****

学生实习报告

实习单位: 上海交通大学

实习时间： 2020.7.3 至 2020.7.31

学院(系): 电子信息与电气工程学院

专 业: 计算机科学与技术

学生姓名: 王梓涵 学号: 517021911179

2020年8月5日

机器学习平台计算图替换的浮点数运算差异分析

# 摘要

机器学习框架中常使用图替换的方式来对模型的计算图进行优化，以提升运行性能，但可能会造成数值上的运算差异。本文就图替换所产生的浮点数运算差异开展研究，提出了预判是否可能存在运算差异，以及对差异进行追踪与分析的方法。同时利用这套方法，在真实的数据集与机器学习模型上进行了实验。此项工作，对于机器学习模型在部署中的运算结果一致性问题，具有一定的借鉴价值。  
  
关键词 机器学习 计算图替换 浮点数运算 软件测试

# 简介

## 研究目的与意义

机器学习框架常使用计算图来表示模型结构，并会对计算图进行一系列优化以提升运行效率。图替换是一种较强的计算图优化，它利用算子的运算性质，将计算图中的某些子图替换为数学上等价的运算，以获得可能的计算性能提升。由于机器学习平台通常是基于浮点数进行运算，某些替换会导致运算的差异。由于机器学习平台常使用较低精度的浮点数进行运算，图替换可能会产生较为显著的预算差异，并影响最终的推理结果。由于机器学习模型可能会被部署到不同平台上，而不同平台上的图替换方案可能不同，这可能会造成模型运算结果的不一致问题。本文对该问题展开研究，对机器学习平台的开发人员，在模型部署上起到一定的帮助。

## 本文贡献

本文从运算差异的角度来研究机器学习模型的替换技术，这在现有的文献中是比较新颖的。具体而言，本文在以下几个方面做出了贡献：

* 分析了对图替换可能导致浮点运算差异的条件；
* 提出了追踪浮点运算差异的思路与方法；
* 选取具有代表性的数据集与网络结构进行了测试，对差异进行了统计和分析。

# 研究内容与方法

## 图替换

在讨论图替换所造成的运算差异之前，有必要对于图替换涉及的相关概念进行明确的定义。定义工作负载为计算图以及与之匹配的参数，即，设所有工作负载的集合为。工作负载在一些机器学习框架中也称为模型、模块等。计算图可表示为以输入数据和模型参数为参数的函数。同时也可理解为表达模型计算过程的有向无环图，其中顶点表示算子，边表示数据的流动方向。这两种表示在计算图的意义下是统一的，图表示可以理解为函数表示的一种具体的实现方式。为模型的参数列表，参数的数量和各参数的维度需要和计算图中的定义相符合。

图替换定义为从工作负载到工作负载的映射。一个图替换不仅会改变计算图的结构，同时还可能会改变参数列表的内容。一个有效的的图替换需要保持模型的语义不发生改变。假设对于任意一个模型的工作负载，经过变换后为，则是有效的当且仅当下列等式成立

在实际应用中，一般仅关心图中具有的特定模式的某些子图，并不会对计算图进行整体的修改。就具体算法而言，图替换一般分为两步：第一步是子图匹配，即定位图中具有某些的结构的子图；第二部是图的改写，生成新的工作负载。在TVM[1]的Relay[2]中间语言编程接口中，提供了数据流模式这一功能进行计算图的模式匹配与改写。而模型参数的变动，则需要开发人员自行处理。

## 运算差异预判

由于机器学习平台中常使用的浮点数进行运算，而浮点数作为实数的有限子集，其运算是不精确的，所以在实数域上式的替换在浮点数域上并不总是满足，这就是运算差异的来源。

由于图替换的多样性，对模型中涉及的所有可能的图替换均进行计算差异的测试是不现实的。需要通过一定的预判，筛选出可能造成运算差异的替换方案。在传统标量或向量程序中，早已讨论过优化所导致的浮点一致性问题。[3]中指出，浮点运算顺序的改变是造成浮点运算差异的根源，并且列举了一系列可能导致差异的标量或向量优化。

在涉及张量运算的机器学习模型中，浮点差异的原因也是类似的，因为张量也是由若干标量以一定的方式排列而成。这里给出一个更加规范的预判运算差异的方法。假设结果张量中各标量元素分别为。对于每个，分别用输入参数中的元素，将替换前后该标量所经过的运算表达出来，分别记为和。如果存在一个结果张量的中元素，仅通过在浮点数域上成立的运算性质（如交换律等）无法使得和在结构上相等，则该替换可能造成浮点运算差异。由于图替换一般只关心某个子图，其余的部分不变，所以只要使用入口处各张量的元素来表达出口处张量各元素即可。

