ICS 35.200

CCS N04

团 体 标 准

T/AII 009—2025

神经网络通用开发环境接口规范

Neural network universal development environment interface specification

（征求意见稿）

2025 - XX - XX发布

2025 - XX - XX实施

深圳市人工智能行业协会 发布

目  次

前言 II

1 范围 1

2 规范性引用文件 1

3 术语和定义 1

4 缩略语 1

5 神经网络开发环境整体设计 2

6 神经网络架构接口模型规范 3

6.1 平台无关算子层 3

6.2 平台无关数据处理层 4

6.3 平台无关模型抽象层 6

6.4 神经网络构建组件及其配置 7

7 神经网络通用训练接口规范 9

7.1 神经网络通用开发框架模型参数保存 9

7.2 神经网络通用开发框架模型结构保存 11

图1　神经网络通用开发环境整体架构 3

图2　平台无关算子层架构 4

图3　数据处理工具整体设计架构图 5

图4　引擎无关模型抽象层整体架构 6

图5　引擎无关模型抽象层类图 7

图6　通用开发接口层 8

图7　兼容多框架的深度学习模型参数实现原理 9

图8　参数解码器实现细节 10

图9　参数解码器实现细节 11

图10　模型连接关系和模型参数 12

图11　兼容多框架的深度学习模型结构保存整体设计 13

前  言

本文件按照GB/T 1.1-2020《标准化工作导则 第1部分：标准化文件的结构和起草规则》的规定起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由深圳市人工智能行业协会（AII）归口。

本文件起草单位：

本文件主要起草人：

神经网络通用开发环境接口规范

1. 范围

本文件确立了面向神经网络通用开发环境接口规范、规定了接口功能及技术要求。

本文件适用于基于神经网络的模型开发和应用过程中所涉及应用编程接口的适配与调用。

1. 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

ISO/IEC 22989:2022 信息技术 人工智能 概念和术语（Information technology - Artificial intelligence - Artificial intelligence concepts and terminology）

1. 术语和定义

ISO/IEC 22989:2022界定的术语和定义适用于本文件。

1. 缩略语

下列缩略语适用于本文件。

ASR：自动语音识别（Automatic Speech Recognition）。

BERT：双向语言表征注意力模型（Bidirectional Encoder Representation from Transformers）。

ByteNet：一种基于CNN的机器翻译模型（Neural Machine Translation based on CNN）。

CNN：卷积神经网络(Convolutional Neural Networks)。

Conformer：一种基于卷积增强的语音识别模型（Convolution-augmented Transformer for Speech Recognition）。

DeepLabv3：一种基于空洞可分离卷积、编解码架构的语义分割模型（Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation）。

EfficientNet：一种高效的卷积网络模型（Efficient Net）。

Faster RCNN：一种基于区域建议网络的实时目标检测模型（Towards Real-Time Object Detection with Region Proposal Networks）。

HRNet：一种高分辨率的人体姿态评估的深度模型（Deep High-Resolution Representation Learning for Human Pose Estimation）。

Mask RCNN：一种图像实例分割模型。

PFLD：实用人脸关键点检测器（A Practical Facial Landmark Detector）。

ResNet：残差网络(Residual Network)。

Resnet3D：基于三维卷积的残差网络(Residual Network with 3D CNN)。

RetinaNet：一种目标检测模型。

RNNT：一种用于语音识别的RNN变换模型（Rnn-Transducer）。

S3D：可分离三维卷积网络（Separable 3D CNN）。

SSD：单步多框检测器（Single Shot MultiBox Detector）。

T5：文本到文本转换注意力模型（Text-To-Text Transfer Transformer）。

TextCNN：一种用于文本分类的卷积网络模型（Convolutional Neural Networks for Sentence Classification）。

Transformer：基于注意力机制的神经网络（Neural Network based on Attention Mechanism）。

TrOCR：基于注意力的光学字符检测（Transformer-based Optical Character Recognition）。

TTS：语音合成（Text-to-Speech）。

UNet：一种用于生物医药领域图像分割的卷积网络（Convolutional Networks for Biomedical Image Segmentation）。

VGGNet：由VGG组提出的图像分类网络模型（Visual Geometry Group Network）。

WaveNet：一种用于生成原始音频波形的神经网络模型（A Generative Model for Raw Audio）。

XLNet：一种自回归预训练语言模型（Generalized Autoregressive Pretraining for Language Understanding）。

