RealTimeTester 系统配置指南

文档版本 1.0

目录

简单介绍

RealTimeTester 系统用于 AI 效果的评估，预览，技术调研等等作用。在我们需要做出大规模投入之前，RealTimeTester 可作可行性参考。对于投资者，技术 leader，测试员，程序员，数据处理员，都可以直接感受到 AI 识别。

RealTimeTester 更像是一个专业 AI 和非专业做人性化的对接工具：千言万语，不如实际体验，然后，你会产生想象力。

组成

RealTimeTester 以 CS 方式组成，后台是 HPC 计算服务，前台使用任意手机即可

RealTimeTester 的后台默认会包含在 Z-AI 的工具链中，系统所有的逻辑处理都在后台，前台只管把识别内容画出来，不需要对手机前台做介绍。

- 后台应用: **RealTimeModelTesterService.exe**
- 32 位安卓前台：**RealTimeModelTesterForMobile32.apk**
- 64 位安卓前台：**RealTimeModelTesterForMobile64.apk**

建模

Z-AI 建模工具链系统很完善，准备好 HPC/GPU 这类电脑，通过视频文档，即可入门建模。之后，使用模型 RealTimeTester 驱动模型查看效果，然后，发挥人类的巨大想象力。

应用

应用模型时，只需要在后台重新驱动几个 Z-AI 核心组件

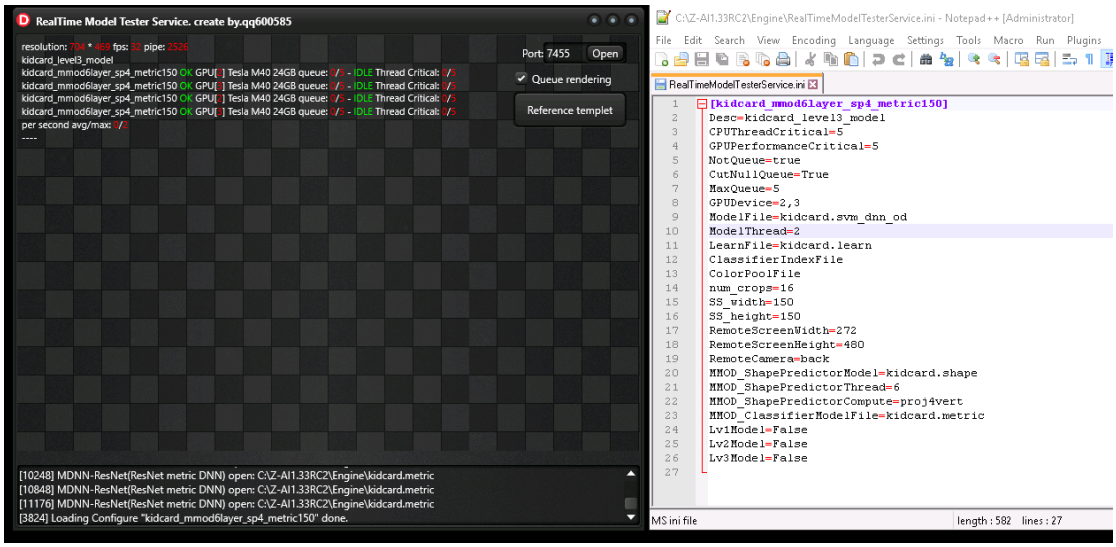
Queue+DNNThread+CS 通讯+FFMPEG，驱动完成这几个东西就可以应用于项目了，这花不了几天时间。不要跑去尝试修改 RealTimeTester 项目源码，编写 RealTimeTester 时并没有考虑支持 2 次开发。因为 RealTimeTester 内部有很多应用识别模式，导致复杂度很高，2 次开发并不现实。验证模型后，新写一个后台驱动反而简单。

RealTimeModelTesterService 后台脚本

所有的脚本以下列方式书写

[模型块的名字]
xxxx，参数

例图



抬头信息，以[xx]表示

显示在服务器状态抬头的信息，同时也会在手机端显示

Desc

模型描述信息，同时也会在手机端显示

CPUThreadCritical

CPU 的线程临界值：GPU 是异构系统，和 CPU 是分离运行的，当 GPU 处理完成一个任务后，系统会从线程池抽取一个线程触发对应的事件，该事件运行于 CPU，这个临界值表示最大可以同时触发 n 个事件给 CPU 运行。
GPU+CPU 要形成流水线，默认配置时建议给 GPUPerformanceCritical*2

