****

**研究生（数据中心技术）课程论文（报告）**

**题 目：认知型SSD-基于新型数据检索存储架构的**

**深度学习引擎**

**学 号 M201977185**

**姓 名 吴 禹**

**专 业 计算机技术**

**课程指导教师 施展、曾令仿**

**院（系、所） 计算机科学与技术学院**

**2019年 11月 8日**

**认知型SSD-基于新型数据检索存储架构的深度学习引擎**

（Cognitive SSD: A Deep Learning Engine for In-Storage Data Retrieval）

计算机硕1910班 M201977185 吴禹

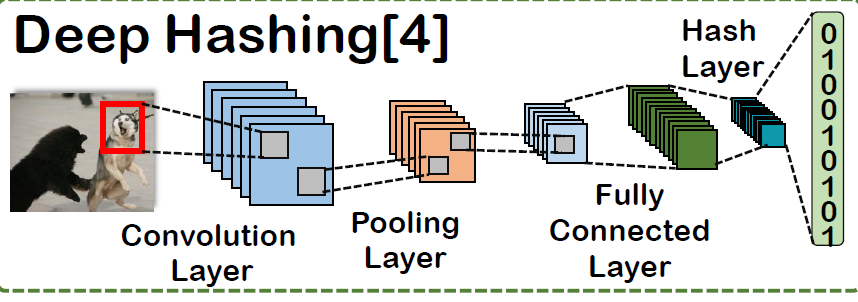
# 引言

本文于2019年7月发表于USENIX会议，由清华大学和中国科学院大学共同参与完成。本文针对传统的数据检索存在的问题进行了改进，提出了一种新的深度学习引擎架构，并通过硬件设计实现对其进行加速，最终设计出以存储设备为核心的用于数据检索的深度学习引擎。本文将对论文内容进行介绍概括，首先针对认知型SSD系统提出背景进行介绍，接着对整个认知型SSD系统的设计概况实现功能以及创新点进行介绍，接着将对其软硬件设计进行讲解，最终将介绍该系统的测试方法和测试结果。

# 1.设计背景及动机

非结构化数据：结构化数据也称作行数据，是由二维表结构来逻辑表达和实现的数据，严格地遵循数据格式与长度规范，主要通过关系型数据库进行存储和管理。与结构化数据相对的是不适于由数据库二维表来表现的非结构化数据，包括所有格式的办公文档、XML、HTML、各类报表、图片和音频、视频信息等。支持非结构化数据的数据库采用多值字段、了字段和变长字段机制进行数据项的创建和管理，广泛应用于全文检索和各种多媒体信息处理领域。目前数据中心中存储用于检索的非结构化数据占百分之八十以上。

传统的非结构化数据检索系统因为其传统框架的限制，将在预处理阶段产生大量的计算，造成大量的时延浪费。基于此，提出了一种基于深度哈希的搜索算法，在传统的深度学习网络的卷积层，激活层，池化层，全连接层后加入了哈希层，通过引入哈希层将可以摆脱复杂的数据预处理阶段，从而实现降低检索功能的时延和提高检索精度。



传统的冯诺依曼体系结构数据的流向存在一定的问题，以计算处理中心为核心，处理器从存储设备中获取数据，经过计算后将数据存入存储设备，这将造成大量的IO开销。过分的以计算为中心将导致性能差，能耗差等问题，基于此，文章提出了以存储设备为核心的近数据处理的体系结构，通过在尽量靠近数据的地方对数据进行处理，减小IO开销，从而降低时延提高计算精度。目前，很多解决方案采用可重构的FPGA实现对硬件的加速，将频繁的操作处理推送到SSD控制器中，本文即在SSD控制器中做了支持近数据处理的可重构的存储控制器。

# 2.解决问题

设计出深度哈希同图型搜索相结合的DLG（deep learning hash + graph search）算法，简化了软件堆栈，避免了数据预处理阶段的大量时延及开销；

提出了以认知型SSD为中心的DLG加速器近数据处理体系结构，通过直接从NAND中访问数据，从而无需跨越多个存储器层次结构从而减少了数据移动带来的时延及开销，最终实现了将深度学习和图型搜索相结合在SSD中进行加速计算，实现精确高效地对非结构化数据进行检索。