YOLO：YOLO单阶段目标检测器（You Only Look Once）。

1. 神经网络开发环境整体设计

神经网络通用开发整体架构的设计可以概括为以下目标：

1. 设计统一的模型抽象层，用于规范化模型构建编程范式，提供模型构建抽象层；
2. 设计统一数据处理层，用于解决各计算框架都有各自的计算流问题；
3. 设计统一低阶算子接口，规范化各框架算子API；
4. 实现各类神经网络构建组件及其配置参数、优化器、代价函数。

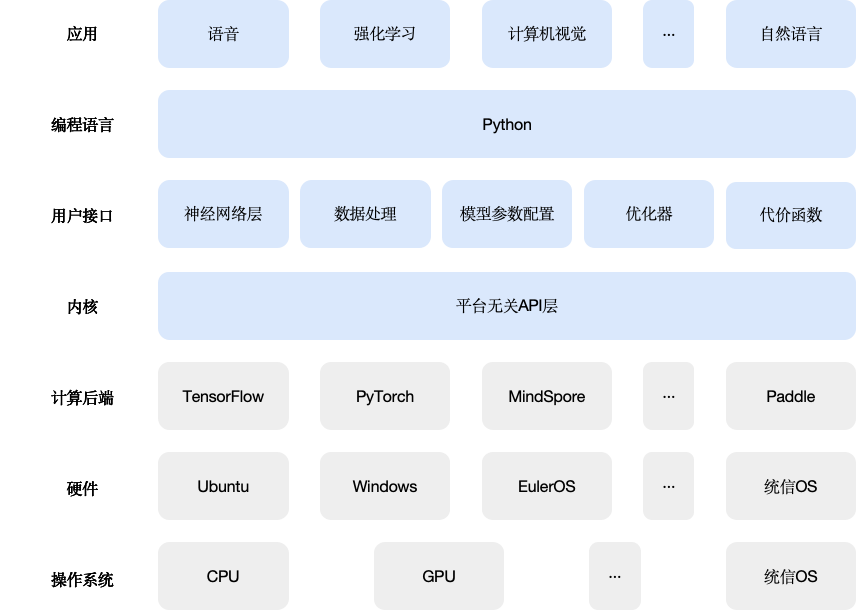


图1　神经网络通用开发环境整体架构

1. 神经网络架构接口模型规范

平台无关算子层

* + 1. 概述

平台无关算子层功能旨在实现一套跨计算引擎的算子层，确保算子的统一性、兼容性和易用性。在神经网络通用开发环境中平台无关算子层应具有图2特性。

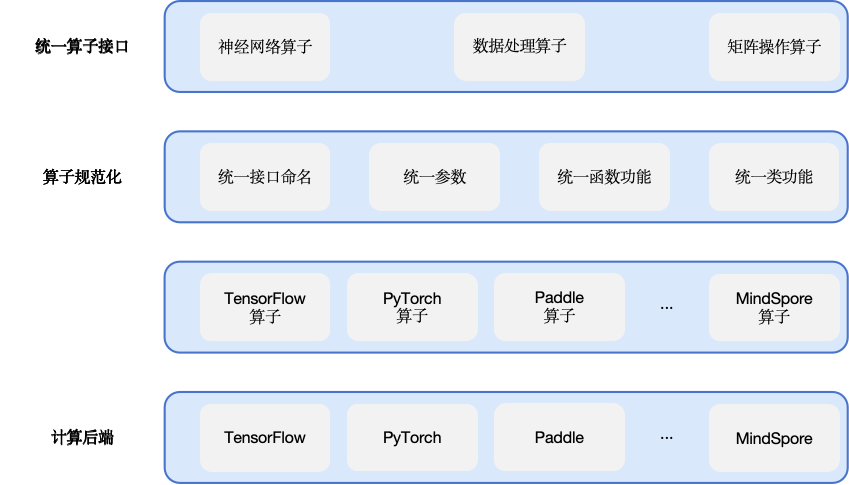


图2　平台无关算子层架构

* + 1. 算子抽象与共性设计

分析不同计算引擎底层算子的实现、调用、参数及功能异同，抽象出各类算子的共性设计特征，并对差异特征进行归纳与转化，形成一套兼容多个计算引擎的算子描述方法。

* + 1. 统一封装与规范化

完成多种类型算子的统一封装，包括统一算子接口命名、统一参数设置以及统一功能表现。通过梳理各引擎底层算子列表，建立映射关系，取共性命名特征，将不同引擎的算子重新抽象封装为统一的命名。

* + 1. 参数与功能一致性

对比不同引擎算子的参数设置，分析参数功能，深入理解由于参数不同带来的计算功能差异，进一步统一输入参数设置与类型，调整算子功能表现，确保引擎无关算子从内而外的统一。

