《Advancing Direct Convolution using Convolution Slicing Optimization and ISA Extensions》笔记

zxp

January 6, 2024

1 论文内容

《通过卷积切片优化和ISA扩展来推进直接卷积》这篇论文提出了SConv，一个适用于SIMD体现结构的直接卷积算法。
针对的问题也是im2col占用的内存太大，这篇论文是基于之前有论文证明直接卷积在一定条件下优于传统的im2Col+GEMM的结论，SConv利用体系结构提高卷积的缓存利用率和利用指令集框架ISA进一步优化直接卷积。比起im2Col+GEMM减少了数据操作开销，并使用了特定的缓存分块技术减少缓存缺失次数。
sconv的编译流程，使用ONNX-MLLR和LLVM实现，运行的时候调用合适的微内核。

1.1 卷积切片

目的是高效的缓存分块。SConv会先通过卷积切片分析（CSA）来计算分块的大小、分布和调度。然后使用卷积切片优化（CSO）根据CSA确定的策略生成用于计算的核。

1.2 卷积切片分析（CSA）

分块策略极其复杂，是这篇论文的最主要的部分。核心是保持一组要计算的输出、输入和核在L1缓存，多组在L2和L3。最大可能让留在L1缓存中的数据进行复用。CSA用了一种启发式算法，基于成本模型（计算缓存和内存之间的数据移动的代价），越低的成本意味着更好的数据重用和和缓存使用，根据不同架构选择最低的值。虽然这个过程需要时间，论文认为这种时间是可以接受的。

1.3 卷积切片优化（CSO）

这篇论文实现直接卷积总共有5层（没计算输入张量和过滤器的批次），第1层调用微内核，第2层在单个集合上迭代，一次打包一个块给最内层使用，层3和1遍历过滤器的所有元素，层4和2历遍输入的所有输入元素，最外层历遍批次。
微内核位于最内层，负责计算形成卷积的算术运算。为了提高效率，微内核应该被视为硬件的扩展，微内核的实现是针对体系结构量身定制的，并使用任何可用的特定指令来增加吞吐量。该论文专注于一个单精度(32位)浮点微内核。并且论文认为和现在的线性代数库一样会有扩展性和战未来。
论文中使用了两个不同的硬件设备，一个IBM的，一个英特尔的。在IBM的设备上使用了基于MMA的微内核。在英特尔的机器上使用了AVX-512。为了更好的使用这些微内核，还对数据进行了打包重塑成适合这些微内核的结构，未说明具体怎么做。但有数据的复制。

1.4 性能评估

使用ONNX-MLIR进行全模型性能评估，将SConv集成ONNX-MLIR框架，并训练七个机器学习模型，每个模型输入一个图像。对比的Base为从Caffe框架中打包的Im2Col程序和OpenBLAS库的GEMM程序实现的卷积算法。SConv使用的微内核也来自OpenBLAS库。在单浮点和单线程下执行，运行100次取平均值。
结果是SConv在所有的模型都优于Base，平均快15%，按加速比排序SConv在90%的卷积上优于Base。
论文的分析为，在x86平台下，SConv的打包时间比Base短，Base平均有26%(20%-32%)用于打包，而SConv平均只有10%(7%-13%)。

1.5 结论

论文的结论是，Im2Col变化是不一定需要的，不使用Im2Col而换一种打包的方式，用SConv的方式将卷积操作简化为GEMM操作会大部分情况会更好。Base优于SConv的场景都是BLAS GEMM最近性能场景（Im2Col转换后的矩阵长宽接近，批次比较小的情况）

2 心得

这篇论文提出了SConv，基于直接卷积，论文认为Im2Col变化后再用线性代数库中的矩阵乘法占用很多内存还浪费了很多时间用于打包，SConv有更好的打包方式，把张量分块成适合结构再打包。