33-解读TPU：设计和拆解一块ASIC芯片

过去几年，最知名、最具有实用价值的ASIC就是TPU了。各种解读TPU论文内容的文章网上也很多。不过，这些文章更多地是从机器学习或者AI的角度，来讲解TPU。

上一讲，我为你讲解了FPGA和ASIC，讲解了FPGA如何实现通过“软件”来控制“硬件”，以及我们可以进一步把FPGA设计出来的电路变成一块ASIC芯片。

不过呢，这些似乎距离我们真实的应用场景有点儿远。我们怎么能够设计出来一块有真实应用场景的ASIC呢？如果要去设计一块ASIC，我们应该如何思考和拆解问题呢？今天，我就带着你一起学习一下，如何设计一块专用芯片。

TPU V1想要解决什么问题？

黑格尔说，“世上没有无缘无故的爱，也没有无缘无故的恨”。第一代TPU的设计并不是异想天开的创新，而是来自于真实的需求。

从2012年解决计算机视觉问题开始，深度学习一下子进入了大爆发阶段，也一下子带火了GPU，NVidia的股价一飞冲天。我们在第31讲讲过，GPU天生适合进行海量、并行的矩阵数值计算，于是它被大量用在深度学习的模型训练上。

不过你有没有想过，在深度学习热起来之后，计算量最大的是什么呢？并不是进行深度学习的训练，而是深度学习的推断部分。

所谓**推断部分**，是指我们在完成深度学习训练之后，把训练完成的模型存储下来。这个存储下来的模型，是许许多多个向量组成的参数。然后，我们根据这些参数，去计算输入的数据，最终得到一个计算结果。这个推断过程，可能是在互联网广告领域，去推测某一个用户是否会点击特定的广告；也可能是我们在经过高铁站的时候，扫一下身份证进行一次人脸识别，判断一下是不是你本人。

虽然训练一个深度学习的模型需要花的时间不少，但是实际在推断上花的时间要更多。比如，我们上面说的高铁，去年（2018年）一年就有20亿人次坐了高铁，这也就意味着至少进行了20亿次的人脸识别“推断“工作。

所以，第一代的TPU，首先优化的并不是深度学习的模型训练，而是深度学习的模型推断。这个时候你可能要问了，那模型的训练和推断有什么不同呢？主要有三个点。

第一点，深度学习的推断工作更简单，对灵活性的要求也就更低。模型推断的过程，我们只需要去计算一些矩阵的乘法、加法，调用一些Sigmoid或者RELU这样的激活函数。这样的过程可能需要反复进行很多层，但是也只是这些计算过程的简单组合。

第二点，深度学习的推断的性能，首先要保障响应时间的指标。我们在第4讲讲过，计算机关注的性能指标，有响应时间（Response Time）和吞吐率（Throughput）。我们在模型训练的时候，只需要考虑吞吐率问题就行了。因为一个模型训练少则好几分钟，多的话要几个月。而推断过程，像互联网广告的点击预测，我们往往希望能在几十毫秒乃至几毫秒之内就完成，而人脸识别也不希望会超过几秒钟。很显然，模型训练和推断对于性能的要求是截然不同的。

第三点，深度学习的推断工作，希望在功耗上尽可能少一些。深度学习的训练，对功耗没有那么敏感，只是

希望训练速度能够尽可能快，多费点电就多费点儿了。这是因为，深度学习的推断，要7×24h地跑在数据中心里面。而且，对应的芯片，要大规模地部署在数据中心。一块芯片减少5%的功耗，就能节省大量的电费。而深度学习的训练工作，大部分情况下只是少部分算法工程师用少量的机器进行。很多时候，只是做小规模的实验，尽快得到结果，节约人力成本。少数几台机器多花的电费，比起算法工程师的工资来说，只能算九牛一毛了。