* + 1. 开发者友好性

解决开发者在使用不同计算引擎时因算子命名规则不同、输出参数命名与作用不一致、计算逻辑不一致等问题带来的不便，提升开发效率。

平台无关数据处理层

* + 1. 概述

平台无关数据处理层的核心需求是抽象统一的数据处理方法、引擎无关的算子支持、高效的数据处理性能、开发者友好的接口设计，同时提供数据集制作、载入、采样、格式转换等功能，确保数据处理流程在不同计算引擎间的一致性、可迁移性和高效性。

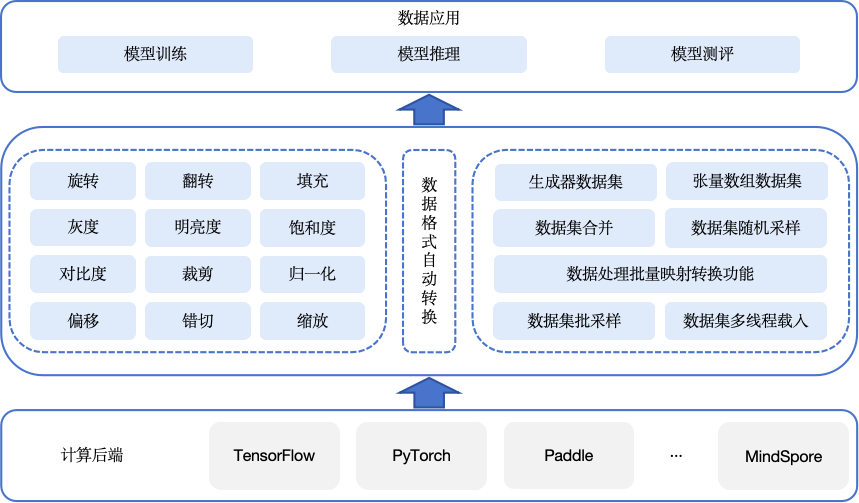


图3　数据处理工具整体设计架构图

* + 1. 数据处理抽象与统一

设计数据处理抽象方法：抽象出通用的数据处理流程，降低多源数据在神经网络训练中的使用难度，减少数据处理在网络训练中的开销。

兼容多种数据格式：兼容不同格式的图像数据（如多维矩阵、数组等），并支持数据流格式转换。

提供数据集制作与载入功能：支持数据集的制作、载入以及数据流格式转换，确保数据在不同计算引擎间的可迁移性。

* + 1. 引擎无关性与算子完备性

梳理不同计算引擎的算子列表：分析各计算引擎支持的数据处理算子，在通用数据处理库中实现各引擎算子的并集，保证算子的完备性。

统一算子功能与参数：由于不同计算引擎中算子参数设置和计算逻辑不同，需要对算子源码进行重构，确保功能一致性。

兼容主流计算引擎：在不修改代码的前提下，实现数据处理流程与各大主流计算引擎的无缝融合。

* + 1. 数据集操作与采样功能

模块化设计：将数据处理工具分为数据集相关算子、采样器、数据集载入等功能模块，便于维护和扩展。

数据集基类：定义数据集基类，提供数据索引与源数据之间的映射关系。

采样器基类：实现数据分批、乱序、权重采样等功能，按照指定规则对数据索引序列进行重排，返回符合开发者定义的新索引序列。

数据集载入基类：利用采样器返回的数据索引与数据集本身，按照批次读取数据，并将数据转化为计算引擎特定数据流传输给计算芯片。

* + 1. 性能优化与高效处理

高效读取与处理源数据：优化数据读取与处理流程，确保数据能够高效地分发给后端计算芯片执行计算。

多线程与多进程支持：实现多线程读取和多进程数据处理功能，提高整体数据传输与处理能力。

