《Design and implementation of Deep Learning 2D Convolutions on modern CPUs》笔记

zxp

January 13, 2024

1 论文内容

《在现代CPU上设计与实现深度学习2D卷积》这篇论文是针对CPU平台的直接卷积的优化。论文的目的是说明在CPU平台上优化卷积有很大的探索空间，为未来基于目标层参数、量化水平和硬件架构的高效卷积提供基础。
论文选择CPU的原因主要是现代多核心CPU为分布式深度学习训练提供竞争力，某些情况下CPU比gpu更适合训练dnn（训练小型模型），CPU容易有更高的内存，CPU比GPU使用更加广泛成本低。
论文指出优化会非常难，优化依赖卷积层参数、量化水平和硬件架构，情况十分多，且都需要手动优化。并且寻找每种情况的最好的优化很麻烦，探索空间太大，需要做的测试会是天文数字。因此需要建立优化过程的分析模型来减少寻找解决方案的时间。论文中提到CPU/GPU生厂商会提供优化库，比如英特尔的oneDNN（不过在这篇文章中是用来对比的）。论文实现的工作会比oneDNN快。

1.1 方法

面对即使针对了特定的层和特定CPU优化的探索空间还是太大和需要的测试程序太多的问题，论文开发了一个分析模型来生成优化参数。通过将执行的Load/Store(L/S)指令数量以及不同级别缓存（L1/L2/L3）和内存的访问数量表示为数学方程，硬件和卷积层参数作为这些方程的输入。通过这样使得优化（比如register blocking和loop tiling）能高效的应用。论文开发了一个程序能生成优化参数并调用适合的优化程序。

1.1.1 向量化

向量化严重受到数据在内存中的布局影响。最流行的三个布局是:
bdyx,byxd,dyxb(b=batch,d=channel,y=hight,x=width)。改变布局会带来较高的开销，但是训练中上一层的输出是下一层的输入，如果输出张量和输入张量

布局能保持一致就只需要在第一层和最后一层改变布局就可以（《高性能零内存直接卷积》那篇论文改变数据布局的时候也提到了这个）。
向量化可以应用在一到两个循环，循环顺序和向量化有很大的关系。有很多向量化的选项，论文在如何选择上面做了理论分析。选择向量化的标准包括明显的并行性、数组的布局、执行的指令数和微内核的内存占用。向量化在output中是高效的，但在input中是不好用的，因为有步长的存在。向量化在不包含input下标的循环中是有力的，这种情况下一次只有一个元素从input数组被加载。有输出张量的通道和卷积核的hight和width，论文中选择了输出的批次，因为只有这个有足够的并行性(《高性能零内存直接卷积》那篇论文中也是选择对通道向量化)。但并不是这样就是最优的，其他情况有其他情况的最优解，论文还对其他情况进行了分析，在批次足够大，且张量未批次在在最内层的时候，向量化批次会更好。总之，因为张量的纬度不是1，但需要将不是一维的张量存储在连续内存中，这使得使用向量化十分复杂。

1.1.2 寄存器阻塞和数据重用

寄存器阻塞可以显著的减少L/S指令的数量。可以获得更高的算术强度。论文认为目前为止寄存器阻塞优化是最关键的。核心是对那几层循环做寄存器阻塞。论文分析了两种，第一种是对历遍output的四个纬度的循环做寄存器堵塞。第二种是也对历遍核的hight和width做寄存器阻塞。对核的hight和width做寄存器阻塞会比较复杂。第一种方法在缓存级别实现了数据重用，第二种方法在寄存器级别实现了数据重用。虽然第二种方法的L/S指令更少，但实际上第一种方法要快一点，第二种方法在缓存中没数据重用，而缓存远比寄存器大。

1.1.3 减少微内核的内存占用

前一节指出，微内核的数据无法全部直接放入缓存的时候，L/S指令更少的寄存器阻塞可能不是最快的。这节指出，减少加载进入寄存器的数据，让参与计算的数据能全部塞进缓存中，会使性能更好。内存占用任何时候都应该能放入L1或者L2级缓存。

1.2 并行化，loop tilling循环展开，循环排序

并行化OpenMP框架实现，线程数等于物理内核数量。并行化的循环是批次或者output的高，取决于参数，需要是足够的并行性，如果并行性不够就需要对多个循环都并行化。被并行化的循环得放在最外层。
循环的顺序，论文中是为了让output不加载只载入内存一次，将input的通道放最内层，核的hight和width放内二三层。（这是在循环分块基础上的，并且论文中默认对核在内存中的顺序做了转换，核在内存中以[批次][hight][width][通道]，向量化的章节进行了说明，论文认为对核进行布局上的变化开销很小，于是对核进行了布局上的变化）。为了并行input的批次被放在最外面。这样只剩下三个循环，output的hight和with，核的批次。只有两种顺序是有效的

（output的hight和with，核的批次）（核的批次，output的hight和with）。论文开发了一种算法来找最优。

1.3 反向传播和梯度更新

论文的工作未把注意放这里。

1.4 性能评估

硬件使用了两颗inter的cpu。评估了在三个流行的cnn（DenseNet-121,Res Net-50,SqueezeNet）上的性能，只评估卷积层和ReLU层(112个不同卷积)，对比了不同优化和onDNN。寄存器分块是目前为止最重要的优化。

1.5 结论

优化过程很复杂，需要建立分析模型。oneDNN效果不好在于采用了JIT策略，生成内核要时间，只有莫一层运行多次或者，负载特别大才好用。

2 心得

使用的优化方法和其他工作是优化直接卷积的论文是一样的。都是，向量化、寄存器阻塞，循环展开和并行化。
这篇文章指出优化的方式很多，不同情况需要不同的优化方式，探索空间很大。如果要得到最高的性能，穷举是需要天文数字，需要建立分析模型。