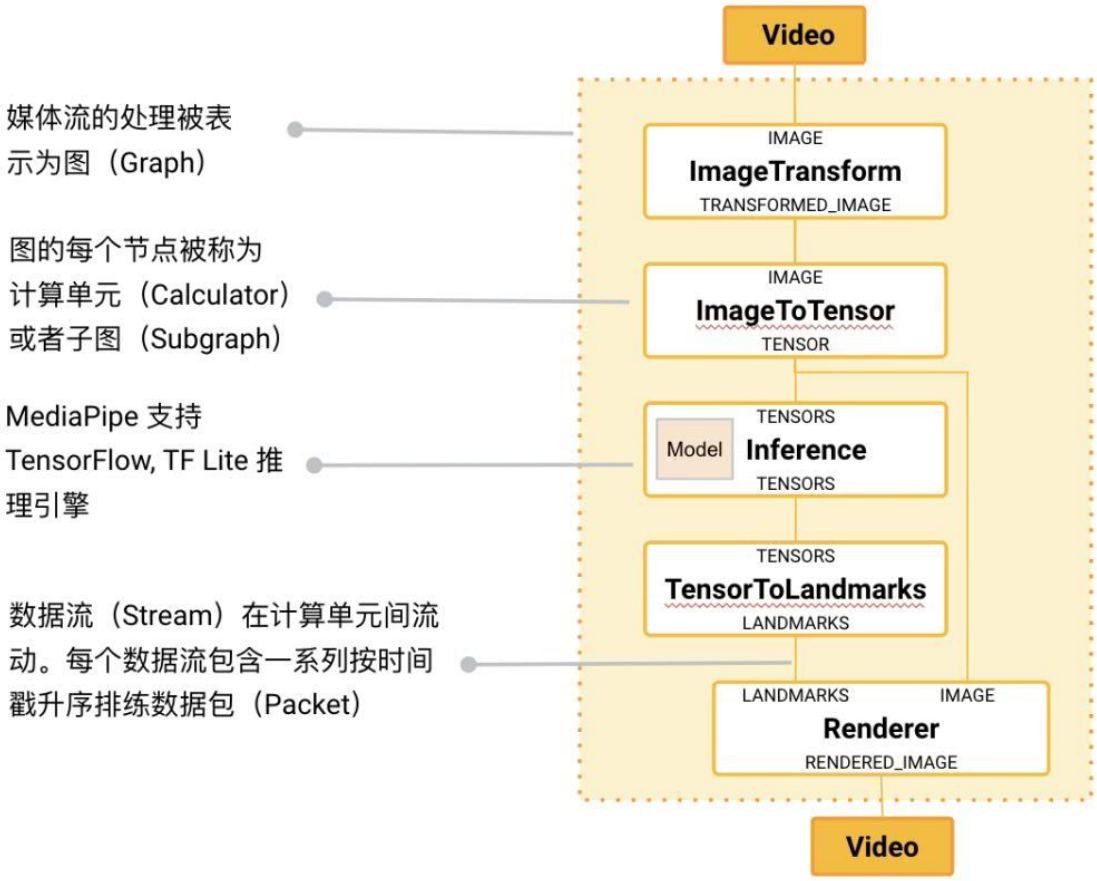
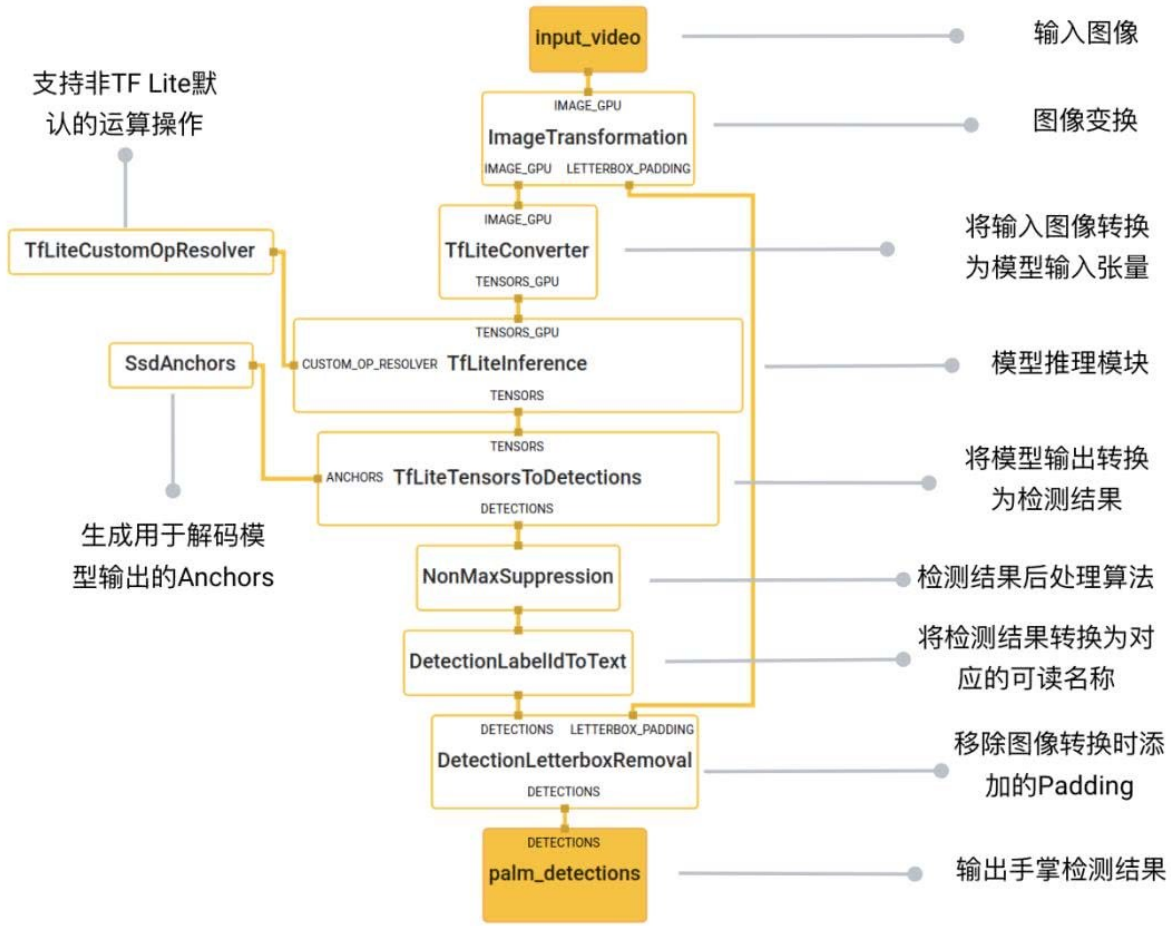
AI学习笔记--MediaPipe--概念

