第 3 章

人工智能应用开发全流程

通过第 2 章可以看出，基于简单模板的人工智能应用开发非常简单。但是，为了更灵活地开发人工智能应用，则需要深入开发流程的每个开发环节中。本章将重点详细介绍人工智能应用开发全流程，以及各个子流程内部的核心模块，然后针对目前人工智能应用开发流程的各种复杂性权衡策略及成本模型展开详细讨论。

3.1 人工智能应用开发全流程解析

人工智能应用开发的全流程大致包括开发态流程和运行态流程。开发态流程是对数据源不断地进行处理并得到人工智能应用的过程；而运行态流程相对简单，主要是将人工智能应用部署起来使用的过程。当人工智能应用在运行态推理效果不好时，需要将推理数据返回给开发态进行进一步迭代调优。

在开发态流程中，每个步骤都会基于一定的处理逻辑对输入数据进行处理，并得到输出数据（中间结果或最终结果），同时也可能会产生模型或知识，或其他一些可能的元信息文件（如配置项文件等）。在处理的过程中，可能会接收外部输入（如用户的输入、配置、其他外部环境的输入等）。每个处理步骤的处理逻辑可以是平台内置的处理逻辑，也可以是开发者自定义的处理逻辑（如开发者利用平台的开发调试环境开发的一套代码）。当数据源经过一系列处理之后，会得到最终的结果数据（如图像识别精度等报表数据）。在这一系列的处理步骤中，可能会出现反复，当我们对某个处理步骤输出的数据不满意时，可以重新修正输入数据或者处理逻辑。

上述一系列的处理步骤结束后，中间所产生的一些模型、知识或者配置可以一起编排成一个人工智能应用。这个人工智能应用就是开发态输出的主要成果。紧接着，就进入运行态流程，将人工智能应用部署为云上的一个推理服务实例，或者打包为一个 SDK，业务客户端就可以调用其接口，发送请求并得到推理结果。同时，平台在被用

户授权的情况下，可以对推理数据和结果进行监测，一旦发现问题，可以将推理数据重新接入开发态的数据源，进行下一步迭代开发，并生成新的人工智能应用。由此可见，人工智能应用的开发流程是一个持续迭代并且不断优化的过程，如图 3-1 所示。

从抽象的角度看，图 3-1 所表达的是一个数据流图，该数据流图有几个常用的核心抽象概念。

（1）数据源。数据源指人工智能应用开发过程的主要输入，可以是原始的文件类型的数据（如在第 2 章中用户所需要离线上传的原始图像数据），也可以是来自某个远程服务的数据流（强化学习经常会用到这种数据源形式，具体介绍见第 6 章），还可以是人工输入的信息。数据源的存储方式多种多样，可以是对象存储，也可以是大数据系统，还可以是客户的业务系统等。

（2）处理。处理指人工智能应用开发全流程中的每个具体环节，根据输入数据和处理逻辑得到输出数据。常用的处理操作包括但不限于数据标注、模型训练、性能监测等。每个操作都有执行历史，保证过程可溯源。

（3）实体对象。实体对象指每个处理环节之间流动的数据内容。数据集、算法或规则、模型或知识应用都是典型的实体对象。

以某证件类 OCR 开发全流程为例，上述人工智能应用开发全流程如图 3-2 所示。可以看出，该流程基本满足图 3-1 中的各类抽象。在该 OCR 开发全流程中，需要通过数据采集模块获取原始数据（即证件类的原始图像），考虑到证件类图像中证件位置可能倾斜，因此需要首先对证件的四个顶点进行标注然后再进行数据处理，将图像中证件位置矫正。紧接着，一方面，可以继续标注证件图像中文字框和文字类别，用于文字框检测和文字识别模型的训练；另一方面，可以根据证件四个顶点的标注信息训练四点标注模型。当这三个模型（矫正、文字框检测和文字识别模型）分别训练完成后，可以通过编排生成一个 OCR 应用，并经过评估之后部署起来使用。在运行态如果有推理不好的数据，则需要通过应用维护模块将其返回开发态进行进一步迭代和优化。在该流程中，数据源包括开发态数据源、运行态数据源、人工输入（如算法编写、数据标注信息、训练超参配置、模型评估检查等输入信息）；处理包括数据采集、数据标注、数据处理、算法选择和开发、模型训练、模型评估和调优、应用生成、应用评估、应用发布和部署、应用维护；实体对象包括数据集、算法、模型、最终生成的应用。

整体而言，如果解决方案已经确定，那么如图 3-3 所示，根据处理操作所属范围的不同，可以将人工智能应用的开发流程分为：①数据准备子流程（包含数据采集、数据处理、数据标注等）；②算法选择和开发子流程；③模型训练子流程；④模型评估和调