GPUPerformanceCritical

GPU 的满载性能临界值：GPU 是异构系统，由 CPU 发送数据给 GPU 进行计算，当 GPU 处理不过来时，数据就会一直积压在服务器，该值表示最多积压 n 个任务，如果达到该临界，继续发送数据给 GPU 那么将会发生等待或则直接反馈数据无效。简单来说，该值主要是保护大规模运算流水线的稳定性，避免大数据计算让 GPU 爆炸。

NotQueue

前台的输入数据不排队，如果给 True，前台发来的数据工作机制是等待识别，如果给 False，前台的工作机制是排队，所有的视频流按 GPUPerformanceCritical 大小排队进行识别。
耗时识别大多是 RNIC/LRNIC/SEG 这类模型，这类模型大多 0.1 秒才能计算完成，不应该排队。
实时识别的模型包括：MMOD6L/MMOD3L/METRIC/LMETRIC/GDCNIC/GNIC，这类模型可以排队识别，但是当 GPU 流水线不流畅出现性能瓶颈时，不排队即可。

排队和不排队的重要差异

排队处理的识别会形成线性数据，线性数据为统计学计算提供了重要参考，例如人脸识别中的活体检测，如果没有线性数据，机器会很难分别是照片还是真人，换句说话，在线性的人脸数据中，人类会眨眼，会动，它会被 AI 系统以线性方式捕获进来识别。

CutNullQueue

识别的结果都存放于队列序结构，该参数表示在识别完成后，如果发现识别不成功，例如对象检测器没有找到目标，这时候是否自动剪掉之前的识别结果。
以下图为例，如果 A 识别失败，那么在 B 完成后，A 和 A 前面的数据将全部被剪掉

该参数为 AI 监控这类系统提供了响应机制，例如，找到某人，开始进行程序，某人离开以后，进入低功耗的待机模式。
另外，也会对线性统计学系统支持，在线性统计中，需要所有的数据都有有效的。

MaxQueue

该参数表示最大保留的识别结果是 5 个，例如，统计学需要使用 5 个结果计算出平均统计结果，那么永远都会有个识别结果输入给统计程序。
统计程序是很有必要的存在，因为现实生活中，AI 识别并不能保证非常准确，因此对识别结果使用统计学支持是提高准确性的做法之一

GPUDevice

以逗号分隔的 ID 表示使用哪些 GPU 设备，查看 GPU 设备 ID 使用 GPU-Z 或则 nvidia-smi 来干

ModelFile

ModelFile 是个自动化的模型文件驱动，模型格式支持如下

Metric: 文件扩展名".metric"

Metric 是度量化网络，训练时候会做弱推理，高速训练
给.Metric 扩展需要附带 LearnFile 值，既 KDTree 模型，该模型在训练网络时会自动生成后缀是.Learn 的模型文件

LMetric: 文件扩展名".large_metric"

LMetric 是大规模度量化网络，K 向量表达空间相比 Metric 容量更宽，LMetric 没有推理功能，高速训练
给.LMetric 扩展需要附带 LearnFile 值，既 KDTree 模型，该模型在训练网络时会自动生成后缀是.Learn 的模型文件

MMOD6L: 文件扩展名".svm_dnn_od"

MMOD6L 是 6 层检测器网络，训练时可以中等推理，以凸裁剪推理为主，例如，口和田的汉字图像，差异在里面的十字，检测器无法推理到内部差异，但是分辨 1 和 2 这种汉字图像，检测器可以推理学会，建模时不推理是中等训练速度，推理以后训练将会非常慢，动则若干小时，步数破百万

MMOD3L: 文件扩展名".svm_dnn_od_3L"

MMOD3L 是 3 层检测器网络，功能与 MMOD6L 一致，检测和训练速度更快

RNIC: 文件扩展名".RNIC"

RNIC 是强推理的残差网络分类器，强调整张照片局部特征，相当于场景识别
RNIC 的额定分类数量是 1000，相当于它能识别 1000 种场景
给.RNIC 扩展需要附带 ClassifierIndexFile 值，既分类表，该模型在训练网络时会自动生成后缀是.index 的模型文件

LRNIC: 文件扩展名".LRNIC"

LRNIC 是中等推理能力的大型残差网络分类器，相当于场景识别
LRNIC 的额定分类数量是 10000，相当于它能识别 10000 种场景
给.LRNIC 扩展需要附带 ClassifierIndexFile 值，既分类表，该模型在训练网络时会自动生成后缀是.index 的模型文件

