分布式计算稿子

**（第2页）**先说，CNN指的是卷积神经网络。我对于这篇论文的报告将从CNN简要介绍、目前已有的CNN学习、论文所介绍的新的CNN分布式学习方式以及结论四个方面介绍这篇论文。

**（第4页）**卷积神经网络(CNN),是一个强大的分类工具，在图片识别方面具有广泛的应用。卷积神经网络是深度学习最常用的模型之一。它包含输入层，卷积层，采样层，全连接层。在这里，我们将简要介绍这四个层，为后面的分布式学习做准备。

**（第5页）**先说输入层。由于机器并不能识别出图片，但是他可以识别出图片的像素。以RGB图片为例子，机器会把一张彩色图片分成三张与原图大小相等R,G,B图，只是RGB相同位置的数值不一样由此可以构成一张彩色图片。如下如所示，将图片转成机器能够识别的图片的过程即为输入层任务。这里仅仅用作表示，因此图片的矩阵规模较小，与实际的差距较大。当把这三个图片合成一张图片时，对应位置数据按权权值相加，即构成我们可见的彩色图片。

**（第6页）**下面我们来说卷积层。卷积层是CNN的核心操作。什么叫卷积呢。比如，现在有一个矢量，那么对它进行乘法操作，能够使它变长活着变短，但是不能改变它的维度。而，卷积操作是对它改变维度的一种操作。以二维的纸为例，相当于把它改变维度变成三维，这个卷起来得过程形象成为卷积。那卷积的具体操作是什么呢？这里指的是选取一定的范围的像素矩阵进行点乘运算。以图片选取的像素为例，那么计算过程是选取的像素矩阵点乘卷积核。得出结果为-8放入最后的输出矩阵。这么做可以使得图片的边缘特征得到弱化，并且能够保留图片原有矩阵的大小。卷积操作具体的效果取决于卷积核的值。如果核边缘都是0，则相当于弱化边缘等等。

**（第7页）**刚才介绍了卷积的基本操作，现在我们介绍对同一个图片执行两次卷积的操作。这么做可以提取图像不同的特征。比如，刚才的卷积核偏向提取主要特征，现在的卷积操作就可以偏向提取背景特征等。最后通过卷积操作得出来几个矩阵，就称此次卷积操作有几个输出通道。输出通道的数目称为卷积核个数。

**（第8页）**采样层又称为池化层，作用是从卷积层的计算结果中选出自己要的，或者具有代表性的结果，这样可以减少计算量。由于一张图片的特征可能众多。因此CNN的结构一般有“卷积层-采样层-卷积层-采样层”等多级结构实现。

**（第9页）**全连接层作为卷积神经网络得最后一层，由多层神经元组成。比如第一层的神经元判断出来这是一个嘴巴，这是一个鼻子等，那么传递给第二层神经元，第二层神经元就能判断出来这是不是一个人。

**（第10页）** 现在已经存在的分布式机器学习框架包含caffe Tensorflow CNTK等等。但是它们各有优缺点。在这里主要想说明一下它们各自的不足。Caffe对于多卷积-采样层的结构不是很灵活，Tensorflow是基于python的编程框架，速度不快，并且，图片必须事先被编译成对应的结构文件才能被处理。并且，Tensorflow的可扩展性不够，虽然它是一个分布式编程框架，但是其文档上的说明的可扩展性针对2-4个GPU的加速比大约是2.6-3.5,加速比不是很高。在这里我提及一下加速逼的概念。加速比，指的是单处理器串行和多处理器并行的时间比率。因此，加速比越高，证明并行效果越好。CNTK没有高级语言接口。基于此，我们需要重新设计自己的较高加速比的，且支持扩展性的新的分布式方案。

**（第11页）**下面我们来看一下方案和测试结果。

**（第12页）**由于卷积运算承担整个机器学习的60-90的计算量，因此我们考虑将卷积计算进行分布式。即论文研究总方案是使用真正异构的CPU和GPU并行计算结构进行模型并行，将卷积的网络工作负载分布在多台机器上。在这里我们提及一下并行的两种思路。一种是数据并行，把图片分割成不同的像素部分，同时记下相对位置以便将来组合成原有相对位置图像，使用CPU群或者GPU群或者CPU&GPU群。在数据并行中，每个节点都计算不同的梯度，需要由master结点对他们进行平均。模型并行指的是每个计算机点都会初始化拿到图片的全部信息，将一个卷积层copy成多个副本划分到多个节点上，根据不同的卷积核大小，分配不同能力的机器结点，相当于进行负载平衡，使其执行不同的卷积任务。

**（第13页）**为了进行不同机器节点的负载平衡，我们首先要做的就是测试不同的机器结点的计算能力，为真正的异构做准备。那么，第一步就是获取CPU，GPU等设备之间的性能比。

具体做法是，通过使得每一个设备都运行N层卷积，其中图像大小和主设备提供的内核大小都在尝试模拟卷积层的一部分，卷积核的值是使用随机值产生的。各个模拟完成后，计算节点会把计算时间会报告给master节点，由master查找设备之间的性能比。master判断设备性能的依据公式如下，max(t)指的是每一台机器的N层卷集中，花费时间最多的那个卷积运算的时间。

