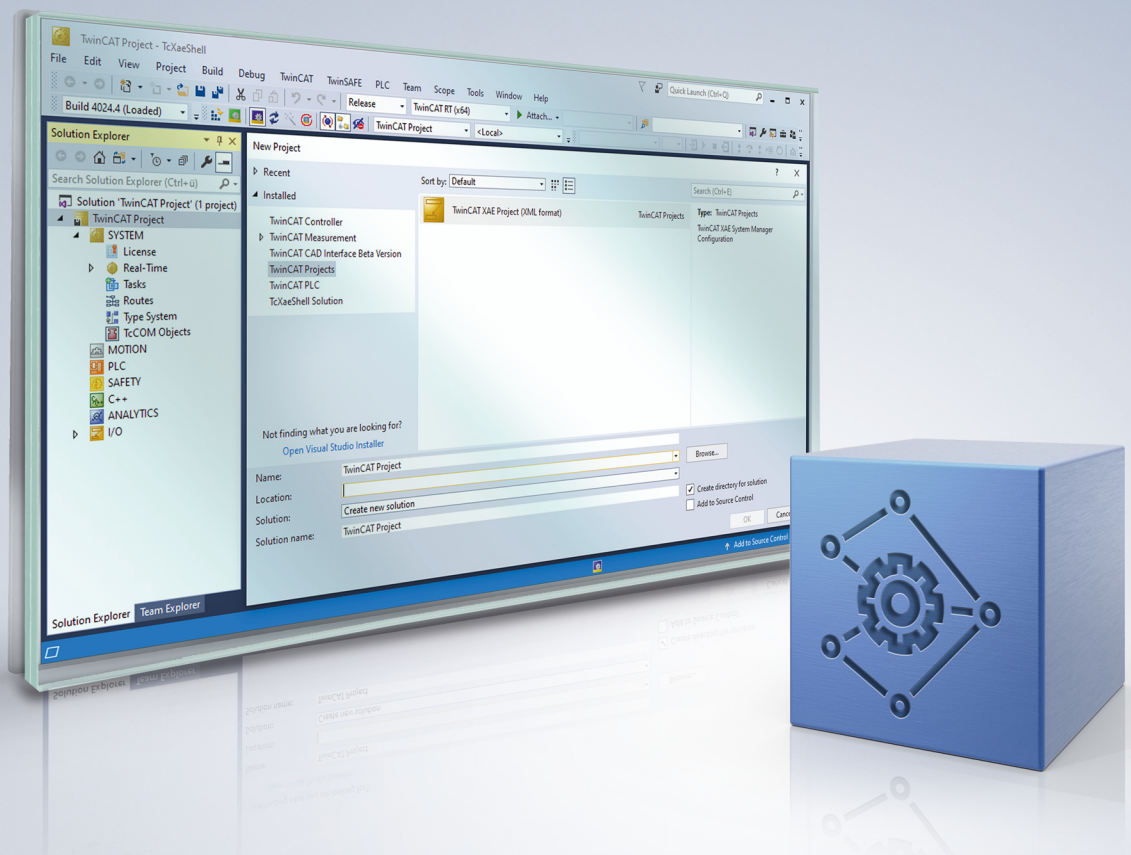
BECKHOFF New Automation Technology

手册 | ZH

TF3820

TwinCAT 3 | Machine Learning Server



2025-03-11 | 版本： 1.0.0

目录

1 前言

1.1 文档说明

本说明仅适用于熟悉国家标准且经过培训的控制和自动化工程专家。
在安装和调试组件时，必须遵循文档和以下说明及解释。
操作人员应具备相关资质，并始终使用最新的生效文档。

相关负责人员必须确保所述产品的应用或使用符合所有安全要求，包括所有相关法律、法规、准则和标准。

免责声明

本文档经过精心准备。然而，所述产品正在不断开发中。
我们保留随时修改和更改本文档的权利，恕不另行通知。
不得依据本文档中的数据、图表和说明对已供货产品的修改提出赔偿。

商标

Beckhoff®、TwinCAT®、TwinCAT/BSD®、TC/BSD®、EtherCAT®、EtherCAT G®、EtherCAT G10®、EtherCAT P®、Safety over EtherCAT®、TwinSAFE®、XFC®、XTS® 和 XPlanar® 是德国倍福自动化有限公司的注册商标并由其授权使用。
本出版物中所使用的其它名称可能是商标名称，任何第三方出于其自身目的使用它们可能会侵犯商标所有者的权利。

EtherCAT® 是注册商标和专利技术，由德国倍福自动化有限公司授权使用

版权所有

第三方商标

本文档可能使用了第三方商标。有关商标信息，可以访问：https://www.beckhoff.com/trademarks。

1.2 安全信息

安全规范

为了确保您的使用安全，请务必仔细阅读
并遵守本文档中每个产品的安全使用说明。

责任免除

所有组件在供货时都配有适合应用的特定硬件和软件配置。严禁未按文档所述修改硬件或软件配置，否则，德国倍福自动化有限公司对由此产生的后果不承担责任。

人员资格

本说明仅供熟悉适用国家标准的控制、自动化和驱动工程专家使用。

警示性词语

文档中使用的警示信号词分类如下。为避免人身伤害和财产损失，请阅读并遵守安全和警告注意事项。

人身伤害警告

⚠ 危险

存在死亡或重伤的高度风险。

⚠ 警告

存在死亡或重伤的中度风险。

⚠ 谨慎

存在可能导致中度或轻度伤害的低度风险。

财产或环境损害警告

注意

可能会损坏环境、设备或数据。

操作产品的信息

这些信息包括：
有关产品的操作、帮助或进一步信息的建议。

1.3 信息安全说明

Beckhoff Automation GmbH & Co.KG (简称 Beckhoff) 的产品，只要可以在线访问，都配备了安全功能，支持工厂、系统、机器和网络的安全运行。尽管配备了安全功能，但为了保护相应的工厂、系统、机器和网络免受网络威胁，必须建立、实施和不断更新整个操作安全概念。Beckhoff 所销售的产品只是整个安全概念的一部分。客户有责任防止第三方未经授权访问其设备、系统、机器和网络。它们只有在采取了适当的保护措施的情况下，方可与公司网络或互联网连接。

此外，还应遵守 Beckhoff 关于采取适当保护措施的建议。关于信息安全和工业安全的更多信息，请访问本公司网站 https://www.beckhoff.com/secguide。

Beckhoff 的产品和解决方案持续进行改进。这也适用于安全功能。鉴于持续进行改进，Beckhoff 明确建议始终保持产品的最新状态，并在产品更新可用后马上进行安装。使用过时的或不支持的产品版本可能会增加网络威胁的风险。

如需了解 Beckhoff 产品信息安全的信息，请订阅 https://www.beckhoff.com/secinfo 上的 RSS 源。

1.4 文档发行状态

版本	变更
1.0.0	首次发布

2 概述

简介

TwinCAT Machine Learning Server 可直接在控制 IPC 或边缘设备上执行 AI 模型。

TwinCAT Machine Learning Server 由两个组件组成：

- PLC 功能块作为该 Machine Learning Server 的客户端。
- Machine Learning Server 作为服务（AI 模型的加载、执行等）的提供者。

这些组件为 PLC 程序提供异步执行功能。异步计算的概念有效地将 AI 模型的执行时间与 PLC 的循环操作解耦。Machine Learning Server 支持在 CPU 和 NVIDIA® GPU 上执行任何复杂的 AI 模型，因此特别适合与 C6043 工业 PC 配合使用。Machine Learning Server 在操作系统的用户模式下运行。这会导致非确定性行为，只能通过对用户模式组件进行相应配置来部分缓解这种情况。

TwinCAT Machine Learning Server 可加载以 ONNX 文件形式提供的 AI 模型。所有相关的 AI 框架，如 TensorFlow、Pytorch、Scikit Learn 等，都支持这一互操作性标准。这将训练环境与执行环境相互独立。任何训练环境都可用于创建 AI 模型，然后使用 TwinCAT Machine Learning Server 执行这些模型。

目标群体与用例

Machine Learning Server 适用于以下用例，但不限于：

- 在处理计算复杂的 AI 模型时， GPU 加速预期节省的计算时间足以弥补因计算时间波动（抖动）带来的时间。
  - 特别是用于图像分类、目标检测或图像分割的视觉 AI 模型。
- 在低优先级任务中使用 AI 模型，这些任务与确定性 PLC 程序之间仅有松散耦合。
  - AI模型的结果不被控制系统使用，但会传达给控制层以上系统。例如，告知设备操作员的过程分析模型、告知维保人员的预测性维护模型等。
  - 控制器在特定时间点不需要其结果的 AI 模型。例如，提供优化或自适应工艺参数的AI 模型。

与同类 TwinCAT 产品的区别和比较

除了 TwinCAT Machine Learning Server之外，还有其他具有类似功能的 TwinCAT 产品，即推理 AI 模型。

- TF3800 TwinCAT Machine Learning Inference Engine
- TF3810 TwinCAT Neural Network Inference Engine
- TF7800 TwinCAT Vision Machine Learning
- TF7810 TwinCAT Vision Neural Network

下表列出了所列产品与 TwinCAT Machine Learning Server 之间的主要区别。

表 1: 产品性能对比

实时性 AI：TF3800、TF3810、TF7810	加速 AI：TF3820
TwinCAT 进程中的实时性 AI 执行	独立进程中的近实时执行
在标准 x64 CPU 上执行	支持在 NVIDIA® GPU 上进行硬件加速
支持选定的 AI 模型和算法操作	支持当前 ONNX Opset 版本，兼容当前各种 AI 模型
标准 PLC 功能块，便于在 TwinCAT 中集成	标准 PLC 功能块，便于在 TwinCAT 中集成
支持 ONNX，可实现互操作性	支持 ONNX，可实现互操作性
授权组合：TF3810 包括 TF3800、TF7800 和 TF7810	也可在有多个客户端的网络中用作服务器

3 安装

要使用 TwinCAT Machine Learning Server，您需要三个组件：

- **TF3820 | TwinCAT Machine Learning Server**
  直接在控制 IPC 或边缘 IPC 上安装并授权此功能模块。
  - 在系统上安装 TcMlServer [▶ 30] 服务。
- **TF3830 | TwinCAT Machine Learning Server Client**
  在工程 PC 上安装此功能模块，以使用所需的 PLC 库。
  - 为 TwinCAT 3 XAE 安装 PLC 库 Tc3_MlServer [▶ 32]。
- **TF38xx | TwinCAT Machine Learning Model Manager**
  在工程 PC 上安装此功能模块，以准备在 TwinCAT 中使用ONNX 文件 [▶ 18]。

系统要求：TwinCAT Machine Learning Server (TF3820)

技术数据	要求
操作系统	Windows10
目标平台	x64
最低 TwinCAT 版本	TwinCAT 3.1 Build 4026
需要的 TwinCAT 设置级别	TwinCAT 3 XAR
所需 TwinCAT 授权	TC1000

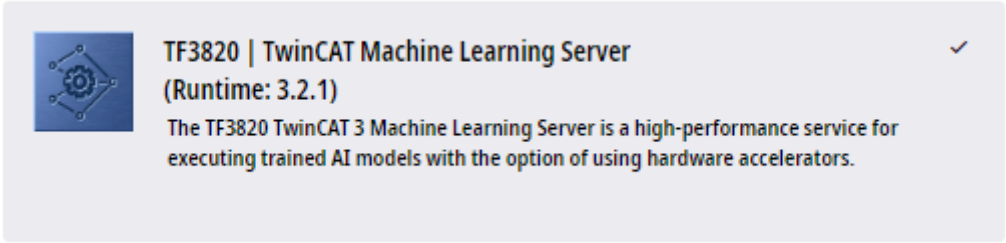
系统要求：TwinCAT Machine Learning Server Client (TF3830)

技术数据	要求
操作系统	Windows10
目标平台	x64
最低 TwinCAT 版本	TwinCAT 3.1 Build 4026
需要的 TwinCAT 设置级别	TwinCAT 3 XAE
所需 TwinCAT 授权	TC1200

TwinCAT Package Manager：安装 (TwinCAT 3.1 Build 4026)

有关安装产品的详细说明，请参阅 TwinCAT 3.1 Build 4026 安装说明中的“安装工作负载”章节。

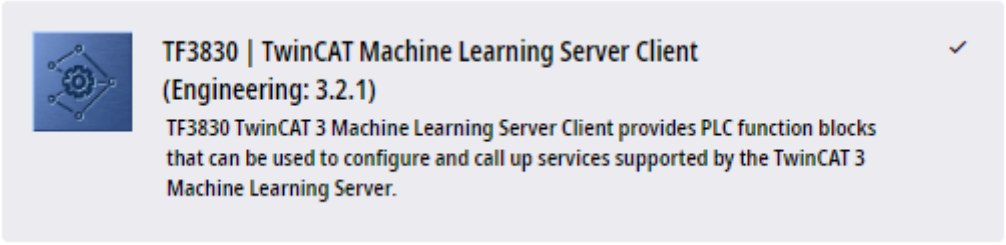
安装以下工作负载才能使用产品：



TwinCAT Package Manager UI: TF3820 | TwinCAT 3 Machine Learning Server

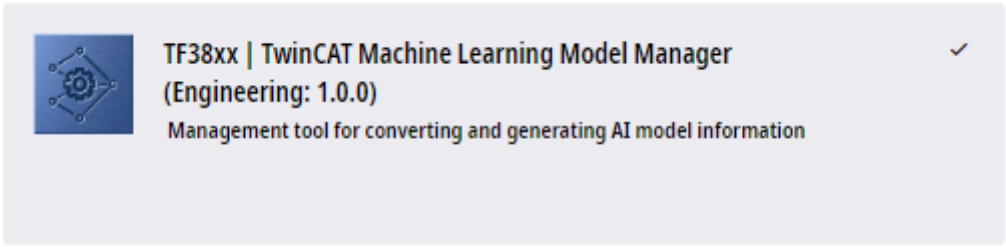
TwinCAT Package Manager CLI:

```
tcpkg install TF3820.MachineLearningServer.XAR
```



TwinCAT Package Manager UI: TF3830 | TwinCAT 3 Machine Learning Server Client

TwinCAT Package Manager CLI:
```
tcpkg install TF3830.MachineLearningServerClient.XAE
```



TwinCAT Package Manager UI: TF38xx | TwinCAT 3 Machine Learning Model Manager

TwinCAT Package Manager CLI:
```
tcpkg install TF38xx.MachineLearningModelManager.XAE
```

在装有 TF3800 或 TF3810 3.2.6 及更高版本的系统上安装

与旧版本的潜在冲突

在安装 TwinCAT Machine Learning Model Manager 的软件包时，可能会出现冲突。卸载旧版本可以快速解决此冲突。

如果安装了 3.2.6 或更低版本的 TF3800.MachineLearningInferenceEngine.XAE 或 TF3810.NeuralNetworkInferenceEngine.XAE 功能模块，则应首先卸载这些功能模块：

```
tcpkg uninstall TF3800.MachineLearningInferenceEngine.XAE --include-dependencies
tcpkg uninstall TF3810.NeuralNetworkInferenceEngine.XAE --include-dependencies
```

然后，才能安装 3.2.10 或更高版本的功能模块。

3.1 授权

TF3820 TwinCAT Machine Learning Server 授权由服务器组件（TcMLServer 服务）请求。授权必须存储在执行服务器进程的 IPC 上。TF3820 授权包括 TF3830 授权，因此无需额外授权即可实现功能块与服务器之间的本地通信。

TF3830 TwinCAT Machine Learning Server Client 授权由 PLC 库 Tc3_MlServer 的功能块查询。需要在远程访问 TwinCAT Machine Learning Server 的运行时系统上配置该授权。功能块与服务器之间的本地通信无需单独 TF3830 授权。