它可以实时，低能耗地独立响应数据检索请求。通过扩展到多SSD系统，能显著减少数据中心中大型存储节点的硬件和电源开销。

降低时延，当Cognitive SSD进行横向扩展以形成包括连接的Cognitive SSD阵列的智能轻量级存储节点时，它的性能优于数据中心中使用的常规计算和存储节点。

# 3.系统设计

## 3.1软件设计

作为认知SSD系统的接口，DLG库通过利用NVMe协议中I / O命令进行建立。它包含一个配置库和一个用户库。配置库使管理员可以根据应用程序需求，快速选择并在Cognitive SSD上部署不同的深度学习模型。

到达主机服务器的数据处理请求可以通过调用用户库提供的API向其发送并建立查询会话。然后，认知SSD的嵌入式处理器上的运行时系统接收并解析请求以激活与用户创建的会话相关联的相应DLG-x模块。

管理员可以通过更新DLG-x指令来更新在认知SSD上运行的学习模型. 更新的指令被发送到在NAND闪存中分配的指令区域，并停留在该区域，直到发出型号更改命令（图2中的DLG\_config）为止。权重和图形结构信息的物理地址记录在DLG-x指令中。以这种方式，DLG-x直接在运行时获取所需数据的物理地址，而不是采用地址转换或查找操作，从而减少额外开销的产生。