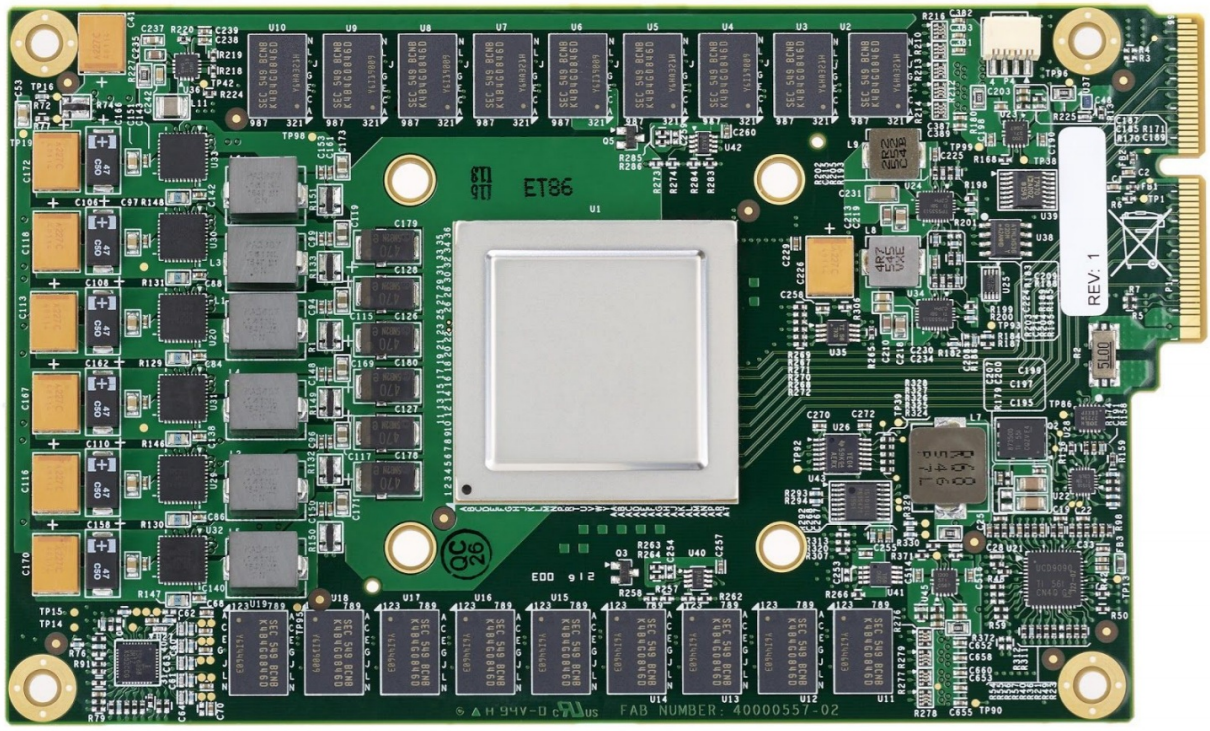
这三点的差别，也就带出了第一代TPU的设计目标。那就是，在保障响应时间的情况下，能够尽可能地提高**能效比**这个指标，也就是进行同样多数量的推断工作，花费的整体能源要显著低于CPU和GPU。