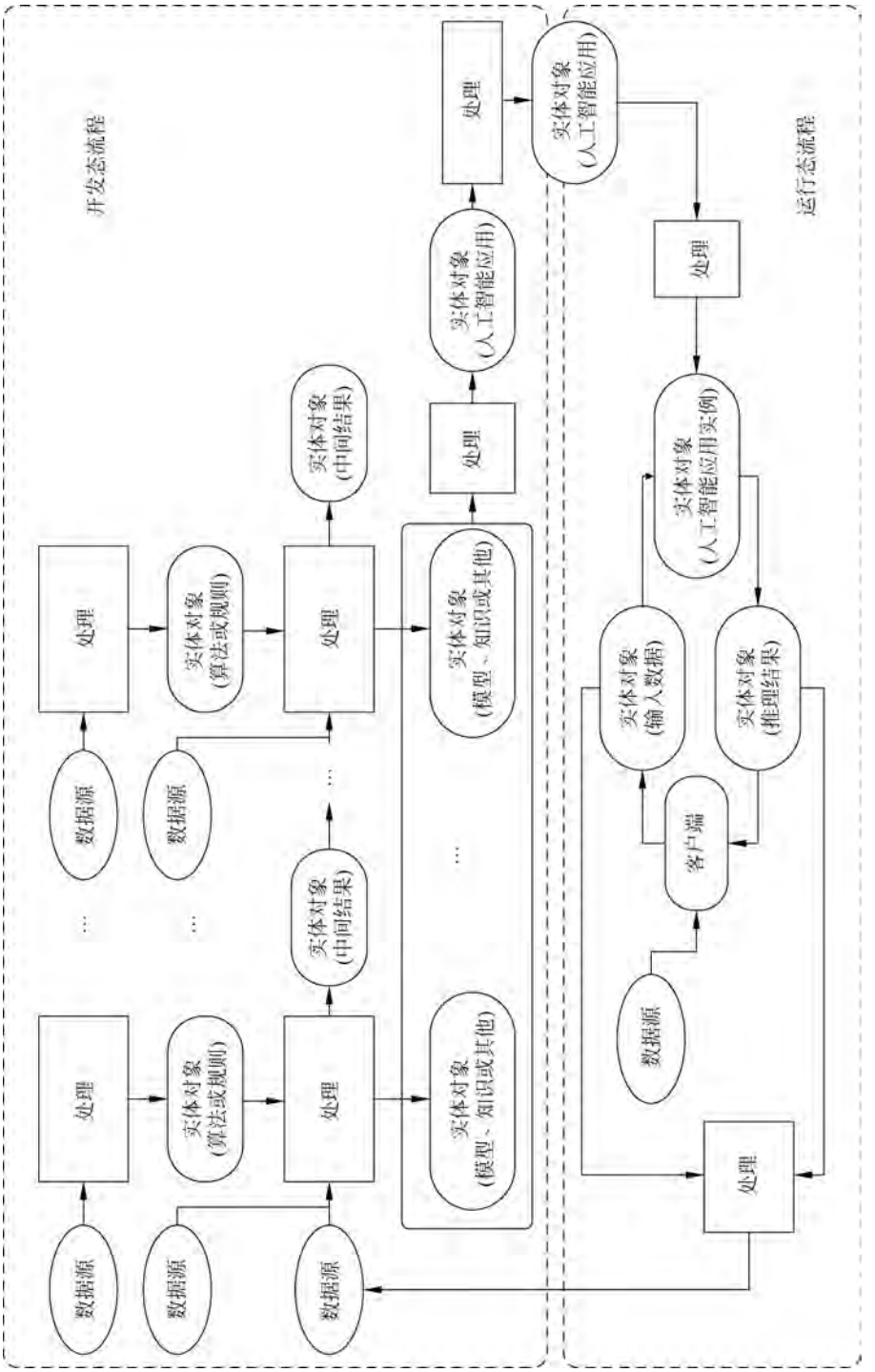

图 3-1　人工智能应用开发全流程

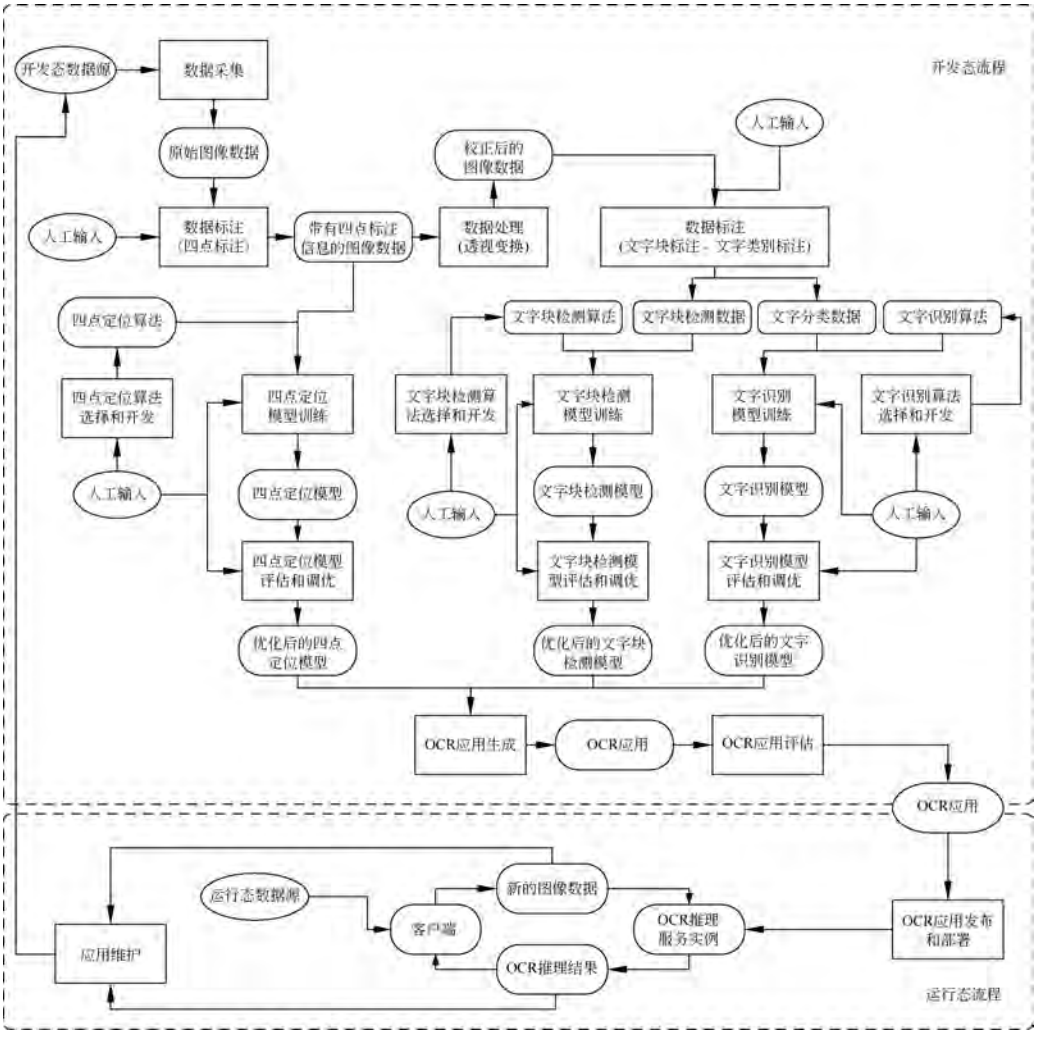

图 3-2　某证件类 OCR 开发的全流程视图

图 3-3　人工智能应用开发全流程所包含的几个主要子流程

优子流程；⑤应用生成、评估和发布子流程；⑥应用维护子流程。由于应用维护子流程会涉及运行态数据回流到开发态，因此这几个子流程之间就形成了一个人工智能闭环。下面将分别介绍这几个子流程。

3.1.1　数据准备子流程

数据准备子流程如图 3-4 所示。在开发人工智能应用之前，应该指定一个数据源，可以通过离线上传或者在线流式读取的方式将数据采集并接入人工智能应用开发平台。

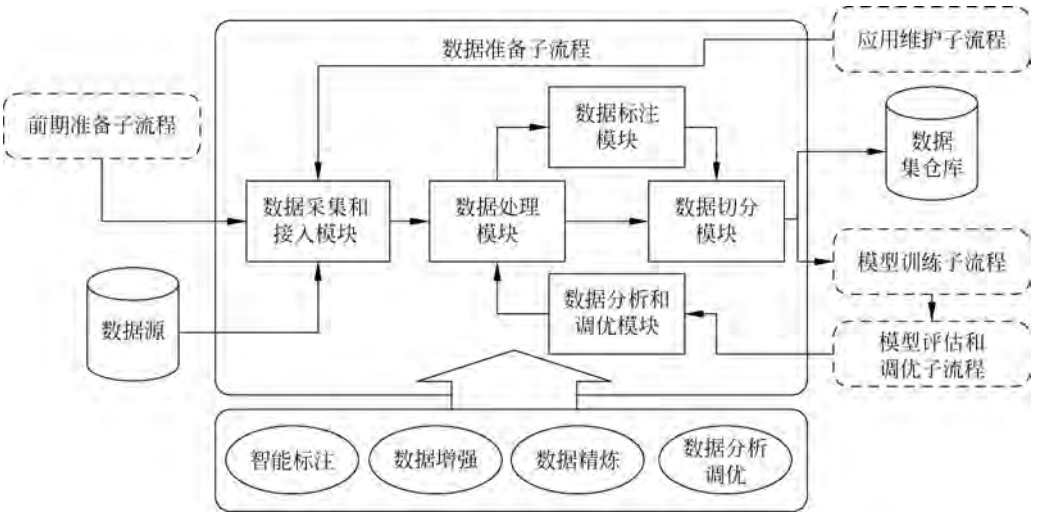

图 3-4　数据准备子流程

将采集的数据存储之后，紧接着就要进行数据处理。大多数情况下，原始数据都是非常杂乱的，有些是结构化数据，有些是非结构化数据（如图像、语音、文本等）。在模型训练前，需要对数据进行校验、清洗、选择、增强等处理。如果数据处理充分，则在模型训练时可以减少很多麻烦。通过一些数据增强、数据精炼等方法，还可以进一步提升模型训练效果。