计算机视觉知识点汇总

ARM 优化

1. ARM Neon Intrinsics 学习指北：从入门、进阶到学个通透
2. 一份朴实无华的移动端盒子滤波算法优化笔记
3. Neon Intrinsics Reference

> `Neon` 是 `ARM` 平台的向量化计算指令集，通过一条指令完成多个数据的运算达到加速的目的，或者说 `Neon` 是 ARM 平台的 `SIMD`（Single Instruction Multiple Data，单指令多数据流）指令集实现。常用于 AI、多媒体等计算密集型任务。

CPU 硬件基础

1. 嵌入式系统 内存模块设计

数字图像处理

1. SIFT 特征详解
2. 使用ffmpeg库进行YUV420到RGB的转化
3. 图像的深度和通道概念区分

深度学习基础知识

1. CNN中参数解释及计算
2. 深度学习推理时融合BN，轻松获得约5%的提速
3. 动图形象理解深度学习卷积

矩阵乘优化

1. OpenBLAS gemm从零入门
2. 通用矩阵乘（GEMM）优化算法
3. 矩阵相乘在 GPU 上的终极优化：深度解析 Maxas 汇编器工作原理
4. 卷积神经网络中的Winograd快速卷积算法

> `Winograd` 是一种快速卷积算法，适用于小卷积核，可以减少浮点乘法的次数。

经典卷积网络

> 1，VGGNet 拥有 5 段 卷积，每一段有 2~3 个卷积层，同时每段尾部会连接一个最大池化层用来缩小图片尺寸，每段内的卷积核数量相同，越靠后的段的卷积核数量越多：64-128-256-512-512。ResNet 网络拥有 4 段卷积， 每段卷积代表一个 残差学习 `Blocks`，根据网络层数的不同， Blocks 的单元数量不同，例如 ResNet18 的 Blocks 单元数量分别为2、2、2 和 2。越靠后的段的卷积核数量越多：64-128-256-512，残差学习 `Blocks` 内的卷积核通道数是相同的。
> 2，ResNet v2 创新点在于通过理论分析和实验证明恒等映射对于残差块的重要性，根据激活函数与相加操作的位置关系，我们称之前的组合方式（ResNet）为“后激活（post-activation）”，现在新的组合方式（ResNet v2）称之为“预激活（pre-activation）”。使用预激活有两个方面的优点：1) `f` 变为恒等映射，使得网络更易于优化；2)使用 `BN` 作为预激活可以加强对模型的正则化。

1. 深度可分离卷积（Xception 与 MobileNet 的点滴）
2. [DL-架构-ResNet系] 002 ResNet-v2
3. ResNet及其变种的结构梳理、有效性分析与代码解读

神经网络量化与压缩

> 1，量化是指用于执行计算并以低于浮点精度的位宽存储张量的技术，或者说量化就是将神经网络的浮点算法转换为定点。 量化模型对张量使用整数而不是浮点值执行部分或全部运算。
> 2，量化简单来说就是将浮点存储（运算）转换为整型存储（运算）的一种模型压缩技术。
> 3，虽然精心设计的 `MobileNet` 能在保持较小的体积时仍然具有与 `GoogleNet` 相当的准确度，不同大小的 `MobileNet` 本身就表明——也许一个好的模型设计可以改进准确度，但同类模型中仍然是更大的网络，更好的效果!
> 4，权重值域调整是另一个机器学习过程，学习的目标是一对能在量化后更准确地运行网络的超参数 `min/max`。

1. 神经网络量化简介
2. 线性量化
3. Int8量化-介绍（一）
4. Int8量化-ncnn社区Int8重构之路（三）
5. ncnn源码学习（六）：模型量化原理笔记
6. 神经网络推理加速之模型量化
7. NNIE 量化感知训练

模型剪枝

论文总结 - 模型剪枝 Model Pruning

轻量网络设计方法

1. 轻量卷积神经网络的设计

> 网络结构碎片化更多是指网络中的多路径连接，类似于 `short-cut`，`bottle neck` 等不同层特征融合，还有如 `FPN` 等结构。拖慢并行的一个很主要因素是，**运算快的模块总是要等待运算慢的模块执行完毕。**

2. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design
3. ShufflenetV2 高效网络的4条实用准则
4. 轻量级神经网络：ShuffleNetV2解读

目标检测网络

1. 一文读懂Faster RCNN
2. 从编程实现角度学习Faster R-CNN（附极简实现）
3. Mask RCNN学习笔记
4. Mask RCNN 源代码解析 (1) - 整体思路
5. 物体检测之Focal Loss及RetinaNet
6. CVPR18 Detection文章选介（下）
7. 2020首届海洋目标智能感知国际挑战赛 冠军方案分享