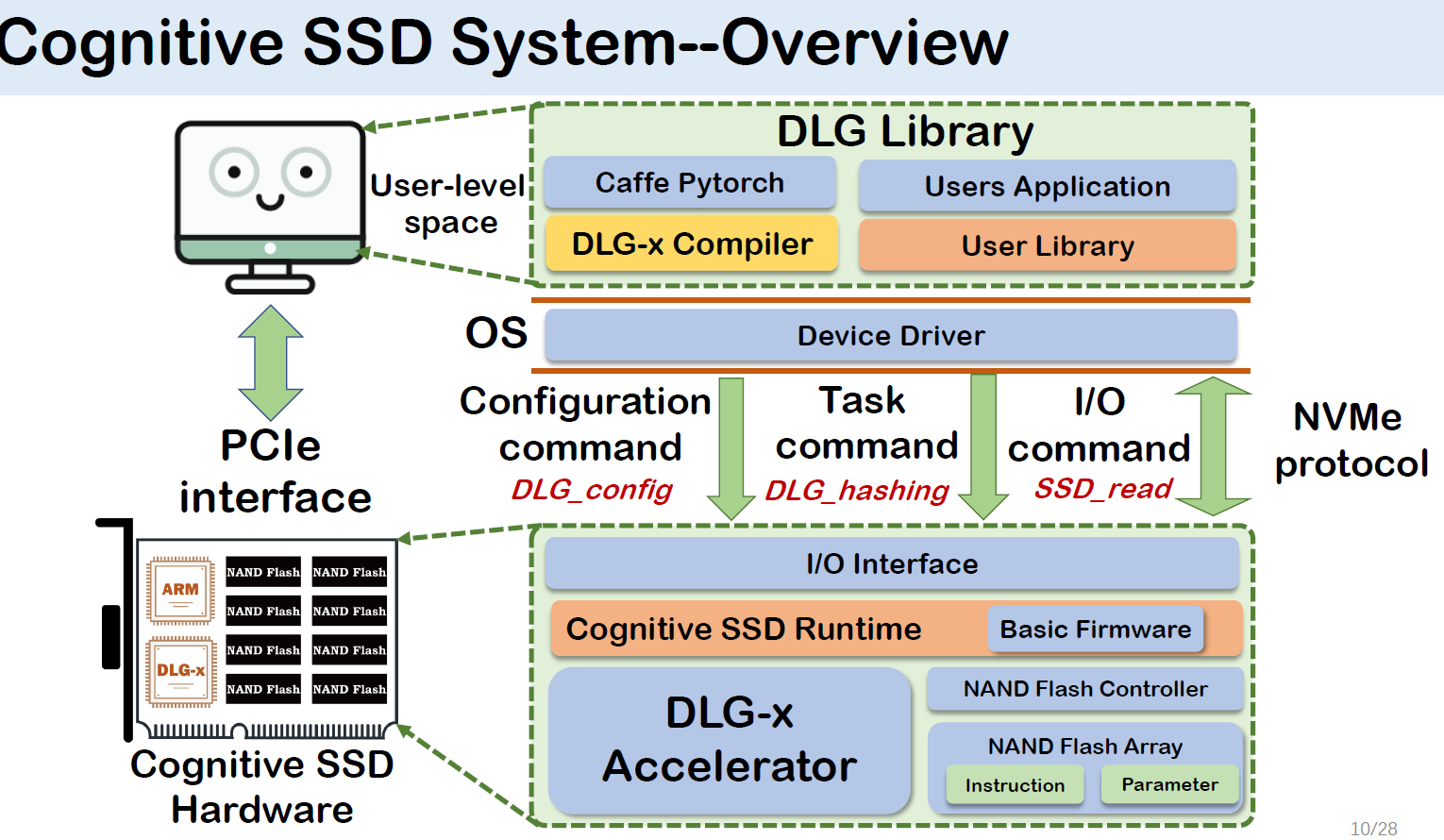


图2

根据功能的不同，将软件系统可进一步细分为数据平面和任务平面。

**数据平面**：数据平面为用户提供SSD\_read和SSD\_write API，以控制主机服务器和Cognitive SSD之间的数据传输。这两个命令绕过闪存转换层直接在物理地址上运行。用户可以调用这些API，以根据数据地址和数据大小的参数将用户发送的数据注入到认知SSD上的数据缓存区域或NAND闪存。

**任务平面**：为了提高支持深度哈希神经网络和图搜索算法的DLG-x加速器的可伸缩性，我们将DLG-x的功能抽象为用户库任务平面中的三个API：DLG\_hashing，DLG\_index和DLG\_analysis 。这些API分别使用NVMe I / O协议的C0h，C1h和C2h命令建立。

首先，DLG\_hashing API旨在提取输入数据的压缩特征并将其映射到哈希或语义空间，这是数据检索系统的基础，对于其他分析功能（如图像分类或分类）很有用。

其次，DLG\_index API是从DLG-x的图形搜索功能中抽象出来的。最后，DLG\_analysis API允许用户使用深度神经网络的数据分析和处理能力来分析输入数据，并且它还具有用于用户定义功能的保留字段。如图3所示，这些任务API是Cognitive SSD提供的关键近数据处理内核的抽象，可以独立或组合调用它们以开发不同的SSD数据处理功能。  
例如，用户可以结合使用DLG\_hashing和DLG\_index API来在大型数据库上完成数据检索，其中DLG\_hashing将查询数据的功能映射到哈希码，而DLG\_index使用它来搜索前T个相似的实例。