另外，数据不仅在模型训练中需要，在模型评估、应用评估等阶段也是需要的，所以通常需要将数据进行切分处理，以满足不同阶段的需求。由于数据通常都是比较敏感的，因此在数据处理中还需要考虑隐私保护等问题。

由于目前常用的很多人工智能算法都会用到机器学习算法，而目前大多数机器学习算法又强依赖于数据的标签，因此数据标注是一个很重要的环节。由于手工标注非常耗时，且成本高昂，因此人工智能应用开发平台通常具备智能标注能力，以减少手工标注工作量。

如果要深入发现数据问题，则需要进行数据分析和调优，这需要一定的业务领域经验和分析技巧。当数据量较小时，可以人工对每一个数据进行查看和分析，但当数据量较大或者数据本身分析难度高时，往往需要借助统计工具对原始数据进行分析。例如，对于结构化数据，可以分析每一列特征的直方图，对于非结构化数据，可以分析其原始数据（如图像像素值）的分布范围，也可以分析其经过结构化信息抽取之后的直方图。另外，对于高维数据，还可以采用 PCA（Principal Components Analysis，主成分分析）等降维方法，将数据映射到二维或三维空间，便于可视化分析。通过数据分析，可能会发现很多潜在的问题，包括数据的类不均衡现象等，然后就可以针对性做优化。

