。代码修改位置的形式可以为数组或列表等。其中，每个向量指示对应词语在第一代码或修改意见信息中的位置以及对应词语与其他词语的相关性等信息。

以用行号表示代码的位置为例，服务器按照代码在前，修改意见信息在后的顺序，拼接代码和修改意见信息。调用代码处理模型，对带有行号的代码和针对该代码的修改意见信息进行分析，得到修改意图和行号的组合及各个组合的概率。在大于概率阈值的多个组合中，随机采样一个组合，将该组合按照“修改意图：XXX；代码修改位置：[XX，XX，…]”的格式输出。例如，代码处理模型经分析后得到的修改意图为“删除”，代码修改位置对应的行号为21、22、23、24和25，则代码处理模型的输出为“修改意图：删除；代码修改位置：[21，22，23，24，25]”，指示删除第21至25行的代码。

本申请实施例是以一对修改意图和代码修改位置对应一个概率为例进行说明的，在一些实施例中，每个修改意图对应一个概率，每个代码修改位置也对应一个概率。在多个修改意图和多个代码修改位置中，各选出一个修改意图和一个代码修改位置，将该修改意图和代码修改位置按照固定格式输出。

上述步骤706至707是基于分析得到的修改意见信息和第一代码进行分析，得到修改意图和第一代码中与修改意见信息对应的代码修改位置的一种可能实现方式，该实现方式在用户对修改意见信息进行确认后再进行后续处理过程，能够保证修改意见信息的准确性，从而提高代码处理的准确性。

708、服务器向代码开发客户端发送第一代码的修改意见信息的修改意图和代码修改位置。

本申请实施例中，服务器与代码开发客户端交互，向代码开发客户端发送代码处理模型输出的修改意图和代码修改位置，后续代码处理工作由代码开发客户端进行，能够分散服务器的工作量，减小服务器的负担。在一些实施例中，服务器基于修改意图和代码修改位置，对第一代码进行处理，得到第二代码，向代码开发客户端发送第二代码，能够在代码处理过程中实现用户无感知，使用户可以输入第一代码后，直

接得到第二代码。

上述步骤705至708是服务器获取修改意见信息以及基于修改意见信息与代码开发客户端之间通过交互来进一步获取修改意图和代码修改位置的一种可能实现方式，在该实现方式中，服务器在获取到修改意见信息后，会将修改意见信息先返回给代码开发客户端，由代码开发客户端上的用户来决定是否进行下一步的修改意图和代码修改位置的获取步骤，而服务器在接收到代码开发客户端触发的继续处理指令后，才会进行修改意图和代码修改位置的获取，并执行后续代码处理过程，能够更加符合用户的代码处理习惯，进而能够提高代码处理的准确性。在一些实施例中，服务器也可以不向代码开发客户端返回修改意见信息，无需用户来确定是否要基于该修改意见信息进行代码处理，而是由服务器在获取到修改意见信息后，直接基于该修改意见信息来获取修改意图和代码修改位置，再将修改意图和代码修改位置通过步骤708返回给代码开发客户端，以执行后续代码处理过程，能够避免人工检查造成的时延等问题，效率较高。

709、代码开发客户端基于修改意图和代码修改位置，在第一代码中的代码修改位置，基于代码段执行与修改意图对应的处理，得到第二代码。

本申请实施例中，代码开发客户端基于代码处理模型输出的修改意图和代码修改位置，对第一代码进行处理，得到第二代码。其中，若修改意图为增加，则在第一代码中代码修改位置处的代码后，插入修改意见信息中的代码段，得到第二代码；或，若修改意图为删除，则从第一代码中删除代码修改位置处的代码，得到第二代码；或，若修改意图为修改，则从第一代码中删除代码修改位置处的代码，在代码修改位置处插入修改意见信息中的代码段，得到第二代码。

在一些实施例中，若修改意图为增加，则代码开发客户端在代码修改位置后新增一空白行，在该空白行中插入修改意见信息中的代码段。若修改意图为删除，则代码开发客户端删除代码修改位置对应的代码行。若修改意图为修改，则代码开发客户端删除代码修改位置对应的代码，将修改意见信息中的代码段输入到代码修改位置中，或，代码开发客户端删除代码修改位置对应的代码行，在删除后的光标所在处后新增一空白行，将修改意见信息中的代码段插入到该空白行中。上述针对代码的修改方式，在新增的空白行中插入修改意见信息中的代码段，符合编程规范，能够避免代码出现语法错误。

例如，用行号表示代码修改位置，若代码处理模型的输出的修改意图为增加，代码修改位置为第21行，则代码开发客户端在代码的第21行后新增一空白行，即新增的第22行，将修改意见信息中的代码段插入到该第22行中。若代码处理模型输出的修改意图为删除，代码修改位置为第21至25行，则代码开发客户端从代码中删除第21至25行的代码行。若代码处理模型输出的修改意图为修改，代码修改位置为第21至25行，则代码开发客户端从代码中删除第21至25行的代码，将修改意见信息中的代码段插入到第21至25行中，或者，从代码中删除第21至25行的代码行，删除后，光标停留在第20行末尾，在第20行后新增一空白行，即新增的第21行，将修改意见信息中的代码段插入到该第21行中。

710、代码开发客户端在代码处理界面的预览窗口中显示第二代码。

在一些实施例中，预览窗口中不仅显示第二代码，还显示同意按钮和拒绝按钮，其中，该同意按钮用于响应于点击操作，触发同意指令，拒绝按钮用于响应于点击操作，触发拒绝指令。当然，上述同意指令和拒绝指令还可以基于其他方式触发，如语音指令等，本申请实施例对此不作限定。

本申请实施例中，在得到第二代码后，代码开发客户端提供预览功能，通过生成第二代码给用户预览，经检查后再执行修改，能够减小意见获取模型或代码处理模型的输出结果存在误差而带来的影响，从而提高代码处理的准确性。尤其对于代码来说，代码中存在一些特殊语法规则，由于这些特殊语法规则应用较少，意见获取模型或代码处理模型难以学习到这些特殊语法规则的相关知识，导致代码处理出错。通过该预览功能，能够判断处理后的代码中是否存在这些特殊语法规则导致的错误，从而保证代码处理的准确性。