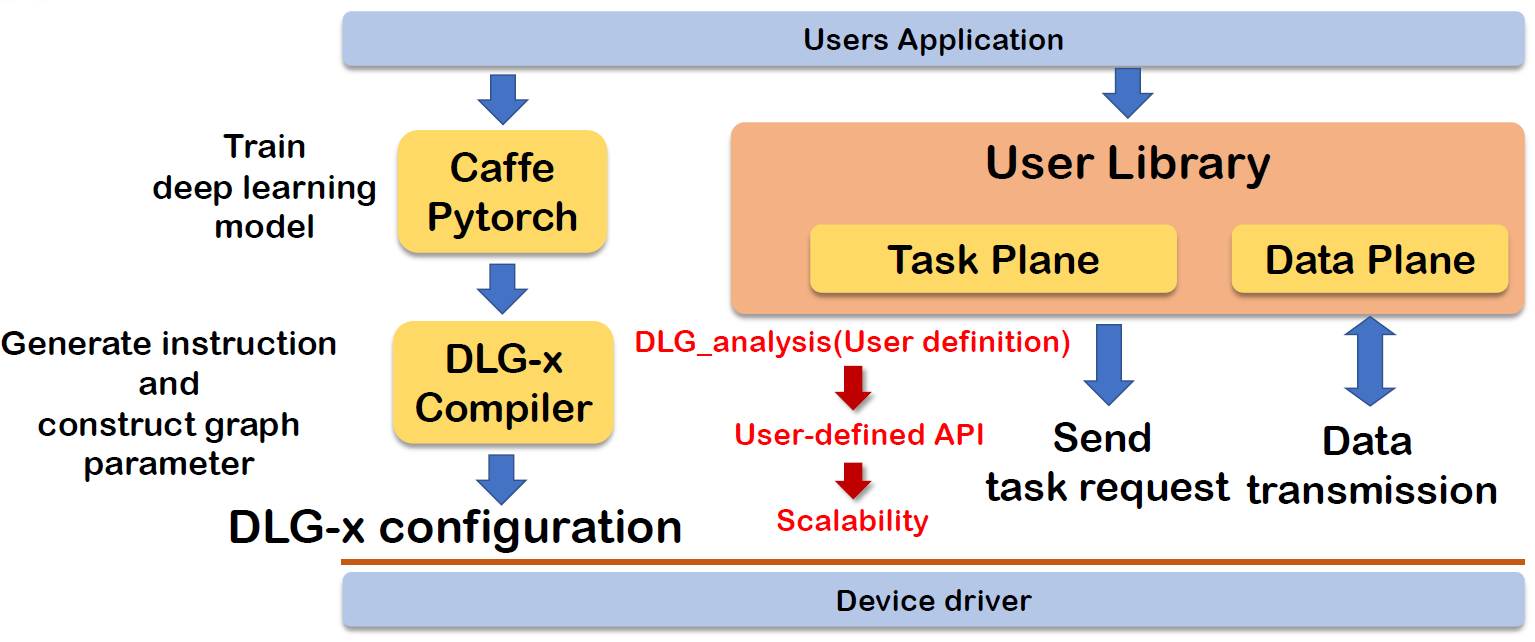


图3

**认知型SSD固件**

部署在Cognitive SSD内部嵌入式处理器上的固件负责通过PCIe接口管理传入的扩展I / O命令。它还将与API相关的命令转换为DLG-x加速器的机器指令，并处理NAND闪存的基本操作，DLG任务计划程序响应任务平面中支持的用户请求，并在Cognitive SSD中启动相应的任务会话。

## 3.2硬件架构

图4

图4描绘了认知性SSD的硬件体系结构，它由嵌入式处理器，DLG-x加速器和连接到闪存芯片的NAND闪存控制器构成。每个NAND闪存控制器连接NAND闪存模块的一个通道，并使用ECC引擎进行纠错。

SSD中的DRAM容量无法满足众多神经单元网络参数的要求，因此这里将NAND闪存直接暴露与DLG-x加速器，该加速器使DLG-x绕过内部DRAM,直接从NAND闪存读取和写入相关的工作数据。

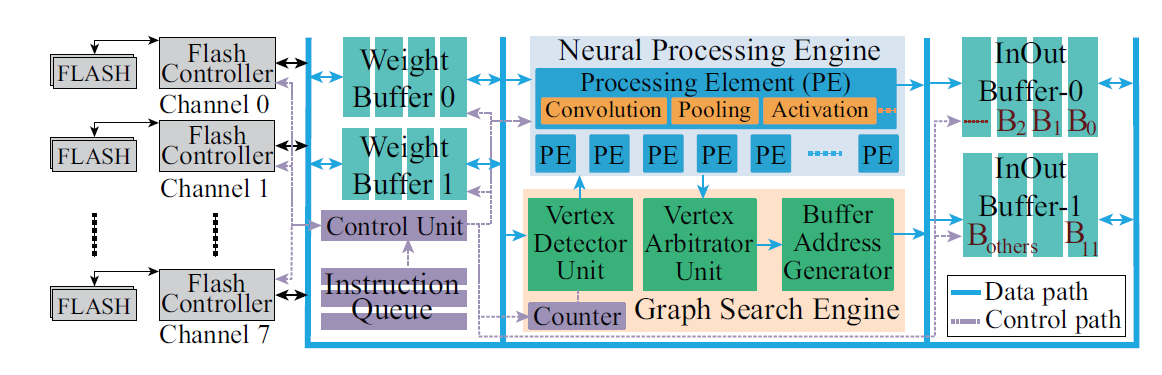


图5

**深度学习单元：**DLG-x加速器具有两个激活的缓冲区（IO缓冲区和权重缓冲区），每个神经网络层的中间结果都临时存储在激活缓冲区里，而权重缓冲区充当神经处理引擎（NPE）和Flash之间的缓冲桥梁，后者将大量的神经参数输出到NPE。NPE包括一组处理引擎（PE），可以执行定点卷积，池化和激活功能操作。 图形搜索引擎（GSE）与NPE合作，NPE和GSE均由从内存中提取指令的控制单元管理。考虑到NAND闪存的I / O操作粒度，这里重新组织了神经网络的数据布局，包括静态参数和中间特征数据，以利用较高的内部闪存带宽。