降低数据处理开销：通过优化数据处理流程，降低数据处理在神经网络训练中的开销占比。

* + 1. 开发者友好性

简化数据处理流程：开发者只需关注图像处理管道的设计，无需关心图像数据格式本身以及不同框架间的差异。

提供统一、便捷的接口：通过前端API提供统一的调用方式，降低开发者的使用难度。

支持算法迁移：确保数据处理流程在不同计算引擎间的一致性，避免因数据处理差异导致的算法迁移失败。

平台无关模型抽象层

* + 1. 概述

平台无关模型抽象层的核心需求是统一模型定义、屏蔽框架差异、动态管理参数与层、支持跨引擎训练与推理，同时提供模型持久化、细粒度训练控制、开发者友好接口等功能。

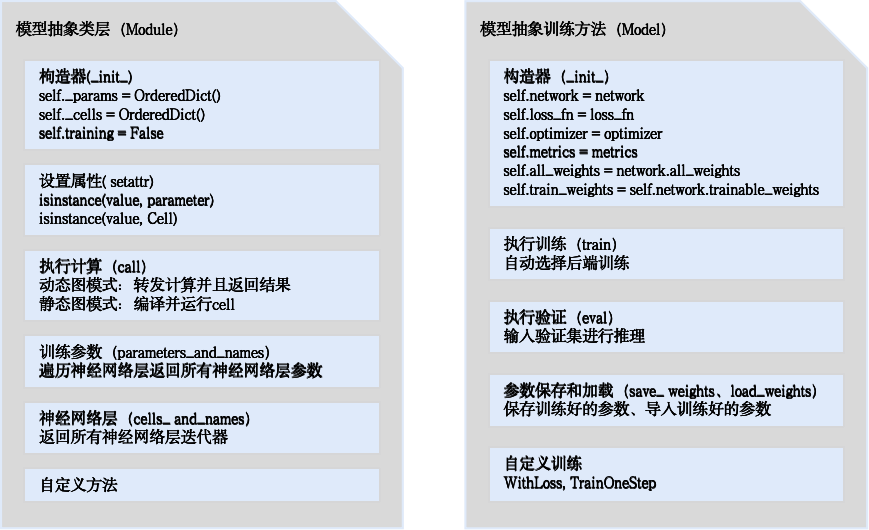


图4　引擎无关模型抽象层整体架构

* + 1. 统一模型抽象与框架差异屏蔽

统一模型基类设计：通过Module基类实现模型抽象，封装各框架模型的核心属性和方法，屏蔽底层差异（如TensorFlow的显式梯度记录、MindSpore的拓扑图构建、PyTorch的前后向分离逻辑）。

兼容多计算引擎：支持以TensorFlow、MindSpore、PyTorch、PaddlePaddle等为后端的模型训练和推理，确保模型定义与执行逻辑的跨框架一致性。

* + 1. 神经网络层与参数管理

神经网络层管理：支持层的动态添加、删除和初始化检测。在构建网络层时自动检测层内参数（如权重、偏置），并将训练参数注册到参数管理模块。

参数管理：支持参数训练状态（是否参与梯度更新）的动态设置与管理。提供参数构建与实例化功能，通过调用后端算子完成参数初始化。

* + 1. 模型构建与执行逻辑

前向传播定义：在forward函数中统一实现模型计算逻辑，开发者只需关注前向过程的定义，无需处理后端框架差异。

模型构建流程：完成训练参数的构建及后端算子的实例化，确保参数与计算逻辑的框架兼容性。实现属性的动态赋值与管理，简化模型定义过程。

* + 1. 训练抽象层设计

统一训练接口：抽象各框架的训练流程差异（如TensorFlow的显式梯度记录、MindSpore的自动构建拓扑图），提供细粒度的训练控制接口。支持在批量数据迭代中输出损失函数、梯度信息，满足高阶开发需求。