3.1.2 算法选择和开发子流程

1.1.2 节介绍了几种不同的人工智能技术领域及其所涉及的经典算法。对于每个领域而言，每年都不断有新的算法出现，尤其是深度学习领域，新算法出现的频率更高。如图 3-5 所示，开发者需要根据数据准备子流程中数据的情况、所要完成的任务及其业务场景来综合考虑如何选择最合适的算法。人工智能应用开发平台内置的很多主流算法库可以被直接订阅使用，这大大简化了人工智能应用开发者的工作量。当然，开发者也可以根据具体需求对某个算法进行深入分析，并自行开发。

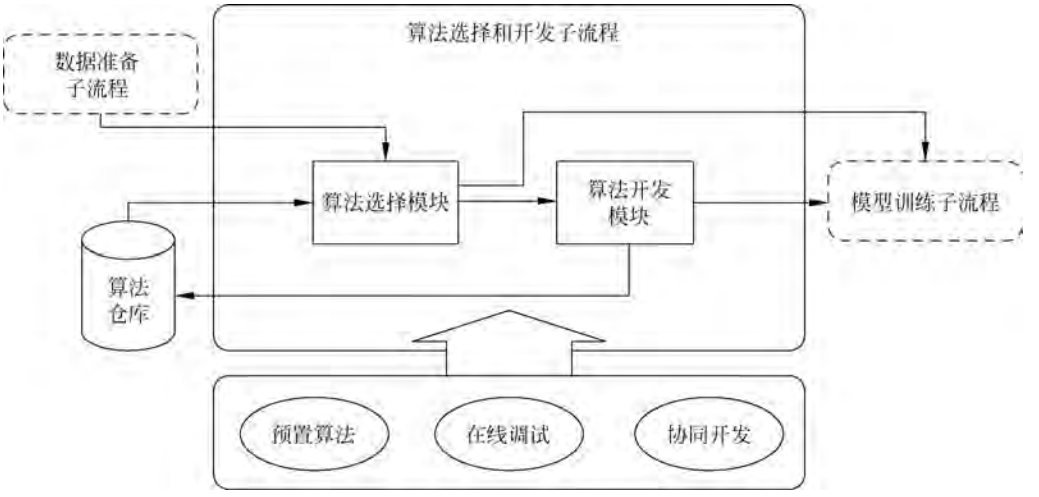

图 3-5 算法选择和开发子流程

人工智能应用开发平台提供常用的算法开发环境，如 Jupyter Notebook、VS Code、PyCharm 等，便于于开发者编写和调试代码。如果开发者本地有 PyCharm 等

环境，可以通过插件将训练作业提交到云上，实现端-云协同开发。算法开发调试后被封装为一个算法对象，可以利用封装好的算法对象和准备好的数据进行训练并得到相应的模型。

3.1.3　模型训练子流程

当前人工智能模型对算力消耗越来越大，其中模型训练是一个很消耗算力的环节，其主要流程如图 3-6 所示。首先需要从数据集仓库读取训练进程内部数据，然后进行数据预处理（如在线数据增强等）。与数据准备子流程里的离线数据处理不同的是，这里的数据预处理模块一般是指在线数据预处理。模型训练模块执行具体的模型迭代计算，最终输出模型进入模型评估和调优子流程。

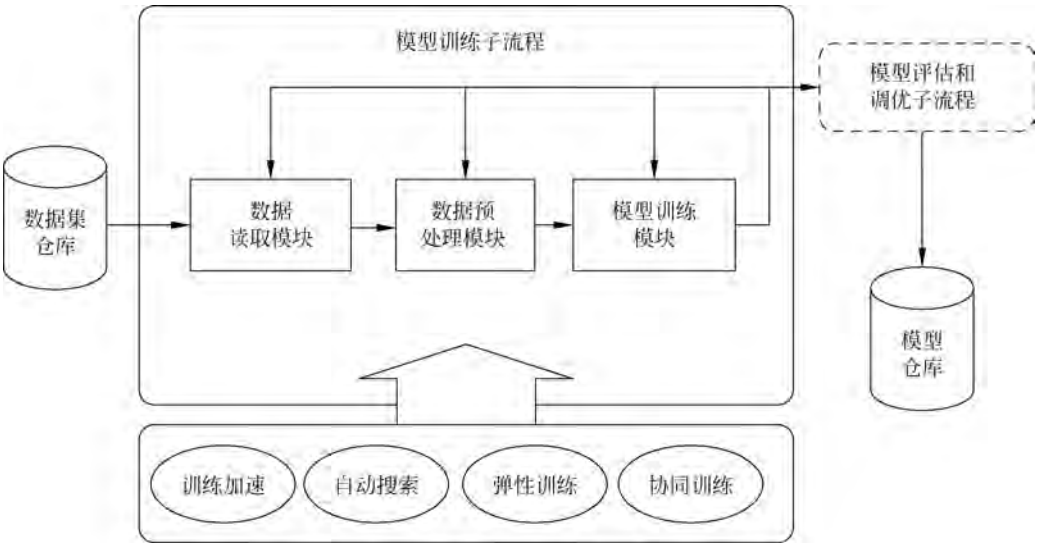

图 3-6　模型训练子流程

为了提升模型性能，模型训练子流程内部的多个模块（数据读取模块、数据预处理模块、模型训练模块）需要被流水线并行起来，并且采用一些其他加速方法，如混合精度、图编译优化、分布式并行加速、调参优化等。除训练加速外，还有一个核心痛点是模型调优。为了减少基于经验的人工调优，可以使用模型评估和调优子流程，根据机器诊断建议进行调优，也可以将该调优流程自动化。另外，在公有云上可以通过弹性训练、协同训练来实现模型训练成本的进一步降低。这些核心能力将在第 6 章中具体介绍。

