根据初始的推理结果，在初期的几层运算结果非常接近，但是随着层数增加逐渐下降。有几个奇怪的现象：

1. shortcut层的相同度显著降低
2. 即使前一层运算结果只有70-80%，下一层也有可能增加，甚至到90%
3. 51层相同度只有30%左右

假设相同度在80%以上的都是相同的只是有一些误差，那么51层的结果非常奇怪，更奇怪的是52层的相同度又恢复到了72%

有几个可以研究的方向：1.shortcut 2. 51层

先研究cupy版本，跑出imagenet的acc

Cupy跑的比numpy还慢

第49-51层

计算出来的M0、M1和读取的几乎一样

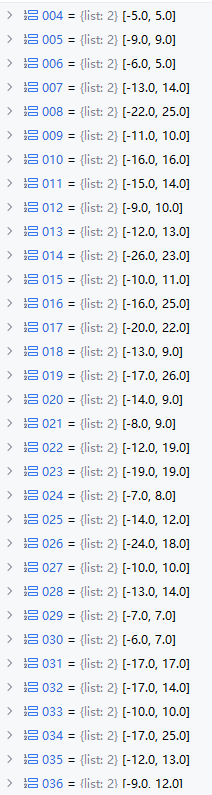
除了M0 M1 和 n是从读取的文件里面得到的，其他的所有数据都来源于上一层或者pytorch模型

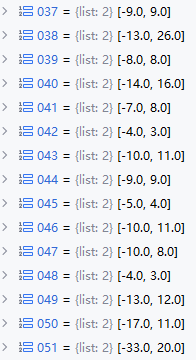
2025年7月9日星期三

通过将input\_quantized进行round取整并转换为uint8类型后，第一层的输出相同度为99.02%

2025年7月10日

统计每一层输出间的差值





2025年7月11日

根据目前为止的研究结果，最大的造成差异的可能原因就是误差积累。

为了验证这个猜想，将pytorch模型中各层对应的输入送到numpy模型对应层中，从而观察每一层对相同度造成的影响。

首先可以观察到shortcut层的效果非常好，十个shortcut层中有6层输出都是100%相同，剩余四层最低相似度为99.92%

对于bottleneck的第一层和第二层，相似度下降也很少，最低为98.1%和98.5%，剩余的输出都在99%以上。第三层下降最多，最低93%

第51层的差值最大达到了-31，其余层的差值都为-1。

猜测pytorch模型中的舍入方式采用了四舍五入的方法。只有使用round时才能达到上述的相似度，使用floor时相似度将下降到70%。但是实际硬件实现时由于右移只能实现floor。