TwinCAT 3 功能组件激活分为完整版和 7 天试用版。两种授权类型均可通过 TwinCAT 3 开发环境 (XAE) 激活。

TwinCAT Machine Learning Server 的循环授权查询

如果 TwinCAT Machine Learning Server 没有授权或只有 7 天试用授权，则每隔 10 秒钟循环执行一次授权查询。

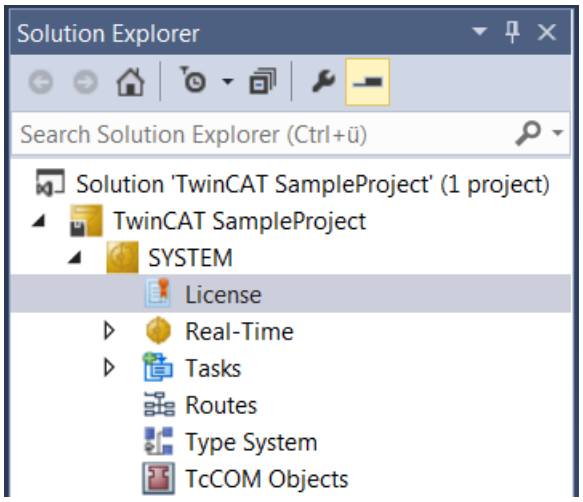
授权使用 TwinCAT 3 功能的完整版本

关于许可证完整版本的程序描述，可参见倍福信息系统的文档“TwinCAT 3 授权”。

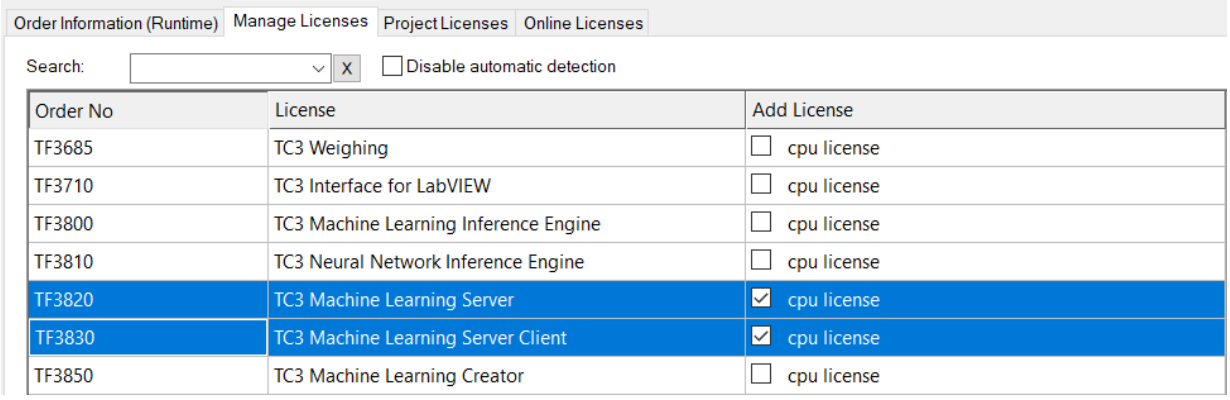
授权使用 TwinCAT 3 功能组件的 7 天试用版

无法为 TwinCAT 3 授权密钥启用 7 天试用版。

1. 启动 TwinCAT 3 开发环境 (XAE)。
2. 打开一个现有的 TwinCAT 3 项目或创建一个新项目。
3. 如果要激活远程设备的授权，请设置所需的目标系统。为此，请从工具栏的 **Choose Target System（选择目标系统）** 下拉列表中选择目标系统。
   ⇨ 授权设置总是指向选定的目标系统。在目标系统上激活项目时，自动将相应的 TwinCAT 3 授权复制到该系统中。
4. 在 **Solution Explorer（解决方案资源管理器）** 中，双击 **SYSTEM（系统）子目录**中的 **License（授权）**。

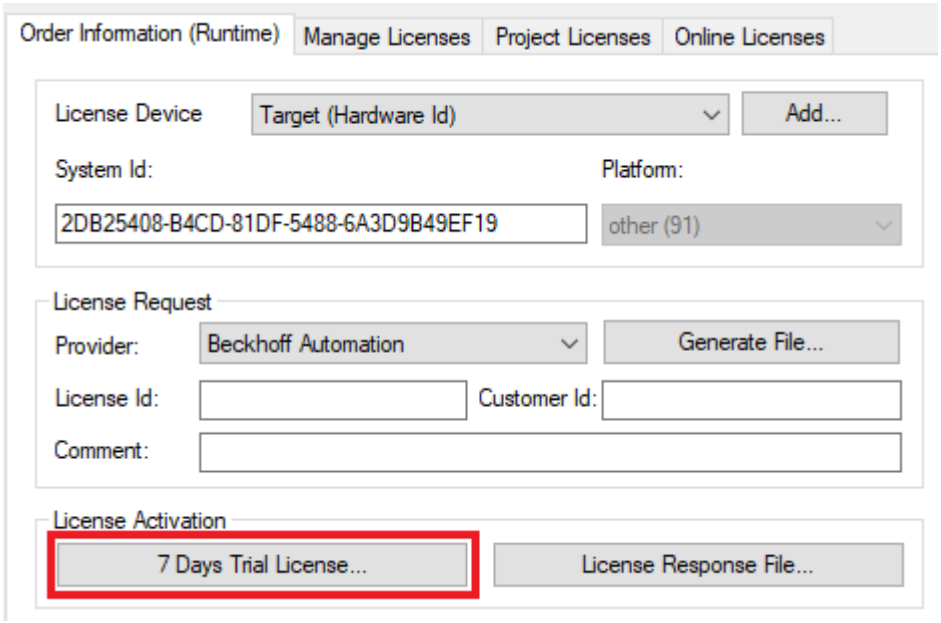


   ⇨ 打开TwinCAT 3 授权管理器。
5. 打开 **Manage Licenses（管理授权）** 选项卡。
6. 在 **Add License（添加授权）** 列中，激活要添加到项目中的授权的复选框（本例中为 "TF3820 TC3 Machine Learning Server" 和/或 "TF3830 TC3 Machine Learning Server Client"）。
   请注意，TF3820 包含本地系统的 TF3830授权。如果本地系统有 TF3820，则客户端不需要 TF3830。只有远程客户端才需要 TF3830。

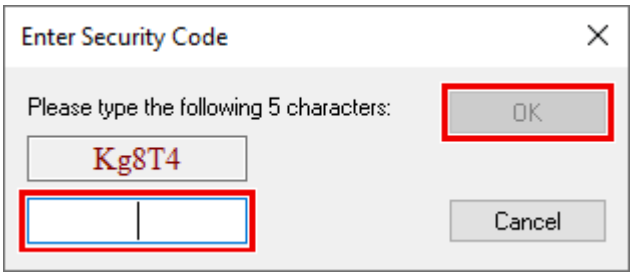


7. 打开 **Order Information (Runtime)（订单信息（运行时））** 选项卡。
   ⇨ 在授权一览表中，之前选定的授权显示为“缺失”状态。

8. 单击 **7 Days Trial License...（7 天试用授权.....）**，激活 7 天试用授权。



⇨ 一个对话框打开，提示输入对话框中显示的验证码。



9. 正确输入验证码，并确认输入信息。
10. 确认对话框，显示激活成功。
    ⇨ 在授权一览表中，授权状态会显示授权的到期日期。
11. 重启 TwinCAT 系统。

⇨ 7 天试用版已启用。

3.2 设置 NVIDIA® 显卡

如果在GPU加速的AI模型执行过程中**没有**使用带有相关Beckhoff镜像的Beckhoff硬件，则客户应自行负责在其系统上创建运行显卡所必需的框架条件。

如果您使用的是带有GPU和相关Beckhoff镜像的Beckhoff IPC，则可以跳过这一部分。

系统要求

系统要求特定得 NVIDIA® 组件。下列软件版本不应高于指定版本（例如，请勿安装 CUDA 12.6），否则可能会不兼容。

NVIDIA® 驱动程序

- 普通 NVIDIA® GPU：版本 560.67
- Quadro/RTX GPU：版本 522.86

CUDA（Compute Unified Device Architecture，统一计算设备架构）

- CUDA 版本 12.5.1

cuDNN

- cuDNN 版本 9.2.1.18（压缩文件，请勿用作 Windows 安装程序）
- 安装说明请访问 NVIDIA® 网站：Tarball 安装。
- 必须将 cuDNN 库的路径添加到系统的（而不是用户的）PATH 变量中。

一般来说，上述措施符合以下系统要求：

- cuda.dll 库的可用性
- nvml.dll 库的可用性

优化的运行时性能表现

在使用倍福硬件（例如带有GPU的C6043工业PC及其相关的Beckhoff镜像）时，NVIDIA® GPU 的运行时行为是为与 TwinCAT 实时交互而优化设计的。使用第三方 GPU 时，根据特定 GPU 及其设置，模型执行期间可能会出现很大的运行时波动。

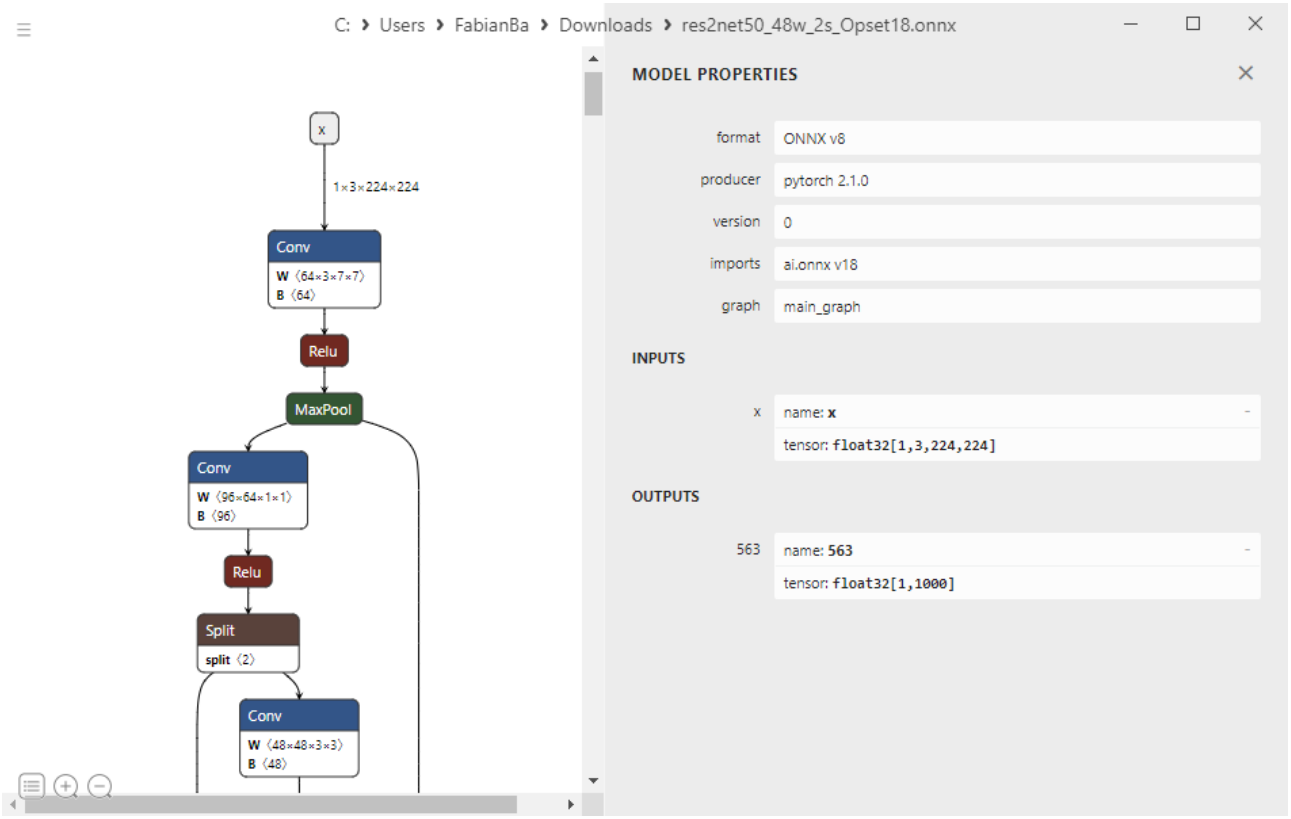
启动服务器后，TwinCAT Machine Learning Server 会立即在日志文件 [▶ 47]中告知所使用显卡可能存在的问题。

4 快速入门

创建或下载 ONNX 文件

如果您没有自己的 ONNX 来进行初始测试，可以使用 GitHub 上的 ONNX Model Zoo 进行测试（举例）。下面以 ONNX Model Zoo 的 ResNet50 为例进行说明。

Netron 可用于轻松检查是否符合 TwinCAT Machine Learning Server 执行的要求 [▶ 29]。



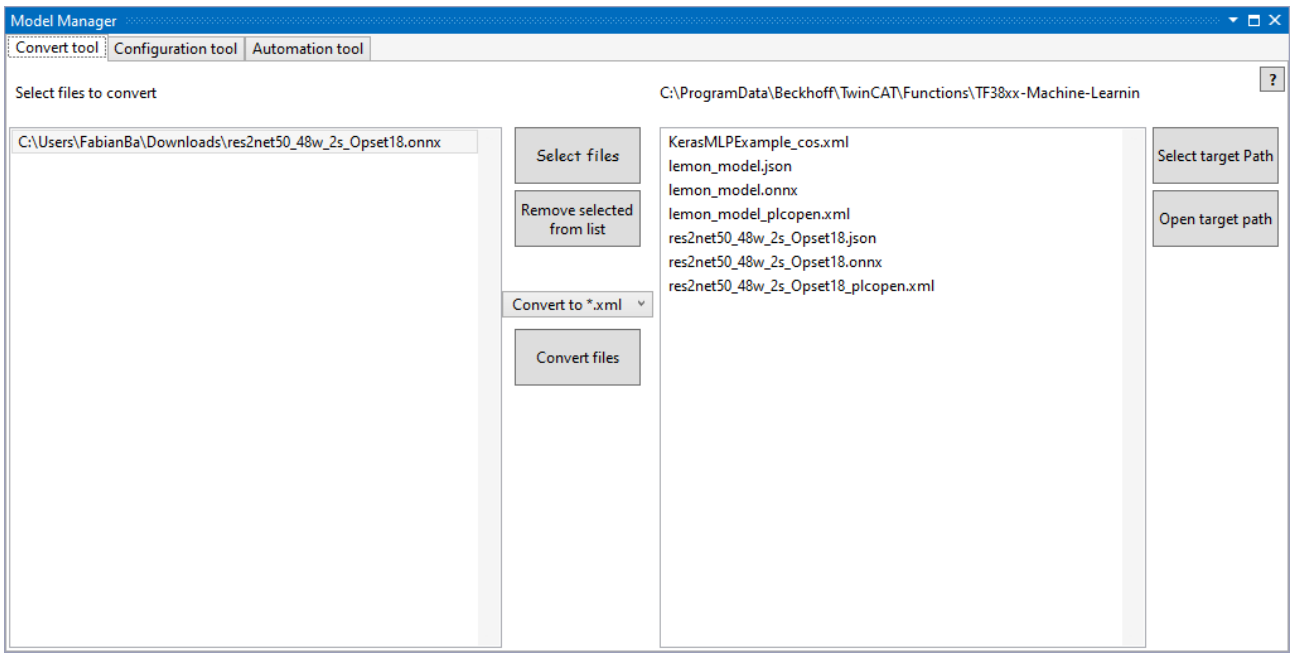
输入节点不是动态的，所使用的 ONNX Opset 也受支持。

使用 TwinCAT Machine Learning Model Manager 准备 ONNX 文件

打开 TwinCAT XAE 并导航至 TwinCAT > Machine Learning(机器学习) > Machine Learning Model Manager（机器学习管理器） [▶ 24]。

用 "Select files"（选择文件）加载下载的 ONNX，然后选择 "Convert files"（转换文件）。转换的 ONNX 及相关 JSON 和 PlcOpenXml 文件会显示在目标路径中。

选择 "Open target path"（打开目标路径），在此路径上打开 File Explorer（文件资源管理器）。



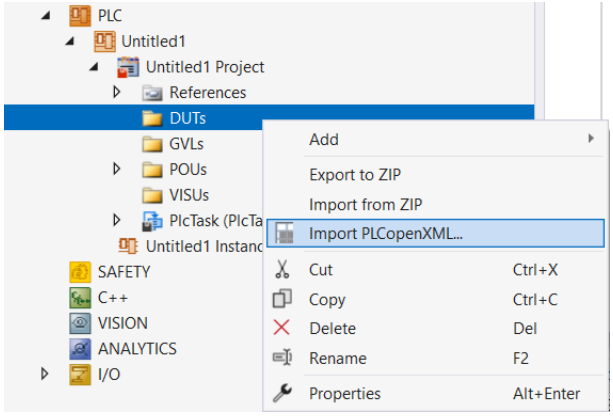
使文件在目标系统上可用

在本快速入门章节，假定 TwinCAT Machine Learning Server 与 PLC 在同一设备上运行。相应地，模型文件（res2net50_48w_2s_Opset18.onnx 和 res2net50_48w_2s_Opset18.json）存储在目标设备的 C:\models 路径下。

有关此步骤的更多信息，请访问：将模型描述文件在服务器设备上可用 [▶ 20]。

编写源代码

从一个空的 PLC 项目开始。首先，右击 DUTs 文件夹并选择 "Import PLCopenXML"（导入 PLCopenXML），导入已创建的 res2net50_48w_2s_Opset18_plcopen.xml。