训练过程控制：封装前向计算、反向传播、参数更新等核心训练步骤，开发者可通过统一接口控制训练细节。隐藏各框架底层计算图构建差异，简化训练流程的代码实现。

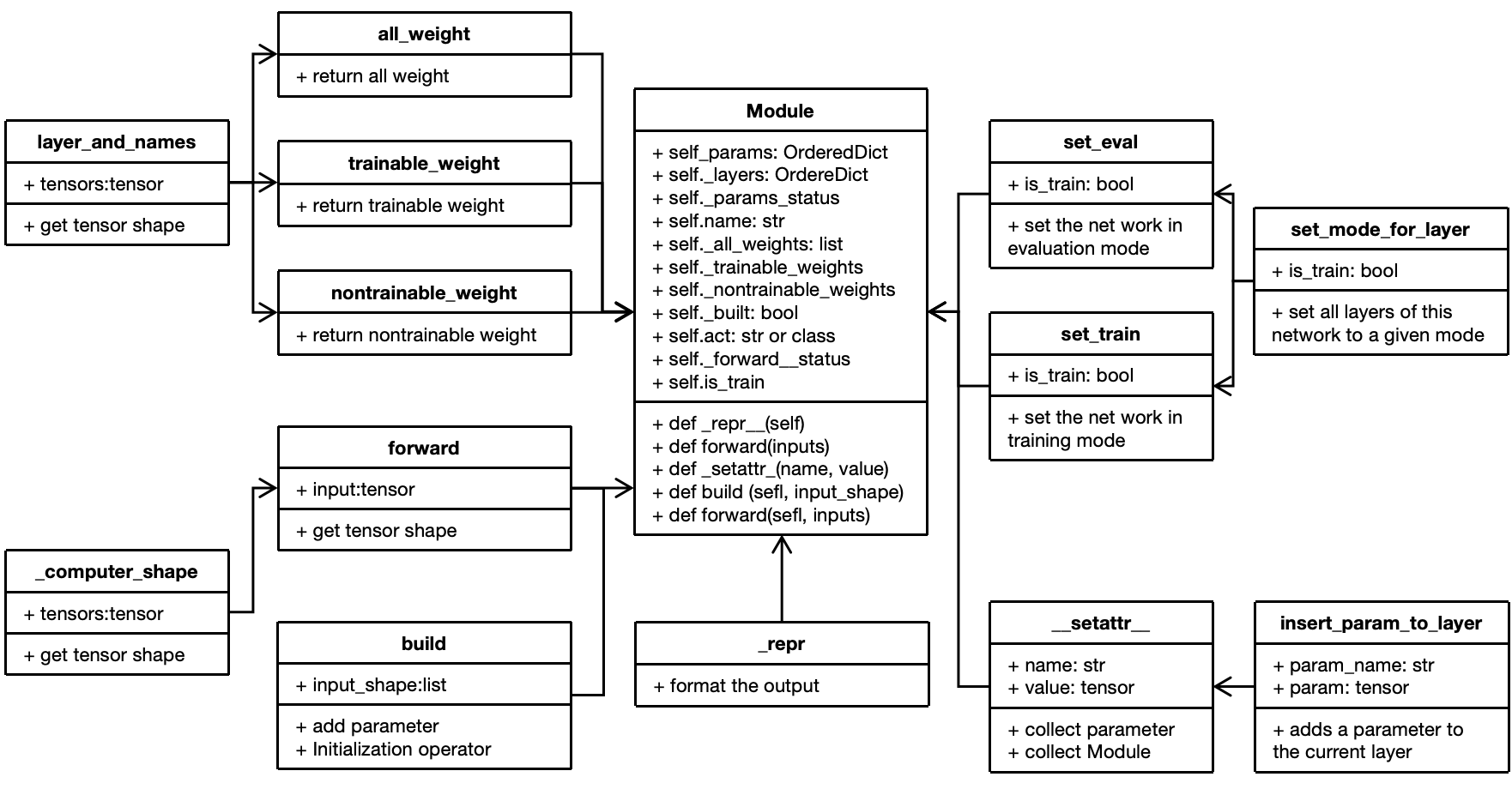


图5　引擎无关模型抽象层类图

神经网络构建组件及其配置

* + 1. 概述

神经网络构建组件的核心需求是提供跨引擎的统一接口、覆盖主流模型的基础组件、支持灵活自定义扩展、优化计算性能，同时实现模型持久化、多领域适配、开发者友好设计，最终达成高效、通用、可扩展的深度学习模型开发体验。