深入理解TPU V1

快速上线和向前兼容，一个FPU的设计

如果你来设计TPU，除了满足上面的深度学习的推断特性之外，还有什么是你要重点考虑的呢？你可以停下来思考一下，然后再继续往下看。

不知道你的答案是什么，我的第一反应是，有两件事情必须要考虑，第一个是TPU要有向前兼容性，第二个是希望TPU能够尽早上线。我下面说说我考虑这两点的原因。



图片来源

第一代的TPU就像一块显卡一样，可以直接插在主板的PCI-E口上

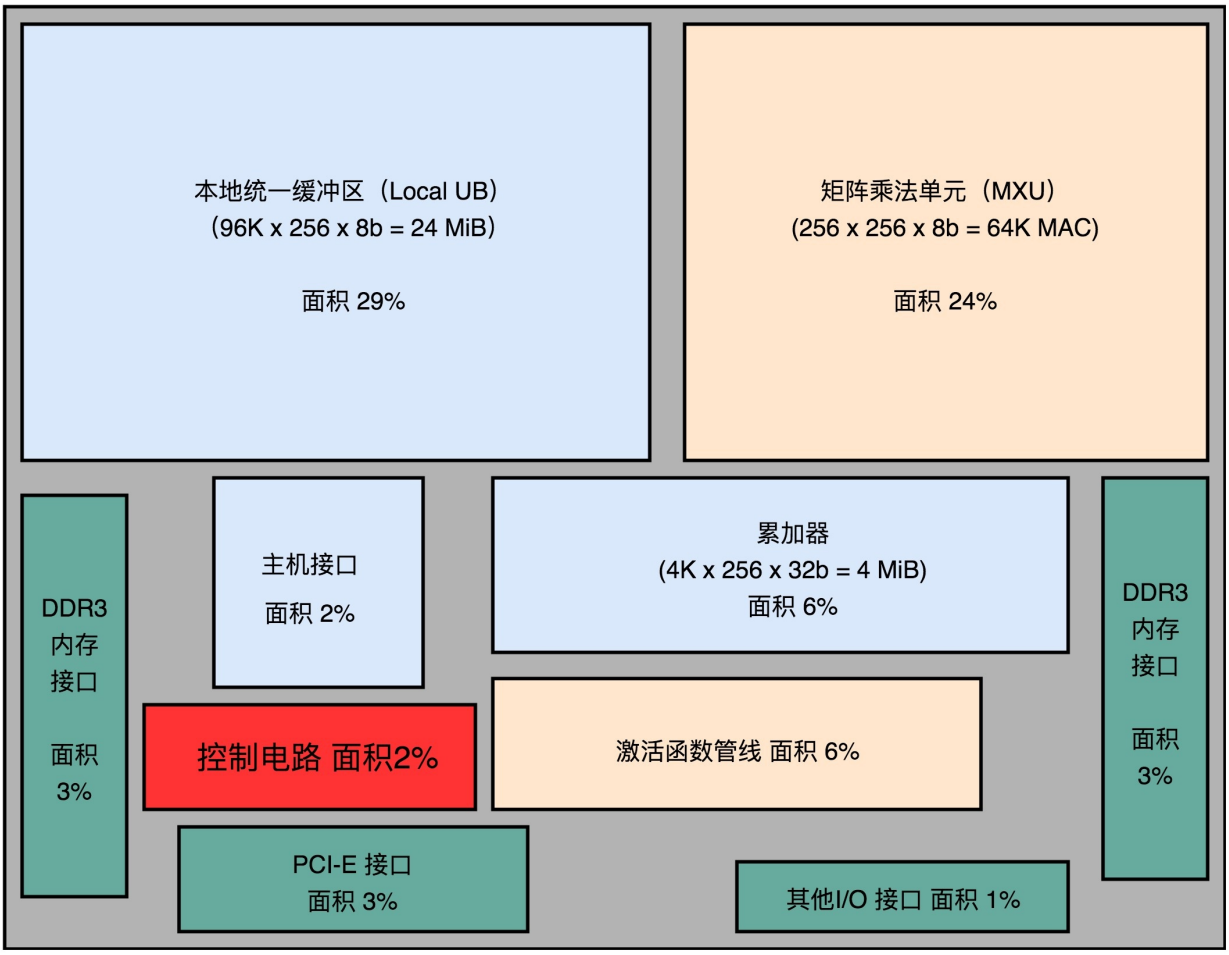
第一点，向前兼容。在计算机产业界里，因为没有考虑向前兼容，惨遭失败的产品数不胜数。典型的有我在第26讲提过的安腾处理器。所以，TPU并没有设计成一个独立的“CPU“，而是设计成一块像显卡一样，插在主板PCI-E接口上的板卡。更进一步地，TPU甚至没有像我们之前说的现代GPU一样，设计成自己有对应的取指令的电路，而是通过CPU，向TPU发送需要执行的指令。

这两个设计，使得我们的TPU的硬件设计变得简单了，我们只需要专心完成一个专用的“计算芯片”就好了。所以，TPU整个芯片的设计上线时间也就缩短到了15个月。不过，这样一个TPU，其实是第26讲里我们

提过的387浮点数计算芯片，是一个像FPU（浮点数处理器）的协处理器（Coprocessor），而不是像CPU和GPU这样可以独立工作的Processor Unit。

专用电路和大量缓存，适应推断的工作流程

明确了TPU整体的设计思路之后，我们可以来看一看，TPU内部有哪些芯片和数据处理流程。我在文稿里面，放了TPU的模块图和对应的芯片布局图，你可以对照着看一下。



图片来源

模块图：整个TPU的硬件，完全是按照深度学习一个层（Layer）的计算流程来设计的

你可以看到，在芯片模块图里面，有单独的矩阵乘法单元（Matrix Multiply Unit）、累加器（Accumulators）模块、激活函数（Activation）模块和归一化/池化（Normalization/Pool）模块。而且，这些模块是顺序串联在一起的。