GDCNIC: 文件扩展名".GDCNIC"

GDCNIC 来自论文“Going Deeper with Convolutions”的改进分类器网络，没有推理功能，训练速度极快，例如 10W 规模的手写数字识别只需要 5 分钟
GDCNIC 更适合识别大量的样本，例如 1 级中文字库
GDCNIC 的额定分类为 10000 个，不支持多 GPU 训练
给. GDCNIC 扩展需要附带 ClassifierIndexFile 值，既分类表，该模型在训练网络时会自动生成后缀是.index 的模型文件
给. GDCNIC 扩展需要附带 SS_width+SS_height 值，既输入尺度，这个尺度在训练时会指定好，在 TesterService 中需要和训练时一致。

GNIC: 文件扩展名".GNIC"

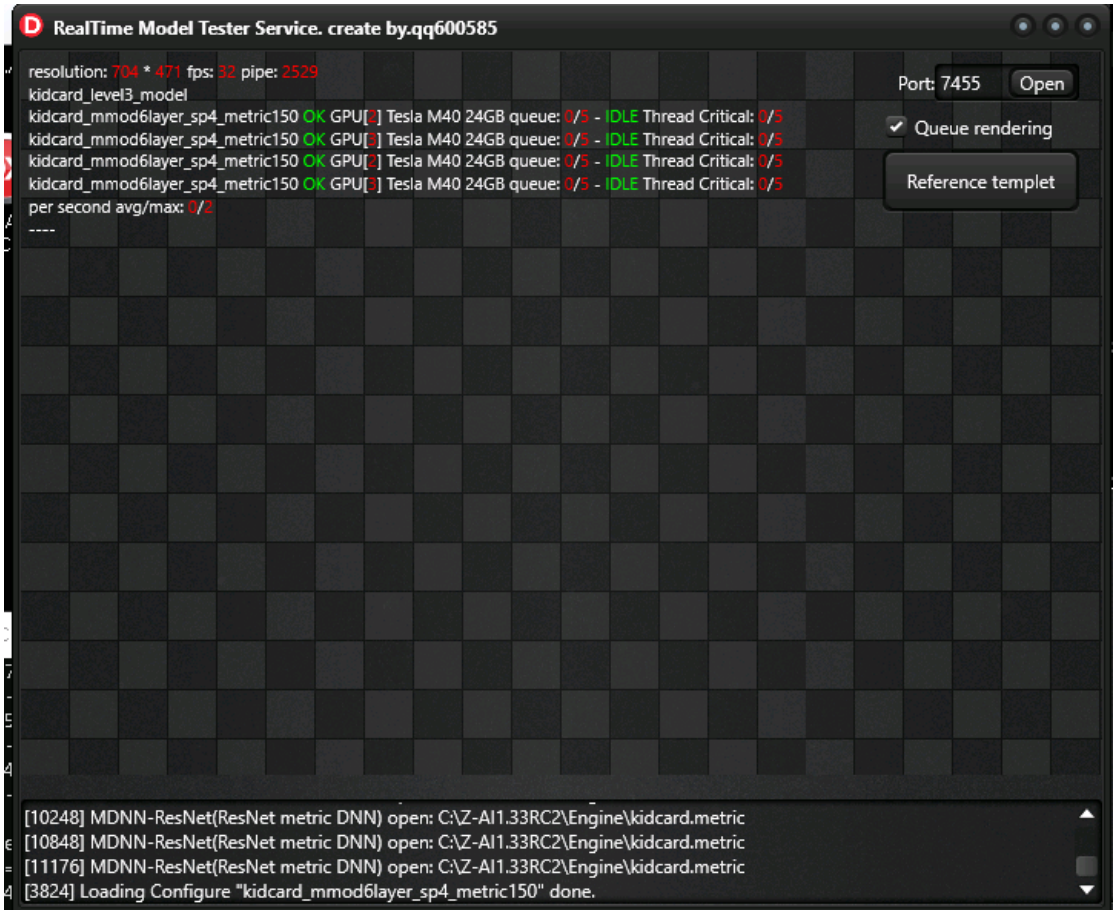
GNIC 与 GDCNIC 类似，GNIC 是 20 年前设计的 OCR 分类器网络，后来翻译成了 GPU 版本，没有推理功能，训练速度极快，例如 10W 规模的手写数字识别只需要 5 分钟
GNIC 更适合识别大量的样本，例如 1 级中文字库
GNIC 的额定分类为 10000 个，不支持多 GPU 训练
给. GNIC 扩展需要附带 ClassifierIndexFile 值，既分类表，该模型在训练网络时会自动生成后缀是.index 的模型文件
给. GDCNIC 扩展需要附带 SS_width+SS_height 值，既输入尺度，这个尺度在训练时会指定好，在 TesterService 中需要和训练时一致。