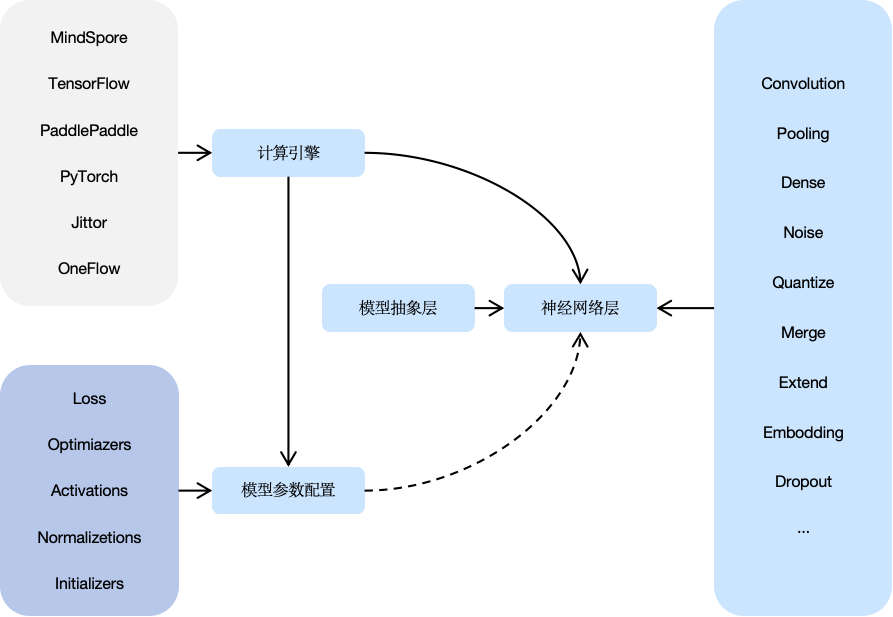


图6　通用开发接口层

* + 1. 基础神经网络组件覆盖

核心层类型支持：提供卷积层、池化层、全连接层、循环神经网络层（RNN/LSTM等）、归一化层（如BatchNorm）、注意力机制层等基础组件，覆盖主流深度学习模型（如CNN、RNN、GAN）的构建需求。

损失函数支持：内置交叉熵、均方差（MSE）等常用代价函数，支持模型训练目标定义。

参数初始化方法：提供正态分布、截断高斯分布等参数初始化策略，确保模型参数初始状态的灵活配置。

激活函数支持：集成ReLU、Tanh、Softmax等常用激活函数，支持非线性变换的多样化选择。

* + 1. 跨引擎的统一接口设计

引擎无关的组件接口：通过分层解耦设计，前端提供统一的神经网络组件接口（如层、损失函数、激活函数），后端调用适配层的算子实现，屏蔽TensorFlow、PyTorch等框架的底层差异。

用户输入与结果反馈：算子层和模型抽象层作为后端，负责收集用户输入并执行计算，最终将结果返回前端接口，确保用户操作与框架无关。

* + 1. 多领域模型支持

支持计算机视觉相关模型定义：

1. 图像分类：ResNet系列、VGG、EfficientNet、Vision Transformer；
2. 目标检测：YOLOv5-v8、SSD、Faster R-CNN、RetinaNet；
3. 图像分割：U-Net、DeepLabv3+、Mask R-CNN；
4. 关键点检测：HRNet、PFLD、OpenPose。

支持自然语言处理相关模型定义：

1. 文本分类：BERT、TextCNN、T5；
2. 机器翻译：Transformer、ByteNet、mBART；
3. 文本生成：GPT系列、BART。

支持多模态与跨领域模型定义：

1. 语音识别：Conformer、RNN-T；
2. 语音合成：WaveNet、Tacotron2；
3. 视频理解：3D ResNet、SlowFast。
4. 神经网络通用训练接口规范

神经网络通用开发框架模型参数保存

* + 1. 概述

不同计算引擎编写的神经网络参数格式不互通，为了保证不同框架之间的兼容性，应该由通用开发框架实现中间格式的转换。具体结构如下图所示：

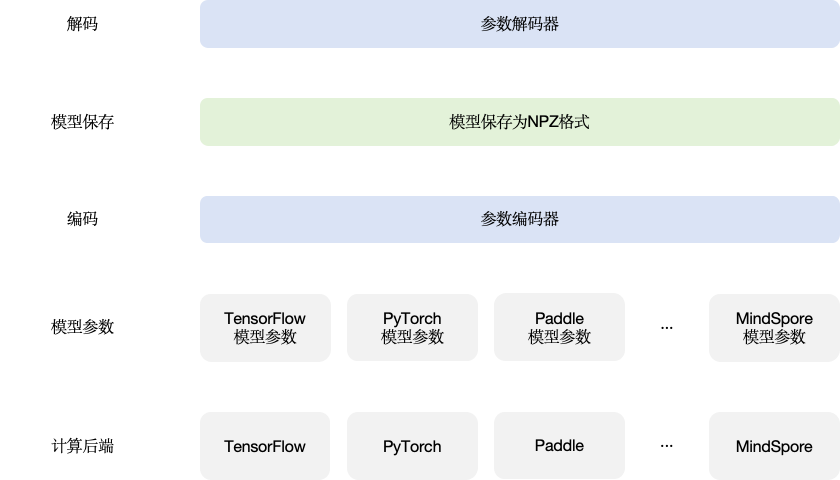


图7　兼容多框架的深度学习模型参数实现原理

* + 1. 参数标准化编码器

统一参数命名规范建立标准参数命名列表（如name/kernel：0表示权重，name/bias:0表示偏置）自动比对不同框架参数名（如PyTorch的name/W和name/b），替换为统一命名格式。

统一参数形状转换定义标准参数形状（如卷积核为滤波器宽×高×输入通道×输出通道）通过Transpose函数转换形状（如将PyTorch的输出通道×输入通道×宽×高转换为标准形状）仅对需要转换的参数（如卷积核）进行形状调整，其他参数保留原状。