下面举两个例子来展示该方法的使用。首先考虑下列关于矩阵转置的运算性质  
将其看作一个图替换方案：在原计算图中，两个矩阵先做乘法再转置；替换后的计算图，两个矩阵左右交换，分别转置后再做矩阵乘法。即设结果矩阵为，将该矩阵中各元素所经过的运算表达出来。在替换前，有  
替换后，则为  
由于浮点数满足交换律，上式可改写为  
可以看到对于任意，式和式总是等价的，说明该替换不可能造成浮点运算差异。注意到求和运算作为一个约减运算，也可能造成浮点运算差异。但这是算子内部调度的问题，在计算图层面无法控制此方面的运算顺序，所以不作考虑。

第二个例子为矩阵乘法的分配律  
设结果矩阵为，分别表示替换前后该矩阵中的各元素

可以看到，任取一组，由于浮点数运算不满足分配律，式无法通过变换和式在结构上等价，所以该替换是可能导致浮点数运算差异的。

## 差异追踪与分析

通过2.2节的差异预判，可以筛选出可能造成差异的图替换方案。下面需要将这些替换作用于实际的工作负载中，对运算过程中差异的变化情况进行追踪。该方法的流程类似于软件测试中的断点调试，但需要根据机器学习模型的特点进行一定的调整。机器学习模型的计算是以图的形式表示的，和传统软件中具有特定顺序的语句序列有一定差异。计算图的“断点”，事实上是一系列符合条件的子图的出口。当这部分子图完成计算时，就触发了一个断点，需要对这部分的结果进行某种操作。

近年来的机器学习模型多为模块化的，整个模型中往往可以拆分成若干更小的基本块。选择断点的时候，可以考虑各模块的出口作为断点。模块的组织在空间上具有一定的顺序，这样可以更方便地将输出的中间结果和计算图的结构对应起来，更为直观。同时总体的断点数量也能控制在一个较为合理的范围内，使测试过程不会耗费过多的计算资源。

断点的设置可以通过以下两种方式进行，一种是在构建计算图时就做好记录，另一种是在以构建好的计算图中通过模式匹配来找到符合断点。本文的实验中的计算图是通过其他机器学习框架编译而来，所以采取的是后一种方式。

在到达断点之后，外部测试程序需要能够访问这样的中间结果。一种方式就是机器学习框架能够以某种方式直接输出结果，让外部测试程序读取。但是在常见的机器学习编译器中往往不具备这样的能力。另一种就是将子图提取出来单独作为一个计算图，计算图的输出就是断点的结果。这样的方式对编译器的技术要求较低，但会造成重复计算，整体的计算量和断点数呈平方级别上涨。由于TVM并不支持中间结果输出，本文实验中采取的是后一种方案。

# 实验过程与结果

## 数据集与模型

本文的实验主要考察的是计算机视觉中的图像分类任务。数据集采用的是CIFAR-10，每个样本为32×32的RGB图像，共有50,000个训练样本，10,000个测试样本。该数据集体量适中，在众多图像分类工作中得到了广泛使用。

实验中选取的模型为中小规模的卷积神经网络，包括ResNet[5]和NASNet-A[6]，它们对于CIFAR-10数据集均有特别设计的结构。具体参数上，ResNet采用的是ResNet-20和ResNet-110，其中20和110为卷积层的数量；NASNet-A采用的是6 @ 768，其中6为普通单元堆叠的数量，768为倒数第二层的特征图数量。实验中模型的参数均为自行训练得到，和原论文中的准确率接近，但不追求达到。

## 替换方案

本实验中涉及三个替换方案：卷积-批归一化替换、卷积-偏置-相加替换、平均池化-相加替换。其中第一个方案是两个模型通用的，后两个方案仅限于NASNet-A。

卷积-批归一化替换是将卷积-批归一化子图变换为卷积-偏置子图，见图1。其原理是把批归一化看作是1×1卷积，需要对卷积的权值和各特征图的偏置值进行预计算，详见[6]。该方法对于减少参数量与计算量有较大帮助。卷积-批归一化在近年来的卷积神经网络中非常常见，具有很广的适用性。