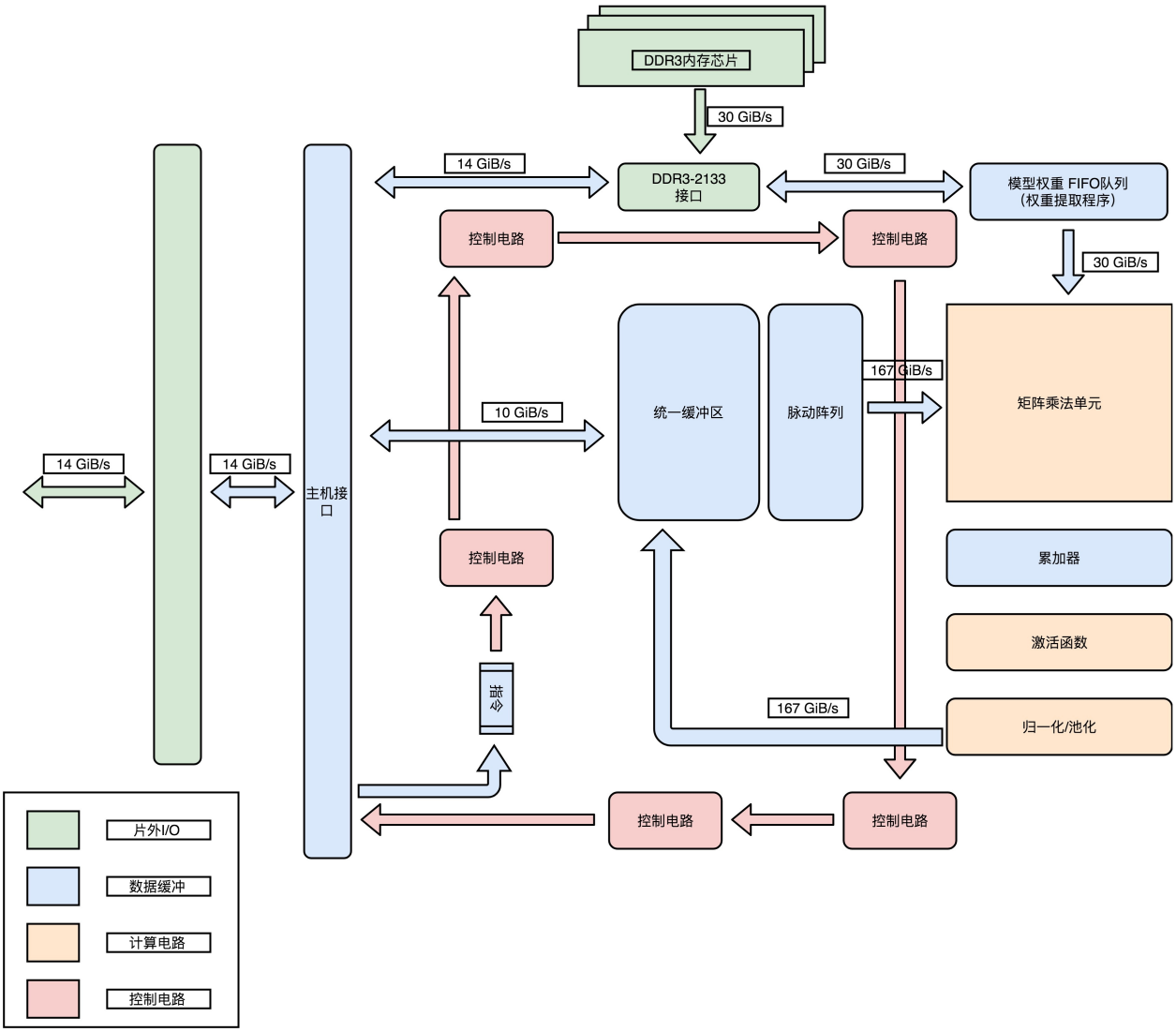
这是因为，一个深度学习的推断过程，是由很多层的计算组成的。而每一个层（Layer）的计算过程，就是先进行矩阵乘法，再进行累加，接着调用激活函数，最后进行归一化和池化。这里的硬件设计呢，就是把整个流程变成一套固定的硬件电路。这也是一个ASIC的典型设计思路，其实就是把确定的程序指令流程，变成固定的硬件电路。