闪存设备中的数据布局：基于多通道数据传输的带宽分析，我们提出了闪存感知数据布局，以利用高级NAND闪存命令读取页面缓存命令来充分利用闪存带宽。NAND闪存制造商提供的读取页面缓存顺序命令可以将块中的下一页连续加载到数据寄存器中，同时将前一页从中读取到DLG-x的缓冲区或SSD的缓存区域中缓存寄存器。因此，基于具有提供的页面缓存命令的NAND闪存架构，我们选择拆分卷积内核并将其存储到闪存设备中以进行并行提取。超过页面大小的卷积内核被放置在NAND闪存中的连续地址空间中，因为cache命令无需任何额外的地址或操作即可自动读取下一页。

以Hash-AlexNet为例分析数据流，当请求到达DLG-x加速器时，输入数据和第一卷积层的第一内核并行传输到InOut缓冲区-0和权重缓冲区- 0。之后，DLG-x加速器开始计算输出特征图，并将其存储到InOut缓冲区1中。  
当处理第一个内核时，第二个内核将从NAND闪存传输到权重缓冲区1，然后依次传输第三个和第四个内核。一旦hash代码生成，如果DLG任务调度程序解码并调度了以下DLG\_index命令，则将其发送到NPE的图形搜索寄存器以查找与查询数据相似的数据结构。

**图形检索单元：**为了快速准确地建立数据库索引，DLG-x加速器将深度学习和图形搜索机制融合到统一的硬件中，并重复使用计算和内存资源以提高效率。一旦生成查询数据的哈希码，DLG-x将使用它来初始索引相应的数据图并从图中搜索最接近的数据条目。

图搜索方法起源于导航展开图（NSG），它非常适合用于大型多媒体数据检索的认知SSD的有限存储空间。NSG算法包括离线阶段和在线阶段。如图6所示。，在离线阶段，NSG方法为要检索的存储数据结构构造了有向的K-NN图。在图中，顶点通过保留其ID和哈希码来表示数据条目。唯一ID代表一个文件，哈希码是该文件的特征向量，可以通过提前调用DLG\_hashing API来获得。D（Wid）和哈希码（Whash\_code）的位宽是API中用户可配置的参数。在图中，一个顶点可以连接到许多顶点，这些顶点彼此之间的距离不同。但是，只有顶点的顶部Knbors最接近的顶点才能被定义为其“邻居”，其中Knbors也是可重新配置的参数，使用户可以在精度和检索速度之间进行权衡。

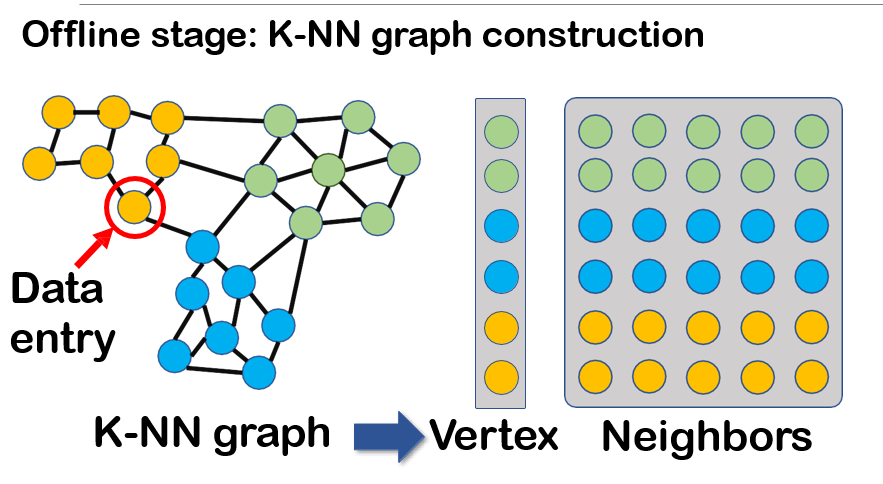


图6 图形检索离线阶段

DLG-x加速器仅加速在线检索阶段，并且数据库更新离线进行，因为后者的任务很少。数据库更新包括哈希码提取阶段和K-NN图构建阶段，其中前者由DLG-x加速器加速，而后者由CPU上的DLG库完成。在离线图形构建中，大约需要10 100秒来更新服务器CPU上百万规模数据的K-NN图形。在线图搜索的架构如图7所示。DLG-x的图形搜索功能从评估图形中随机初始顶点的距离开始，然后将整个图形从邻域的顶点移动到顶点，以找到最接近的结果。