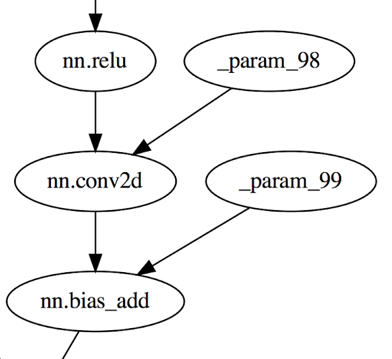
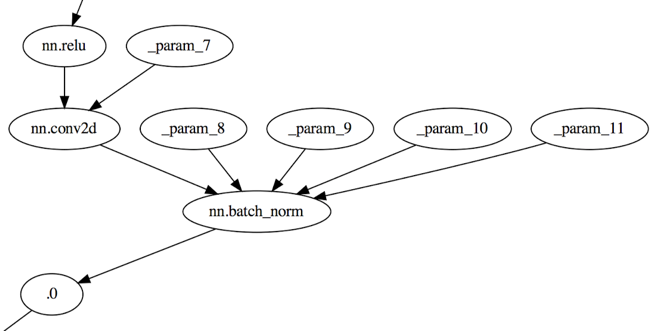


图1 卷积-批归一化替换

卷积-偏置-相加替换是在NASNet-A中，将具有卷积-偏置-相加模式子图的两次卷积和两次偏置合并成一次的替换，见图2。变换的过程中，需要将两个卷积的权重连接，两个偏置的值相加，具体细节在[8]中有提及。