在一些实施例中，代码处理界面还提供人工修改选项，响应于对该人工修改选项的触发操作，代码开发客户端显示人工修改界面，在该人工修改界面中显示第二代码，接收针对第二代码的修改操作。代码开发客户端接收到针对修改后的第二代码的提交操作后，使用修改后的第二代码替换第一代码，以实现对第一代码的处理。

711、若代码开发客户端接收到针对预览窗口所显示的第二代码的同意指令，则将第一代码替换为第二代码。

在一些实施例中，代码开发客户端响应于通过同意按钮触发的针对第二代码的同意指令，将第一代码替换为第二代码。或者，代码开发客户端响应于通过拒绝按钮触发的针对第二代码的拒绝指令，执行该拒

绝指令对应的操作，该拒绝指令对应的操作可以是获取针对第二代码的反馈意见信息，将第二代码的反馈意见信息和第一代码的修改意见信息一起发送给服务器。服务器基于第二代码的反馈意见信息和第一代码的修改意见信息再次执行后续代码处理过程。上述拒绝指令对应的操作也可以是打开人工修改界面，在该人工修改界面中展示第一代码，接收针对该第一代码的修改操作。

其中，将第一代码替换为第二代码可以是，代码开发客户端将上述输入界面中的第一代码替换为第二代码，或，基于上述输入界面中第一代码的相关信息，用第二代码替换第一代码等，本申请实施例对此不作限定。

针对不同的场景，代码的替换方式不同。例如，本申请实施例提供的代码处理方法可以是代码发布、代码上线和代码测试过程中的一种代码调试手段，能够将原本的代码替换为调试后的代码。若代码开发客户端提供代码发布功能，则处理前的代码可以为待发布到代码库中的代码，代码开发客户端接收到同意指令后，将处理后的代码发布到代码库中，以实现对处理前的代码的替换。处理前的代码也可以为已经存储在代码库中的代码，代码开发客户端接收到同意指令后，用处理后的代码替换代码库中的代码。若代码开发客户端提供代码上线功能，则处理前的代码为待上线的代码，代码开发客户端接收到同意指令后，将处理后的代码上线到平台，以实现对处理前的代码的替换，实现平台功能的增加。处理后的代码也可以是已在平台上线的代码，代码开发客户端接收到同意指令后，用处理后的代码替换平台中对应位置、对应名称或对应功能的代码。若处理前的代码可以为待测试的代码，代码开发客户端接收到同意指令后，用处理后的代码代替处理前的代码进行代码测试。代码的替换方式还可以是，代码开发客户端运行有IDE客户端，IDE客户端中包括用于输入代码的编辑区，编辑区中为处理前的代码。在代码调试过程中，若代码开发客户端接收到同意指令，则代码开发客户端将编辑区中的代码替换为处理后的代码。上述过程表明，本申请实施例提供的代码处理方法能够为编程的不同阶段提供相应的代码调试功能，还能够在不同阶段对代码做出不同处理，应用范围广泛。

本申请实施例是以将第一代码替换为第二代码为例进行说明的，在一些实施例中，代码开发客户端接收到同意指令后，不是用第二代码替换第一代码，而是按照服务器返回的修改意图和修改位置对第一代码进行处理，能够在第二代码较大的情况下，减小代码处理所占用的CPU资源。

上述步骤709至711是基于修改意图和代码修改位置，对第一代码进行处理的一种可能实现方式，该实现方式通过将第二代码提供给用户，能够使用户检查该第二代码，从而保证代码处理的准确性。根据用户反馈的检查结果，可以选择使用新的数据集对代码处理所使用的模型进行训练，也可以修改模型结构后重新进行训练，从而调整代码处理所使用的模型，以实现模型的优化，进而提高模型的准确性。上述均是以代码开发客户端提供预览功能为例进行说明的，该预览功能通过用户检查，能够提高代码处理的准确性。在一些实施例中，代码开发客户端不提供预览功能，直接对第一代码进行处理，能够避免人工检查造成的时延等问题，效率较高。

上述图7所示的实施例是以代码开发客户端获取第一代码，服务器获取修改意图和代码修改位置为例进行说明的，该场景仅为示例性场景，在一些实施例中，计算设备具有独立完成代码处理任务的能力，整个代码处理功能均在该计算设备上进行，不用与其他设备进行交互，代码处理的效率较高。在另一些实施例中，多台计算设备各自负责代码处理任务中的部分步骤，协作完成该代码处理任务，尤其是在基于云服务实现代码处理任务时，云服务器协同工作，能够实现用户无感知处理，使用户输入第一代码后，直接得到第二代码。

本申请实施例提供的代码处理方法，在代码处理过程中，获取第一代码和修改意见信息，由于该修改意见信息的语义以及第一代码和修改意见信息之间的关联等，与修改意图和代码修改位置之间存在一定规律，因此，根据该第一代码和修改意见信息，能够获取修改意图和代码修改位置，以实现对修改意见信息的自动化理解。基于该修改意图和代码修改位置，对第一代码进行处理。由于待处理的代码量可能巨大，因此，与人工理解上述修改意见信息后再对代码进行处理相比，本申请实施例提供的代码处理方法能够降低代码处理过程中的出错率，提高代码的处理效率。

上述图7所示的代码处理方法通过代码处理模型获取修改意图和代码修改位置，以该代码处理模型为LLM为例，该代码处理模型的训练过程包括预训练阶段和微调阶段。在预训练阶段，使用大量的训练数据训练语言模型，得到代码处理模型。此时的代码处理模型是一种通用语言模型，已经学习到基础的语言知

识，能够通用于多种应用场景。在微调阶段，使用相关领域的文本数据，对预训练阶段得到的代码处理模型进行微调，从而使代码处理模型学习具体领域中的知识，进而能够更加准确地处理该具体领域中的代码处理任务。图9是根据本申请实施例提供的一种代码处理模型训练方法的流程图，下面结合图9，以服务器执行该模型训练方法为例进行说明。其中，步骤900为预训练阶段，步骤901至904为微调阶段。上述方法包括以下步骤：