3.1.4 模型评估和调优子流程

人工智能应用开发全流程的每个处理步骤都可能会产生一个或多个模型、知识或其他内容(如配置项、脚本等)。模型一般分为两种：一种是参数化模型，另一种是非参数化模型。大多数参数化模型的参数是用一定的算法逻辑不断处理输入数据而生成的，因此这些模型更容易受到数据变化的影响。不同的数据集训练得到的参数化模型可能会有很大的不同。因此，模型的评估就显得更为必要。

另外，如前文所述，AI 模型具有多个方面的特点(如性能、精度、鲁棒性等)，因此模型评估比较复杂。开发者需要对每个模型进行评估，才能够知道其是否满足要求。

模型评估需要加载评估数据集，并进行模型预测结果的计算，如图 3-7 所示。

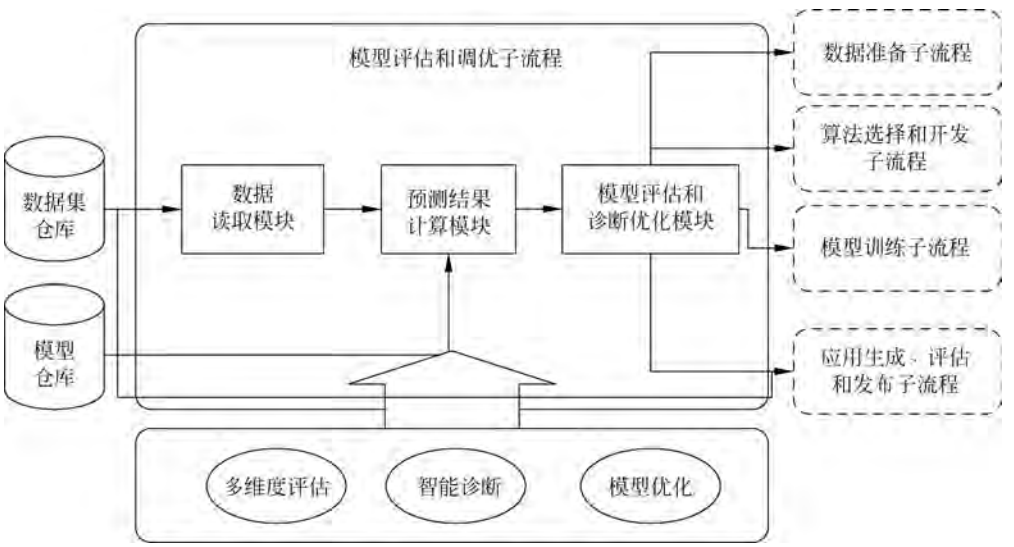

图 3-7 模型评估和调优子流程

模型评估模块针对模型预测结果与真实结果之间的差异，输出一系列不同维度下模型表现的效果，便于开发者分析。当模型的某些指标没有达到期望时，开发者往往需要深入理解并定位其原因。由于算法和模型的复杂性，开发者通常需要具备非常高的技能才可以找到原因。因此，人工智能应用开发平台提供模型诊断功能，针对模型的每个指标，通过一系列工具链自动分析来辅助模型诊断过程。开发者根据平台反馈的诊断建议可进行进一步的模型调优。二次调优会涉及数据准备的调优、算法的重新选择或开发，以及模型的重新训练，因此模型评估和诊断模块后续可能会分别对接这

些子流程。如果模型的全部指标都达到期望，则可以进入应用生成、评估和发布子流程。

3.1.5　应用生成、评估和发布子流程

应用生成、评估和发布子流程如图 3-8 所示。一个复杂的人工智能应用通常包括多个模型及其他配置文件或脚本。当所有的模型都评估并且调优之后，需要对这些模型进行编排和优化，才可以形成一个完整人工智能应用。因此，本子流程中第一个模块就是应用生成模块。