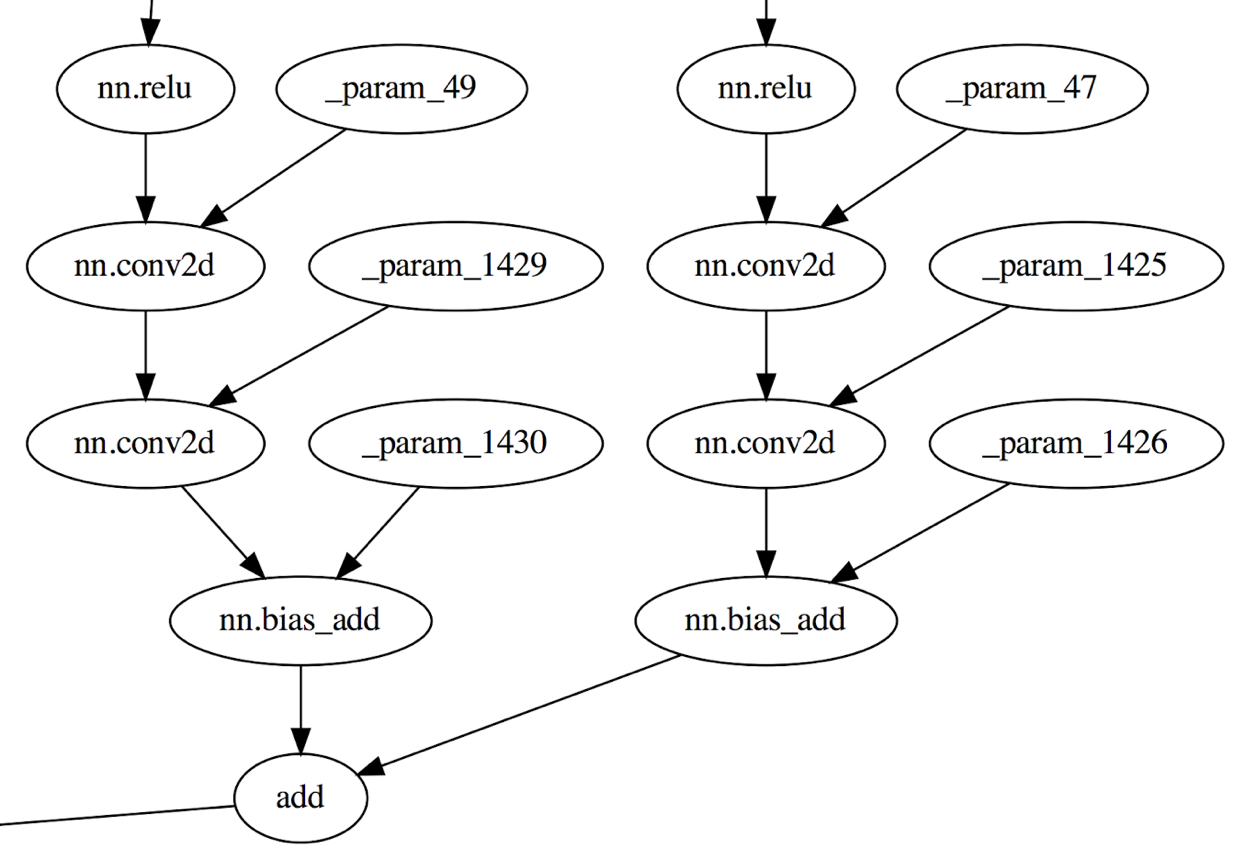
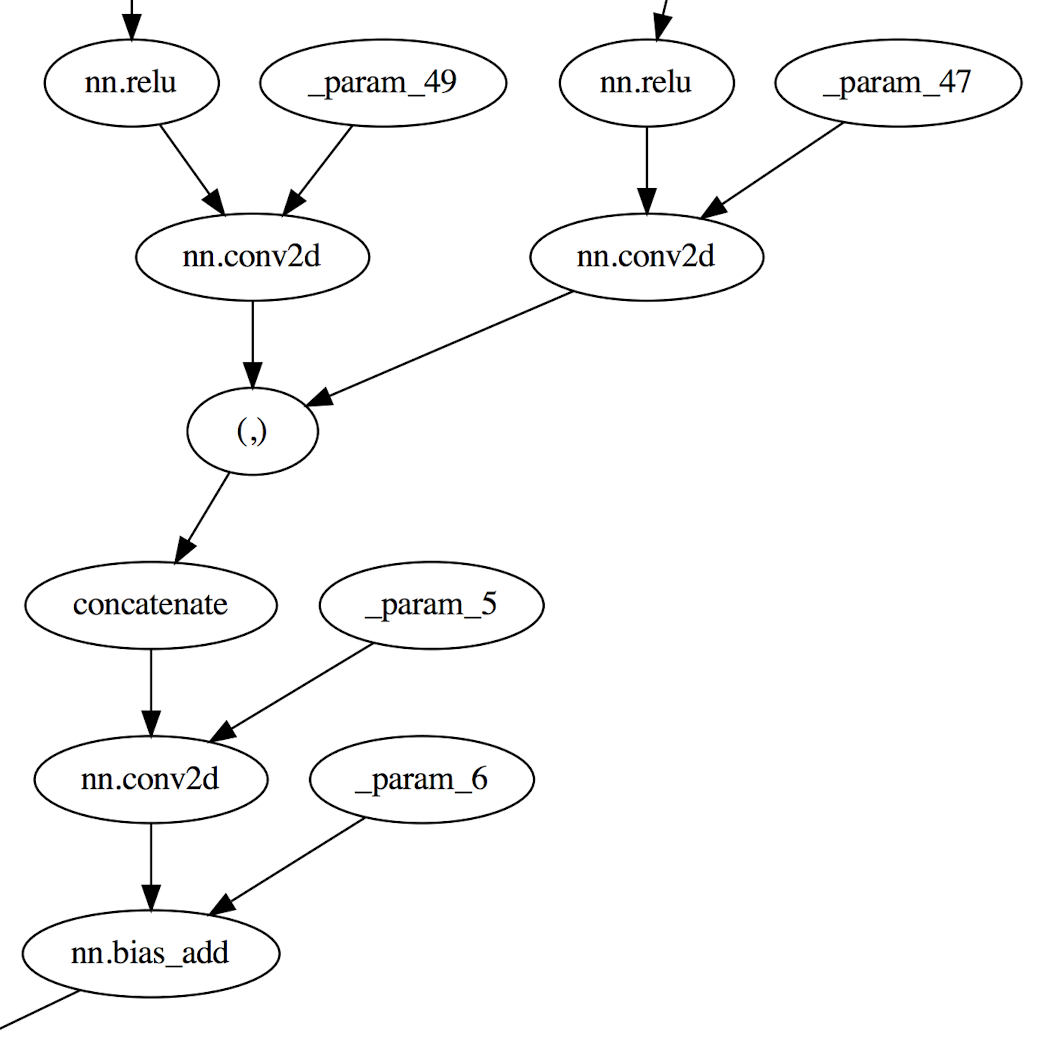