900、服务器获取代码处理模型。

其中，该代码处理模型是一种语言模型，该语言模型通过大量训练数据训练得到，能够在训练过程中学习到语言的结构、语法和语义等语言模式和表征。本申请实施例中，服务器获取之前下载的语言模型，或，接收其他设备发送的该语言模型，将该语言模型作为代码处理模型，以进行后续训练。以该语言模型为基础进行微调，只用在通用语言知识的基础上，学习相关领域的知识即可，不用再从头开始训练新的模型，从而节省时间和计算资源。

本申请实施例是以服务器将已经完成预训练的语言模型作为代码处理模型为例说明的，在一些实施例中，服务器基于大量训练数据，对语言模型重新进行预训练，得到代码处理模型。其中，预训练为无监督训练，包括多轮训练过程。训练前，服务器对大量训练数据进行文本分词，得到多个词语，将这多个词语转换为向量，每个向量指示对应词语在大量训练数据中的位置以及对应词语与其他词语的相关性等信息。在任一轮训练过程中，服务器调用语言模型，对该向量进行计算，得到各个词语的多个下一词语及对应概率，若该概率符合训练停止条件，则将对应语言模型作为代码处理模型，若该概率不符合训练停止条件，则基于该概率对语言模型进行反向传播训练，从而调整该语言模型中的参数，再进行下一轮训练。其中，训练停止条件可以为上述概率在一定数值范围内，或，模型训练的轮数达到预设轮数，本申请实施例对此不作限定。服务器使用大量训练数据对语言模型重新进行预训练，能够使文字处理模型学习到的知识更加符合代码处理任务所对应的领域，使代码处理的准确性更高。

901、服务器获取第一训练数据集，该第一训练数据集包括多组第一样本数据和第一标签，第一样本数据包括一组第一代码和修改意见信息，该组的第一标签包括一组与该组第一样本数据对应的修改意图和代码修改位置。

其中，第一样本数据为代码处理模型的输入数据，第一标签为第一样本数据对应的真实标签，该第一样本数据可以是人工采集得到的，第一标签可以是人工标注得到的。本申请实施例中，服务器本地读取预先下载好的第一训练数据集，或，从开源库中获取第一训练数据集。本申请实施例是以人工标注第一样本数据，以获取第一训练数据集为例进行说明的，在一些实施例中，服务器记录历史代码处理过程中的相关信息，得到第一训练数据集，本申请实施例对此不做限定。

在一些实施例中，服务器将获取的数据按照预设比例，分为第一训练数据集和第一测试数据集，简称训练集和测试集。训练集用于训练模型，测试集用于检验模型训练结果，通过该模型训练结果，能够对模型进行进一步训练，从而提高模型性能。

在对代码处理模型进行训练的过程中包括多轮训练，下述步骤902至904是对一轮训练过程的说明。

902、每轮训练过程中，服务器将第一样本数据输入到代码处理模型中，得到第一输出数据。

其中，第一输出数据为多个组合及各个组合的概率，每个组合中包括一组修改意图和代码修改位置，该概率指示输入为第一样本数据的情况下，输出为对应组合的概率。在一些实施例中，服务器基于第一训练数据集，通过每一轮的训练，对代码处理模型进行微调，使微调后的模型能够被应用于预设领域。训练前，服务器对第一样本数据进行文本分词，得到多个词语，将这多个词语转换为向量，每个向量指示对应词语在第一样本数据中的位置以及对应词语与其他词语的相关性等信息。每轮训练过程中，服务器调用代码处理模型，对第一样本数据对应的向量进行计算，得到第一输出数据。

本申请实施例是以第一输出数据为多个组合及各个组合的概率为例说明的，在一些实施例中，上述第一输出数据为一组修改意图和修改位置。

903、服务器基于第一输出数据和第一标签，获取第一损失函数值。

其中，第一输出数据为代码处理模型预测的修改意图和代码修改位置，第一标签为真实的修改意图和代码修改位置，第一损失函数值指示代码处理模型预测的准确性。本申请实施例中，服务器获取第一输出数据的多个组合的概率分布和第一样本数据的真实标签的概率分布，根据预设函数，对这两个概率分布进行变换，得到第一损失函数值。

本申请实施例是以第一输出数据为多个组合及各个组合的概率为例说明的，在一些实施例中，上述第一输出数据为一组修改意图和修改位置，则服务器将第一输出数据和第一标签之间的误差作为第一损失函数值。

904、若该轮训练不满足训练停止条件，则服务器基于第一损失函数值，调整代码处理模型中的参数，若该轮训练满足训练停止条件，则将该模型输出为代码处理模型。

其中，调整的参数可以为代码处理模型中的部分参数，也可以为代码处理模型中新增的参数，本申请实施例对此不作限定。第一损失函数值指示：预测的概率分布和真实的概率分布之间的差异，也即是，第一输出数据的多个组合的概率分布和第一样本数据的真实标签的概率分布之间的差异。若该轮训练不满足训练停止条件，则服务器基于第一损失函数值，调整代码处理模型中的参数，再进行下一轮训练，使得第一损失函数值所指示的差异更小，直至满足训练停止条件。训练停止条件可以为第一损失函数值在一定数值范围内，也可以为模型训练的轮数达到预设轮数，本申请实施例对此不作限定。若该轮训练满足训练停止条件，则将该模型输出为代码处理模型。

本申请实施例是以第一损失函数值指示预测的概率分布和真实的概率分布之间的差异为例说明的，在一些实施例中，第一损失函数值指示代码处理模型的预测标签与第一标签之间的误差。

在一些实施例中，服务器通过模型监督微调（supervised fine-tuning，SFT）和提示工程（prompt）调整代码处理模型，使代码处理模型具备端到端的处理能力，从而能够同时生成修改意图和代码修改位置。

上述步骤902至904是基于第一训练数据集，对代码处理模型进行训练的一种可能实现方式，该实现方式中，代码处理模型为LLM，能够在现有的语言模型上进行训练，不用再从头开始训练模型，能够节省计算时间和计算资源。在一些实施例中，该代码处理模型也可以是其他具有自然语言处理能力或强大分类能力的模型。

上述图9所示的代码处理模型，通过大量基础训练数据和具体领域的训练数据训练得到，具有强大的针对具体领域的自然语言处理能力，能够较为准确地根据第一代码和修改意见信息，同时输出修改意图和代码修改位置，提高代码处理的效率。