在 References下添加 PLC 库 Tc3_MlServer。

在简化的示例中，该代码由两个步骤组成。首先，在 TwinCAT Machine Leanring Server 上创建会话，然后执行加载模型的推理。

声明

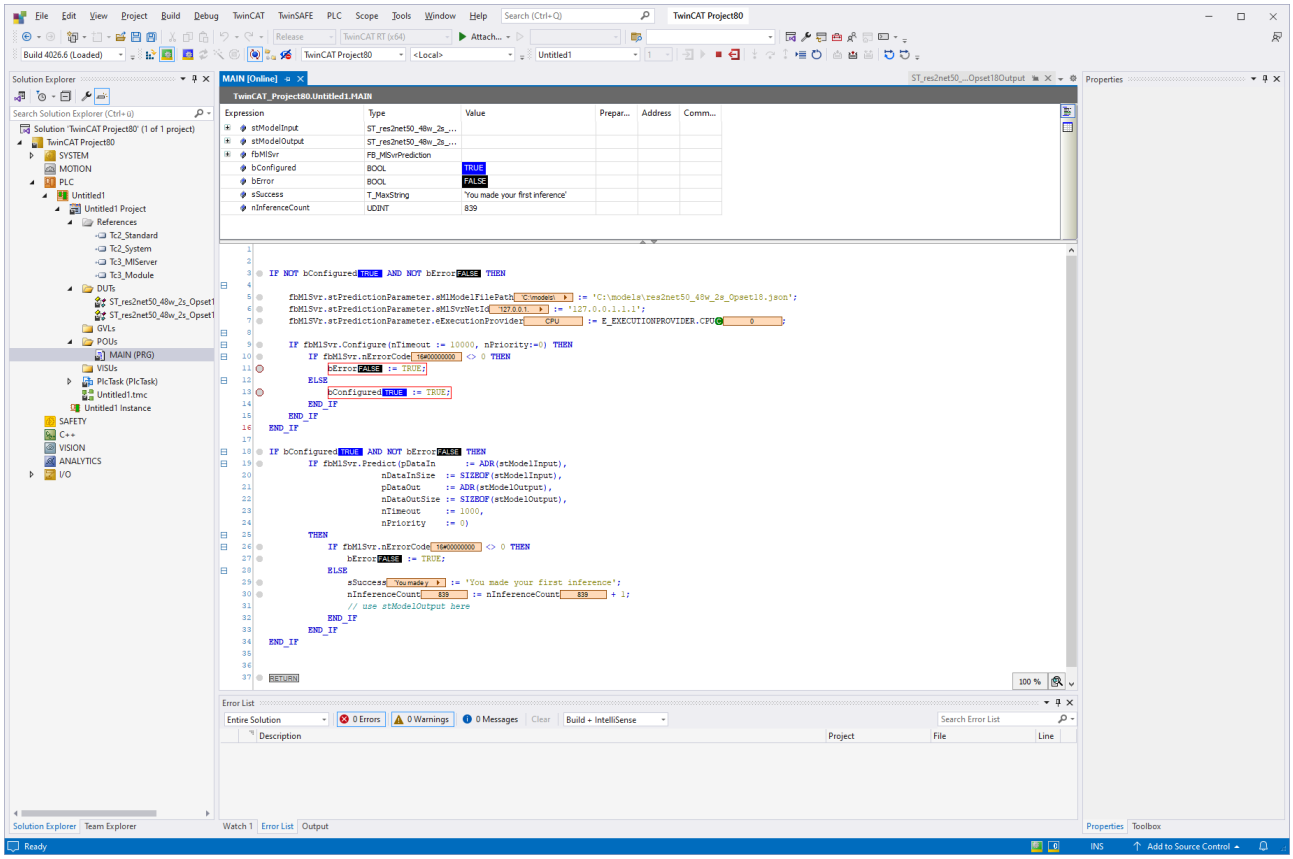
```
stModelInput : ST_res2net50_48w_2s_Opset18Input;
stModelOutput : ST_res2net50_48w_2s_Opset18Output;

fbMlSvr     : FB_MlSvrPrediction;
bConfigured : BOOL := FALSE;
bError      : BOOL := FALSE;

sSuccess        : T_MaxString;
nInferenceCount : UDINT := 0;
```

代码

```
IF NOT bConfigured AND NOT bError THEN

   fbMlSvr.stPredictionParameter.sMlModelFilePath := 'C:\models\res2net50_48w_2s_Opset18.json';
   fbMlSvr.stPredictionParameter.sMlSvrNetId := '127.0.0.1.1.1';
   fbMlSVr.stPredictionParameter.eExecutionProvider := E_ExecutionProvider.CPU;

   IF fbMlSvr.Configure(nTimeout := 10000, nPriority:=0) THEN
      IF fbMlSvr.nErrorCode <> 0 THEN
         bError := TRUE;
      ELSE
         bConfigured := TRUE;
      END_IF
   END_IF
END_IF

IF bConfigured AND NOT bError THEN
      IF fbMlSvr.Predict(
                  pDataIn      := ADR(stModelInput),
                  nDataInSize  := SIZEOF(stModelInput),
                  pDataOut     := ADR(stModelOutput),
                  nDataOutSize := SIZEOF(stModelOutput),
                  nTimeout     := 1000,
                  nPriority    := 0)
      THEN
         IF fbMlSvr.nErrorCode <> 0 THEN
            bError := TRUE;
         ELSE
            sSuccess := 'You made your first inference';
            nInferenceCount := nInferenceCount + 1;
            // use stModelOutput here
         END_IF
      END_IF
END_IF
```

激活配置

激活配置，启动 PLC。结果显示如下。计数器值 `nInferenceCount` 增加，代码 `sSuccess` 成功运行。



5 技术简介

5.1 工作流程

工作流程包括以下步骤：

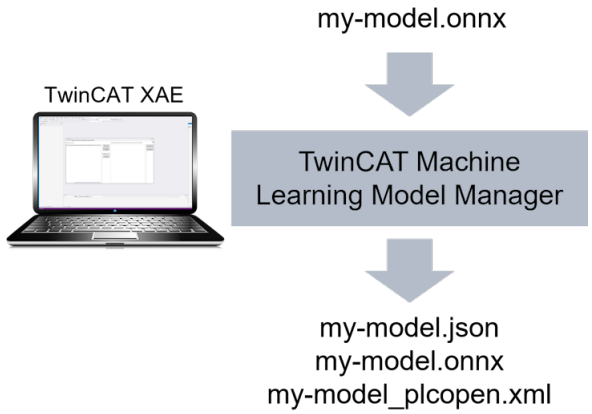
- 从 ONNX 文件中提取 PLC 接口说明 [▶ 18]（使用 TwinCAT Machine Learning Model Manager（TwinCAT机器学习模型管理器） [▶ 24]）。
  - 从 ONNX 文件创建一个 PlcOpenXml文件，它为 PLC 提供了模型的输入和输出数据类型。
  - 生成的 JSON 文件包含模型的元数据。这些元数据一方面包含与TwinCAT平台相关的配置信息，另一方面由用户根据具体应用场景定义的自定义信息。
- 在目标系统上提供 JSON 文件和 ONNX 文件 [▶ 20]。
- 在 TwinCAT 工程环境中导入 PlcOpenXml [▶ 18]，并通过FB_MlSvrPrediction功能块 [▶ 32] 将其集成到 PLC 程序。
- PLC 配置 TwinCAT Machine Learning Server [▶ 21] 并加载 AI 模型。
- 将加载的AI模型异步调用到PLC任务周期中 [▶ 22]。
- 在设备运行时交换/更新 AI 模型 [▶ 24]。

5.1.1 为TwinCAT机器学习服务器准备ONNX文件

生成PLC的接口描述

要在 TwinCAT PLC 中使用带有 FB_MlSvrPrediction [▶ 32] 的 ONNX，需要接口信息。这些都是由 TwinCAT Machine Learning Model Manager（TwinCAT机器学习模型管理器） [▶ 24] 生成的。

有关支持的 ONNX Opset版本 和限制的信息，请访问：ONNX 支持 [▶ 29]。



JSON 文件

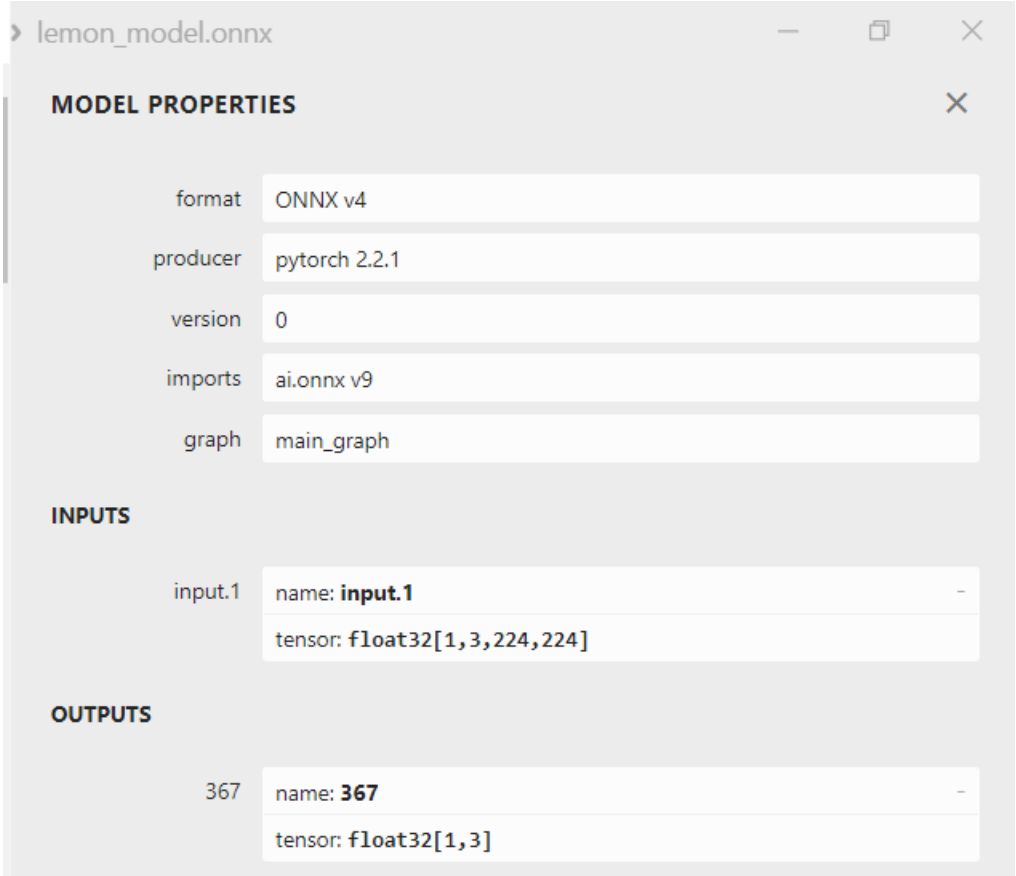
JSON 文件包含功能块 `FB_MlSvrPrediction` 所需的元数据。用户还可以通过自定义属性 [▶ 40]在 JSON 文件中添加自己的元数据。JSON 文件由功能块加载，参见 Configure 方法 [▶ 33]。JSON 文件和 ONNX 文件都必须可以在运行时 PC 上加载，参见 将模型描述文件在服务器设备上可用 [▶ 20]。

PlcOpenXml

PlcOpenXml 包含输入和输出节点的 PLC 类型描述。自动生成的输入/输出结构 (DUT) 反映了您提供的 ONNX 文件的输入/输出定义。因此，请确保为 ONNX 中的输入和输出节点使用*有意义的名称*。

严格要求使用这些创建的数据类型。

在 PlcOpenXml 文件中生成名称的示例：



文件名：**lemon_model**.onnx

名称输入节点：**input.1**

名称输出节点：**367**

生成两个 STRUCT 类型的 DUT，名称分别为 ST_**lemon_model**Input 和 ST_**lemon_model**Output。每个 STRUCT 都有一个头部（不要更改！）和一个 data area。data area 元素根据输入和输出节点命名，在上述情况中为 in_**input1** 和 out_**367**。PLC 中不允许使用的字符会被自动删除。如果一个 AI 模型有多个输入或输出节点，则每个节点都会座位 STRUCT 的一个元素显示。

生成的输入数据类型示例

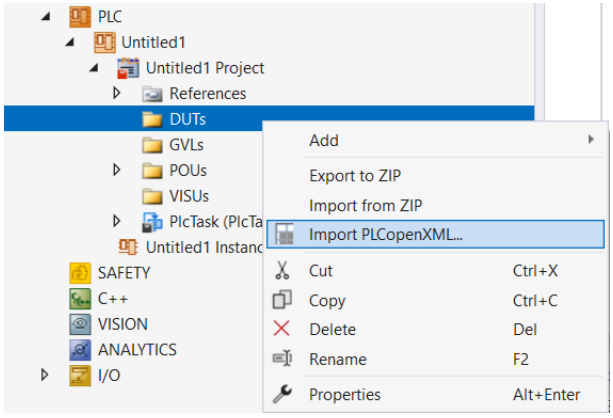
```
TYPE ST_lemon_modelInput :
STRUCT
   {attribute 'hide'} _header_DO_NOT_CHANGE : ARRAY[0..63] OF SINT := [-1,-112,120,86,52,18,-1,-1,1,
-1,0,0,-1,-1,-112,-112,104,-72,-97,-115,-17,4,98,-85,-50,-67,-12,-50,-6,33,-10,-11,88,80,121,12,119,
-56,-24,8,110,-32,-69,113,-21,3,102,-60,64,48,9,0,0,0,0,0,0,0,0,0,0,0,0];
   in_input1 : ARRAY[0..0,0..2,0..223,0..223] OF REAL;
END_STRUCT
END_TYPE
```

头部表示 AI 模型输入张量（输出数据类型则为输出张量）形状的哈希值。这样可以确保数据类型与正确 的 ONNX 一起使用。这也意味着，具有相同输入和输出形状和字典顺序的模型具有相同的哈希值，因此他们数据类型相互兼容。这对于模型更新 [▶ 24]尤为重要。

头部的字节数组不得更改!
如果需要，可以调整 STRUCT 中元素的名称，但不能调整它们在 STRUCT 中的顺序。

在 TwinCAT 3 中导入 PlcOpenXml

右击PLC项目中文件夹（如 DUTs），通过 "Import PLCopenXML"（导入 PLCopenXML）字段，将生成的 PlcOpenXml 导入到 PLC 项目。



注意

验证的签名：Machine Learning Server仅通过生成的输入和输出模型类型进行操作

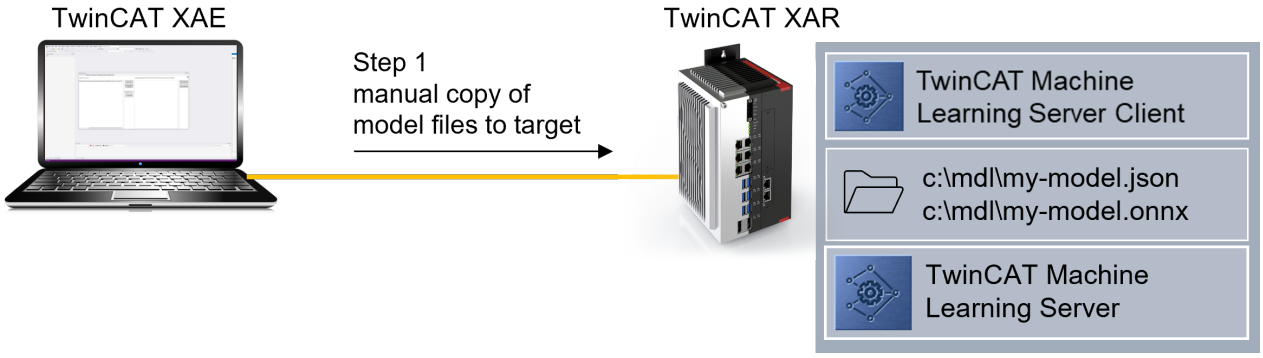
请注意，必须使用提供的PlcOpen文件中定义的输入/输出模型类型。类型的签名由 TcMlServer 验证，以确保推理操作的安全性。

5.1.2 将模型描述文件在服务器设备上可用

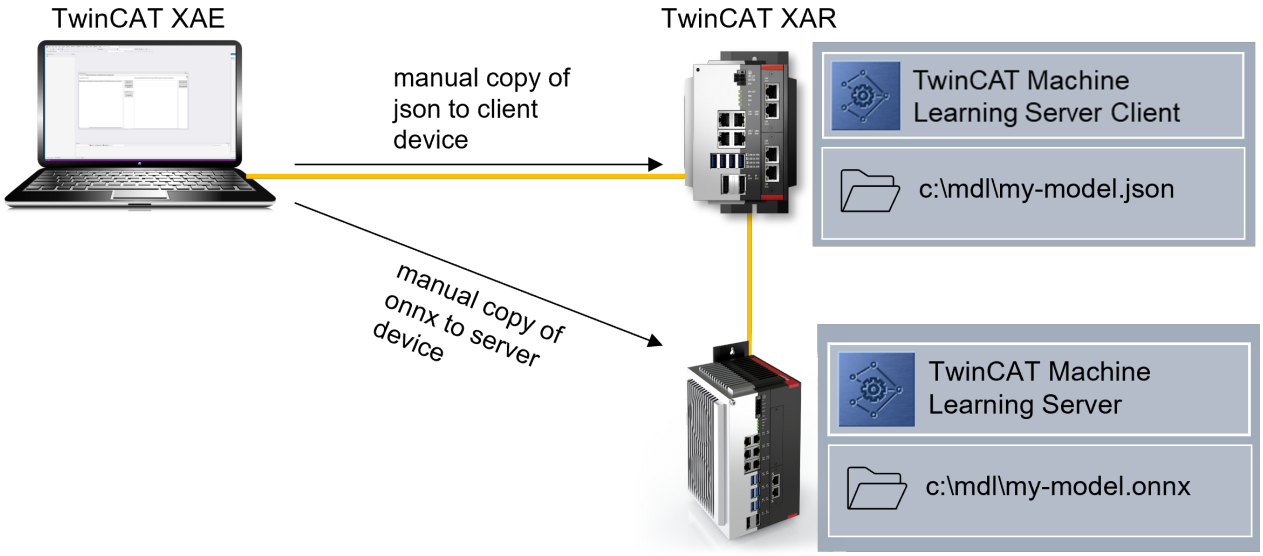
为了能够加载模型，必须让 TwinCAT Machine Learning Server 知道这些模型。这意味着服务器必须知道创建的 JSON 和 ONNX 文件在文件系统中的定位，才能成功加载 AI 模型。该配置使用 Configure [▶ 33] 方法发布。该方法需要传入JSON文件的完整路径，该文件包含接口描述信息。相关的 ONNX 文件必须存储在同一文件夹路径下。哈希值始终用于验证 JSON 与 ONNX 之间的关联性。

ONNX文件需要安装在运行TwinCAT Machine Learning Server 的设备上。JSON 文件则需要在客户端设备上提供。存储路径本身是任意的；但 JSON 和 ONNX 必须具有相同的路径。