图2 卷积-偏置-相加替换

平均池化-相加替换是将NASNet-A中具有平均池化-相加模式的子图替换成一个单独的卷积，其权重为一个常量，见图3。需要注意的是，只有当平均池化算子中，计算特征图边缘值时包括所填充的零值，该替换在语义上才是等价的，因为卷积计算永远是包括所填充的零值的。

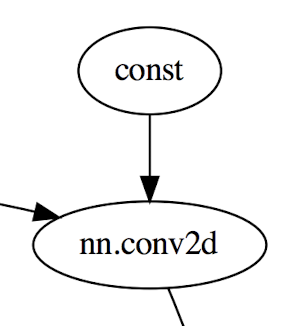
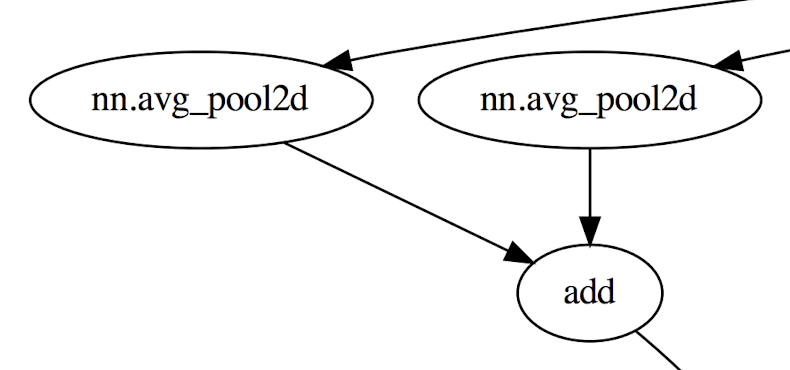


图3 平均池化-相加替换

## 差异分析

本节中，将上述替换方案作用于模型的工作负载，分别测试模型在测试集上的准确率，并追踪各断点输出的中间结果，进行统计分析。测试准确率时使用测试集的全部数据；追踪断点时，由于计算量大，仅使用测试集中的部分样本。所有测试均在单精度（32位）及半精度（16位）上分别进行测试。

对于ResNet模型，该本实验中仅有一种替换可用，即卷积-批归一化替换。首先进行测试的是ResNet-20，其在各配置下的准确率见表1，各断点的运算差异统计见图4和图5，差异测试使用了测试集的前1000个样本。本文中所有运算差异统计图均分为左右两个子图，左侧为最大差异，右侧为平均差异。可以看到，对单精度浮点数，图替换所产生的计算差异很小，在数量级，最终的准确率没有任何任何影响。半精度浮点数，计算差异在数量级。在10,000个测试集中，有个位数样本的推理结果出现了差异。就变化趋势而言，在各卷积模块中，运算差异呈现不断上升的趋势，而最终出现明显的下降。这是因为最后一层Softmax会将数值压缩到之间，运算差异也随之压缩了。对于不同精度的浮点，其差异的变化趋势类似，只是数量级会有所不同。

表1 ResNet-20准确率

|  |  |
| --- | --- |
| 配置 | 准确率 |
| 单精度，原始 | 0.8944 |
| 单精度，替换 | 0.8944 |
| 半精度，原始 | 0.8947 |
| 半精度，替换 | 0.8943 |

|  |  |
| --- | --- |
|  |  |

图4 ResNet-20单精度在各断点的运算差异统计

|  |  |
| --- | --- |
|  |  |

图5 ResNet-20半精度在各断点的运算差异统计

为了研究运算差异和模型规模的关系，对ResNet-110进行了测试。各配置下的准确率见表2，各断点的运算差异统计见图6和图7，差异测试使用了测试集的前1000个样本。该模型的准确率并未受浮点数精度以及图替换的影响。中间结果运算差异比ResNet-20略高一些，但差别不大。在单个模型中，差异的变化整体呈上升趋势，但是中间也有下降，比ResNet-20呈现出更为复杂的变化过程。