评价指标

1. ROC和AUC介绍以及如何计算AUC

语义分割

1. 2019年最新基于深度学习的语义分割技术讲解
2. U-Net 论文笔记

3D 视觉

1. 3D视觉CV界的终极体现形式，计算机如何「看」这个三维世界

计算机视觉基础知识

Pytorch 框架

1. pytorch自定义层如何实现？超简单！
2. 【PyTorch】torch.nn.Module 源码分析
3. 详解Pytorch中的网络构造，模型save和load，.pth权重文件解析
4. 半小时学会 PyTorch Hook
5. 详解Pytorch中的网络构造
6. 深度学习与Pytorch入门实战（九）卷积神经网络&Batch Norm
7. Pytorch 里 nn.AdaptiveAvgPool2d(output_size) 原理是什么?

Caffe 框架

1. Message type "caffe.PoolingParameter" has no field named "ceil_mode"
2. Caffe Pooling层对ceil mode选择的支持
3. caffe源码解析-开篇

编程语言

1. http://www.cplusplus.com/reference/stl/

模型部署

1. 海思AI芯片(Hi3519A/3559A)方案学习（二十五）初识 mapper_quant 和mapper_param
2. 部署PyTorch模型到终端
3. 多场景适配，TNN如何优化模型部署的存储与计算
4. 模型转换、模型压缩、模型加速工具汇总
5. 深度学习模型转换与部署那些事(含ONNX格式详细分析)
6. ONNX初探

AI 芯片

1. 看懂芯片原来这么简单（二）：AI为什么聪明？什么是华为自研架构NPU？
2. 【专利解密】如何提高AI资源利用率？华为卷积运算芯片
3. 从GTX到RTX NVIDIA GPU架构的变迁史
4. CPU、GPU、NPU等芯片架构、特点研究
5. 什么是异构并行计算？CPU与GPU的区别是什么？

内存优化

1. 优化 TensorFlow Lite 推理运行环境内存占用

关于科研和研发

1. 中国人民大学赵鑫：AI 科研入坑指南

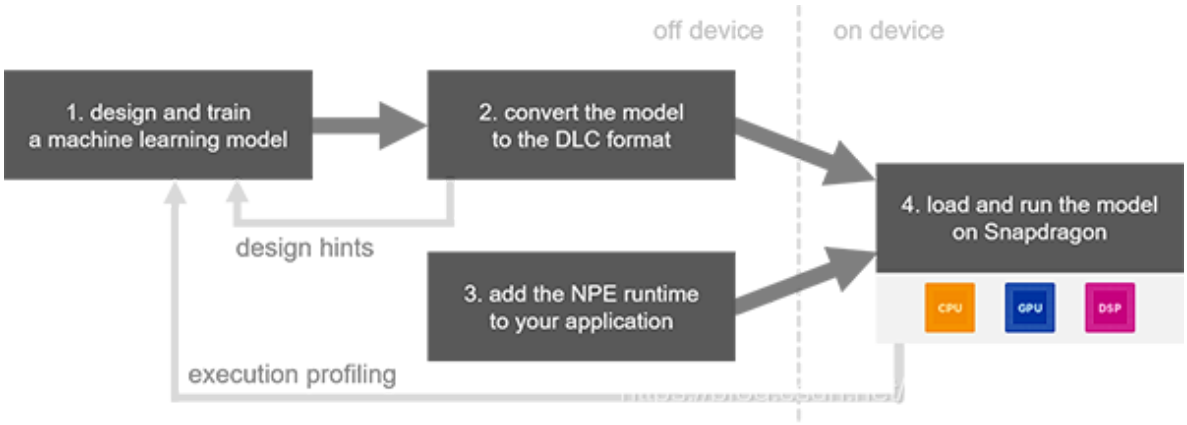
3D 视觉技术

1. 3D成像方法 汇总（原理解析）--- 双目视觉、激光三角、结构光、ToF、光场、全息
2. 关于双目立体视觉的三大基本算法及发展现状的总结

知识点输出

1，为了尽可能地提高 `MAC阵列` 的利用率以及卷积运算效率，阵列控制模块会根据第一卷积参数矩阵的行数和第一卷积数据阵列的行数来确定第一乘法累加窗口的列数。

2，`SNPE` 开发流程：



3，目标检测模型效果提升方法：

- 以 `Cascade RCNN` 作为 `baseline`，以 `Res2Net101` 作为 `Backbone`；
- `Albumentation` 库做数据集增强-用在模型训练中；
- 多尺度训练(`MST` Multi-scale training/testing)的升级版-`SNIP`方法(Scale Normalization for Image Pyramids)，用在 `baseline` 模型训练和测试中：解决模板大小尺度不一的问题；
- `DCN` 可变性卷积网络-用在 `baseline` 模型的 `backone` 中；
- `soft-NMS`：解决目标互相重叠的问题；
- `HTC` 模型预训练， `Adam` 优化算法可以较好的适应陌生数据集，学习率热身(`warm-up`)来稳定训练过程。