接着，我们再来看下面的芯片布局图，其中控制电路（Control）只占了2%。这是因为，TPU的计算过程基本上是一个固定的流程。不像我们之前讲的CPU那样，有各种复杂的控制功能，比如冒险、分支预测等等。

你可以看到，超过一半的TPU的面积，都被用来作为Local Unified Buffer（本地统一缓冲区）（29%）和矩阵乘法单元（Matrix Mutliply Unit）了。

相比于矩阵乘法单元，累加器、实现激活函数和后续的归一/池化功能的激活管线（Activation Pipeline）也用得不多。这是因为，在深度学习推断的过程中，矩阵乘法的计算量是最大的，计算也更复杂，所以比简单的累加器和激活函数要占用更多的晶体管。

而统一缓冲区（Unified Buffer），则由SRAM这样高速的存储设备组成。SRAM一般被直接拿来作为CPU的寄存器或者高速缓存。我们在后面的存储器部分会具体讲。SRAM比起内存使用的DRAM速度要快上很多，但是因为电路密度小，所以占用的空间要大很多。统一缓冲区之所以使用SRAM，是因为在整个的推断过程中，它会高频反复地被矩阵乘法单元读写，来完成计算。



图片来源

芯片布局图：从尺寸可以看出，统一缓冲区和矩阵乘法单元是TPU的核心功能组件

可以看到，整个TPU里面，每一个组件的设计，完全是为了深度学习的推断过程设计出来的。这也是我们设计开发ASIC的核心原因：用特制的硬件，最大化特定任务的运行效率。

细节优化，使用8 Bits数据

除了整个TPU的模块设计和芯片布局之外，TPU在各个细节上也充分考虑了自己的应用场景，我们可以拿里

面的矩阵乘法单元（Matrix Multiply Unit）来作为一个例子。

如果你仔细一点看的话，会发现这个矩阵乘法单元，没有用32 Bits来存放一个浮点数，而是只用了一个8 Bits来存放浮点数。这是因为，在实践的机器学习应用中，会对数据做归一化（Normalization）和正则化（Regularization）的处理。咱们毕竟不是一个机器学习课，所以我就不深入去讲什么是归一化和正则化了，你只需要知道，这两个操作呢，会使得我们在深度学习里面操作的数据都不会变得太大。通常来说呢，都能控制在-3到3这样一定的范围之内。

因为这个数值上的特征，我们需要的浮点数的精度也不需要太高了。我们在第16讲讲解浮点数的时候说过，32位浮点数的精度，差不多可以到1/1600万。如果我们用8位或者16位表示浮点数，也能把精度放到2^6或者2^12，也就是1/64或者1/4096。在深度学习里，常常够用了。特别是在模型推断的时候，要求的计算精度，往往可以比模型训练低。所以，8 Bits的矩阵乘法器，就可以放下更多的计算量，使得TPU的推断速度更快。

用数字说话，TPU的应用效果

那么，综合了这么多优秀设计点的TPU，实际的使用效果怎么样呢？不管设计得有多好，最后还是要拿效果和数据说话。俗话说，是骡子是马，总要拿出来溜溜啊。

Google在TPU的论文里面给出了答案。一方面，在性能上，TPU比现在的CPU、GPU在深度学习的推断任务上，要快15～30倍。而在能耗比上，更是好出30～80倍。另一方面，Google已经用TPU替换了自家数据中心里95%的推断任务，可谓是拿自己的实际业务做了一个明证。

总结延伸

这一讲，我从第一代TPU的设计目标讲起，为你解读了TPU的设计。你可以通过这篇文章，回顾我们过去32讲提到的各种知识点。

第一代TPU，是为了做各种深度学习的推断而设计出来的，并且希望能够尽早上线。这样，Google才能节约现有数据中心里面的大量计算资源。

从深度学习的推断角度来考虑，TPU并不需要太灵活的可编程能力，只要能够迭代完成常见的深度学习推断过程中一层的计算过程就好了。所以，TPU的硬件构造里面，把矩阵乘法、累加器和激活函数都做成了对应的专门的电路。