表2 ResNet-110准确率

|  |  |
| --- | --- |
| 配置 | 准确率 |
| 单精度，原始 | 0.9079 |
| 单精度，替换 | 0.9079 |
| 半精度，原始 | 0.9079 |
| 半精度，替换 | 0.9079 |

|  |  |
| --- | --- |
|  |  |

图6 ResNet-110单精度在各断点的运算差异统计

|  |  |
| --- | --- |
|  |  |

图7 ResNet-110半精度在各断点的运算差异统计

为了考察多次替换对运算差异的影响过程，对NASNet-A 6 @ 768进行了测试。该模型有三个可用的替换方案。在每个配置测试过程中，依次进行卷积-批归一化、卷积-偏置-相加、平均池化-相加替换，分别测试每个工作负载下的准确率和中间结果差异（与原始工作负载相比）。中间结果的准确率见表3，各断点的差异见图8和图9，差异测试使用了测试集的前300个样本。在表和图中，0表示原始工作负载，1、2、3分别表示进行过卷积-批归一化、卷积-偏置-相加、平均池化-相加替换之后的工作负载。单精度在准确率上依然没有变化，半精度在两至三个样本上发生了差异。断点的差异不再有明显的上升的趋势。多次替换后，断点的差异整体上有所上升。

表3 NASNet-A 6 @ 768准确率

|  |  |  |  |
| --- | --- | --- | --- |
| 配置 | 准确率 | 配置 | 准确率 |
| 单精度，0 | 0.9183 | 半精度，0 | 0.9181 |
| 单精度，1 | 0.9183 | 半精度，1 | 0.9182 |
| 单精度，2 | 0.9183 | 半精度，2 | 0.9183 |
| 单精度，3 | 0.9183 | 半精度，3 | 0.9182 |

|  |  |
| --- | --- |
|  |  |

图8 NASNet-A 6 @ 768单精度在各断点的运算差异统计

|  |  |
| --- | --- |
|  |  |

图9 NASNet-A 6 @ 768半精度在各断点的运算差异统计

综合上述实验结果，可以看到，图替换对于模型的输出结果会有影响，但影响很小。对于单精度浮点，几乎可以确保其推理结果不会出现任何变化。而对于半精度浮点，在一万个样本中也仅有某几个会出现差异，对模型的准确率的影响微乎其微。可能的问题在于，替换后的模型可能无法和先前保持一致的行为。这进一步地印证了深度学习模型对于浮点数精度的要求较低，对中间结果有一定的容错能力。

# 结论

本文提出了通过张量元素表达式的结构，来预判计算图替换是否可能存在运算差异的分析方法。同时，本文对获取中间结果差异的思路和方法进行了探讨。利用这套方法，在真实的数据集与机器学习模型上进行了实验。图替换所造成的运算差异很小，但在浮点数精度较低时推理结果仍会有细微区别。此项工作，对于机器学习模型在部署中的运算结果一致性问题，具有一定的借鉴价值。

# 参考文献

1. Chen TQ, Moreau T, Jiang ZH, Zheng LM, Yan E, Cowan M, Shen HC, Wang LY, Hu YW, Ceze L, Guestrin C, Krishnamurthy A. TVM: An Automated End-to-End Optimizing Compiler for Deep Learning. 13th USENIX Symposium on Operating Systems Design and Implementation, 2018: 579-594.
2. Roesch J, Lyubomirsky S, Weber L, Pollock J, Kirisame M, Chen TQ, Tatlock Z. Relay: A New IR for Machine Learning Frameworks. Proceedings of the 2nd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, 2018.
3. Corden M. Consistency of Floating-Point Results Using the Intel Compiler. <https://software.intel.com/content/dam/develop/external/us/en/documents/pdf/fp-consistency-121918.pdf>
4. Learning Multiple Layers of Features from Tiny Images. Krizhevsky A, 2009. <http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>
5. He KM, Zhang XY, Ren SQ, Sun J. Deep Residual Learning for Image Recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
6. Zoph B, Vasudevan V, Shlens J, Le QV. Learning transferable architectures for scalable image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
7. Fusing Batch Normalization and Convolution in Runtime. <https://tehnokv.com/posts/fusing-batchnorm-and-conv/>
8. Jia ZH, Padon O, Thomas J, Warszawski T, Zaharia M, Aiken A. TASO: Optimizing Deep Learning Computation with Automatic Generation of Graph Substitution.

|  |  |
| --- | --- |
| 指指导教师对学生实习情况的评价意见 | 指导教师（签名）：  年 月 日 |
| 实 习 报 告 评 阅 人 意 见 | 评阅人（签名）：  年 月 日 |
| 系 主 任 意 见 | （实习成绩按等级制（A、B、C、D和F）方式记载）  评 定 成 绩 ：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  系 主 任(签名)：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  年 月 日 |

周记一

本周为生产实习的第一周，工作主要以学习相关文献为主，为后续实验的开展提供参考资料。文献调研分为三方面开展。一方面是实验所涉及的机器学习模型，另一个方面是计算图优化的相关文献，这是本周工作的核心。第三个方面则是TVM在线文档的学习，由于后续的实验都会在TVM上开展，所以这一部分也非常重要。