图10是基于意图识别模型和定位模型来进行代码处理的流程示意图，图10中包括三个过程，一个是获取第一代码和修改意见信息，将第一代码和修改意见信息输入到意图识别模型中，得到修改意图，再将第一代码、修改意见信息和修改意图输入到定位模型中，得到代码修改位置，再根据该修改意图和代码修改位置，对第一代码进行处理的过程。另一个是基于训练数据集来对意图识别模型进行训练的过程。最后一个是基于训练数据集来对定位模型进行训练的过程。下面，基于图11至图13来对这三个过程进行进一步说明。

图11是根据本申请实施例提供的一种代码处理方法的流程图。下面结合图11，以代码开发客户端为代码开发客户端，由代码开发客户端获取第一代码，服务器执行上述代码处理方法为例，对基于意图识别模型和定位模型实现的代码处理方法进行更加详细的说明，该方法包括以下步骤：

1100、代码开发客户端显示代码处理界面，该代码处理界面用于提供代码处理功能。

1101、代码开发客户端响应于在该代码处理界面上对该代码处理功能的操作，获取开发得到的第一代码。

1102、代码开发客户端响应于对该第一代码的提交操作，向服务器发送代码处理指令，该代码处理指令携带第一代码。

1103、服务器接收代码处理指令。

1104、服务器基于代码处理指令，获取第一代码。

1105、服务器基于第一代码，通过意见获取模型，获取针对第一代码的修改意见信息，向代码开发客户端返回第一代码的修改意见信息，该修改意见信息包括描述性文本和待处理的代码段，描述性文本指示基于代码段对第一代码的修改意图，其中，修改意图包括从第一代码中删除该代码段、在第一代码中增加该代码段以及用该代码段对第一代码中的代码段进行替换。

1106、代码开发客户端显示针对第一代码的修改意见信息，响应于针对该修改意见信息的触发操作，向服务器发送继续处理指令，或者，响应于针对该修改意见信息的触发操作，停止代码处理过程。

上述步骤1100至1106与步骤700至706同理，本申请实施例在此不作赘述。

1107、若接收到继续处理指令，服务器基于第一代码和针对第一代码的修改意见信息，通过意图识别模型，获取针对第一代码的修改意图。

本申请实施例中，若接收到继续处理指令，服务器按照预设顺序，拼接第一代码和修改意见信息，调用意图识别模型，对拼接后的第一代码和修改意见信息进行处理，得到修改意图。其中，预设顺序与训练意图识别模型的数据中，第一代码和修改意见信息的拼接顺序一致，意图识别模型用于获取修改意图，即判断修改意图为增加、删除和修改中的哪一类，模型的输入为第一代码和修改意见信息，输出为增加、删除和修改这三类修改意图中的一类。上述获取修改意图的过程可以是，对拼接后的第一代码和修改意见信息进行文本分词，得到多个词语，将这多个词语转换为多个向量，调用意图识别模型，对这多个向量进行计算，得到修改意图。

例如，服务器按照代码在前，修改意见信息在后的顺序，拼接代码和修改意见信息，调用意图识别模型，对带有行号的代码和针对该代码的修改意见信息进行分析，得到修改意图。

本申请实施例是以意图识别模型直接输出一个修改意图为例进行说明的，在一些实施例中，意图识别模型先获取多个修改意图和各个修改意图的概率，再从多个修改意图中选出一个修改意图，输出该修改意图。其中，从多个修改意图中选出一个修改意图可以是选择其中概率最大的修改意图，也可以是在大于概率阈值的多个修改意图中，随机采样一个修改意图，该概率指示输入为第一代码和对应修改意见信息的情况下，输出为对应修改意图的概率。

上述实施例均是以意图识别模型输出一个修改意图为例进行说明的，在另一些实施例中，意图识别模型输出多个修改意图和各个修改意图对应的概率，服务器会基于概率，从多个修改意图中选出一个修改意图，根据该修改意图继续执行后续代码处理过程，本申请实施例对此不作限定。

1108、服务器基于第一代码、针对第一代码的修改意见信息和修改意图，获取第一代码中与修改意见信息对应的代码修改位置。

本申请实施例中，服务器按照预设顺序，拼接第一代码、修改意见信息和修改意图，调用定位模型，对拼接后的第一代码、修改意见信息和修改意图进行处理，得到与修改意见信息对应的代码修改位置，该定位模型用于获取代码修改位置，即判断应该在第一代码的哪个位置实施修改。定位模型的输入为第一代码、修改意见信息和修改意图，输出为代码修改位置。上述获取代码修改位置的过程可以是，对拼接后的第一代码、修改意见信息和修改意图进行文本分词，得到多个词语，将这多个词语转换为多个向量，调用定位模型，对这多个向量进行计算，得到代码修改位置。

以用行号表示代码的位置为例。服务器按照代码、修改意见信息和修改意图的顺序，拼接代码、修改意见信息和修改意图，调用定位模型，对带有行号的代码、修改意见信息和修改意图进行分析，得到代码修改位置，即行号。

本申请实施例是以定位模型的输出为一个代码修改位置为例进行说明的，在一些实施例中，定位模型通过分析第一代码、修改意见信息和修改意图，先获取多个代码修改位置和各个代码修改位置的概率，再从多个代码修改位置中选出一个代码修改位置，输出该代码修改位置。其中，从多个代码修改位置中选出一个代码修改位置可以是选择其中概率最大的代码修改位置，也可以是在大于概率阈值的多个代码修改位置中，随机采样一个代码修改位置，该概率指示输入为第一代码、对应修改意见信息和修改意图的情况下，输出为对应代码修改位置的概率。

上述实施例均是以定位模型输出一个代码修改位置为例进行说明的，在另一些实施例中，定位模型输出多个代码修改位置和各个代码修改位置对应的概率，服务器会基于概率，从多个代码修改位置中选出一个代码修改位置，根据该代码修改位置继续执行后续代码处理过程，本申请实施例对此不作限定。