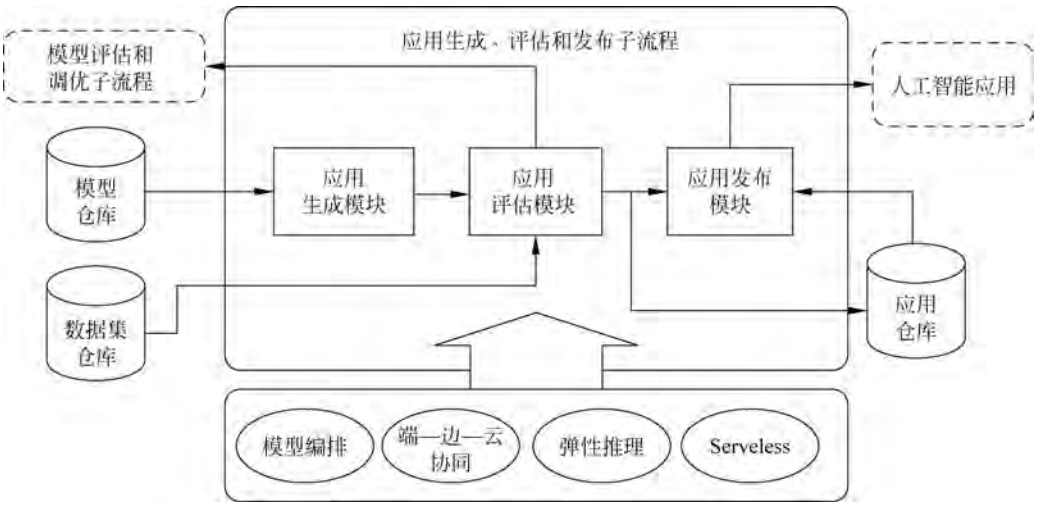

图 3-8　应用生成、评估和发布子流程

同样，类似于模型评估和诊断，也需要对整个人工智能应用进行评估。如果所有指标都满足要求，开发者可以启动人工智能应用的发布；如果有一些指标不满足要求，开发者可以返回模型评估和调优子流程进行二次调优。一个人工智能应用出现的问题可能由其内部的一个或多个模型的问题引起。因此，需要进一步查看哪些模型出现了问题，并做相应的调优。

当人工智能应用评估通过之后，开发者需要将其发布并进行使用。如第 1 章所述，可以选择将其部署为一个在线服务，或者打包为一个 SDK（Software Development Kit，软件开发工具包）直接被其他应用集成。人工智能应用的使用者需要准备一个客户端，可以调用这个部署好的在线推理服务的 API，也可以直接调用 SDK 的推理 API，输入推理数据之后得到推理结果。

目前云化时代需要考虑将应用部署在端、边、云，而且可以互相之间协同推理。弹性推理和 Serverless 是推理服务化的重要能力，具体内容将在第 8 章讲述。

3.1.6 应用维护子流程

正如前面所述，参数化模型一般会与数据分布强相关。然而在实际应用中，运行态数据与开发态数据的分布一般有很大的不同。因此，当某个人工智能应用推理效果不好时，该应用（尤其是带有参数的人工智能应用）需要重新调优。部署后重新调优的过程称为人工智能应用的维护。人工智能应用的维护比传统软件应用的维护更加复杂。

在应用维护子流程（见图 3-9）中，首先需要应用指标监控模块对人工智能应用的表现进行监控，其次在用户授权的情况下通过数据采集模块进行数据采集，并且筛选出推理效果不好的数据，并进入应用迭代模块，进行进一步调优。当需要进一步调优时，则会从数据准备子流程开始，将上述所有子流程都执行一遍。在应用维护子流程中，涉及难例挖掘（难例是指推理效果不好的样本）、模型迁移和自调优，这些能力将在第 9 章讲述。

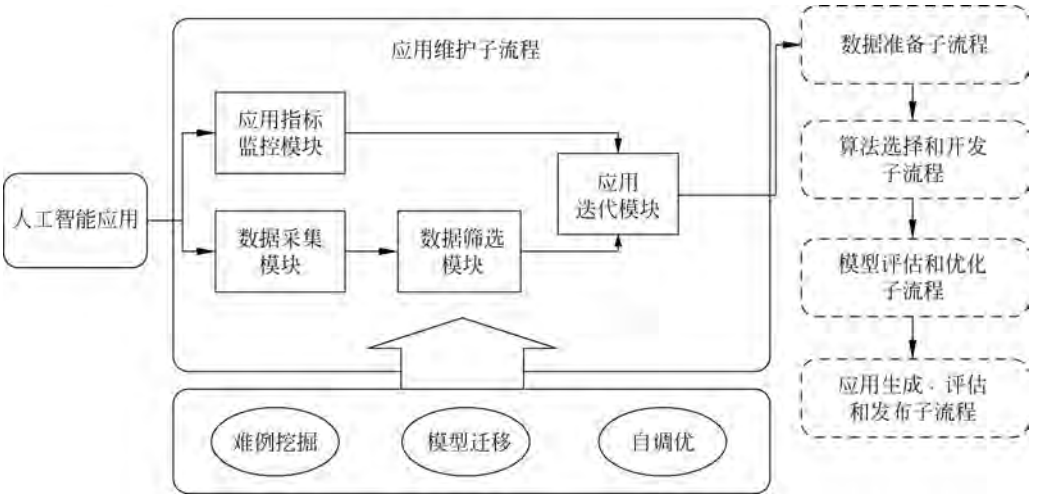

图 3-9 应用维护子流程

总体上，人工智能应用开发的过程就是不断进行数据和模型的处理，并且最终生成满足预期的人工智能应用的过程。当人工智能应用部署后又需要及时维护以保证其能够正常使用。

3.2　人工智能应用开发流程的权衡

从 3.1 节可以看出，人工智能应用开发过程的挑战很多，主要表现在三个方面：①开发流程复杂冗长；②算法技能要求高，需要应用开发者熟悉算法；③应用维护很频繁，可能超过传统软件应用。

因此，考虑到这些挑战，往往就需要在开发过程中做一些权衡。下面将针对这三种挑战，依次分析如何有效利用平台优势和业务具体场景，做出最佳权衡。

3.2.1　复杂和简单的取舍

由于人工智能应用无处不在，可以与各行各业相结合，所以人工智能应用的开发需要足够灵活，能够适应各种行业的需求。但是往往灵活背后的代价就是复杂，尤其对于人工智能应用开发来说，其天然具备较高的复杂度。