SS: 文件扩展名".SS"

分割器网络，数据科学家使用的模型
SS 设计从最底层出发，直接采用像素级训练，因此 SS 有替代 Z-AI 所有模型的能力，SS 训练速度极慢，驱动 SS 分割器对 GPU 的硬件要求偏高，同时需要专业标注人员和专业系统支撑。
给. SS 扩展需要附带 ColorPoolFile 值，既分类颜色表，在训练网络时会自动生成后缀是. colorPool 的分类颜色表文件

ModelThread

每个 GPU 设备启动的模型线程，例如，
GPUDevice=2,3
ModelThread=2
那么 TestService 启动以后，将会在 GPU2/GPU3 上分别载入 2 个模型，总共 4 个模型来处理识别流水线，如下图



LearnFile

如果模型是 Metric 或则 LMetric，必须要给 LearnFile，这是与度量化网络相互对应的 KDTree 模型。

ClassifierIndexFile

如果模型是 RNIC/LRNIC/GDCNIC/GNIC，必须要给 ClassifierIndexFile，这是分类器的分类表，这个文件在训练模型时会自动生成。

ColorPoolFile

如果模型 SS 语义分割器，必须要给 ColorPoolFile，这是语义像素的分类颜色表，在训练网络时会自动生成后缀是. colorPool 的分类颜色表文件

num_crops

当模型为 RNIC/LRNIC 时候，num_crops 表示照片的拆分数，也也可以理解成识别深度
默认为 16，它的工作含义是，将照片随机裁剪成 16 个小像素块，然后给 RNIC/LRNIC 网络识别，最后取平均统计值，如果照片很大，例如 1080p，那么将会被拆分成 16 张小图，然后识别 16 次。一般来说，该值不要超过 64，不要低于 8。

SS_width

这是泛参数，GDCNIC/GNIC/SP，这类模型，都会使用它

SS_height

这是泛参数，GDCNIC/GNIC/SP，这类模型，都会使用它

RemoteScreenWidth

直接控制手机端的屏幕分辨率，也是输入到 TesterService 做识别的分辨率，270*480 表示 270p，这是很小的光栅，几乎不怎么消耗宝贵的网络流量。

RemoteScreenHeight

直接控制手机端的屏幕分辨率，也是输入到 TesterService 做识别的分辨率，270*480 表示 270p，这是很小的光栅，几乎不怎么消耗宝贵的网络流量。

RemoteCamera

手机端的实时摄像头捕获方式，back 表示后置摄像头，front 表示前置摄像头

MMOD_ShapePredictorModel

自动化的 2 级 SP 模型文件
MMOD 在检测出目标以后，检测结果数据会自动流向 SP 模型做 2 级别识别

MMOD_ShapePredictorThread

自动化的 2 级 SP 模型的工作线程数量，这个数量不应该超过 cpu 的核心总数，如果 CPU 带有 numa 节点，例如志强的多节点 CPU，这里给单核总数，否则 CPU 将会桥起来计算，一般来说，windows server 会自动调度，一个 app 使用 1 个物理 cpu，不会自动桥

MMOD_ShapePredictorCompute

自动化的 2 级 SP 模型到 3 级分类器模型的计算函数接口
这里给的值是 proj4vert，表示将 4 个顶点以 2 个三角方式重新投影生成一张规范光栅
这里的值也可以直接不给，如果 sp 模型反馈 4 个顶点，它也会自动投影
如果 sp 模型反馈出的是银行/支付宝这类平台使用的人脸规范算法，它会自动对齐人脸，然后进入 3 级分类器模型识别
简单来说，该值是自动化的，可以投影对齐 4 顶点，也可以投影人脸

MMOD_ClassifierModelFile

3 级别分类器模型
这里如果是 Metric/LMetric 模型，需要指定 LearnFile
这里如果是 RNIC/LRNIC/GDCNIC/GNIC 模型，需要指定 ClassifierIndexFile

Lv1Model

强制 1 级识别模型处理程序

Lv2Model

强制 2 级识别模型处理程序

Lv3Model

强制 3 级识别模型处理程序