方案 1 - 客户端和服务器安装在同一 IPC 上：JSON 和 ONNX 文件必须以任意路径存储在 IPC 上。

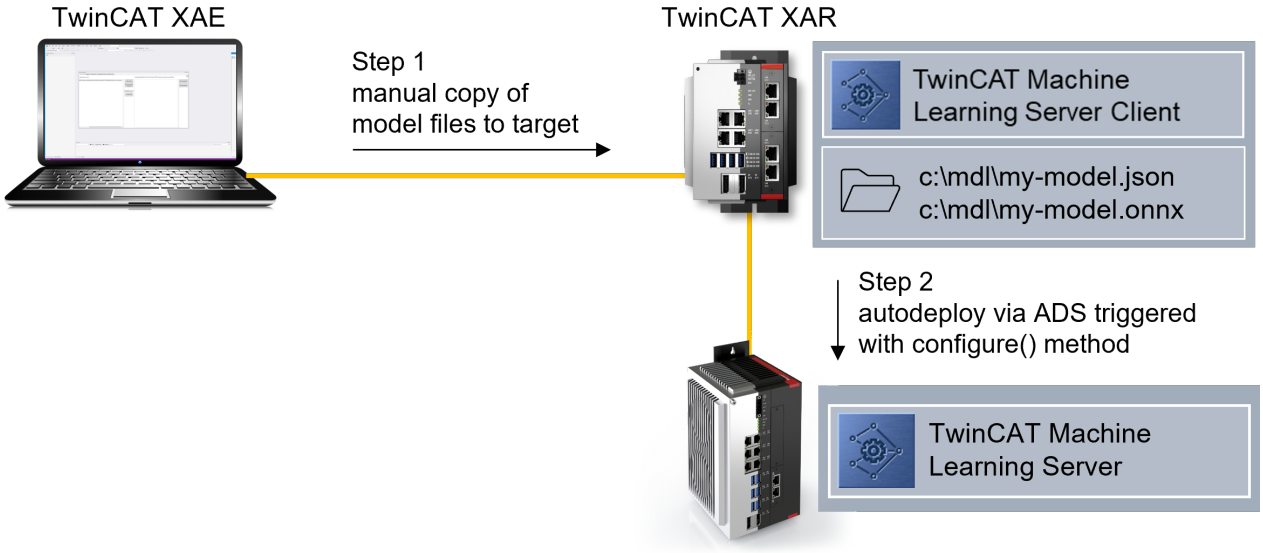


方案 2 - 客户端和服务器安装在不同的 IPC 上：ONNX 文件可直接存储在服务器上。JSON 文件必须存储在客户端设备的相同路径下。



方案 3 - 客户端和服务器安装在不同的 IPC 上：模型文件（ONNX 和 JSON）也可以同时存储在客户端。调用 Configure 方法时，首先检查服务器设备上的路径。如果在那里找不到 ONNX 文件，则检查客户端设备上的路径。如果在客户端设备上找到 JSON 和 ONNX，则通过 ADS 将 ONNX 文件传输到服务器设备，然后从服务器加载。该方案耗时较长，但仅需针对该模型执行一次。之后，模型将直接从服务器设备加载。

ADS 路由器内存必须足够大，以便能够通过 ADS 发送模型。



5.1.3 从 PLC 客户端配置服务器

调用方法 configure [▶ 33] 会在 TcMlServer 中为功能块 `FB_MlSvrPrediction` 的相应实例实例化一个会话。FB 成员 stPredictionParameter [▶ 36] 中定义的配置用于实例化。通常，每个 `FB_MlSvrPrediction` 实例在服务器上都分配有自己的会话。不过，会话也可以通过参数 `bExclusiveSession` 共享使用。

在配置调用过程中，特别定义了以下内容：

- TwinCAT Machine Learning Server 在哪里?
  - 通过服务器设备的 AMS Net Id 指定。默认值为“本地”。
- 应加载哪个 AI 模型?
  - 通过相应 JSON 文件的路径指定。
- AI 模型应该在哪个硬件上执行?

- 通过执行提供程序 (E_ExecutionProvider [▶ 36]) 和 可选的GPU设备ID来定义。

每个打开的会话都会在服务器设备上分配资源。并行会话的数量在软件端没有限制，仅受可用硬件资源的限制。如果没有足够的可用内存（RAM 或 vRAM）来打开另一个会话，配置命令将失败。

deconfigure [▶ 33] 方法可用于关闭会话，从而释放资源。

如果客户端未在定义的时间段内（即所谓的会话超时时间）向服务器发送请求，服务器就会认为该客户端不再活跃。当达到配置的会话超时 [▶ 36]时，会话将自动关闭。

示例

声明

```
fbMlSvr : FB_MlSvrPrediction();
```

代码

```
// configure session paramaters
fbMlSvr.stPredictionParameter.sMlModelFilePath := 'C:\mdl\lemon_model.json';
fbMlSvr.stPredictionParameter.sMlSvrNetId := '127.0.0.1.1.1';
fbMlSVr.stPredictionParameter.eExecutionProvider := E_ExecutionProvider.CPU;

// Submit configuration request to the TcMlServer
// Provide a generous nTimeout, as the configuration can take a substantial amount of time
IF fbMlSvr.Configure(nTimeout := 1000, nPriority:=0) THEN
// check for error
// change state
END_IF;
```

5.1.4 执行 AI 模型

带有 FB_MlSvrPrediction [▶ 32] 的 AI 模型的执行通过 Predict [▶ 34] 方法（或 batched [▶ 34] 调用时的 *PredictBatched*）触发，与 PLC 任务循环异步。

输入数据类型和输出数据类型（参见 Machine Learning Model Manager [▶ 24] 章节的 PlcOpenXml）以及超时和优先级都会传递给该方法。

发送推理命令：

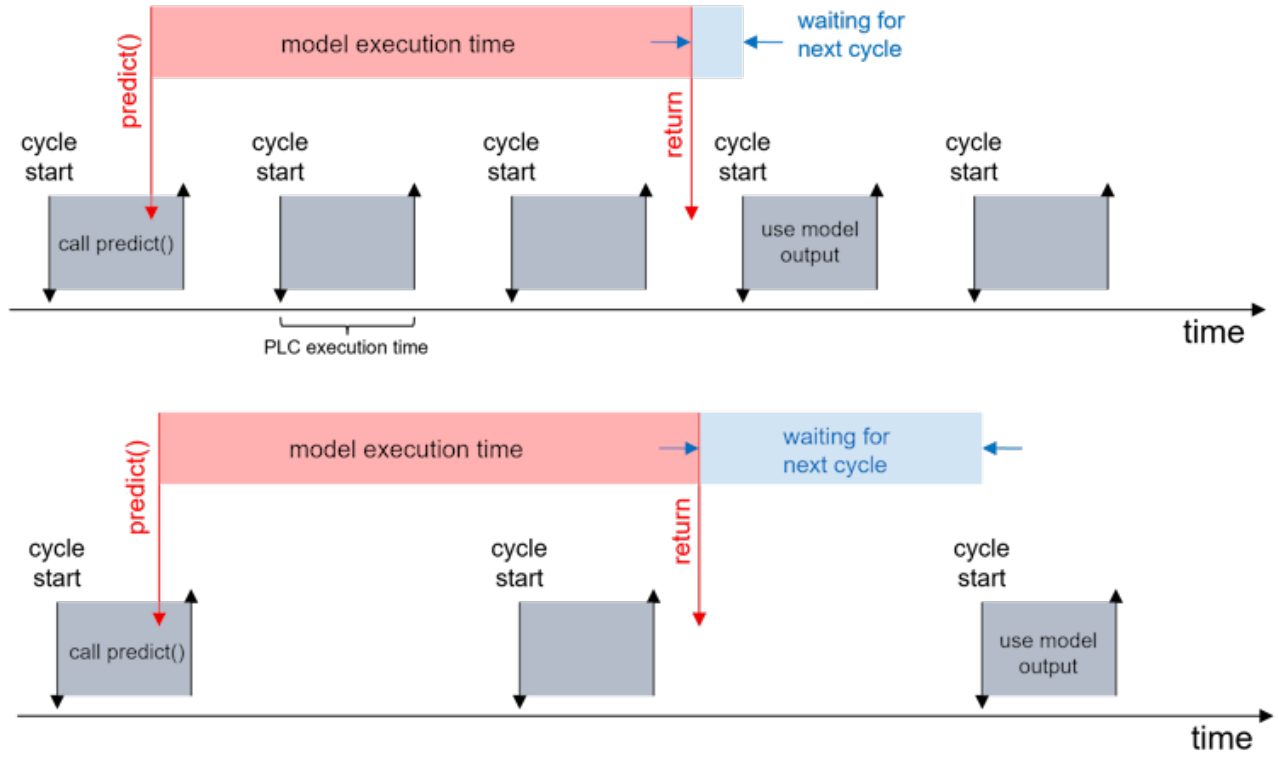
调用该方法时，输入数据区域会通过ADS发送到服务器。为此所需的复制操作在任务循环中同步进行。请注意，较大的数据量会需要更多时间。传输大量数据（如大图像数据）时，必须配置 PLC 任务循环时间，以防止循环超时。

编辑推理命令：

随后 TwinCAT Machine Learning Server 接受该推理命令。如果有多个请求无法同时处理，就会创建一个队列，其中优先级较高的请求会在队列中提前处理。基于 CPU 的推理命令总是顺序处理，即队列在这里特别重要。而基于 GPU 的推理则可以并行处理。每个 FB 实例在服务器上只能有一个待处理请求，即服务器上的最大请求数就是活跃客户端的数量。

请求推理结果：

一定要注意，异步请求的处理只能以 PLC 任务循环时间的时间粒度进行监控。因此，延迟时间预算较低的应用程序必须尽可能缩短循环时间，并相应提高采样率，以便最大限度地减少由时间分辨率引起的延迟。这一点在与其他计算密集型同步执行算法交互时尤为重要，例如在预处理或后处理数据时。



Sample Predict()

声明

```
stModelInput : ST_LemonModelInput; // DUT from PlcOpenXml produced with TC ML Model Manager
stModelOutput : ST_LemonModelOutput; // DUT from PlcOpenXml produced with TC ML Model Manager
fbMlSvr : FB_MlSvrPrediction();
```

代码：

```
// Submission of an inference request at the TcMlServer
// and subsequent postprocessing of the inference result
// Submission of the asynchronous inference request to the TcMlServer
IF fbMlSvr.Predict(
                pDataIn       := ADR(stModelInput),
                nDataInSize   := SIZEOF(ST_LemonModelInput),
                pDataOut      := ADR(stModelOutput),
                nDataOutSize  := SIZEOF(ST_LemonModelOutput),
                nTimeout      := 100,
                nPriority     := 0) THEN
      IF fbMlSvr.nErrorCode <> 0 AND NOT fbMlSvr.bConfigured  THEN
         // If nErrorCode -1 is encountered, increase nTimeout
          eState := E_State.eError;
      ELSE
         // Postprocessing of the inference results
   END_IF
END_IF
```

Sample PredictBatched()

声明

```
stModelInputBatch : ARRAY[0..3] OF ST_LemonModelInput;
stModelOutputBatch : ARRAY[0..3] OF ST_LemonModelOutput;
```

代码

```
IF fbMlSvr.PredictBatched(
                pDataIn       := ADR(stModelInputBatch),
                nDataInSize   := SIZEOF(ST_LemonModelInput),
                nBatchSize    := 4,
                pDataOut      := ADR(stModelOutputBatch),
                nDataOutSize  := SIZEOF(ST_LemonModelOutput),
                nTimeout      := 100,
                nPriority     := 0) THEN
```

在示例调用中，每次推理请求的超时都设置为 100 个周期。在上图中，3 个周期（上图）或 2 个周期（下图）后即可得到结果。可通过 `fbMlSvr.nMaxInferenceDuration` 查询可能出现的抖动。这显示了执行推理所需的最大 PLC 循环数。根据该值，通常可以很好地解释超时值。

5.1.5 更新 AI 模型

可以在运行时交换 AI 模型。下面介绍了两个案例及操作步骤。

案例 1：不改变模型接口的模型更新

案例定义：

在本例中，AI 模型的输入和输出接口在更换模型时保持不变。为了实现这一点，输入和输出节点的顺序（如果有多个节点）和形状必须保持不变。

建议在更新模型时不要改变整个模型架构，即在现有 AI 模型上进行迁移训练/微调。因此，模型的接口不变，运行时行为也保持不变。

模型更新无需编译进程，也无需停止 TwinCAT。

更新模型的步骤：

- 使用 TwinCAT Machine Learning Model Manager 创建 JSON 和 PlcOpenXml。
- 在相关系统中提供 JSON 和 ONNX。如果完整路径已更改，请通过ADS（例如）将新路径存储在一个变量中。
- 输入和输出数据类型的哈希值不变。这意味着**无**需读取新的 PlcOpenXml。新的 PlcOpenXml 仍未使用。
- 改变正在运行的 PLC 中的状态，例如通过 ADS 设置相应的变量，然后调用 Deconfigure [▶ 33]() 关闭当前会话。
- 调用“配置”加载新的 JSON。

从 Deconfigure 到 Configure 方法完成期间，不能通过此功能块向服务器发送推理调用。

案例 2：带有模型接口变更的模型更新

案例定义：

在这种情况下，当AI模型被替换时，其输入和输出接口会发生变化。因此，PLC 中的输入和输出数据类型不再与模型相匹配。通常情况下，源代码中有多处地方需要修改。

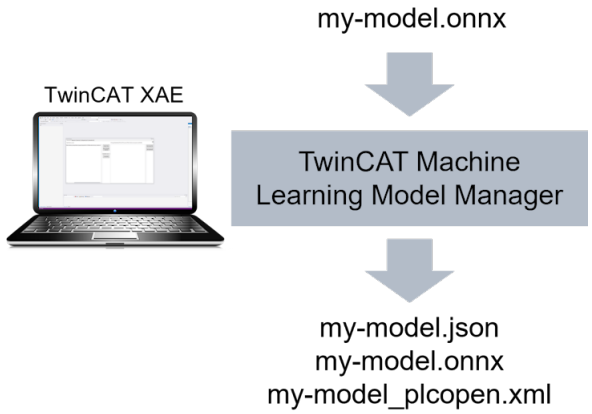
建议在 TwinCAT XAE 中修改源代码，重新测试项目，然后再将其加载到设备上。请注意，AI 模型的运行时行为也可能发生了变化。

在本例中，当加载修改后的新 TwinCAT 项目时，模型更新与 TwinCAT同时停止。

5.2 TwinCAT Machine Learning Model Manager（TwinCAT机器学习模型管理器）

TwinCAT Machine Learning Model Manager 用于管理 ONNX 模型文件，并准备这些文件，以供与 TwinCAT 一起使用。TwinCAT Machine Learning Model Manager 的功能总结如下：

- 创建元数据文件（JSON 文件）。
- 创建 PlcOpenXml，用于描述模型输入和输出的PLC数据类型。
- 可选：插入特定应用的元数据（自定义属性）。用户可以在 JSON 中添加自定义属性。在 PLC 运行时使用 FB_MlSvrPrediction [▶ 32] 的方法读出这些属性。



TwinCAT Machine Learning Model Manager 有三种不同的接口：

1. 图形用户界面（Visual Studio 插件）
2. Python 接口（Python 软件包）
3. **C**ommand **L**ine **I**nterface（命令行接口，CLI）

这些信息在下文有详细说明。

5.2.1 图形用户界面

TwinCAT 3 Machine Learning Model Manager 是编辑 ONNX 文件的核心用户界面。该工具集成在 Visual Studio 中，可通过菜单栏中的 **TwinCAT > Machine Learning(机器学习) > Machine Learning Model Manager（机器学习模型管理器）** 打开。