在开发人工智能应用之前，同时需要业务经验知识和人工智能经验知识，这样才能设计出合理的方案。对于人工智能应用开发全过程的每个处理步骤而言，输入数据的统计分布、输入数据的覆盖范围、最适合的处理逻辑、输出都是不确定的。这种不确定性会不断传递给后续的处理步骤。随着处理步骤的增多和数据的不断变化，可能需要增加、减少或改变后续的处理步骤，或者改变某个处理步骤中的具体逻辑。因此，人工智能应用开发过程其实是一个不断试错、不断调优、不断迭代的过程，很难一次性开发出一个可以满足要求并直接部署的人工智能应用。这就是人工智能应用开发过程天然具备的复杂性。

正如第 1 章和第 2 章所述，为了降低这种复杂性，通常需要固化一些开发流程模板，可以基于模板来开发自己的人工智能应用，不需要全部的灵活度，但是有时候足以解决当前面临的问题。当然这种模板也可以被用来二次加工、不断迭代和优化。这种基于已有模板的开发方式更加简单，也更容易解决相对受限领域的具体问题。

3.2.2　人与机器的平衡

人工智能应用开发需要利用人工智能算法来处理数据，因此开发人员必须同时具

备软件工程和人工智能方面的知识和技能，开发门槛相对较高。虽然基于工作流模板的开发方式可以大幅降低人工智能应用开发门槛，但是开发者（工作流的使用者）仍然需要按照工作流的每个处理步骤不停地迭代。如上所述，人工智能应用开发过程其实是一个反复迭代的过程，并且需要较强的人工干预。

大多数情况下，人工干预的程度也跟待解决问题的难度强相关。如果问题没有特别复杂，一般采用一些简单的参数调优即可，和软件工程师对数据库性能参数调优一样。对于一些非常经典和成熟的机器学习算法，算法的架构基本相对稳定，即使是算法工程师也未必会对其进行大幅度的修改，更多的是一些小范围优化。这些超参数包括但不限于算法本身的一些阈值选择或训练策略选择等。因此，大部分开发者为了快速将算法应用到实际问题中，通常基于经验对这些参数进行调节，从而找到更好的算法和模型。但是如果有更强的机器，人工只需定义好规则和搜索空间，就可以利用机器强大的算力来做参数的自动选择和调优。这个调优过程就转变为一个自动化搜索过程。

现在一些传统的人工智能算法都逐渐成熟，大多数可以借助大集群算力和一定的搜索调优算法来完成最优算法的自动选择、优化和训练，具体可以参考第 6 章。

因此，很多人工不断进行调优、迭代的实验过程，逐渐地都可以交给机器来完成，尽量减少开发者的负担，这就是人与机器的平衡。如果要在算力上多投入一些，就可以在人工上少投入一些，反之亦然。开发人工智能应用需要在人和机器方面做一个平衡。人工智能应用开发平台所能够提供的是更多的灵活性和层次性，能够适应不同比例的人力投入和机器投入。

3.2.3 开发和运行的融合

从 3.1 节可以看出，在人工智能应用开发和部署之后，需要及时维护。在维护阶段，用户可以选择应用指标监控模块来实时查看人工智能应用的推理效果。如果推理效果不满足要求，则需要手工或者自动维护，将不合适的数据回流到开发态。然后开发者可以重新查看和理解这些数据，并基于这些数据对已有人工智能应用进行迭代优化。

由于数据的变化会严重影响人工智能应用推理效果的好坏，因此人工智能应用的迭代需要非常及时。这也就使得人工智能应用的开发态和运行态紧密结合，形成一个闭环。对于有些可以自动维护并自动进行迭代优化的场景，这个闭环基本可自动运行，仅需在人工智能应用版本更迭时进行人工审核。

未来，随着人工智能应用的进一步复杂化，包括其内部模型本身的复杂，以及运行态环境的复杂（包括端、边、云），进行人工智能应用开发态和运行态的融合将更为必要，并且这种融合通过人工智能应用开发平台体现出来，可以进一步简化维护人工智能应用的难度。

总体上看，以上三个层面的权衡，其实本质上对人工智能应用开发平台提出了非常高的要求。只有提供足够多的领域模板、足够多的自动化调优能力，以及足够强大的人工智能应用开发态和运行态闭环能力，并在具体业务场景中做出最佳权衡，才能真正提升整体开发效率、降低整体开发成本，给业务方带来最终价值。

3.3　人工智能应用开发全流程的成本分析

人工智能应用开发的成本很大程度上会影响人工智能在各个行业的渗透率。成本越低，则渗透率越高，人工智能对行业的影响速度也越快。然而，人工智能应用开发的总体成本模型非常复杂，但大致包括以下几个层面。

3.3.1　设计和开发成本

如前文所述，如果结合开发流程模板来开发人工智能应用，则相对比较简单。而且，随着机器学习、深度学习等人工智能算法的发展，人工智能应用的使用门槛正在逐步降低，并且结合大算力做最优算法的选择和搜索变得越来越可行，因此可以把更多成本交给机器，进一步降低人工成本。对于不同的人工智能应用，以及相同人工智能应用的不同阶段而言，人工成本和机器成本的比例都是不一样的，这需要人工智能应用开发者按照成本预算自行决策。

然而，人工智能应用开发的最主要难点还在于如何识别业务问题，并将业务问题与最匹配的应用开发流程模板联系起来，即如何进行端到端的设计。这一点是很难靠机器来代替的，目前主要以人工为主。例如，某客户想做一个智能门禁系统，以更好地管理人员的出入，保证安全。对于这样一个问题，人工智能应用开发工程师可以想到多种可能的方案，如指纹识别、人脸识别、虹膜识别等。每种识别方案背后的算法技术