上一篇我们简单介绍了一下 mediapipe 从 clone 到 build 的全部过程，接下来我们屡下 MediaPipe 当中的概念性名词。大致的流程：



上图是大致的一个框架结构，可以看到几个基本的构成元素。其中 ImageTransformationCalculator 将输入的图像调整到模型可以接受的尺寸，用以送入 TF Lite 模型的推理模块；使用 TfLiteTensorsToDetectionsCalculator，将模型输出的 Tensor 转换成检测结果；运用 NonMaxSuppressionCalculator 等计算单元做后处理；最终从 HandDetection子图输出检测结果给主图。



The basics

- Packet

MediaPipe当中的基本数据格式，每一个 Packet 当中都包含了一个时间戳和一个共享的 share 指针，用于指向对应的物理内存地址数据块。这个数据块返回的是 void*，可以被强制转换成任意的数据类型。并且，Packet 也包含了对应数据对象说明的 Type 以便于对 Packet 进行描述。每一个对象都有自己的对象所述空间，并且都 copy 了一段时间戳以便于上层操作。

Packet 更详细的说明：
https://mediapipe.readthedocs.io/en/latest/packets.html

- Graph

MediaPipe 处理图形对象是在对应的图形中进行处理的。Graph 在 video和 Node 处理的过程中被定义。一个 Graph 对象可以在输入和输出过程中被拆分成任意的数据格式，也可以被组合打包合并。一般来说字节流的时间戳都是正向的，但是在 MediaPipe 中也可能存在倒置的情况。

- Node

Node 用于消费输入产生的 Packets 对象。他出现在整个图形运算的过程中，他们也可以被称为“Calculators”,由于早期 MediaPipe 的设计，每一个 Node 接口都具有一个Input 和 output 端口，并且使用一个 type 或者 index 来标记这些对象。

- Streams

在两个 Node 之间用于方阵Packet 交换的对象。数据包的时间戳必然要单调递增。

- Side packets

一个 Side Packets 是用于两个 Node 之间进行数据传输，交流的途径（Packet 并未定义时间戳）。他可以用来提供一些数据需要保存，但是 Packet 会随着Stream 变换的场景，

- Packet Ports

一个 Packet Port 具有一个关联性的 Type,Packets 之间进行传输必然会在 Port 中提供一个相关性的描述类型。一个 Output Stream Port 可以与任意一种相同类型的输入 Stream 进行连接通信。并且每个消费者都会接受一个输出 Packets 的 clone 对象，且他具有自己的方阵数据，所以消费者可以在他们的空间内消费这些数据。同理 Side Packets 的数据包。

一个Port 被定义出来后，必然会建立一个连接，并且可以确定的，一旦一个 graph 变为了空，则这个连接就会处于非连接的状态。

值得注意的是，一个 stream 连接被开启后，可能会有任意时间戳的 packet 被传输过来。

- Input and Output

Data数据可以来源于原始的Nodes，这些Node 没有固定输入源和生产 Packet ，可以自发地产生数据包(例如从文件中读取);或者来自图形输入流，它允许应用程序将数据包输入到Graph中。

类似地，有接收数据并将其写入不同目的地(例如文件、内存缓冲区等)的接收节点，应用程序也可以使用回调接收Graph输出。

Runtime behavior

- Graph lifetime

一旦一个Graph被初始化完成后，它就可以开始数据处理的流程，并且可以处理一个数据包流，直到每个用到的 stream被关闭或者Graph 对象调用了 cancled 方法。然后可以选择销毁 Graph 或重新启动Graph 处理。

- Node lifetime

框架在节点上调用的主要生命周期方法有三个:

- Open:在其他方法之前调用。当它被调用时，节点所需的所有 Side Packets都是可用的。
- Process:当一组新的输入产生后，根据节点的输入策略多次调用。
- Close:调用以后结束 Node。

此外，每个calculator都可以定义构造函数和析构函数，这对于创建和释放独立于处理数据的资源非常有用。

- Input policies

默认的输入策略是根据时间戳确定数据包的序列 。节点在调用其流程方法时，同时接收同一时间戳的所有输入;并按时间戳顺序接收连续的输入集。这可能需要延迟一些包的处理，直到在所有输入流上接收到具有相同时间戳的包，或者直到可以保证具有该时间戳的包不会到达没有接收到它的流为止。

其他策略也可用，使用一种称为InputStreamHandler的单独组件实现。