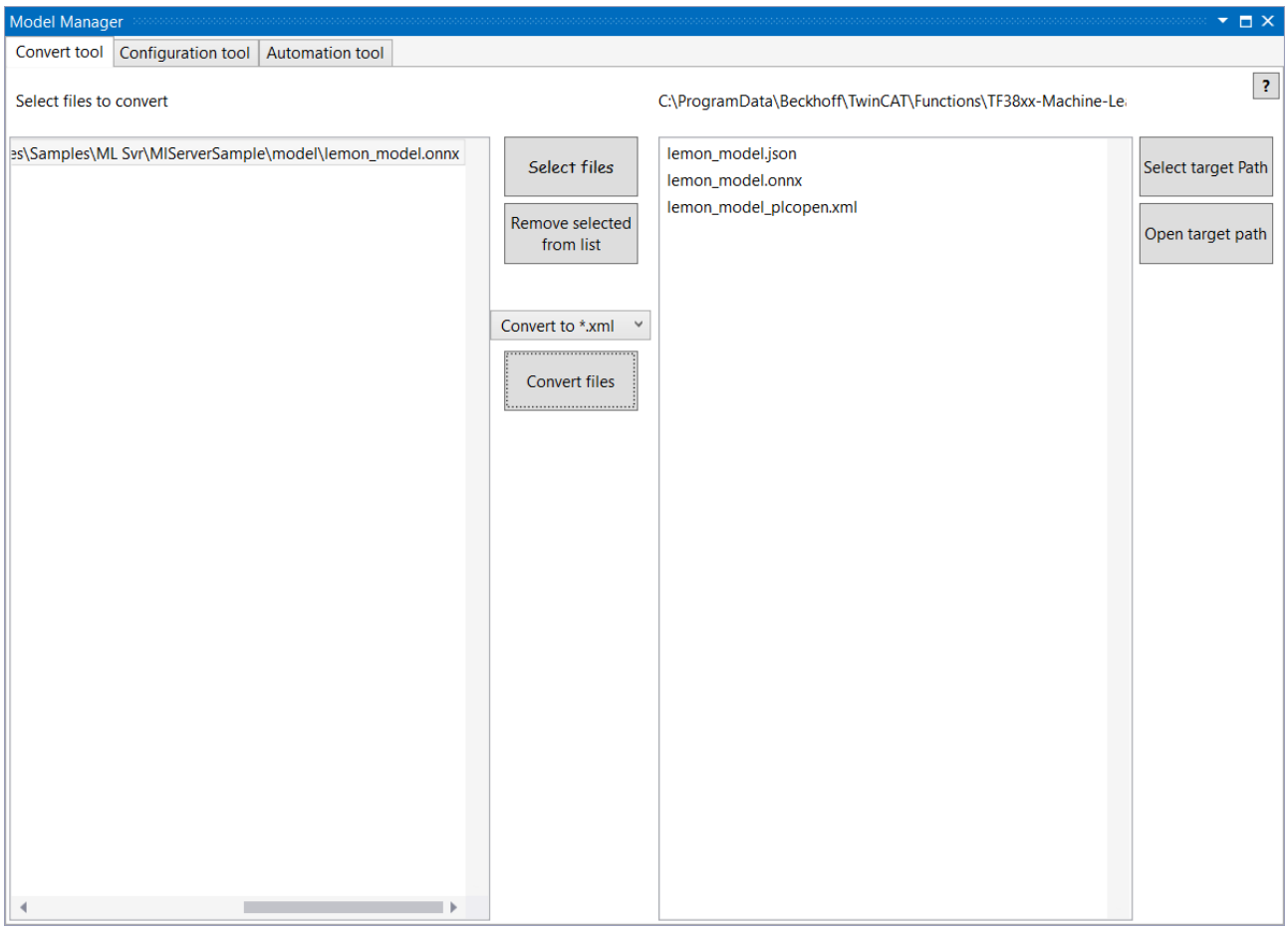
必需的 Visual Studio 版本

TwinCAT 3 Machine Learning Model Manager 的图形界面与 Visual Studio 2017、2019、2022 和 TcXaeShell 兼容。

创建 JSON 和 PlcOpenXml

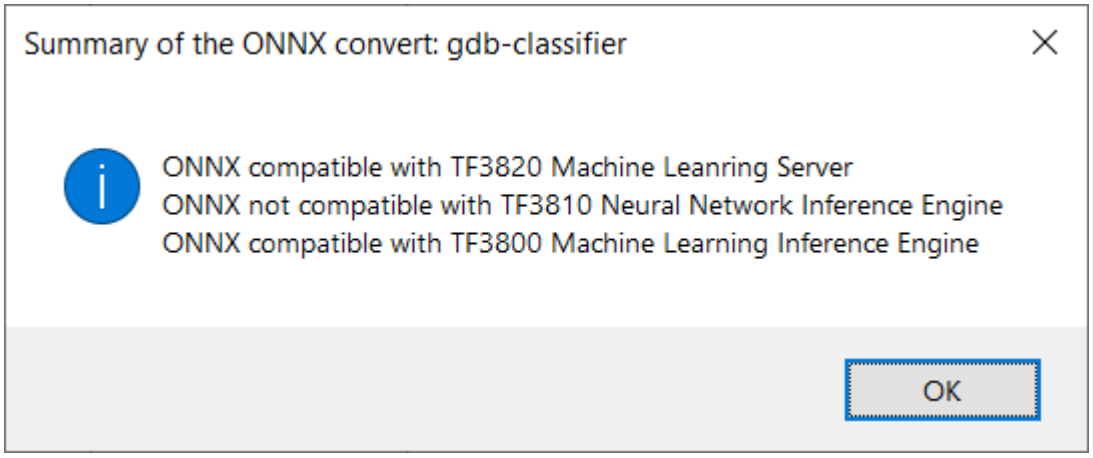
1. 打开 **Convert Tool（转换工具）**选项卡。
   ⇨ 单击 **Select files（选择文件）**打开文件浏览器。
2. 选择 ONNX 文件（按住 Ctrl 键可进行多选）。
   ⇨ 选定的 ONNX 文件列在左侧，并标有路径和文件名。
3. 如有需要，您可以选择文件，然后单击 **Remove selected from list（从列表中删除选定文件）**按钮，再将其从列表中删除。
4. 点击 **Convert files（转换文件）**，创建所需的 JSON 和 PlcOpenXml。
   ⇨ 文件存储在转换文件路径中。默认路径为 <TwinCATPath>\Functions\TF38xx-Machine-Learning\ConvertToolFiles。
5. 点击 **Open target path（打开目标路径）**，在文件浏览器中打开转换后的文件路径。

⇨ 可以使用 **Select target path（选择目标路径）**更改路径。即使重启 PC，也会保留更改。



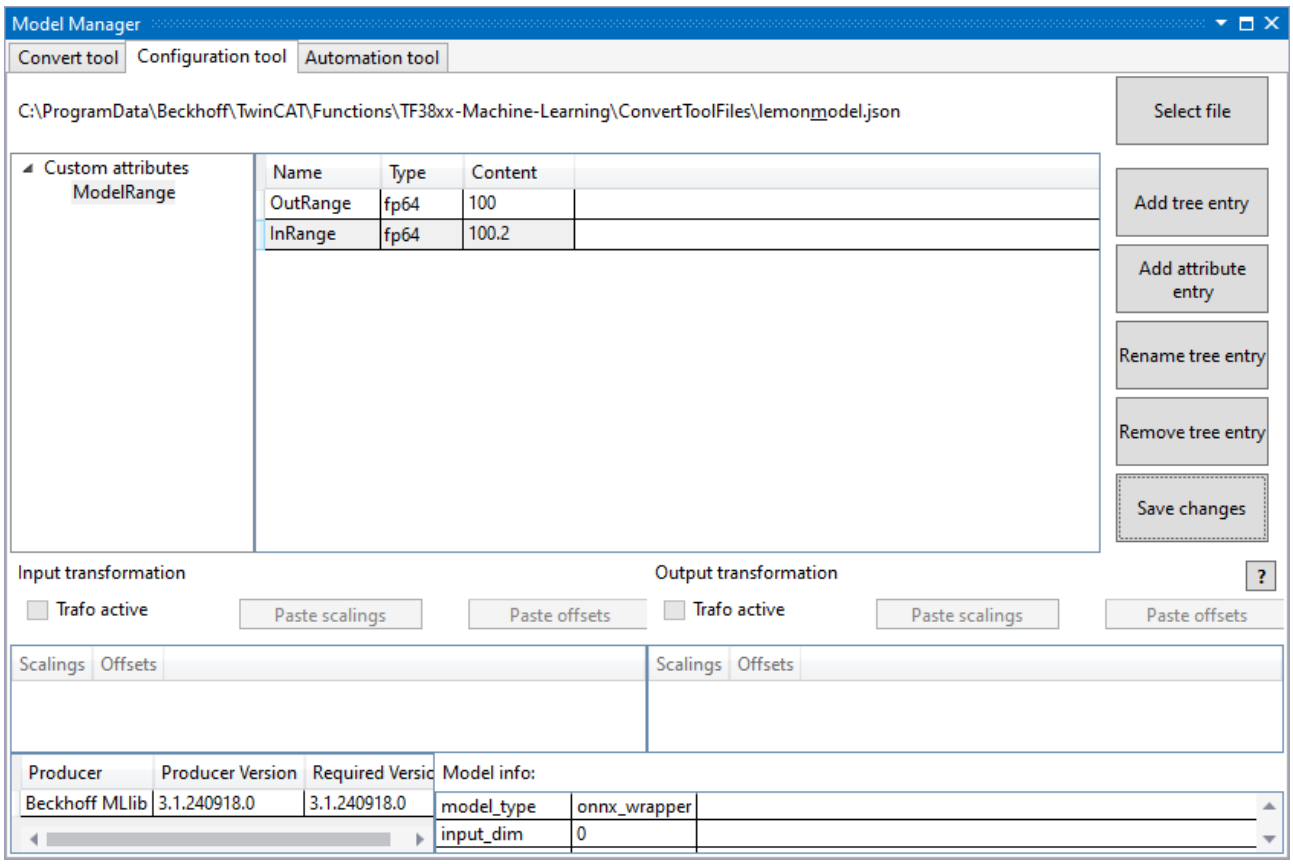
这里介绍的工具也是管理TwinCAT机器学习推理引擎和TwinCAT神经网络推理引擎的ONNX文件的核心工具。因此，如果 ONNX 兼容，这些产品也会生成用于可选执行AI模型的描述文件。转换为 *.xml 下拉菜单与这些产品相关。这与 TwinCAT Machine Learning Server 无关。

每个转换后的 ONNX 都会弹出一个窗口，显示 ONNX 与之兼容的 TwinCAT 产品。



创建自定义属性

可通过 **Select File（选择文件）**选择 JSON，然后进行编辑。编辑后，使用 **Save changes（保存更改）**可替代原文件。



可使用以下按钮编辑自定义属性：

- **Add attribute entry（添加属性条目）**：向选定的树形结构项中添加一个属性。必须在左侧列表中选择树项。
  - 如果创建了属性，则必须分别指定名称、数据类型和值。
  - 选择属性并按下 Delete（删除）按钮即可删除属性。
- **Add tree entry（添加树条目）**：在选定的树项下添加一个树项（作为子树项）。
- **Rename tree entry（重命名树条目）**：可对选定的树项进行重命名。
- **Remove tree entry（移除树条目）**：删除选定的树项，包括子树项。

5.2.2 Python 接口

安装 Python 软件包

Python 软件包以 whl 文件的形式保存在 <TwinCatInstallDir>\Functions\TF38xx-Machine-Learning\Utilities\ModelManagerAPI\PythonPackage 文件夹中。

要安装该软件包，请使用 `pip install` <TwinCatInstallDir>\Functions\TF38xx-Machine-Learning\Utilities\ModelManagerAPI\PythonPackage\<whl-file-name>。该文件夹可能包含不同版本的软件包（仅当您在旧的TwinCAT机器学习安装基础上安装了新版本时）。

确保始终使用当前版本。

使用 API

软件包中装有

```
import  beckhoff.toolbox as  tb
```

从 ONNX 文件**创建 JSON 和 PlcOpenXml**

```
tb.onnxprep("C:\\PathTo\\myONNX.onnx")
```

显示**模型信息**

```
tb.info("C:\\PathTo\\myONNX.json")
```

添加**自定义属性**

```
new_ca = { 'nID' : -34234, 'bTested' : True, 'fNum' : 324.3E-12, 'AnotherTreeItem' : { 'fPi' : 3.134
12, 'bFalseFlag' : False }}
tb.modify_ca("C:\\PathTo\\myONNX.json","C:\\PathTo\\myONNX_ca.json", new_ca)
```

添加**模型说明**（模型的名称、版本、作者等）

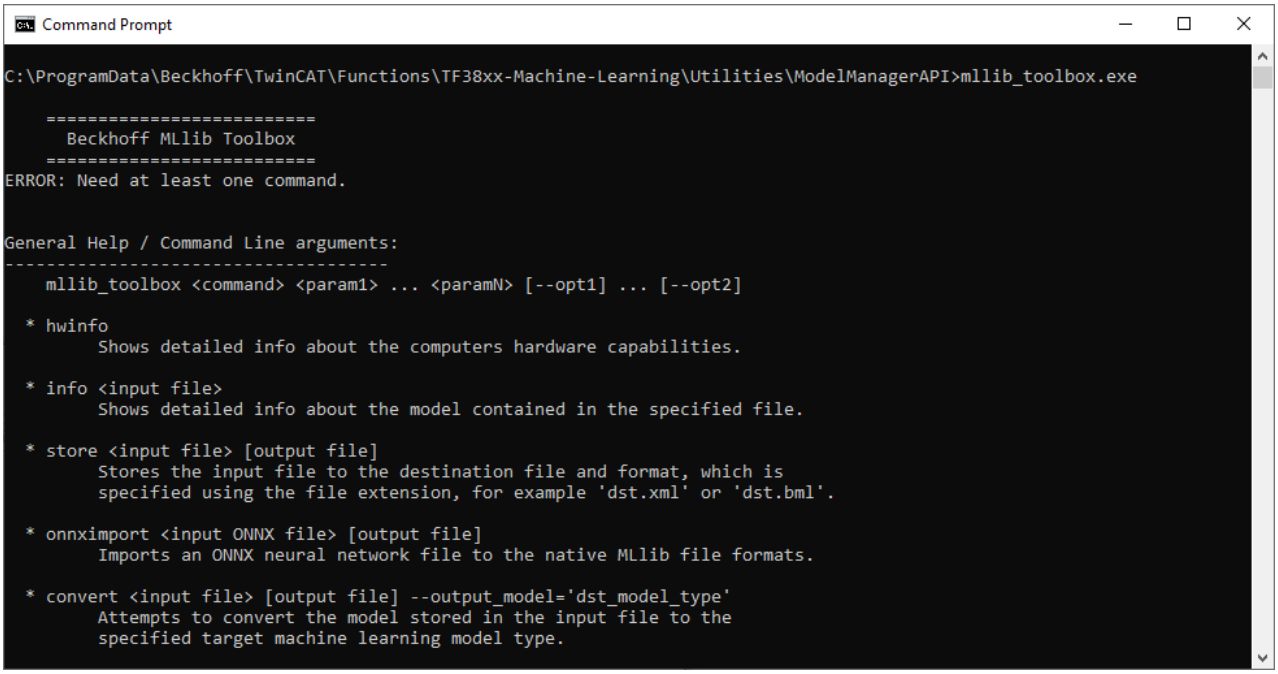
```
model_description = {
    "new_version" : "2.3.1.0",
    "new_name" : "CurrentPreControlAxis42",
    "new_desc":"This is the most awesome model to control Axis42",
    "new_author":"Max",
    "new_tags":"awesome, ingenious, astounding",
}
tb.modify_md("C:\\PathTo\\myONNX.json", "C:\\PathTo\\myONNX_md.json",
             **model_description)
```

5.2.3 命令行接口

模型管理器也可以通过命令提示符使用。为此可以使用mllib_toolbox.exe工具。

可执行文件位于 *<TwinCatInstallDir>|Functions|TF38xx-Machine-Learning|Utilities|ModelManagerAPI*。

运行不带参数运行 exe 会显示帮助信息。



使用 API

从 ONNX 文件**创建 JSON 和 PlcOpenXml**

```
mllib_toolbox.exe onnxprep C:\PathTo\mymodel.onnx
```

显示**模型信息**

```
mllib_toolbox.exe info C:\PathTo\mymodel.json
```

CLI 不支持创建自定义属性。

5.3 ONNX 支持

支持的 ONNX Opset 版本

TwinCAT Machine Learning Server，更精确地说是 TcMlServer 服务，支持 ONNX Opset 版本 21。ONNX 通常提供与 ONNX Opset 较新版本的向后兼容性。

所使用的 Opset 版本通常列于 ONNX 文件的 *imports（导入）*部分。如下图所示，以 ONNX Opset 版本 18 为例，使用 Netron 进行可视化。

对支持的 ONNX 属性的限制

不支持动态输入或输出形状

不支持动态输入或输出形状。仅首位的批量大小（batch size）参数可以是动态值（见下一节）。

允许情况示例：

```
float32[244, 244, 1]

float32[1, 3, 244, 244]

float32[5, 244, 244, 3]
```

不允许情况示例：

```
float32[244, 244, ?]

float32[244, height, 3]

float32[?, 244, 244, ?]

float32[1, 244, 244, unknown]
```

您可以快速固定 ONNX 节点的形状，例如使用 Python 中的 onnxruntime 软件包，参见固定动态输入形状 | onnxruntime。

形状…

```
float32[-1, 3, ?, ?]
```

变成带有…

```
python -m onnxruntime.tools.make_dynamic_shape_fixed --input_name x --input_shape 1,3,960,960
model.onnx model.fixed.onnx
```