机器学习模型方面，我准备从计算机视觉中的图像分类模型入手。其中卷积网络占模型结构的绝大部分，这部分结构具有相当的多样性，优化的机会比较多。我浏览了网上关于近年来图像分类模型的多篇综述，对各模型的大致结构有了一个全局的把握。

计算图优化是近年来机器学习系统方面比较受关注的方向，有一定数量的论文以此作为课题。本课题所需要关注的是那些会产生浮点运算差异的图优化，这样就将很多优化排除在外了。子图替换是比较强的计算图优化，是通过算子的数学性质来将计算图中的某个子图替换成与其在数学上等价另一个子图。由于浮点数运算是不精确的，这样的替换有时就会产生运算差异。TASO这个工作通过自动化的方式去列举可能的替换方式，其论文里通过案例研究的方式列举了其在常用模型上的替换结果，给了我很大的启发。此外我还在互联网上找到了一篇详细解释卷积-批归一化融合的博客，这个结构在新兴的卷积神经网络上很常见，也可能会导致计算差异。

通过调研，我准备将ResNet和NASNet-A作为实验的对象。ResNet的结构非常经典也相对简单，可以作为初步上手之用。NASNet结构就复杂得多，但其潜在的优化空间也很大，适合方法成熟后进行研究。

TVM的学习上，我首先把一个模型编译的整个流程重新过了一遍，然后学习了Relay IR的使用方式。Relay是TVM的计算图中间表示语言，其本质上是一个λ演算，其对于程序的变换也是和LLVM一样通过遍来进行。由于我对编译原理和基础的程序设计语言理论都比较熟悉，所以这一部分开展地都比较顺利。我编写了一个转换数据类型的遍作为练习。

本周的工作介绍到这里。下周我将搭建整个实验的基础程序框架，尝试实现一个子图替换并进行测试。

周记二

本周主要工作在于搭建实验的程序框架，并进行了初步的子图替换的尝试。本周的工作非常重要，所完成的程序以及积累的经验将对后续的工作有很大的帮助。

由于算子替换常常需要同时改变计算图和模型参数，所以需要将它们作为一个整体对待，在我的代码将其封装为一个“工作负载”，对象的方法实现了编译、运行等基本操作。

实现子图替换的前提是能找到和所有匹配的子图，所以需要对计算图进行模式匹配。在我目前的实现中，模式匹配部分的代码是自己手写的，输入的“模式”就是另一个Relay表达式。这样的好处是这个表达式本身也可以运算，便于开展测试。不足在于，目前的模式匹配是将所有的表达式都当作树状结构，对于菱形结构可能匹配不准确。同时所有表达式中不确定的表达式都用变量代替，模糊了变量和任意表达式的界限。这些问题可能会在后续得到改善。

子图替换是通过对工作负载的变换来实现的，这一变换抽象为一个遍。从计算图中找出所有匹配的子图并执行替换的流程是相同的，这一步骤可以提取出来，只需要实现从匹配的子图到新子图的替换，同时改变模型参数即可。可以将实现了子图替换方法的子类类型作为参数传给替换遍。计算图替换遍在作用于一个具体的工作负载时，动态地构造具体的替换对象，作用于已匹配的子图。

我成功地实现了卷积-批归一化融合，并且在随机的输入上进行了测试。我明确了ResNet-20的具体结构，将其编译为一个工作负载。我将这一替换作用于其计算图，的确能发现运算差异。但是目前的网络的参数都是随机初始化的，可能并不能反应在真实数据上的差异表现，这是后续工作中需要完成的任务。

下一周，我将使用有效的ResNet-20参数，在真实的CIFAR-10数据集上进行测试，同时在模型中设置一些断点，追踪模型的中间结果差异。

周记三

本周的工作主要在于使用有效的参数、真实的数据对计算图优化的运算差异进行测试，通过设置断点的方式获得模型的中间结果并进行比较。

在数据集方面我选择了广泛使用的CIFAR-10。选择的原因有两点，一个是很多模型都有对此数据集专门设计的结构，不需要做额外的预处理，便于简化工作流程。第二是该数据集体量适中，相比ImageNet这类大型数据集来说更适合在个人计算设备上使用。