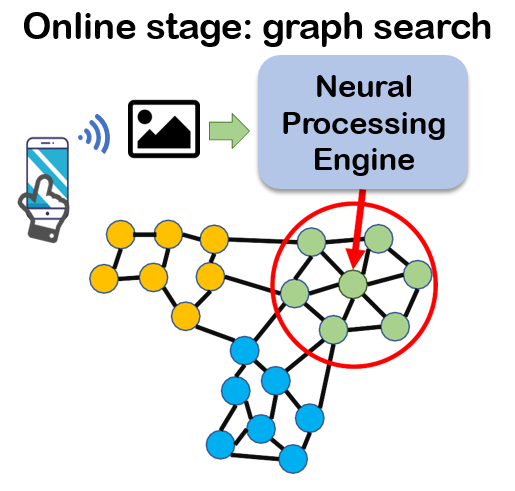


图7 图形检索在线阶段

## 3.3运行流程

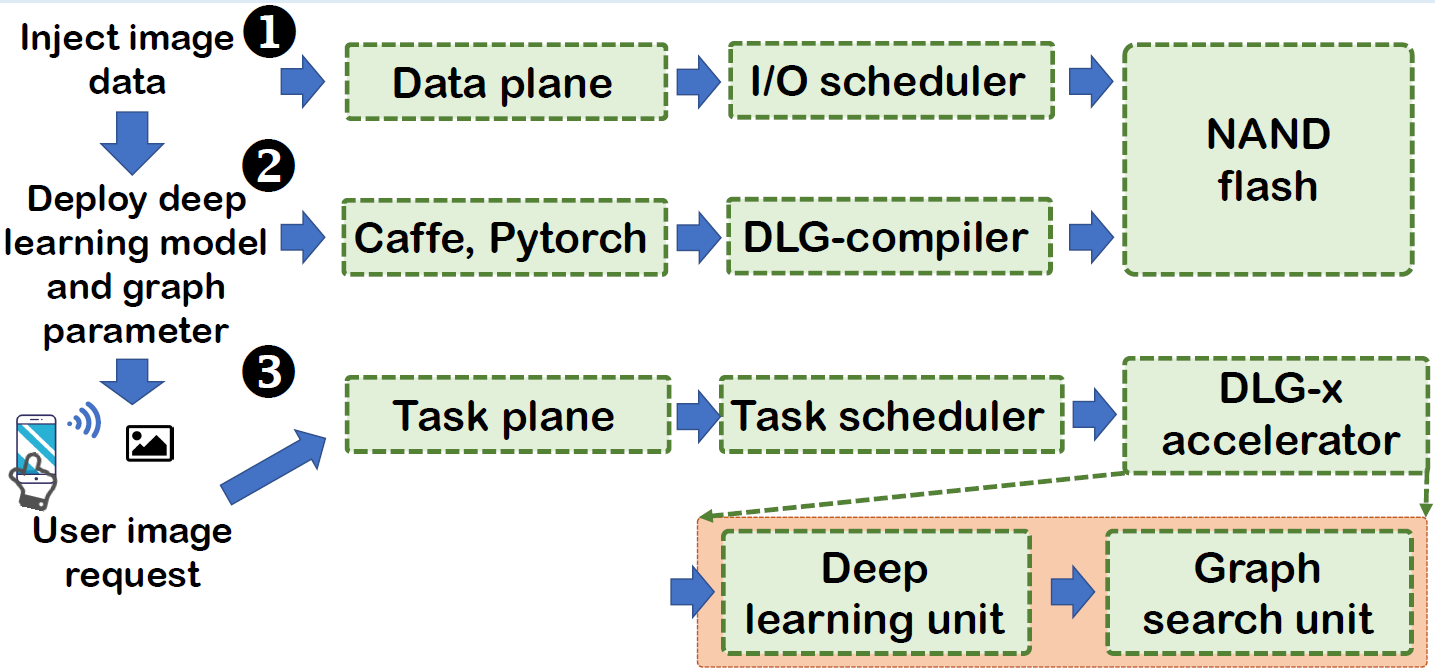
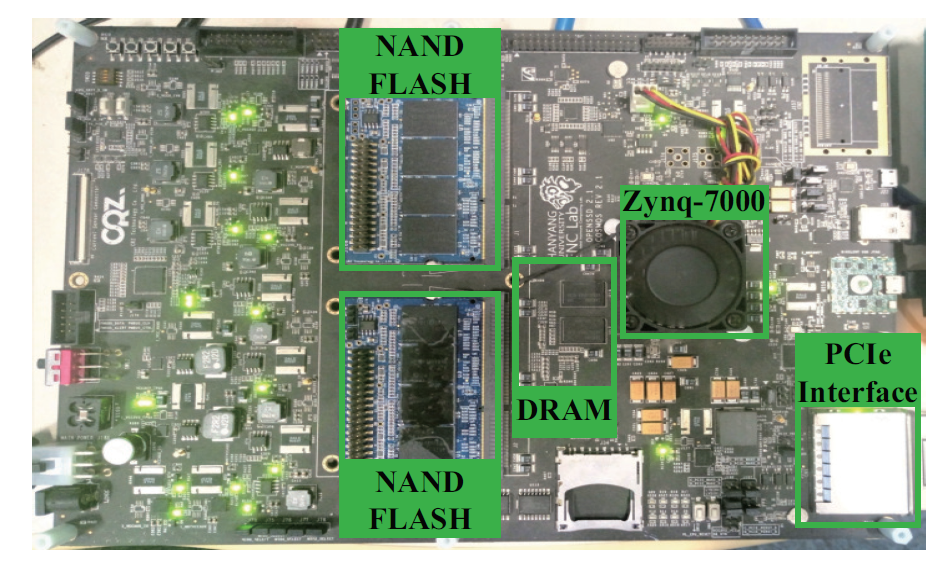
****