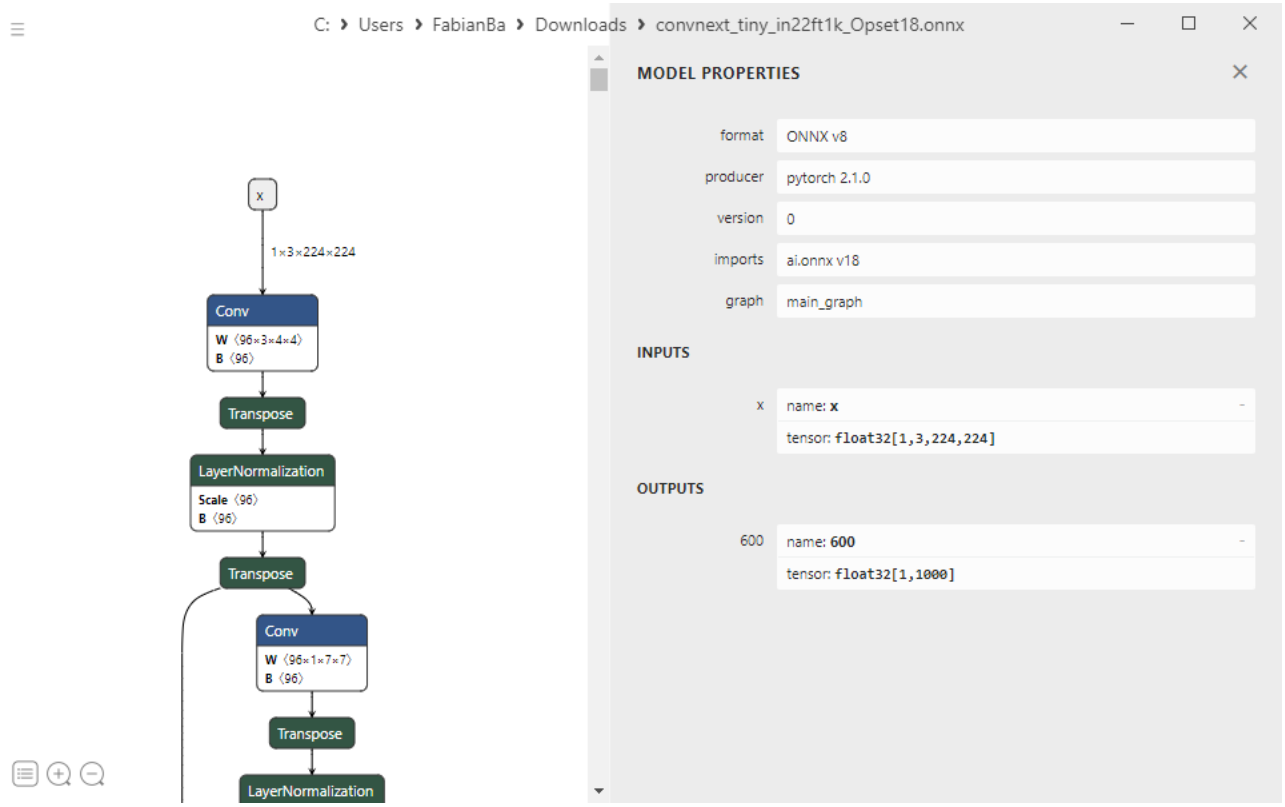
```
float32[1, 3, 960, 960]
```

批量大小必须处于首要位置

TwinCAT Machine Learning Server 仅支持以批量大小作为输入节点首要参数的模型。批量大小参数可以是动态的（与所有其他参数不同）。允许情况示例：

```
float32[batch, 3, 244, 244]

float32[?, 3, 244, 244]

float32[unknown, 244, 244, 3]
```

PredictBatched() [▶ 34] 方法只有在模型包含作为首位参数的动态批量大小时才能使用。



ONNX 文件大小

目前，可加载 ONNX 文件的大小限制为 2 GB。如果该限制对您的应用程序构成挑战，请联系倍福（参见 技术支持和服务 [▶ 49]）。

5.4 TcMlServer 服务

可以通过检查是否有 TcMlServer.exe 服务来验证 TF3820 的安装情况。请注意，TcMlServer 服务在重启时提供延迟自主启动。请注意这里的延迟启动时间。 TcMlServer 在启动时会在系统上分配 ADS 端口 19900。

系统要求

除了 64 位 Windows 和 TC1000 TwinCAT 3 ADS 安装（功能模块 TC1000.ADS.XAR）外，TcMlServer 对系统没有其他正式要求，只要求有约 500 MB 的硬盘。不过，为了实现 TcMlServer 的高性能运行，建议尽可能多地使用 UM 核心。

授权查询

启动时，TcMlServer 会检查 TF3820 授权是否可用。如果授权最初不可用，则每 10 秒钟查询一次授权状态。因此，授权可能需要一小段时间才能生效。如果 TcMlServer 未获得正确授权，将为请求发出相应的错误代码。

安装路径和日志文件：

C:\ProgramData\Beckhoff\TwinCAT\Functions\TF38xx-Machine-Learning\TcMlServer

在支持情况下，日志文件夹中会提供包含所有记录消息的日志文件。

日志文件夹结构如下：

每次重启后，TcMlServer 都会创建一个新目录，该目录的名称由创建日期和时间组成。日志以 JSON 文件格式存储在该目录中，并由 TcMlServer（如有）定期写入。因此，在出现错误时，可能会有多个 JSON 文件，其文件名指定了这些文件包含的错误或警告的区间范围。

5.4.1 执行提供程序

目前可用的 ExecutionProvider 如下：

- CPU
- CUDA

ExecutionProvider [▶ 36] 是通过 FB_MlSvrPrediction [▶ 33] 的 Configure [▶ 32] 方法传输的。

CPU

加载的 AI 模型在 IPC 的 CPU 资源上运行。如果同一设备上正在运行 TwinCAT 运行时，那么只能使用未被 TwinCAT 占用的 CPU 资源（隔离核心只能由 TwinCAT 使用，共享核心只能使用一定时间）。操作系统负责在所有可用线程上并行化计算任务。

如果多个客户端在一个服务器上使用执行提供程序 CPU 创建会话，则推理请求会依次得到处理。调用 Predict 时请注意 `Priority` 参数。

CUDA

加载的 AI 模型在 IPC 的 GPU 资源上执行。如果 GPU 资源充足，可以在一个 GPU 上并行运行多个会话。

使用 `nDeviceId` 字段可以区分可能安装的多个显卡。对于最多只有一个显卡的计算机，该值可保留为默认值，否则该字段对应的是显卡的 CUDA 计算索引。如果使用装有多个显卡的计算机，则应设置以下**系统环境变量**：

```
CUDA_DEVICE_ORDER='PCI_BUS_ID'
```

此外，TcMlServer 允许在不同的功能块（FB）之间共享推理资源，这些功能块为其推理会话提供相同的规范。在预期会出现瓶颈的情况下（例如，显卡内存不足），可以使用共享推理引擎来避免这一问题 (`bExclusiveSession = FALSE`)。然而，对于有状态模型（如循环模型），应避免这种配置。

6 API

6.1 功能块

6.1.1 FB_MlSvrPrediction

`FB_MlSvrPrediction` 是一个 TcMlServer 客户端，为 PLC 提供异步和可选的硬件加速 AI 模型推理。

该功能块位于 **PLC 库 Tc3_MlServer**。

语法

声明：

```
fbMlSvr : FB_MlSvrPrediction;
```

定义：

```
FUNCTION_BLOCK FB_FTR_IIRCoeff
VAR_INPUT
    stPredictionParameter : ST_PredictionParameter;
END_VAR
VAR_OUTPUT
    bError                : BOOL;
    nErrorCode            : HRESULT;
    bConfigured           : BOOL;
    nMaxInferenceDuration : UDINT;
END_VAR
```

输入

名称	类型	描述
stPredictionParameter [▶ 36]	ST_PredictionParameter	Machine Learning Server 会话的配置结构

输出

名称	类型	描述
bError	BOOL	如果当前挂起的异步请求到 TcMLServer 因错误而被取消，则为 TRUE。
nErrorCode	HRESULT	当前挂起的异步请求到 TcMlServer 的返回代码 [▶ 44]。
bConfigured	BOOL	如果配置成功，则为 TRUE。表示功能块是否与 TcMlServer 有一个有效会话。如果为 FALSE，则必须调用 "configure"。请注意：由于在 TcMlServer 执行请求时出错，bConfigured 可能会变为 FALSE。
nMaxInferenceDuration	UDINT	执行推理所需的最大 PLC 循环数。另请参见 执行 AI 模型 [▶ 22]。

方法

名称	定义定位	描述
Configure [▶ 33]	本地	根据 stPredictionParameter [▶ 36] 中定义的配置，在 TcMlServer 上为 FB 实例创建会话。
Deconfigure [▶ 33]	本地	在 TcMlServer 上结束功能块的会话。服务器上已分配的资源将再次释放。
GetCustomAttribute_array [▶ 35]	本地	获取 AI 模型 ARRAY 类型的自定义属性

名称	定义定位	描述
GetCustomAttribute_bool [▶ 35]	本地	获取 AI 模型 BOOL 类型的自定义属性
GetCustomAttribute_fp64 [▶ 36]	本地	获取 AI 模型 LREAL 类型的自定义属性
GetCustomAttribute_int64 [▶ 36]	本地	获取 AI 模型 LINT 类型的自定义属性
GetCustomAttribute_str [▶ 36]	本地	获取 AI 模型 STRING 类型的自定义属性
Predict [▶ 34]	本地	向 TcMlServer 传送异步推理请求
PredictBatched [▶ 34]	本地	向 TcMLServer 传送异步捆绑推理请求

6.1.1.1 异步方法

`FB_MlSvrPrediction` 的异步方法在签名上的特点是接受一个 *timeout（超时）* 参数并返回一个表示异步请求执行状态的 *bool（布尔值）*。因此，PLC 程序必须等待异步方法执行完毕，并根据应用程序的要求配置 *timeout（超时）*，以便应用程序能够对 TcMlServer 的任何延迟做出适当响应。请注意，指定的 *timeout（超时）* 以 PLC 任务周期为单位。

还需要注意的是，异步请求的处理只能以 **PLC 任务周期时间**的时间粒度进行监控。因此，延迟时间预算较低的应用程序必须尽可能缩短循环时间，并相应提高采样率，以便最大限度地减少由时间分辨率引起的延迟。这一点在与其他计算密集型同步执行算法交互时尤为重要，例如在预处理或后处理数据时。

然而，PLC任务的最小周期时间受限于需要从客户端传输到服务器的数据量。传输大量数据（如大图像数据）时，必须配置 PLC 任务循环时间，以防止循环超时。

6.1.1.1.1 Configure()

调用方法 *configure* 会在 TcMlServer 中为功能块 `FB_MlSvrPrediction` 的相应实例实例化一个会话。FB 成员 `stPredictionParameter` 中定义的配置用于实例化。

会话实例化可能需要很长时间，特别是因为要进行多次推理来预热推理引擎 *（模型预热）*。在选择方法调用的超时参数时，应考虑到这一事实。

还应考虑，在配置 CUDA 加速会话时，调用该方法暂时需要对 GPU 进行独占访问。因此，这种会话的配置会严重干扰其他并行运行的 FB 的推理性能。

该方法通过返回 TRUE 表示异步实例化调用处理完成后，可通过 FB 成员 bError 对结果进行评估，如果出现错误，则通过 nErrorCode 提供错误信息。一旦推理会话成功实例化，FB 就会将成员 `bConfigured` 设置为 TRUE。

另请参见 从 PLC 客户端配置服务器 [▶ 21]。

	参数	类型	默认	描述
输入	nTimeout	ULINT		返回超时错误前的 PLC 任务周期数。
输入	nPriority	UDINT	0	请求的优先级。值越大意味着优先级越高。
输出	Configure	BOOL		返回值。当异步调用的结果可用时立即返回 TRUE。然后可以使用 'bError' 和 'nErrorCode' 属性检查调用结果。

6.1.1.1.2 Deconfigure()

Deconfigure 方法会关闭 TcMlServer 中 FB 的推理会话。成功调用 Deconfigure 后，FB 可以使用 Configure 方法建立新的推理会话。

成功配置推理会话后，在调用 Deconfigure 之前，所有对 Configure 的调用都将被忽略。

	参数	类型	默认	描述
输入	nTimeout	ULINT		返回超时错误前的 PLC 任务周期数。
输出	Deconfigure	BOOL		返回值。当异步调用的结果可用时立即返回 TRUE。然后可以使用 'bError' 和 'nErrorCode' 属性检查调用结果。

6.1.1.1.3 Predict()

Predict 方法向 TcMlServer 提交异步推理命令。

该方法根据 ONNX 文件的规范提供输入数据，参见相关说明 为TwinCAT机器学习服务器准备ONNX文件 [▶ 18]。

所提供的输出数据指针必须有效，并根据创建的 PlcOpenXml 指向输出数据类型的实例。异步推理成功完成后，传输的输出内存区域中的数据就会生效，并被释放，以供进一步处理。

另请参见 执行 AI 模型 [▶ 22]。

	参数	类型	默认	描述
输入	pDataIn	PVOID		输入数据类型实例的指针
输入	nDataInSize	UDINT	0	输入数据类型的大小
输入	pDataOut	PVOID		输出数据类型实例的指针
输入	pDataOutSize	UDINT		输出数据类型的大小
输入	nTimeout	ULINT		返回超时错误前的 PLC 任务周期数。
输入	nPriority	UDINT	0	请求的优先级。值越大意味着优先级越高。
输出	Predict	BOOL		返回值。当异步调用的结果可用时立即返回 TRUE。然后可以使用 'bError' 和 'nErrorCode' 属性检查调用结果。

6.1.1.1.4 PredictBatched()

PredictBatched 方法向 TcMlServer 提交异步批量推理命令。

该方法根据 ONNX 文件的规格预计模型输入数据类型数组的提供情况，参见 为TwinCAT机器学习服务器准备ONNX文件 [▶ 18]。

所提供的输出数据指针必须有效，并根据创建的 PlcOpenXml 指向输出数据类型数组的实例。异步推理成功完成后，传输的输出内存区域中的数据就会生效，并被释放，以供进一步处理。

另请参见 执行 AI 模型 [▶ 22]。

	参数	类型	默认	描述
输入	pDataIn	PVOID		输入数据类型数组实例的指针
输入	nDataInSize	UDINT	0	输入数据类型的大小（元素的大小，而不是数组的大小）
输入	nBatchSize	UINT		批量的大小
输入	pDataOut	PVOID		输出数据类型实例的指针
输入	pDataOutSize	UDINT		输出数据类型的大小
输入	nTimeout	ULINT		返回超时错误前的 PLC 任务周期数。
输入	nPriority	UDINT	0	请求的优先级。值越大意味着优先级越高。
输出	PredictBatched	BOOL		返回值。当异步调用的结果可用时立即返回 TRUE。然后可以使用 'bError' 和 'nErrorCode' 属性检查调用结果。

6.1.1.2 同步方法

6.1.1.2.1 GetCustomAttribute_array

	参数	类型	描述
输入	sCustomAttributeName	T_MaxString	自定义属性的名称
输入	fmtAttributeDataType	引用 ETcMllDataType	自定义属性的数据格式
输入	pDataBuffer	PVOID	复制用户定义属性的目标数据缓冲区。
输入	nDataBufferLen	UDINT	目标数据缓冲区的长度（单位：字节）
输入	nArrayLength	引用 UDINT	用户定义属性中数据类型元素的数量（即 fp32 值的数量）。
输入	pnBytesWritten	UDINT 的指针	返回写入目标缓冲区的字节数。
输出	GetCustomAttribute_array	BOOL	如果方法中出现错误，则为 TRUE。

6.1.1.2.2 GetCustomAttribute_bool

	参数	类型	描述
输入	sCustomAttributeName	T_MaxString	自定义属性的名称
输入	nDataOut	引用 BOOL	Bool 类型自定义属性的输出值
输出	GetCustomAttribute_bool	BOOL	如果方法中出现错误，则为 TRUE。

6.1.1.2.3 GetCustomAttribute_fp64

	参数	类型	描述
输入	sCustomAttributeName	T_MaxString	自定义属性的名称
输入	nDataOut	引用 LREAL	自定义属性 fp64 的输出值
输出	GetCustomAttribute_fp64	BOOL	如果方法中出现错误，则为 TRUE。

6.1.1.2.4 GetCustomAttribute_int64

	参数	类型	描述
输入	sCustomAttributeName	T_MaxString	自定义属性的名称
输入	nDataOut	引用 LINT	自定义属性 int64 的输出值
输出	GetCustomAttribute_int64	BOOL	如果方法中出现错误，则为 TRUE。

6.1.1.2.5 GetCustomAttribute_str

	参数	类型	描述
输入	sCustomAttributeName	T_MaxString	自定义属性的名称
输入	nDataOut	引用 T_MaxString	枚字符串类型自定义属性的输出值
输出	pnStringLen	UDINT 的指针	（可选）字符串属性的实际长度
输出	GetCustomAttribute_string	BOOL	如果方法中出现错误，则为 TRUE。

6.2 数据类型

6.2.1 E_ExecutionProvider

枚举说明了 TcMlServer 支持的所有执行模式。

名称	类型	值	描述
CPU	USINT	0	在 CPU 上执行
CUDA	USINT	1	在支持 CUDA 的 NVIDIA® GPU 上执行

6.2.2 ST_PredictionParameter

TcMlServer 上推理会话的配置选项。

名称	类型	默认	描述
sMlModelFilePath	STRING(255)		创建的 JSON 文件的完整路径（参见 Model Manager [▶ 24]）
eExecutionProvider	E_ExecutionProvider [▶ 36]	ExecutionProvider.CPU	sMlModelFilePath 下命名的 AI 模型将在此指定硬件上执行。

名称	类型	默认	描述
nDeviceId	UDINT	0	使用 "CUDA" ExecutionProvider 时所需 GPU 设备的索引。该索引与 CUDA 计算索引对应，仅与具有多个 GPU 的 IPC 相关。
bExclusiveSession	BOOL	TRUE	确定在 TcMlServer 上为 FB 实例而创建的推理会话的独占性。如果为 TRUE，TcMlServer 会创建一个独占会话，这对于依赖状态性模型（如 RNN）来说是必要的，以避免干扰。如果为 FALSE，会话可以与其他请求相同配置的 FB 实例共享，这样可以减少内存负载。
nSessionTimeout	ULINT	72	TcMlServer 上 FB 会话的非活动超时持续时间（小时）。达到该时长后会话将过期，服务器将释放对应客户端已分配的资源。
sMlSvrNetId	T_AmsSvrNetId	'127.0.0.1.1.1'	可访问 TcMlServer 服务的设备的 AMS Net Id。默认为‘本地’。