为了满足深度学习推断功能的响应时间短的需求，TPU设置了很大的使用SRAM的Unified Buffer（UB），就好像一个CPU里面的寄存器一样，能够快速响应对于这些数据的反复读取。

为了让TPU尽可能快地部署在数据中心里面，TPU采用了现有的PCI-E接口，可以和GPU一样直接插在主板上，并且采用了作为一个没有取指令功能的协处理器，就像387之于386一样，仅仅用来进行需要的各种运算。

在整个电路设计的细节层面，TPU也尽可能做到了优化。因为机器学习的推断功能，通常做了数值的归一化，所以对于矩阵乘法的计算精度要求有限，整个矩阵乘法的计算模块采用了8 Bits来表示浮点数，而不是像Intel CPU里那样用上了32 Bits。

最终，综合了种种硬件设计点之后的TPU，做到了在深度学习的推断层面更高的能效比。按照Google论文里

面给出的官方数据，它可以比CPU、GPU快上15～30倍，能耗比更是可以高出30～80倍。而TPU，也最终替代了Google自己的数据中心里，95%的深度学习推断任务。

推荐阅读

既然要深入了解TPU，自然要读一读关于TPU的论文In-Datacenter Performance Analysis of a Tensor Processing Unit。

除了这篇论文之外，你也可以读一读Google官方专门讲解TPU构造的博客文章 An in-depth look at Google’s first Tensor Processing Unit(TPU)。

课后思考

你能想一想，如果我们想要做一个能够进行深度学习模型训练的TPU，我们应该在第一代的TPU的设计之上做怎么样的修改呢？

欢迎留言和我分享你的想法。如果这篇文章对你有收获，你也可以把他分享给你的朋友。

精选留言：

- 胖胖胖 2019-07-10 09:52:20
  训练的话，大量的池化卷积，而且很多网络都是对称的，反向传播损失。虽然矩阵乘法可以并行但一层一层的训练迭代的参数更新的考虑时序信息，可以考虑之前处理进位的方法，在硬件上实现，减少等待前面运算的时间，加快它参数更新吧 [1赞]

- 胖胖胖 2019-07-10 09:43:11
  感觉老师的专栏要结束了呀，收获很大，特别是老师推荐的那些书，比之前自己乱买的书简直好了一万倍，真的学到了理解了一点东西，不像以前完全囫囵吞枣，看了就忘 [1赞]

- 靠人品去赢 2019-07-10 15:00:41
  突然想起来，前一阵挖矿潮，当时候退出的一些挖矿机就是AISC的，就是对挖矿专门处理的TPU，现在深

度学习这方面有没有类似专门的比较出名的TPU，感觉现在大多数还是用显卡来跑深度学习。

- 胖胖胖 2019-07-10 09:40:56
  信息时代，数据的爆炸增长，使得深度学习的方法开始发挥作用，反过来又push计算能力的提升，对于计算的实现，由于大量简单重复，直接搭为固定的电路结构（其实就是之前讲的各种门电路，寄存器的组合加上时钟信号和控制信号），就像微机原理里面提到的 硬件软化和软件硬化，按需求，资源稀缺和收益比决定是硬件实现还是软件实现，但在硬件的改进的过程中还得考虑市场的情况，毕竟要落地之后有收益才能存活下去

- missingmaria 2019-07-10 09:38:33
  搜了一下，竟然没有搜到第二代TPU的技术细节介绍。但是新闻里开发者透露了一句话，“在芯片进行学习训练的过程中，只需要采用固定的模型即可，不需要变动算法”，猜测二代TPU是针对固定算法开发的，在训练具体模型的时候，将几个算法搭载在一起即可

- xindoo 2019-07-10 08:46:24
  训练和推断最大的不同就是训练需要大量的迭代，所以针对训练的tpu肯定是优化迭代，但我具体想不出如何在硬件层面优化迭代。

- Sentry 2019-07-10 08:46:23
  除了响应时间，效能比，还有就是兼容性，尺寸，成本……

- Linuxer 2019-07-10 08:26:43
  进入每个字都认识系列了，硬着头皮看