4，`SNIP` 论文解读：

`SNIP` 非常 `solid` 地证明了就算是数据相对充足的情况下，`CNN` 仍然很难使用所有 `scale` 的物体。个人猜测是由于 CNN 中没有对于 scale invariant 的结构，CNN 能检测不同 scale 的“假象”，更多是通过CNN 来通过 `capacity` 来强行 `memorize` 不同 `scale` 的物体来达到的，这其实浪费了大量的 `capacity`，而 `SNIP` 这样只学习同样的 scale 可以保障有限的 capacity 用于学习语义信息。论文的关键贡献：发现现在的 CNN 网络无法很好的解决 scale invariance 的问题，提出了一个治标不治本的方法。

5，高效模型设计（模型压缩）方法：

一般而言，高效模型的设计有 6 大基本思路：1）轻量级架构、2）模型裁剪、3）模型搜索、4）低精度量化、5）知识蒸馏、6）高效实现。

6，网络深度与宽度的理解及意义

> 更多理解参考知乎网络宽度对深度学习模型性能有什么影响？

在一定的程度上，网络越深越宽，性能越好。宽度，即通道(`channel`)的数量，网络深度，及 `layer` 的层数，如 `resnet18` 有 `18` 层网络。注意我们这里说的和宽度学习一类的模型没有关系，而是特指深度卷积神经网络的(通道)宽度。

- `网络深度的意义`：CNN 的网络层能够对输入图像数据进行逐层抽象，比如第一层学习到了图像边缘特征，第二层学习到了简单形状特征，第三层学习到了目标形状的特征，网络深度增加也提高了模型的抽象能力。
- `网络宽度的意义`：网络的宽度（通道数）代表了滤波器（`3` 维）的数量，滤波器越多，对目标特征的提取能力越强，即让每一层网络学习到更加丰富的特征，比如不同方向、不同频率的纹理特征等。

7，所有 `Inception` 模型都具有一个重要的性质——都是遵循 拆分-变换-合并（`split-transform-merge`） 的设计策略。

8，对于某种指令，延迟 `latency` 主要关注单条该指令的最小执行时间，吞吐量 `throughout` 主要关注单位时间内系统（一个CPU核）最多执行多少条该指令。因为 AI 计算的数据量比较大，所以更关注吞吐量。

9，CPU 高性能通用优化方法包括：

- 编译选项优化
- 内存性能和耗电优化：内存复用原则，小块快跑是内存设计的重要原则。
- 循环展开：循环的每次迭代都有一定的性能损失（分支指令）。但是现代 ARM 处理器具有分支预测的能力，它可以在执行条件之前预测是否将进入分支，从而降低性能损耗，这种情况下全部循环展开的的优势就减弱了。
- 并行优化和流水线重排：并行优化分为多线程核与核之间数据处理，以及单核心内部并行处理。从本质上讲，流水线重排也是一种并行优化。

10，卷积性能优化方式：卷积的计算方式有很多种，通用矩阵运算（`GEMM`）方式有良好的通用性，但是仅使用 GEMM 无法实现性能最优。除 GEMM 外，常用的优化方法还包括滑窗（`Sliding Window`）、快速傅里叶变换（Fast Fourier Transform, `FFT`）、`Winograd` 等。不同的方法适合不同的输入输出场景，最佳的办法就是对算子加入逻辑判断，将不同大小的输入分别导向不同的计算方法，以最合适的方法进行卷积计算。

- 大多数情况下，使用滑窗方法的计算性能还是无法和 `GEMM` 方法比较，但是一般当输入小于 32×32 时，可以考虑采用滑窗的优化方式。
- `Winograd` 是存在已久的性能优化算法，在大多数场景中，`Winograd` 算法都显示了较大的优势，其用更多的加法运算代替部分乘法运算，因为乘法运算耗时远高于加法运算。`Winograd` 适用于乘法计算消耗的时钟周期数大于加法运算消耗的时钟周期数的场景，且常用于 3×3 卷积计算中。对于 `CPU`，一般来说，一次乘法计算消耗的时间是一次加法计算消耗时间的 `6` 倍。
- `FFT` 方法不适合卷积核较小的 `CNN` 模型。