所依赖的软硬件的成熟度、成本，以及算法本身的成熟度都各不一样。这时就需要与业务需求方进行沟通，从成本、研发难度、精度要求、体验等各个维度来综合考虑并选择出一种最佳方案。即便是具体到某一个方案，也有很多细节需要选择。假设客户选择了人脸识别方案，那么人工智能应用开发工程师会想到一系列问题，包括并不限于以下几点：①采用什么类型和型号的摄像头，以及摄像头如何布局和安装？②光照的变化怎么处理？如何处理强光和弱光场景？③所需识别人员有多少？④如果待识别人员名单发生变动如何处理？⑤整个软硬件系统方案是什么？⑥目标识别精度和速度是多少？⑦如果识别不了某些人，怎么处理？⑧如何对待识别人员进行动作约束？例如，需要对准并正视摄像头才可以识别，如果待识别人员不配合，需要如何处理？

这就涉及如何针对业务问题和场景，将客户需求层层分解，并转换为具体应用开发流程模板的选择问题，从而形成一个端到端的解决方案。这个阶段需要反复沟通和设计或实验验证，进而也增加了开发的成本。

从降低人工智能整体设计和开发成本的角度看，人工智能应用开发平台会按照三个阶段不断演进：第一阶段，大部分依赖于人工设计和开发；第二阶段，平台提供大量的应用开发流程模板，开发者仅需要负责业务问题的转换和需求分解，以及基于模板开发时的部分参数选择或调节；第三阶段，开发流程模板会覆盖部分业务问题和需求，更贴近领域具体问题，并且平台会结合更强的优化算法和大集群算力来加速调参。随着人工智能服务单位算力的成本越来越低，以及平台的积累越来越多，人工智能应用的设计和开发成本会逐步降低。

3.3.2 部署和维护成本

在人工智能应用部署方面，部署成本体现在多设备部署方面。未来的人工智能推理一定是端-边-云协同的，因此一次开发和任意部署的能力尤为必要。

如 3.2 节所述，在部署完成后，人工智能应用的维护往往非常重要。人工智能应用本身的脆弱性导致其维护成本非常高。在人工智能应用的运行态，推理数据量可能会很大，返回训练集中做重新训练时，重新标注的成本会很高，并且重新训练的算力成本也比较高。因此，如何自动判断人工智能应用推理表现的恶化，自动对造成这种恶化的关键数据做选择、标注并重训练模型，是大幅度降低维护成本的关键。

从降低人工智能部署和维护成本的角度看，人工智能应用开发平台会按照三个阶

段不断演进：第一阶段，依赖纯人工部署和维护；第二阶段，具备端-边-云多场景化部署能力，并基于自动难例发现算法，采集对应用恶化起关键作用的数据，然后基于这些数据做半自动标注和重新训练，降低应用维护成本；第三阶段，可以采用纯自动方式进行模型部署和自适应更新，仅需在重新部署时引入人工确认。

3.3.3　边际成本

人工智能应用开发的边际成本主要体现在两个方面：一是将人工智能开发流程模板进行跨场景复制时总成本的增量；二是将人工智能应用本身进行跨场景部署和维护时总成本的增量。

对于人工智能开发者而言，如果将已开发好的开发流程模板不断扩大以支持更多的业务场景，当然边际成本就会很低。但是，通常这些模板（尤其是专业模板）跟业务问题有很强的关联，而业务问题和场景差异很大。比如，同样是一个面向图像目标识别的开发流程模板，有的业务场景比较简单，如检测某个固定场景、固定光照条件下单一的、清晰的目标物体，就可以套用一个简单的模板解决；而有的业务场景比较复杂，如远距离视频监控目标物体，远距离造成目标物体不清晰，并且物体较小，如果光照条件变化大，待识别的目标有多个种类并且类别间差异非常小时，算法的复杂度将急剧上升，这时就需要套用一个复杂的模板，或者重新开发一个面向此类场景的模板。因此，现有人工智能开发流程模板必须确定其所能覆盖的业务问题范围及其局限性。任何的人工智能开发流程模板都是有局限性的，只是局限性的大小不同。为了尽可能扩大模板覆盖业务问题的范围，就需要预先对很多场景进行针对性设计和抽象，并且结合算力自动选择适合当前问题的方案。

当人工智能应用开发好之后部署在不同场景时，不同环境造成的推理数据的差异是一个很大的挑战。正如前文所述，人工智能应用需要根据推理数据的变化而不断进行维护。如果维护能够尽可能自动化，那么边际成本就会更低。

从降低人工智能边际成本的角度看，人工智能应用开发平台会按照三个阶段不断演进：第一阶段，依赖已有的人工智能开发流程模板和应用，手工进行跨场景优化和复制；第二阶段，在已有开发流程模板和应用的基础上，增加一定程度的跨场景自适应能力；第三阶段，开发流程模板和应用所能支持的场景更丰富，并自动给用户的新场景提供最优模板变种，自动更新应用。

综上可以看出，当前人工智能应用的设计、开发、部署、维护阶段本身的可复制性

都比较差，这使得边际成本难以降低，也造成了当前人工智能应用可复制性差的问题。因此，人工智能应用开发更需要借助大集群算力、模板库、业务知识库，以及每个模板内依赖的半自动标注、自动算法选择、自动模型训练和优化等人工智能应用开发平台的基础能力，才可以真正降低人工智能应用开发全生命周期的成本，使得人工智能应用更加普及，实现人工智能无处不在。