1109、服务器向代码开发客户端发送第一代码的修改意见信息的修改意图和代码修改位置。

1110、代码开发客户端基于修改意图和代码修改位置，在第一代码中的代码修改位置，基于代码段执行与修改意图对应的处理，得到第二代码。

1111、在代码处理界面的预览窗口中显示第二代码。

1112、若代码开发客户端接收到针对预览窗口所显示的第二代码的同意指令，则将第一代码替换为第二代码。

上述步骤1109至1112与步骤708至711同理，本申请实施例在此不作赘述。

上述图10所示的代码处理方法通过意图识别模型获取修改意图，该意图识别模型可以是多种自然语言处理模型。图12是根据本申请实施例提供的一种意图识别模型训练方法的流程图，下面结合图12，以服务器执行该模型训练方法为例进行说明，该方法包括以下步骤：

1201、服务器获取第二训练数据集，该第二训练数据集包括多组第二样本数据和第二标签，第二样本数据包括一组第一代码和修改意见信息，该组的第二标签包括一个与该组第二样本数据对应的修改意图。

其中，第二样本数据为意图识别模型的输入数据，第二标签为第二样本数据对应的真实标签。步骤1201中获取第二训练数据集的过程与上述步骤901中获取第一训练数据集的过程同理，本申请实施例不再赘述。

在对意图识别模型进行训练的过程中包括多轮训练，下述步骤1202至1204是对一轮训练过程的说明。

1202、每轮训练过程中，服务器将第二样本数据输入到意图识别模型中，得到第二输出数据。

其中，该意图识别模型为根据代码处理任务选择的模型。训练前，服务器对第二样本数据进行文本分词，得到多个词语，将这多个词语转换为向量，每个向量指示对应词语在第二样本数据中的位置以及对应词语与其他词语的相关性等信息。每轮训练过程中，服务器调用意图识别模型，对第二样本数据对应的向量进行计算，得到第二输出数据，即意图识别模型预测的修改意图。

在一些实施例中，该意图识别模型为循环神经网络（recurrent neural network，RNN）模型。在一轮训练过程中，第二样本数据对应的向量在RNN模型中进行前向传播计算，得到第二输出数据。其中，第二样本数据对应的向量经过RNN模型中循环层、隐藏层和全连接层等神经网络层的计算，得到第二输出数据。

在一些实施例中，该意图识别模型为Transformers模型。Transformers模型包括多个编码模块（encoder）和解码模块（decoder）。encoder模块用于对输入数据编码，从而将输入数据转换为中间数据，decoder模块用于对中间数据解码，从而将该中间数据转换为输出数据。encoder模块和decoder模块的结构相同，参数可能不同。每个encoder模块或decoder模块中，包括自注意力层（self-attention，SA）和前馈神经网络层（feed forward neural network，FFNN）。以encoder模块为例，SA层用于在对输入数据中的词语编码时，让编码后的词语包含输入数据中其他词语的信息，即包含该词语与其他词语之间的关系。FFNN层为前向网络层，用于对SA层的输出进行近一步的特征提取。在一轮训练过程中，调用encoder模块，对第二样本数据对应的向量进行处理，得到中间数据，调用decoder模块，对该中间数据进行处理，得到第二输出数据。

在一些实施例中，该意图识别模型为朴素贝叶斯分类模型。在一轮训练过程中，调用朴素贝叶斯分类模型，对第二样本数据对应的向量进行计算，得到至少一个修改意图和各个修改意图对应的概率，该概率指示在输入为第二样本数据的情况下，输出为对应修改意图的概率，将至少一个修改意图中概率最大的修改意图作为第二输出数据。

1203、服务器基于第二输出数据和第二标签，获取第二损失函数值。

其中，第二输出数据为意图识别模型预测的修改意图，第二标签为真实的修改意图，第二损失函数值指示第二输出数据与第二标签之间的误差。服务器对调用预设的损失函数，基于第二输出数据和第二标签进行计算，得到第二损失函数值。

1204、若该轮训练不满足训练停止条件，则服务器基于第二损失函数值，调整意图识别模型中的参数，若该轮训练满足训练停止条件，则将该模型输出为意图识别模型。

本申请实施例中，若该轮训练不满足训练停止条件，则服务器基于第二损失函数值，计算模型中各个参数误差。基于该误差调整意图识别模型中的参数，再进行下一轮训练，使得第二损失函数值更小，直至满足训练停止条件。训练停止条件可以为第二损失函数值在一定数值范围内，也可以为模型训练的轮数达到预设轮数，本申请实施例对此不作限定。若该轮训练满足训练停止条件，则将该模型输出为意图识别模型。

上述步骤1202至1204是基于第二训练数据集，对意图识别模型进行训练的一种可能实现方式，该实现方式中提供了多种可以作为意图识别模型的模型，该实现方式中所提到的模型仅为示例性的，意图识别模型还可以为其他自然语言处理模型。

上述图10所示的代码处理方法通过定位模型获取代码修改位置，该定位模型可以是多种自然语言处理模型。图13是根据本申请实施例提供的一种定位模型训练方法的流程图，下面结合图13，以服务器执行该模型训练方法为例进行说明，该方法包括以下步骤：

1301、服务器获取第三训练数据集，该第三训练数据集包括多组第三样本数据和第三标签，第三样本

数据包括一组第一代码、修改意见信息和修改意图，该组的第三标签包括一个与该组第三样本数据对应的代码修改位置。

其中，第三样本数据为定位模型的输入数据，第三标签为第三样本数据对应的真实标签。步骤1301中获取第三训练数据集的过程与上述步骤901中获取第一训练数据集的过程同理，本申请实施例不再赘述。

在对定位模型进行训练的过程中包括多轮训练，下述步骤1302至1304是对一轮训练过程的说明。

1302、每轮训练过程中，服务器将第三样本数据输入到定位模型中，得到第三输出数据。

1303、服务器基于第三输出数据和第三标签，获取第三损失函数值。

1304、若该轮训练不满足训练停止条件，则服务器基于第三损失函数值，调整定位模型中的参数，若该轮训练满足训练停止条件，则将该模型输出为定位模型。

上述步骤1302至1304是基于第三训练数据集，对定位模型进行训练的一种可能实现方式，该实现方式中提供了多种可以作为定位模型的模型，该实现方式中所提到的模型仅为示例性的，定位模型还可以为其他自然语言处理模型。