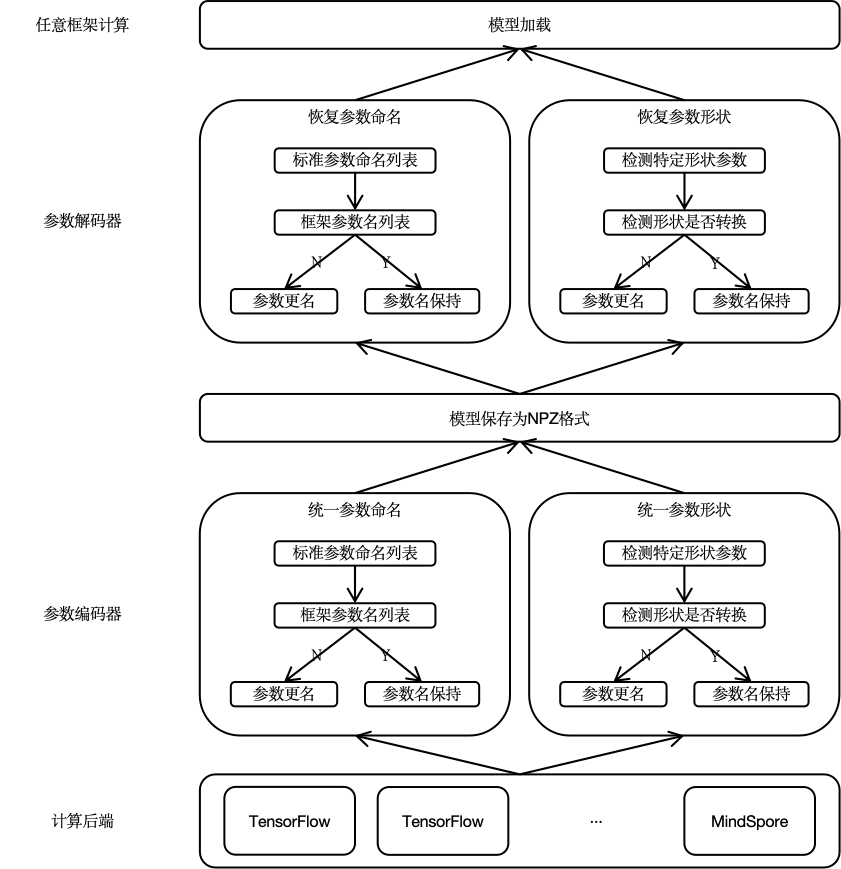


图8　参数解码器实现细节

* + 1. 参数适配解码器

恢复框架特定参数名：根据目标框架的命名规则动态映射参数名（如将标准名kernel:0转换为PyTorch的W）。

恢复框架特定参数形状：逆转换标准形状至目标框架形状（如将标准卷积核形状转回PyTorch的通道优先格式）仅对需逆转换的参数操作，非结构参数（如全连接层偏置）直接加载。

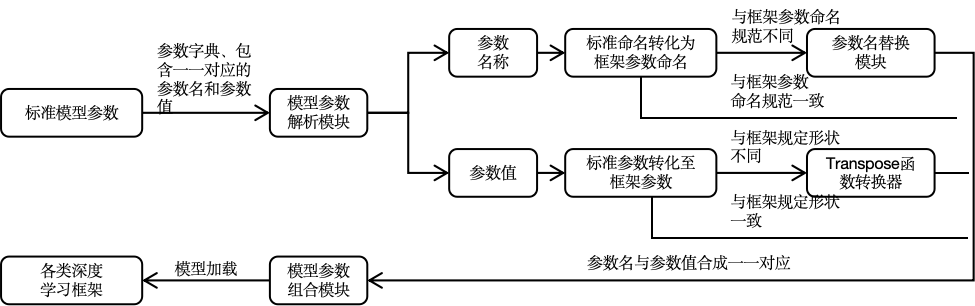


图9　参数解码器实现细节

* + 1. 多框架兼容性设计

支持主流框架参数格式覆盖TensorFlow/Keras、PyTorch、MindSpore、PaddlePaddle等框架的命名与形状差异识别框架特有的参数类型并转换为标准结构。

神经网络通用开发框架模型结构保存

* + 1. 概述

设计突破了框架间模型结构互操作的壁垒，使得不同深度学习框架训练的模型可通过统一表示实现"一次导出，多端部署"。这种结构存储方案与参数编码器/解码器（前文所述）共同构成完整的跨框架模型迁移生态系统。