由于网上要找到和我使用的模型结构、格式完全相同的参数非常困难，我选择自行对模型进行训练。输入的数据按样本进行归一化，训练的数据集还需要进行平移和镜像变换进行增强。学习率从0.1开始，如果十轮内验证集损失不下降则除以10，最小为0.001。这些设置是为了使模型尽可能快地收敛，同时不至于出现严重的过拟合，使其参数达到基本可用的水平，并不追求达到原论文中所列出的准确率。

数据和模型都准备完成后，下面需要从模型中设置断点进行中间结果的输出。由于模型的计算图需要进一步编译成可在目标设备上运行的内核代码，其中间结果对用户往往是不可见的，Relay本身也不提供获取中间结果的功能。这意味着不能像调试普通程序那样设置断点，在程序运行过程中输出中间结果。而应该直接将所期望断点的子表达式提取出来，直接让这个子表达式的值作为函数的返回值。但这样就意味着会有比较严重的重复计算问题，也是目前条件下难以避免的问题。我选择采用的各模块的出口点作为断点，这样的好处是逻辑简单，符合直觉，方便对应和比较。由于近几年的神经网络多采用模块化的设计，这样的策略适用性是很广的。断点的寻找和使用的正是先前实现的子图匹配技术。每找到一个断点，就构造一个对应的工作负载。

我成功地在ResNet-20上输入数据集并进行了评估和中间结果的测试。注意到在使用单精度的情况下，模型中间结果的差异都在数量级，对于最终准确率没有影响。而对于半精度，模型中间结果的差异在数量级，在测试集的某几个样本上会出现分类的差异。我还尝试了ResNet-110这个体量更大的网络，其差异的范围类似，但在这个网络上替换前后却没有出现分类结果的差异。

下一周，我将在NASNet-A上进行进一步的实验。有了前三周的积累，希望后续的实验能够相对顺利地开展。

周记四

本周是生产实习的最后一周，我在NASNet-A上进行实验。本周在完成实验目标的同时，对之前框架里的一些内容进行了一定程度的调整，并且也增加了一些实用功能，使得实习资料更为完善。

在NASNet-A上，除了通用的卷积-批归一化之外，还包括两个额外的替换，这两个替换都是该模型特有的。一个是将深度可分离卷积-相加替换成深度卷积-连接-1x1卷积，第二个是将平均池化-相加替换成用常量卷积核卷积。

在这两个替换的实现过程中，我均发现了一些新的问题。在实现第一个替换中，我发现原先的图替换流程无法处理有多个后继节点的情况。由于图是按照后深度优先的顺序来进行搜索的，当访问到第一个后继节点时，将图改写，但这一改写并未反映到其他后继节点中。这一错误虽然不会改变计算结果，但是会使整体的计算量翻倍。解决方法是在计算图的替换过程中记录所有源表达式到目标表达式的引用映射。在找到可替换的子图时，先在查找表里寻找是否之前已经进行过类似的替换。如果没有进行过，再运行相应的子图替换算法。

第二个替换中，我发现替换前后运算的差异明显增大，对最终的分类结果也有比其他替换明显更大的差异。我进行简单的测试后，发现大的差异主要出现在每个特征图的边界值上。问题的根源在于平均池化中，如果进行填充的话，填充的区域默认是不算在内的。在很多机器学习框架里，仅有这个行为，而在TVM里可以进行选择。而卷积是一定会把填充值算入的。这样替换的话，语义就不等价了。所以我写了一个函数遍，先把原先所有的平均池化的属性全部改掉，再进行后续的替换。这样和模型的参数可能就不那么匹配了，准确率也会下降，但为了测试该替换，也是不得不做的事情。

此外我还做了一些其他的改进或增强。首先我把模式匹配换成了Relay中提供的数据流模式语言，这样模式的定义更准确，同时替换的时候也更好提取相应的结构。第二点是我通过GraphViz可以将Relay表示的计算图绘制出来，这样更方便调试，以及在报告中呈现结果。

和ResNet一样，我对这一系列优化所造成的计算差异进行了测试，得到了相应数据。我使用RStudio将本次和上次的实验结果绘制了出来，便于进一步分析。

为期一个月的生产实习接近尾声了。本次实习我充分发挥主观能动性，独立设计流程、调研并完成实验，是对我科研能力一次小规模的检验。本次的实习也对我今后的科研工作起到了一定的帮助作用，我会继续在系统软件、机器学习、程序分析与测试方面进行探索，做出更多工作。