上述步骤1301至1304的定位模型训练方法与步骤1201至1204的意图识别模型的训练方法同理，本申请实施例不再赘述。

上述图11至图13说明的代码处理方法，使用两个模型，分别确定修改意图和代码修改位置，模型的目标较为明确，预测的准确率较高。

以上介绍了本申请实施例的方法，以下介绍本申请实施例的装置。应理解，以下介绍的装置具有上述方法中计算设备的任意功能。上文结合图6至图13详细描述了根据本申请实施例提供的代码处理方法以及该代码处理方法所使用的模型的训练方法。基于同一发明构思，下面将结合图14描述根据本申请实施例提供的代码处理装置。应理解，方法实施例所描述的技术特征同样适用于以下装置实施例。

参见图14，本申请实施例提供了一种代码处理装置，该装置包括：

代码获取模块1401，用于获取开发得到的第一代码；

修改意见信息获取模块1402，用于响应于代码分析指令，对获取到的第一代码进行分析，得到针对第一代码的修改意见信息，修改意见信息包括描述性文本和待处理的代码段，描述性文本指示基于代码段对第一代码的修改意图，其中，修改意图包括从第一代码中删除该代码段、在第一代码中增加该代码段以及用该代码段对第一代码中的代码段进行替换；

意见识别模块1403，用于基于分析得到的修改意见信息和第一代码进行分析，得到修改意图和第一代码中与修改意见信息对应的代码修改位置；

代码处理模块1404，用于基于修改意图和代码修改位置，在第一代码中的代码修改位置，基于代码段执行与修改意图对应的处理。

在一些实施例中，意见识别模块1403用于：

响应于针对修改意见信息的触发操作，基于修改意见信息和第一代码进行分析，得到修改意图和代码修改位置。

在一些实施例中，上述代码处理模块1404包括：

第二代码获取单元，用于基于修改意图和代码修改位置，在第一代码中的代码修改位置，基于代码段执行与修改意图对应的处理，得到第二代码；

显示单元，用于在预览窗口中显示第二代码；

替换单元，用于若接收到同意指令，则将第一代码替换为第二代码。

在一些实施例中，上述第二代码获取单元用于：

若修改意图为增加，则在第一代码中代码修改位置处的代码后，插入修改意见信息中的代码段，得到第二代码；或，

若修改意图为删除，则从第一代码中删除代码修改位置处的代码，得到第二代码；或，

若修改意图为修改，则从第一代码中删除代码修改位置处的代码，在代码修改位置处插入修改意见信息中的代码段，得到第二代码。

在一些实施例中，上述代码处理装置还包括：

代码处理模型训练模块，用于获取第一训练数据集，第一训练数据集包括多组第一样本数据和第一标

签，第一样本数据包括一组第一代码和修改意见信息，第一标签包括一组与第一样本数据对应的多组修改意图和代码修改位置；基于第一训练数据集，对代码处理模型进行训练。

在一些实施例中，上述代码处理装置还包括：

意图识别模型训练模块，用于获取第二训练数据集，第二训练数据集包括多组第二样本数据和第二标签，第二样本数据包括一组第一代码和修改意见信息，第二标签包括一个与第二样本数据对应的修改意图；基于第二训练数据集，对意图识别模型进行训练。

在一些实施例中，上述代码处理装置还包括：

定位模型训练模块，用于获取第三训练数据集，第三训练数据集包括多组第三样本数据和第三标签，第三样本数据包括一组第一代码、修改意见信息和修改意图，第三标签包括一个与第三样本数据对应的代码修改位置；基于第三训练数据集，对定位模型进行训练。

应理解，代码处理装置对应于上述方法实施例中的代码开发客户端，装置中的各模块和上述其他操作和/或功能分别为了实现方法实施例中的代码开发客户端所实施的各种步骤和方法，具体细节可参见上述方法实施例，为了简洁，在此不再赘述。

其中，代码获取模块1401、修改意见信息获取模块1402、意见识别模块1403和代码处理模块1404均可以通过软件实现，或者可以通过硬件实现。示例性的，接下来以代码获取模块1401为例，介绍代码获取模块1401的实现方式。类似的，修改意见信息获取模块1402、意见识别模块1403和代码处理模块1404的实现方式可以参考代码获取模块1401的实现方式。

模块作为软件功能单元的一种举例，代码获取模块1401可以包括运行在计算实例上的代码。其中，计算实例可以包括物理主机（计算设备）、虚拟机、容器中的至少一种。进一步地，上述计算实例可以是一台或者多台。例如，代码获取模块1401可以包括运行在多个主机/虚拟机/容器上的代码。需要说明的是，用于运行该代码的多个主机/虚拟机/容器可以分布在相同的region中，也可以分布在不同的region中。进一步地，用于运行该代码的多个主机/虚拟机/容器可以分布在相同的AZ中，也可以分布在不同的AZ中，每个AZ包括一个数据中心或多个地理位置相近的数据中心。其中，通常一个region可以包括多个AZ。

同样，用于运行该代码的多个主机/虚拟机/容器可以分布在同一个VPC中，也可以分布在多个VPC中。其中，通常一个VPC设置在一个region内。同一region内两个VPC之间，以及不同region的VPC之间跨区通信需在每个VPC内设置通信网关，经通信网关实现VPC之间的互连。

模块作为硬件功能单元的一种举例，代码获取模块1401可以包括至少一个计算设备，如服务器等。或者，代码获取模块1401也可以是利用ASIC实现、或PLD实现的设备等。其中，上述PLD可以是CPLD、FPGA、GAL或其任意组合实现。

代码获取模块1401包括的多个计算设备可以分布在相同的region中，也可以分布在不同的region中。代码获取模块1401包括的多个计算设备可以分布在相同的AZ中，也可以分布在不同的AZ中。同样，代码获取模块1401包括的多个计算设备可以分布在同一个VPC中，也可以分布在多个VPC中。其中，所述多个计算设备可以是服务器、ASIC、PLD、CPLD、FPGA和GAL等计算设备的任意组合。

需要说明的是，在其他实施例中，代码获取模块1401可以用于执行代码处理方法中的任意步骤，修改意见信息获取模块1402可以用于执行代码处理方法中的任意步骤，意见识别模块1403可以用于执行代码处理方法中的任意步骤，代码处理模块1404可以用于执行代码处理方法中的任意步骤，代码获取模块1401、修改意见信息获取模块1402、意见识别模块1403和代码处理模块1404负责实现的步骤可根据需要指定，通过代码获取模块1401、修改意见信息获取模块1402、意见识别模块1403和代码处理模块1404分别实现代码处理方法中不同的步骤来实现代码处理装置的全部功能。