7 示例

7.1 基于 AI 的图像处理

本示例演示了如何处理：

- 使用 TwinCAT Vision 从本地硬盘加载图像数据，并在 PLC 中提供这些数据。
- 使用 TwinCAT Vision 库对图像进行预处理。
- 使用 FB_MlSvrPrediction 在本地安装的 TwinCAT Machine Learning Server 上启动会话，并加载 AI 模型（分类模型）。
- 在 TwinCAT Machine Learning Server 上执行推理。
- 继续在 PLC 中处理结果。

文件下载和概览

您可在此处下载该项目：https://infosys.beckhoff.com/content/1033/TF3820_TC3_Machine_Learning_Server/Resources/17328164363.zip

- ZIP 中包含一个 **tnzip**，您可以在 TwinCAT XAE 中通过 *File > Open > Open Solution from Archive...* 打开。
- 模型文件夹包含一个 **ONNX** 以及已创建的 **JSON** 和 PlcOpenXml。
- 数据集文件夹包含待处理的**示例图像**。

要求

安装以下功能模块：

- TwinCAT 标准版
- TF3820 | TwinCAT Machine Learning Server
- TF3830 | TwinCAT Machine Learning Server Client
- TF7xxx | TwinCAT 3 Vision

项目搭建

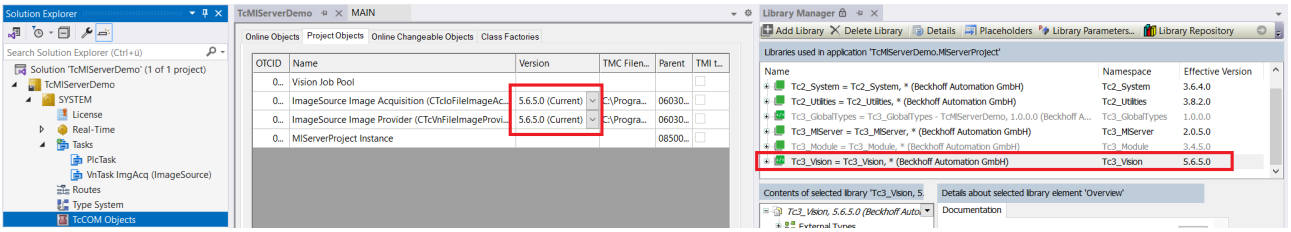
- 打开 tnzip 并保存项目。
- 设置 FileSpource：
  - 在 System Manager 中，选择树状项 *Vision > FileSource > ImageSource*
  - 添加模型文件夹中的图像。
  - 将 ImageSource 的循环时间设置为 100 ms。
- 在 MAIN 第 9 行中指定模型 JSON 的 FullPath。ONNX 必须在同一文件夹中。
- 确保所有软件至少有 7 天试用授权：
  - TC1200 TwinCAT PLC
  - TF3820 TwinCAT Machine Learning Server
  - TF7100 TwinCAT Vision Base

可选：TwinCAT Vision 版本的调整

该示例是在 5.6.5.0 版本下创建的。若想使用其他版本，请**手动**选择您已安装并希望使用的版本。为此，首先要选择相应的版本。右击 OTCID 列，从而打开上下文菜单。选择 “Reload TMI/TMC Description(s) with changed version”（用更改后的版本重新加载 TMI/TMC 说明）。另请参见 TwinCAT Vision 文档。

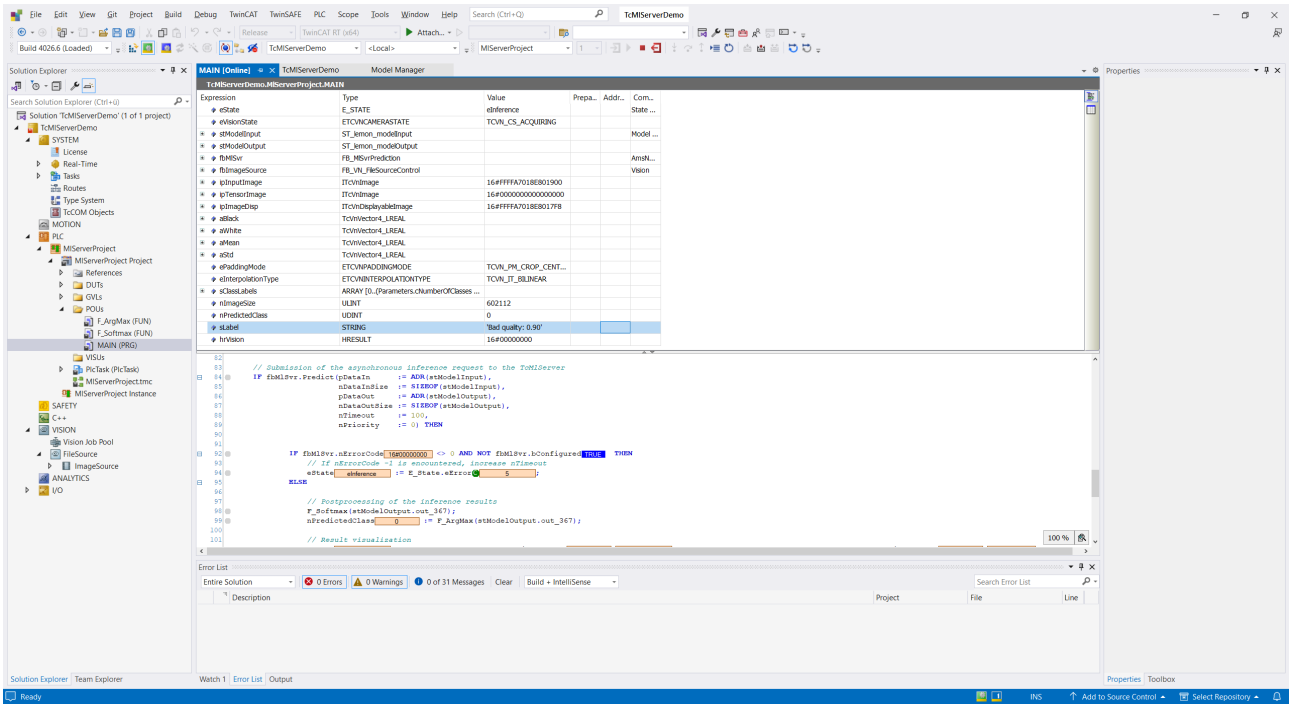
如果未安装版本 5.6.5.0，也未手动更改当前使用版本的设置，则会收到 “Error loading Repository driver”（加载存储库驱动程序时出错）错误消息。

系统将自动使用 PLC 库的最新版本。



运行项目

在目标系统上使用 Activate Configuration（激活配置）启动应用程序。如果所有配置均设置正确，稍后，`eState` 应设置为 `eInference`，变量 `sLabel` 应显示当前推理的结果。



如果发生错误，`eState` 将设置为 Error。然后，您可以打开 fbMlSvr 实例并读出错误代码。使用错误代码表 [▶ 44]缩小问题范围。

PLC 程序摘录

声明

在本声明中，关于处理 TwinCAT Machine Learning Server 处理的主要内容是 AI 模型的输入和输出数据类型，以及连接到 Machine Learning Server 的客户端实例。

```
stModelInput : ST_lemon_modelInput;      //model input datatype, imported via PlcOpenXml
stModelOutput : ST_lemon_modelOutput;    //model output datatype, imported via PlcOpenXml
fbMlSvr : FB_MlSvrPrediction();   // Instance of Client to TcMlServer
```

数据类型已通过 PlcOpenXml 读入 TwinCAT 项目（参见模型文件夹）。说明可在 DUTs 文件夹中找到。

会话配置

在本示例中，客户端在 E_State.eMlSvrConfiguration 状态下打开了 TwinCAT Machine Learning Server 上的一个会话。这指定了访问服务器的系统、会话中加载的模型以及执行模型的硬件。

```
fbMlSvr.stPredictionParameter.sMlModelFilePath := 'C:
\models\lemon_model.json';                          // fullpath to model
fbMlSvr.stPredictionParameter.sMlSvrNetId := '127.0.0.1.1.1';                                    //
Server on local system
fbMlSVr.stPredictionParameter.eExecutionProvider := E_EXECUTIONPROVIDER.CPU;
       // CPU execution

// Submit configuration request to the TcMlServer
// Provide a generous nTimeout, as the configuration can take a substantial amount of time
```

```
IF fbMlSvr.Configure(nTimeout := 1000, nPriority:=0) THEN
   IF fbMlSvr.nErrorCode <> 0 THEN
      // If nErrorCode -1 is encountered, increase nTimeout
      eState := E_State.eError;
   ELSE
      eState := E_State.eImageAcquisition;
   END_IF
END_IF
```

调用该方法 `Configure()` 会向服务器发送会话请求。该调用与 PLC 任务异步，当会话设置成功完成后，会确认为 `TRUE`。

执行模型

在状态 `E_State.eInference` 中，`Predict` 调用被发送到 Machine Learning Server。该调用相对于 PLC 任务也是异步的。如果结果可用，该方法返回 TRUE。

在本示例中，在调用推理之前，先使用 `ITcVnImage` 函数将类型 `F_VN_ExportImage` 的图像复制到模型输入数据类型中。

```
F_VN_ExportImage(ipTensorImage, ADR(stModelInput.in_input1), nImageSize, hrVision);
// Submission of the asynchronous inference request to the TcMlServer
IF fbMlSvr.Predict(pDataIn       := ADR(stModelInput),
                   nDataInSize   := SIZEOF(stModelInput),
                   pDataOut      := ADR(stModelOutput),
                   nDataOutSize := SIZEOF(stModelOutput),
                   nTimeout      := 100,
                   nPriority     := 0) THEN
      IF fbMlSvr.nErrorCode <> 0 AND NOT fbMlSvr.bConfigured  THEN
         // If nErrorCode -1 is encountered, increase nTimeout
          eState := E_State.eError;
      ELSE

         // Postprocessing of the inference results
         F_Softmax(stModelOutput.out_367);
         nPredictedClass := F_ArgMax(stModelOutput.out_367);
```

在成功完成错误检查后，可以使用推理的结果。

7.2 自定义属性

通过自定义属性，可以为 AI 模型提供在 PLC 运行时需要考虑的元数据。元数据概念同样存在于 ONNX 中，但这些元数据不能在 PLC 运行时进行评估。

原则上，您可以同时使用这两种元数据区域。需区分 PLC 运行时所需信息（如自定义属性区）与仅供 PLC 程序员在工程阶段使用的信息（如 ONNX 元数据或自定义属性）。

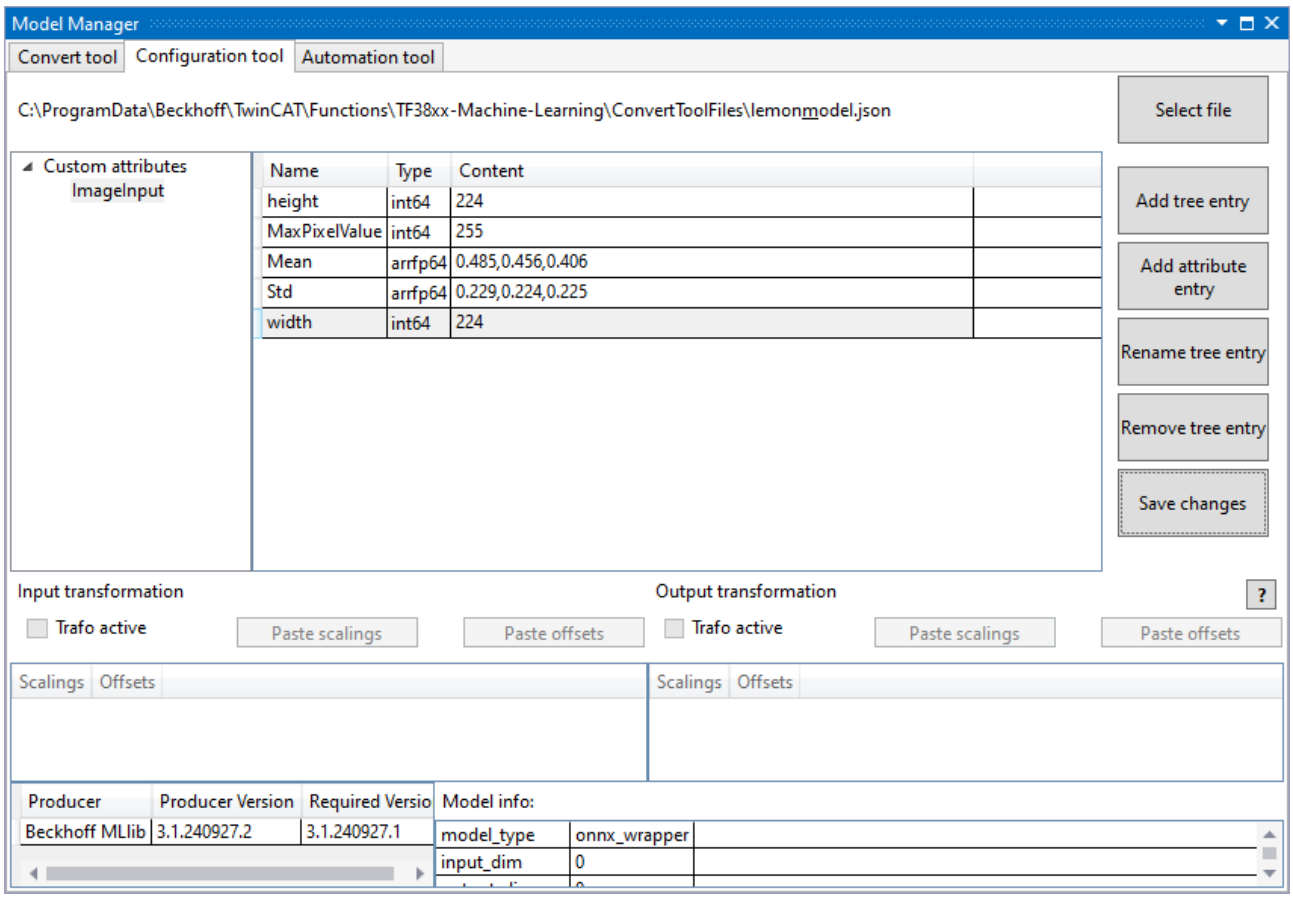
无论如何，目标是使 ONNX 的创建者能够高效地向 ONNX 的用户传递足够的信息，以便用户能够正确且地使用 AI 模型。

用于图像预处理的自定义属性

下面将扩展示例 基于 AI 的图像处理 [▶ 38]。添加了输入图像的相关信息。目的是根据元数据调整图像预处理。

在 TwinCAT Machine Learning Model Manager 中实现（TwinCAT机器学习模型管理器）

可在图形环境中使用 Configuration Tool（配置工具），输入自定义属性。

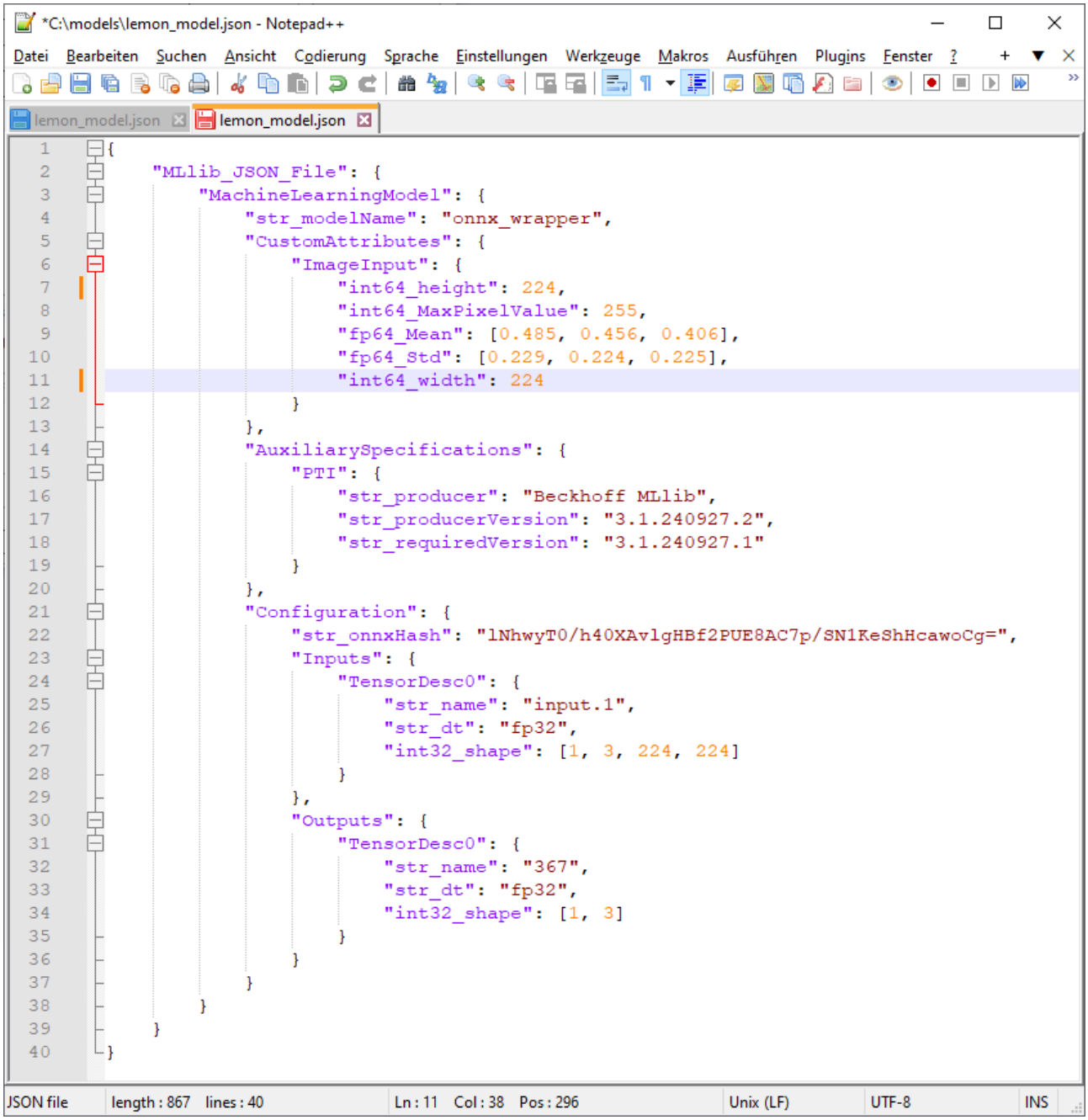


在 Python中实现

从工作流程的角度来看，在模型训练后直接在 Python 中输入所有所需的自定义属性通常更为方便。

```
new_ca = { 'ImageInput' : {'nwidth' : 244, 'nheight' : 244, 'nMaxPixelValue' : 255, 'fMean' : [0.485
, 0.456, 0.40], 'fStd' : [0.229, 0.224, 0.225]} }
tb.modify_ca("C:\\models\\lemon_model.json", "C:\\models\\lemon_model.json", new_ca)
```