图8 用户使用认知性SSD系统流程

图8展示了用户使用认知性SSD系统时，系统的处理过程及数据的流向。

1. 用户通过data plane和IO接口将图形数据存入NAND Flash；
2. 通过DLG-compiler将深度学习框架进行编译存入NAND-Flash
3. 收到请求时，Task scheduler 将启动DLG-x 计数器；
4. DLG-x生成哈希码和查询结果；
5. 通过PCIe接口将NAND-Flash 中的图像ID传输到主机服务器；

# 4.性能评估



实验采用了如下配置：

1.zynq FPGA Chip 作为DLG-x 和闪存的控制器，FPGA内部集成了Dual Cortex A9 处理器；

2. 1GB DRAM

3. 8-channels NAND flash

4.以太网端口

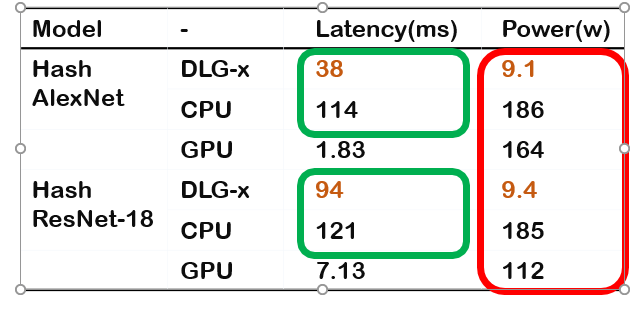
5.PCIe接口

实验平台：

Ubuntu 14.04, Caffe[9], Crow web framework[10].

任务负载：

Content-Based Image Retrieval System (CBIR)



通过在DLG-x和GPU及CPU上运行不同的神经网络模型，比较对应的时延和功耗得出，DLG-x在功耗上具有优势，对比CPU计算神经网络时，亦能提供更低的时延迟。

# 5.结论

* 认知型SSD为非结构化数据提供更好的解决方案
* DLG-x加速器将深度学习和图形搜索集成到一个芯片中，可以直接访问NAND闪存中的数据，而无需跨越多个层次结构。
* 基于FPGA的原型评估表明，Cognitive SSD在功率效率方面优于其他解决方案。

感谢施老师和曾老师这两个多月的认真付出，一周两次的数据中心课程极大地丰富了视野，对研究生初期的我们大有裨益，课堂研讨和实验环节更是让我们受益匪浅。