本申请实施例还提供了一种计算机可读存储介质。所述计算机可读存储介质可以是计算设备能够存储的任何可用介质或者是包含一个或多个可用介质的数据中心等数据存储设备。所述可用介质可以是磁性介质，（例如，软盘、硬盘、磁带）、光介质（例如，高密度数字视频光盘（digital video disc，DVD））、或者半导体介质（例如固态硬盘）等。该计算机可读存储介质包括计算机程序指令，当所述计算机程序指令由计算设备集群执行时，所述计算设备集群执行代码处理方法。

本申请实施例还提供了一种包含指令的计算机程序产品，能够运行在计算设备上或被储存在任何可用介质中的软件或程序产品。当所述计算机程序产品在至少一个计算设备上运行时，使得至少一个计算设备

执行代码处理方法。

另外，本申请的实施例还提供一种装置，这个装置具体可以是芯片，组件或模块，该装置可包括相连的处理器和存储器；其中，存储器用于存储计算机执行指令，当装置运行时，处理器可执行存储器存储的计算机执行指令，以使芯片执行上述各方法实施例中的代码处理方法。

其中，本实施例提供的装置、计算设备集群、计算机可读存储介质、计算机程序产品或芯片均用于执行上文所提供的对应的方法，因此，其所能达到的有益效果可参考上文所提供的对应的方法中的有益效果，此处不再赘述。

通过以上实施方式的描述，所属领域的技术人员可以了解到，为描述的方便和简洁，仅以上述各功能模块的划分进行举例说明，实际应用中，可以根据需要而将上述功能分配由不同的功能模块完成，即将装置的内部结构划分成不同的功能模块，以完成以上描述的全部或者部分功能。另外，与上述实施例提供的代码处理方法实施例属于同一构思，其具体实现过程详见方法实施例，这里不再赘述。

在本申请所提供的几个实施例中，应该理解到，所揭露的装置和方法，可以通过其它的方式实现。例如，以上所描述的装置实施例仅仅是示意性的，例如，所述模块或单元的划分，仅仅为一种逻辑功能划分，实际实现时可以有另外的划分方式，例如多个单元或组件可以结合或者可以集成到另一个装置，或一些特征可以忽略，或不执行。另一点，所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口，装置或单元的间接耦合或通信连接，可以是电性，机械或其它的形式。

所述作为分离部件说明的单元可以是或者也可以不是物理上分开的，作为单元显示的部件可以是一个物理单元或多个物理单元，即可以位于一个地方，或者也可以分布到多个不同地方。可以根据实际的需要选择其中的部分或者全部单元来实现本实施例方案的目的。

另外，在本申请各个实施例中的各功能单元可以集成在一个处理单元中，也可以是各个单元单独物理存在，也可以两个或两个以上单元集成在一个单元中。上述集成的单元既可以采用硬件的形式实现，也可以采用软件功能单元的形式实现。

所述集成的单元如果以软件功能单元的形式实现并作为独立的产品销售或使用时，可以存储在一个可读取存储介质中。基于这样的理解，本申请实施例的技术方案本质上或者说对现有技术做出贡献的部分或者该技术方案的全部或部分可以以软件产品的形式体现出来，该软件产品存储在一个存储介质中，包括若干指令用以使得一个设备（可以是单片机，芯片等）或处理器（processor）执行本申请各个实施例所述方法的全部或部分步骤。而前述的存储介质包括：U盘、移动硬盘、ROM、RAM、磁碟或者光盘等各种可以存储程序代码的介质。

在本申请的描述中，除非另有说明，“/”表示“或”的意思，例如，A/B可以表示A或B。本文中的“和/或”仅仅是一种描述关联对象的关联关系，表示可以存在三种关系，例如，A和/或B，可以表示：单独存在A，同时存在A和B，单独存在B这三种情况。此外，“至少一个”是指一个或多个，“多个”是指两个或两个以上。“第一”、“第二”等字样并不对数量和执行次序进行限定，并且“第一”、“第二”等字样也并不限定一定不同。

本申请中，“示例性的”或者“例如”等词用于表示作例子、例证或说明。本申请中被描述为“示例性的”或者“例如”的任何实施例或设计方案不应被解释为比其他实施例或设计方案更优选或更具优势。确切而言，使用“示例性的”或者“例如”等词旨在以具体方式呈现相关概念。

需要说明的是，本申请所涉及的信息（包括但不限于用户设备信息、用户个人信息等）、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）以及信号，均为经用户授权或者经过各方充分授权的，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。例如，本申请中涉及到的代码都是在充分授权的情况下获取的。

上述所有可选技术方案，可以采用任意结合形成本公开的可选实施例，在此不再一一赘述。

最后应说明的是：以上实施例仅用以说明本申请的技术方案，而非对其限制；尽管参照前述实施例对本申请进行了详细的说明，本领域的普通技术人员应当理解：其依然可以对前述各实施例所记载的技术方案进行修改，或者对其中部分技术特征进行等同替换；而这些修改或者替换，并不使相应技术方案的本质脱离本申请各实施例技术方案的保护范围。

本领域普通技术人员可以理解实现上述实施例的全部或部分步骤可以通过硬件来完成，也可以通过程序来指令相关的硬件完成，所述的程序可以存储于一种计算机可读存储介质中，上述提到的存储介质可以是只读存储器，磁盘或光盘等。

以上所述仅为本申请的示例性实施例，并不用以限制本申请，凡在本申请的精神和原则之内，所作的任何修改、等同替换、改进等，均应包含在本申请的保护范围之内。

权利要求书

1.一种代码处理方法，其特征在于，所述方法包括：

代码开发客户端获取开发得到的第一代码；

所述代码开发客户端响应于代码分析指令，对获取到的所述第一代码进行分析，得到针对所述第一代码的修改意见信息，所述修改意见信息包括描述性文本和待处理的代码段，所述描述性文本指示基于所述代码段对所述第一代码的修改意图，其中，所述修改意图包括从所述第一代码中删除所述代码段、在所述第一代码中增加所述代码段以及用所述代码段对所述第一代码中的代码段进行替换；