* + 1. 跨框架模型拓扑抽象

统一层次化组件定义：将各框架神经网络层(Layer)抽象为通用ModuleNode对象封装层类型、输入输出维度、激活函数等核心元数据。

方向性连接追踪：通过前向传播自动记录数据流向动态捕获层间输入输出依赖关系。

* + 1. 结构-参数解耦设计

分离式存储架构：模型结构独立存储于HDF5文件模型参数单独保存为标准.npz文件。

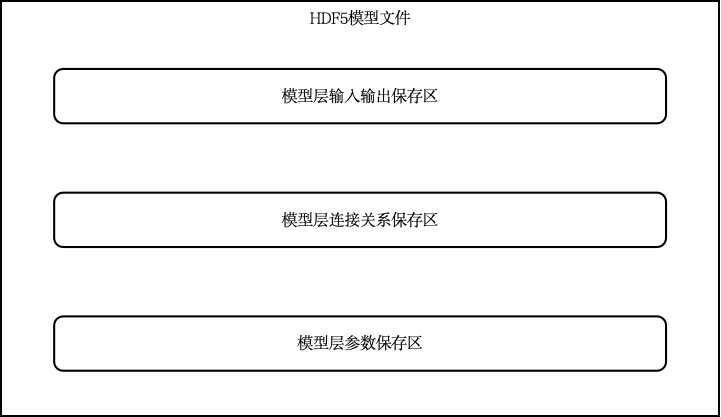


图10　模型连接关系和模型参数

双向关联机制：通过层唯一ID实现结构与参数的精确匹配支持参数热替换（如不同训练阶段的参数加载）。

* + 1. 多后端兼容性支持

统一编程接口层：高层API兼容TensorFlow/PyTorch/MindSpore等后端屏蔽各框架Layer实现的底层差异。

动态适配能力：自动转换框架特有层类型（如TensorFlow的DepthwiseConv2D→通用SeparableConv）支持自定义层注册与类型映射。

标准化存储格式：采用HDF5作为统一容器格式，分层存储模型元数据、连接图、参数数据。

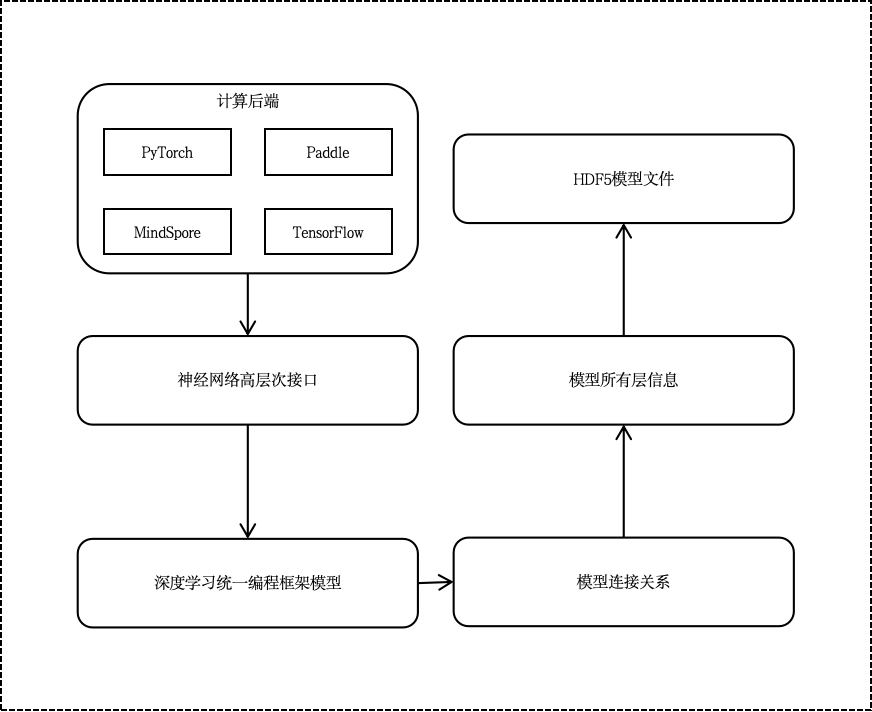


图11　兼容多框架的深度学习模型结构保存整体设计

* + 1. 多阶段应用支持

训练/推理一体化：保存优化器状态、学习率策略等训练上下文，支持从检查点恢复完整训练状态。

跨框架推理部署：导出ONNX等中间格式的转换基础，实现HDF5+NPZ→目标框架模型的端到端转换。

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_