结果可以在 JSON 中以纯文本形式追踪。



在 TwinCAT 中读取自定义属性

下面是 PLC 项目的扩展示例。

在 TwinCAT Machine Learning Server 上成功配置会话后，会立即读出存储的自定义属性。

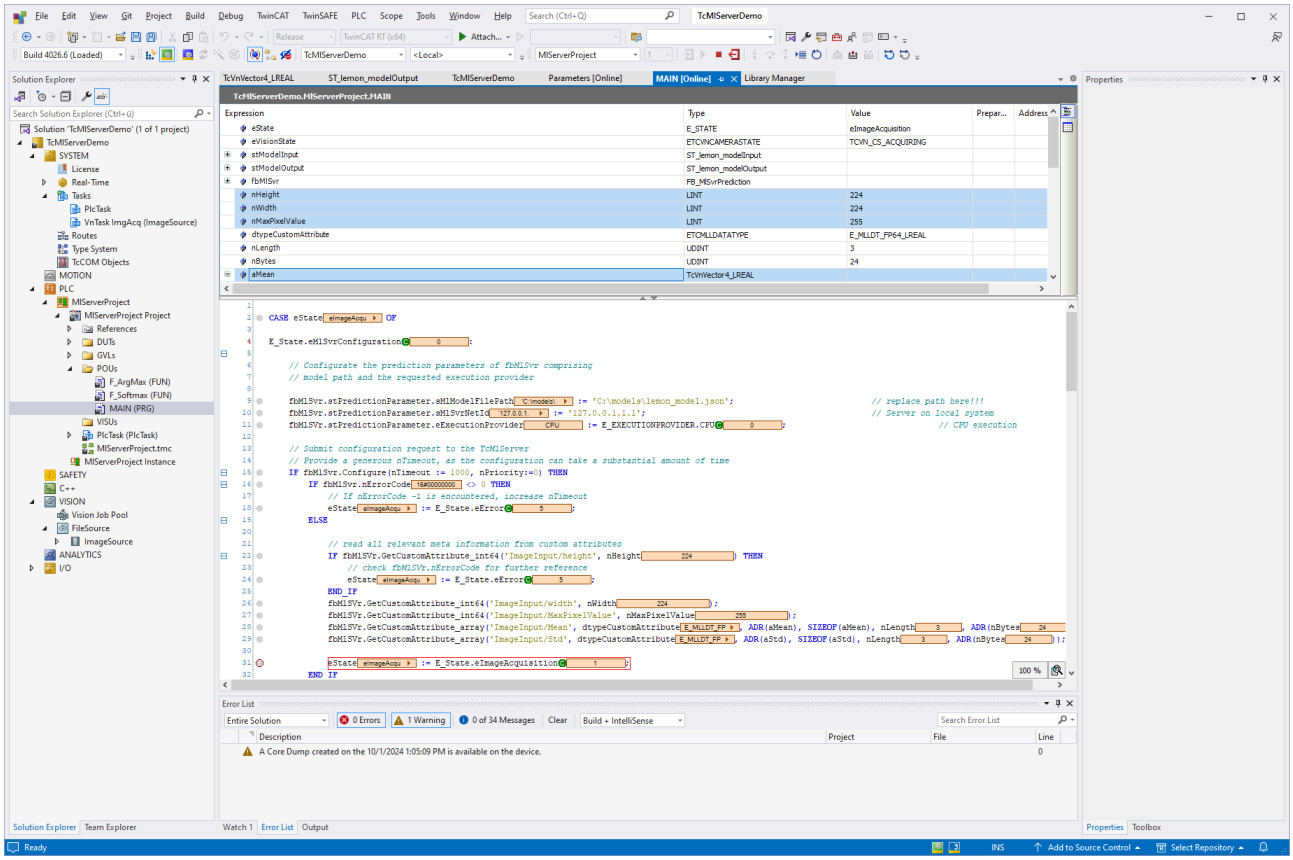
```
IF fbMlSvr.Configure(nTimeout := 1000, nPriority:=0) THEN
   IF fbMlSvr.nErrorCode <> 0 THEN
      // If nErrorCode -1 is encountered, increase nTimeout
      eState := E_State.eError;
   ELSE

      // read all relevant meta information from custom attributes
      IF fbMlSVr.GetCustomAttribute_int64('ImageInput/height', nHeight) THEN
         // check fbMlSVr.nErrorCode for further reference
         eState := E_State.eError;
      END_IF
      fbMlSVr.GetCustomAttribute_int64('ImageInput/width', nWidth);
      fbMlSVr.GetCustomAttribute_int64('ImageInput/MaxPixelValue', nMaxPixelValue);
      fbMlSVr.GetCustomAttribute_array('ImageInput/
Mean', dtypeCustomAttribute, ADR(aMean), SIZEOF(aMean), nLength, ADR(nBytes));
      fbMlSVr.GetCustomAttribute_array('ImageInput/
Std', dtypeCustomAttribute, ADR(aStd), SIZEOF(aStd), nLength, ADR(nBytes));

      eState := E_State.eImageAcquisition;
```

```
    END_IF
END_IF
```

然后，存储在 PLC 中的变量中的自定义属性随后被用于预处理流程中。基本上，这里需要考虑的只是必要的类型转换。

```
// Adjust the dimensions of the input image to match the model input requirements
hrVision := F_VN_ResizeImageExp(ipInputImage, ipTensorImage, LINT_TO_UDINT(nWidth), LINT_TO_UDINT(nH
eight), eInterpolationType, ePaddingMode, aBlack, hrVision);
// Convert the image to type REAL and scale to the range [0.0, 1.0]
hrVision := F_VN_ConvertElementTypeExp(ipTensorImage, ipTensorImage, TCVN_ET_REAL, 1.0 / LINT_TO_REA
L(nMaxPixelValue), 0, hrVision);
// Normalization
hrVision := F_VN_SubtractVectorFromImage(ipTensorImage, aMean, ipTensorImage,hrVision);
hrVision := F_VN_DivideImageByVector(ipTensorImage, aStd, ipTensorImage, hrVision);
```

激活配置后，可以使用 Online View（在线视图）检查是否正确应用了数值。



8 附录

8.1 Error Codes

Error Code (dec)	Error Code (hex)	Description	Hints
0	0000	Operation successful	-
-1	FFFF	ML server timeout	Increase the timeout argument of your request. Check, that the TcMlServer service is running.
-3	FFFD	ML server malformed data	-
-5	FFFB	PLC protocol error	Make sure to call the methods of the FB in a canonical order, i.e. make sure the FB is configured before making inference calls.
-8	FFF8	Invalid AMS Net ID	Check the syntactical correctness of the provided AmsNetId.
-9	FFF7	Input pointer is null	Check the validity of the input data pointer provided to the predict methods.
-11	FFF5	Output pointer is null	Check the validity of the output data pointer provided to the predict methods.
-13	FFF3	Batch size is zero	Provide a valid batch size (>= 1).
-15	FFF1	Input size is zero	Provide a valid input data size (>0).
-17	FFEF	Output size is zero	Check your installation of TF3830.
-18	FFEE	Driver instantiation failed	Check your installation of TF3830.
-20	FFEC	Driver state propagation to state OP failed	Check the validity and integrity of your model file (existing path, valid file format).
-22	FFEA	Driver loading failed	Check your installation of TF3830.
-24	FFE8	Driver state depropagation failed	Check your installation of TF3830.
-26	FFE6	Driver parameter configuration failed	TwinCAT-internal error
-28	FFE4	Could not instantiate file accessor	TwinCAT-internal error
-30	FFE2	Could not propagate file accessor state	TwinCAT-internal error
-32	FFE0	Could not access model file	Check the validity of your model file (existing path, valid file format).
-34	FFDE	Mismatch in read model file bytes	TwinCAT-internal error
-36	FFDC	Could not parse model file	Check the integrity of your model file.
-38	FFDA	String buffer overflow	Check the integrity of your model file, especially the length of the contained model hash.

Error Code (dec)	Error Code (hex)	Description	Hints
-40	FFD8	Could not find model config in file	Check the integrity of your model file, especially the defined model configuration.
-42	FFD6	Could not retrieve model attributes	Check the integrity of your model file, especially the availability of a model hash.
-43	FFD5	ADS server connection error	ADS messages could not be sent. Check integrity of AmsRouter.
-44	FFD4	Engine mismatch	The model file you intended to load is not designated for the TcMlServer.
-45	FFD3	Invalid CST attribute name	Check the name of the attribute in your model file or your query. The name must not be empty.
-47	FFD1	CST attribute retrieval failed	The queried attribute could not be found. Check the queried attribute name in query and model file.
-49	FFCF	CST attribute invalid buffer	Check the destination pointer provided to the retrieval function. Must not be a null pointer.
-50	FFCE	Versioned model could not be resolved	Check the valditiy of the model version you are querying.
-51	FFCD	If a PLC online change is detected, a running predict is discarded with the error	
-101	FF9B	Invalid PLC request variant	Internal error
-102	FF9A	Invalid device index	Verify the device index provided in prediction parameters of the FB.
-103	FF99	Invalid request data	Internal error
-201	FF37	Invalid data	Internal error
-202	FF36	Invalid model type	Requested tensor of unsupported datatype. See log files.
-206	FF32	Invalid model	The provided model file is not supported. check especially, that no dimension parameters are left except for an optional batch dimension.
-300	FED4	Invalid execution provider	The requested execution provider is invalid. Make sure to select an EP from the enum E_ExecultionProvider.
-302	FED2	Unsupported execution provider	The requested execution provder is valid but not supported on your system (for instance due to a lack of GPU support).
-400	FE70	Session config file not found	Internal error
-402	FE6E	Environment config file not found	Internal error
-404	FE6C	Run config file not found	Internal error
-406	FE6A	Server config file not found	Internal error

Error Code (dec)	Error Code (hex)	Description	Hints
-408	FE68	Device config files not found	Internal error
-500	FE0C	Invalid session config file	Internal error
-502	FE0A	Invalid environment config file	Internal error
-504	FE08	Invalid run config file	Internal error
-506	FE06	Invalid server config file	Internal error
-508	FE04	Invalid device config files	Internal error
-600	FDA8	Targeted inference provider unavailable	Internal error
-602	FDA6	Inference provider session timeout	Increase the session timeout in the prediction parameters.
-701	FD43	Dynamic loaded library not found	Internal error
-703	FD41	Symbol not loadable from dynamic library	Internal error
-801	FCDF	Unknown job execution error	Internal error
-803	FCDD	Unknown error	Internal error
-901	FC7B	Failed backend execution	Internal error
-1100	FBB4	License server connection error	Internal licensing error
-1102	FBB2	License server ADS error	Internal licensing error
-1104	FBB0	License ADS error	Internal licensing error
-1106	FBAE	License invalid	Make sure, that a license TF3820 is available.
-1108	FBAC	License invalid unknown	Internal licensing error
-1201	FB4F	Expected tensor collection type	Make sure, that only types defined in the PLCopen files generated by the Beckhoff model management products is used. Do not manipulate the contained header
-1203	FB4D	Mismatching tensor collection hashes	Make sure, that only types defined in the PLCopen files generated by the Beckhoff model management products is used. Do not manipulate the contained header
-1205	FB4B	Invalid tensor collection header	Make sure, that only types defined in the PLCopen files generated by the Beckhoff model management products is used. Do not manipulate the contained header
-1207	FB49	Tensor collection offset feature not supported	Make sure, that only types defined in the PLCopen files generated by the Beckhoff model management products is used. Do not manipulate the contained header
-1209	FB47	Internal allocation special type header	Internal error
-1300	FAEC	Server startup cannot build log directory	No Logs directory available and could not be generated.
-1801	F8F7	NVIDIA NVML library function failure	Make sure the Nvidia nvml.dll is avilable.

Error Code (dec)	Error Code (hex)	Description	Hints
-1803	F8F5	NVIDIA NVML library extraction failure	Make sure the Nvidia nvml.dll is avilable.
-1901	F893	NVIDIA CUDA library function failure	Make sure the Nvidia cuda.dll is available.
-1903	F891	NVIDIA CUDA library extraction failure	Make sure the Nvidia cuda.dll is available.
-2000	F830	Model registry file storage error	Internal error
-2002	F82E	Model registry file deletion error	Internal error
-2003	F82D	Model registry invalid reference	Internal error
-2004	F82C	Model registry entry failed to load model	Internal error
-2006	F82A	Model registry inconsistent model hashes	Internal error
-2008	F828	ADS could not open remote file	Make sure, that an .onnx file with the same name as your .json model description file is located in the same directory.
-2010	F826	ADS could not retrieve size of remote file	Internal error
-2012	F824	ADS could not load remote file	Make sure, that your AMS router has sufficient memory for your .onnx file.
-2014	F822	ADS could not close remote file	Internal error
-2018	F81E	ADS could not open client port	Make sure your TwinCAT installation is valid.
-2201	F767	Buffer base is null	Internal error
-2203	F765	Not enough memory for head	Internal error
-2205	F763	Buffer out of memory	Internal error
-2207	F761	Buffer cannot read from null	Internal error
-2209	F75F	Buffer cannot write to null	Internal error
-2211	F75D	Buffer corrupted with invalid string	Internal error
-2020	F81C	Insufficient AMS Router Memory on Client	Increase Router Memory on the Client system
-2022	F81A	Insufficient AMS Router Memory on Server	Increase Router Memory on the Server system
-2024	F818	Could not read router memory on client	Internal error
-2026	F816	Could not read router memory on server	Internal error
-2301	F703	Invalid CUDA setup detected	Check your CUDA and cuDNN installation.

8.2 日志文件

TwinCAT Machine Learning Model Manager 日志：C:\ProgramData\Beckhoff\TwinCAT\Functions\TF38xx-Machine-Learning\Logs

TwinCAT Machine Learning Server 日志：C:\ProgramData\Beckhoff\TwinCAT\Functions\TF38xx-Machine-Learning\TcMlServer\Logs

8.3 Third-party components

This software contains third-party components.
Please refer to the license file provided in the following folder for further information:
C:\Program Files (x86)\Beckhoff\Legal\TwinCAT-XAR-MlServer
C:\Program Files (x86)\Beckhoff\Legal\TwinCAT-XAE-ModelManagerCore

8.4 技术支持和服务

倍福公司及其合作伙伴在世界各地提供全面的技术支持和服务，对与倍福产品和系统解决方案相关的所有问题提供快速有效的帮助。

下载搜索器

我们的下载搜索器包含我们供您下载的所有文件。您可以通过它搜索我们的应用案例、技术文档、技术图纸、配置文件等等。

可供下载的文件格式多种多样。

倍福分公司和代表处

若需要倍福产品的本地支持和服务，请联系倍福分公司或代表处！

倍福遍布世界各地的分公司和代表处地址可在倍福官网上找到：http://www.beckhoff.com.cn

该网页还提供更多倍福产品组件的文档。

倍福技术支持

技术支持部门为您提供全面的技术援助，不仅帮助您应用各种倍福产品，还提供其他广泛的服务：

- 技术支持
- 复杂自动化系统的设计、编程和调试
- 以及倍福系统组件的各种培训课程

热线电话：	+49 5246 963-157
电子邮箱：	support@beckhoff.com

倍福售后服务

倍福服务中心提供所有售后服务：

- 现场服务
- 维修服务
- 备件服务
- 热线服务

热线电话：	+49 5246 963-460
电子邮箱：	service@beckhoff.com

倍福公司总部

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

电话：	+49 5246 963-0
电子邮箱：	info@beckhoff.com
网址：	www.beckhoff.com

Trademark statements

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.

Third-party trademark statements

Microsoft, Microsoft Azure, Microsoft Edge, PowerShell, Visual Studio, Windows and Xbox are trademarks of the Microsoft group of companies.

NVIDIA, NVIDIA RTX and CUDA are trademarks of NVIDIA Corporation.

更多信息:
www.beckhoff.com/tf3820

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
电话号码: +49 5246 9630
info@beckhoff.com
www.beckhoff.com