《Anatomy_of_High-Performance_Deep_Learning_Convolutions_on_SIMD_Architectures》笔记

zxp

December 23, 2023

1 论文内容

《SIMD架构上高性能深度学习卷积的剖析》，这篇论文的工作是优化直接卷积，这篇论文选择直接卷积的原因也是矩阵乘法的内存开销太大了还有矩阵乘法对内存带宽要求高，直接卷积没有这些开销。论文中的直接卷积也是在cpu上实现的。该论文还使用了一个编译策略降低编译花费的时间（这部分完全看不懂）

1.1 实现

论文中使用的优化有，向量化和寄存器阻塞，缓存阻塞和循环排。对融合乘加(FMA)操作进行矢量化，将这些操作放在最内层循环。然后寄存器堵塞和缓存堵塞都是为了使用寄存器和缓存的高速读取，然后提高数据复用，FMA指令有延迟，使用寄存器堵塞足以隐藏FMA指令的延迟（没用过FMA，不懂FMA也不懂寄存器堵塞，我猜测他是想将数据分成足够放入寄存器的小块）。然后和《高性能零内存卷积》一样依靠这些优化对卷积的循环重新排列了顺序
然后使用了卷积微内核的代码。这个微内核基本上有三个参数:一个指向由内核调用计算的输出子张量的指针，以及所需的输入子张量和权重子张量的相应指针。
论文指出，这些优化严重依赖运行的机器和卷积操作使用的输入张量的特征。这篇论文的目标是目标是Intel AVX512支持的平台。
现代CPU架构中的一个重要优化是软件预取，旨在减少缓存未命中的延迟开销。这篇论文丰富了预取功能。
论文中使用的软件预取指令散布在整个FMA指令中，并有效地预取未来FMA指令使用的子张量。在实现中，设计了一个两级的预取策略。在第一级，通过相同的微内核调用发出L1缓存预取，将数据拉入“稍后”(在可调的时间距离内)使用。在第二级，发布L2缓存预取，涉及未来微内核调用的子张量。为了适应第

二级的预取，用三个额外的参数扩展了微内核API:一个指向将被未来调用使用的输出子张量的指针，以及需要预取的输入和权重子张量的指针。这样的二级预取策略实际上减少了来自关键路径的缓存未命中延迟开销。然而，找到将在未来微内核调用中使用的子张量的正确指针并将其用作卷积参数需要一个复杂的、分支的逻辑。
论文也使用了并行化，因为循环很多，有丰富的并行策略。论文选择了mini batch迭代来划分（不知道什么是mini batch，不过论文说看算法三第一行，就是按批次来分），通过这种方式，线程共享了卷积核，这些张量随后可以从共享缓存中重用。
层融合，现代DNN架构不仅包含卷积层，还包含ReLU、Pooling、LRN、Normalization和Bias等层。这些层也许用的上卷积层的数据，当涉及的数据在缓存中很热时，通过利用这种时间局部性，节省了这些层将消耗的内存带宽。
内核流，论文使用了一种内核流框架，以此来减少面对不同情况选择用什么内核引入的复杂的、有条件的代码的开销。
该框架由两个阶段组成:dryrun和replay阶段，dryrun阶段生成算法的先决参数，在CNN层的设置过程中，干运行阶段只需要执行一次;在运行时，执行内核流框架的replay阶段。

2 心得

尚未看完，这篇论文的名词好多，不过有些优化和《零内存高性能直接卷积》一样。