**（第14页）**具体工作量的分配是这样的，master负责收集slave机器计算的卷积图像结果，卷积核大小以及数量，不同的节点接收不同的内核数量。master结点负责所有slave结点的卷积调度，slave结点只负责卷积计算。考虑到方案的主题是进行分布式卷积计算，因此在所有slave结点进行卷积计算完成后，由master结点完成其余的运算步骤。

**（第15页**）这是测试CPU集群和GPU集群的硬件设备参数。

**（第16页）**第一个实验是CPU集群实验。PC1是master,其余结点是slave机器。依次考虑增加CPU集群个数，图片数量，卷积核数目对加速比的影响。

**（第17页）**如图是实验一的结果，现在我们来分析一下。对于同样批次的图片，CPU个数越多，加速比越大。但是由于CPU是异构，各个设备和master之间有通信，但是加速比仍然在加大，则说明，此时的通信的量对加速比影响较小。同时，我们也可以看到，对于同样数目的CPU，同一批图片，卷积核越多，则加速比越大。由于卷积核代表着参数的通信量，且卷积核通信量占总CNN通信量的10%,则参数通信量对此实验加速比仍然影响不明显。换句话说，CPU个数和内核数决定了加速比的上限。现在，我们想具体考虑一下通信时间和计算时间随集群节点增加，内核个数增加的变化。

**（第18页）**以1024章图片为例，对于同一组内核，随着集群节点的增加，通信时间的确是增加的，但是由于卷积计算时间在在并行情况下减少的更明显，则他们相互抵消。由四张图的对比，得知，实际上卷及占比越高，集群增加结点的加速比效果越好。

**（第19页）**第二个实验是GPU集群实验。由于PC1不支持CUDA,PC2是master,其余结点是slave机器。依次考虑增加CPU集群个数，图片数量，卷积核数目对加速比的影响。

**（第20页）**现在我们来分析一下实验二的结果。当卷积核较小时，当涉及集群中有三个GPU中时候，其加速比的分布为1.45\* - 2.45\* 。第二张图片的加速范围是1.5-2.2倍。但是到了最后一张结果图片，即卷积内核数木最大时候，其加速范围是1.7-2倍。可以看到，随着卷积核书越大，加速比范围反而缩小。由于卷积核的个数代表着传参时候的网络通信量，由此我们考虑是否是因为网络通信量越来越大，影响到了加速比。对此，我们进行了具体通信时间，卷积计算时间，以及其余计算时间的对比。

**（第21页）**实验二的附加小实验如下，我们以1024张图片为例，查看CPU集群个数，卷积内核数对图片计算各个时间的影响。两个GPU的时候，通信的占比约20%，但是当三个GPU的时候，通信占比约30%。即GPU的集群结点数目越大，通信量占比越大，则它适用于对卷积核数目敏感的卷积运算。

**（第22页）**现在我们横向对比 CPU集群和GPU集群的加速比分布范围，我们找的是不同图片的最好加速比。如图所示，对于CPU集群，规模为2个CPU的时候加速比分布范围是1.4-1.98，为4个CPU的时候加速比分布范围是1.56-3.28.由此可见，CPU集群适用于对卷积核数目不敏感，即核数目多的CNN运算。但是对于GPU集群，规模为2个GPU的时候，加速比分布范围1.96-1.66,规模为3个GPU的时候，加速比分布范围是2.45-2.0.由此，我们可以看到，随着计算规模增加，计算带来的性能优化远不如通信带来的时间膨胀。由此，GPU集群更适合于对卷积规模敏感的CNN运算。

**（第23页）**上述无论是CPU实验还是GPU运算，都是局限在集群数目较少的情况。由此，我们还需要验证集群的可扩展性。论文作者进行了多结点的性能模拟，我们查看CPU集群的可扩展性，可以看到，在集群规模小于4的是时候，加速比性能提升明显，但是4之后，加速比逐渐趋于稳定，但是也没有破坏机器性能。

**（第24页）**我们现在查看GPU集群的可扩展性。在GPU集群规模大于8之后，加速效果几乎停滞不前。综上CPUGPU 模拟实验，在规模大于8之后，通信对时间的影响略微加大，但是计算卷积的时间略微减少，由此，总的CNN时间几乎不变。

**（第25/26页）**现在我们查看该实验的结论。将卷积进行分布式运算是实现CNN分布式运算的一个有效工具。它能够加速计算模型，具有良好的可扩展性。但是文章有一个方面还可以探索，由于GPU计算一律使用CUDA,即只能使用NvidiaGPU。如果实验可以扩展成支持OPENCL的ADM形式GPU集群，那么就可以构建不同种类的GPU构成的计算集群了。