所述代码开发客户端基于分析得到的所述修改意见信息和所述第一代码进行分析，得到所述修改意图和所述第一代码中与所述修改意见信息对应的代码修改位置；

所述代码开发客户端基于所述修改意图和所述代码修改位置，在所述第一代码中所述代码修改位置，基于所述代码段执行与所述修改意图对应的处理。

2.根据权利要求1所述的方法，其特征在于，所述代码开发客户端基于分析得到的所述修改意见信息和所述第一代码进行分析，得到所述修改意图和所述第一代码中与所述修改意见信息对应的代码修改位置包括：

所述代码开发客户端响应于针对所述修改意见信息的触发操作，基于所述修改意见信息和所述第一代码进行分析，得到所述修改意图和所述代码修改位置。

3.根据权利要求1所述的方法，其特征在于，所述代码开发客户端基于所述修改意图和所述代码修改位置，在所述第一代码中所述代码修改位置，基于所述代码段执行与所述修改意图对应的处理包括：

所述代码开发客户端基于所述修改意图和所述代码修改位置，在所述第一代码中所述代码修改位置，基于所述代码段执行与所述修改意图对应的处理，得到第二代码；

所述代码开发客户端在预览窗口中显示所述第二代码；

若接收到同意指令，则所述代码开发客户端将所述第一代码替换为所述第二代码。

4.根据权利要求3所述的方法，其特征在于，所述代码开发客户端基于所述修改意图和所述代码修改位置，在所述第一代码中所述代码修改位置，基于所述代码段执行与所述修改意图对应的处理，得到第二代码包括：

若所述修改意图为增加，则所述代码开发客户端在所述第一代码中所述代码修改位置处的代码后，插入所述修改意见信息中的代码段，得到所述第二代码；或，

若所述修改意图为删除，则所述代码开发客户端从所述第一代码中删除所述代码修改位置处的代码，得到所述第二代码；或，

若所述修改意图为修改，则所述代码开发客户端从所述第一代码中删除所述代码修改位置处的代码，在所述代码修改位置处插入所述修改意见信息中的代码段，得到所述第二代码。

5.根据权利要求1至4中任一项所述的方法，其特征在于，所述修改意图和所述代码修改位置通过代码处理模型得到，所述代码处理模型的训练过程包括：

获取第一训练数据集，所述第一训练数据集包括多组第一样本数据和第一标签，所述第一样本数据包括一组第一代码和修改意见信息，所述第一标签包括一组与所述第一样本数据对应的多组修改意图和代码修改位置；

基于所述第一训练数据集，对所述代码处理模型进行训练。

6.根据权利要求1至4中任一项所述的方法，其特征在于，所述修改意图通过意图识别模型得到，所述意图识别模型的训练过程包括：

获取第二训练数据集，所述第二训练数据集包括多组第二样本数据和第二标签，所述第二样本数据包

括一组第一代码和修改意见信息，所述第二标签包括一个与所述第二样本数据对应的修改意图；

基于所述第二训练数据集，对所述意图识别模型进行训练。

7.根据权利要求1至4中任一项所述的方法，其特征在于，所述代码修改位置通过定位模型得到，所述定位模型的训练过程包括：

获取第三训练数据集，所述第三训练数据集包括多组第三样本数据和第三标签，所述第三样本数据包括一组第一代码、修改意见信息和修改意图，所述第三标签包括一个与所述第三样本数据对应的代码修改位置；

基于所述第三训练数据集，对所述定位模型进行训练。

8.一种代码处理装置，其特征在于，所述装置包括：

代码获取模块，用于获取开发得到的第一代码；

修改意见信息获取模块，用于响应于代码分析指令，对获取到的所述第一代码进行分析，得到针对所述第一代码的修改意见信息，所述修改意见信息包括描述性文本和待处理的代码段，所述描述性文本指示基于所述代码段对所述第一代码的修改意图，其中，所述修改意图包括从所述第一代码中删除所述代码段、在所述第一代码中增加所述代码段以及用所述代码段对所述第一代码中的代码段进行替换；

意见识别模块，用于基于分析得到的所述修改意见信息和所述第一代码进行分析，得到所述修改意图和所述第一代码中与所述修改意见信息对应的代码修改位置；

代码处理模块，用于基于所述修改意图和所述代码修改位置，在所述第一代码中所述代码修改位置，基于所述代码段执行与所述修改意图对应的处理。

9.根据权利要求8所述的装置，其特征在于，所述意见识别模块用于：

响应于针对所述修改意见信息的触发操作，基于所述修改意见信息和所述第一代码进行分析，得到所述修改意图和所述代码修改位置。

10.根据权利要求8所述的装置，其特征在于，所述代码处理模块包括：

第二代码获取单元，用于基于所述修改意图和所述代码修改位置，在所述第一代码中所述代码修改位置，基于所述代码段执行与所述修改意图对应的处理，得到第二代码；

显示单元，用于在预览窗口中显示所述第二代码；

替换单元，用于若接收到同意指令，则将所述第一代码替换为所述第二代码。

11.根据权利要求10所述的装置，其特征在于，所述第二代码获取单元用于：

若所述修改意图为增加，则所述代码开发客户端在所述第一代码中所述代码修改位置处的代码后，插入所述修改意见信息中的代码段，得到所述第二代码；或，

若所述修改意图为删除，则所述代码开发客户端从所述第一代码中删除所述代码修改位置处的代码，得到所述第二代码；或，

若所述修改意图为修改，则所述代码开发客户端从所述第一代码中删除所述代码修改位置处的代码，在所述代码修改位置处插入所述修改意见信息中的代码段，得到所述第二代码。

12.一种计算设备集群，其特征在于，包括至少一个计算设备，每个计算设备包括处理器和存储器；

所述至少一个计算设备的处理器用于执行所述至少一个计算设备的存储器中存储的指令，以使得所述计算设备集群执行如权利要求1至权利要求7中任一项所述的方法。

13.一种计算机可读存储介质，其特征在于，包括计算机程序指令，当所述计算机程序指令由计算设备集群执行时，所述计算设备集群执行如权利要求1至权利要求7中任一项所述的方法。

14.一种包含指令的计算机程序产品，其特征在于，当所述指令被计算设备集群运行时，使得所述计算

设备集群执行如权利要求1至权利要求7中任一项所述的方法。