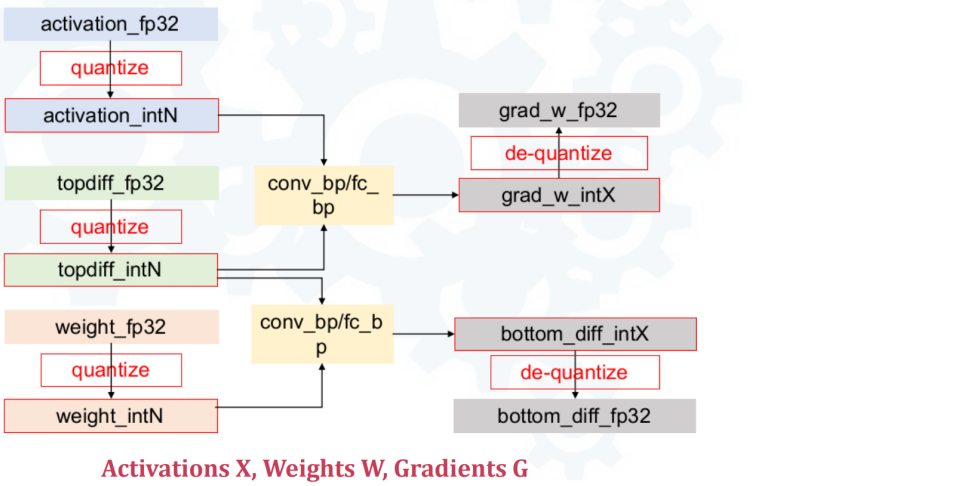
ASP-DAC21 Tutorial

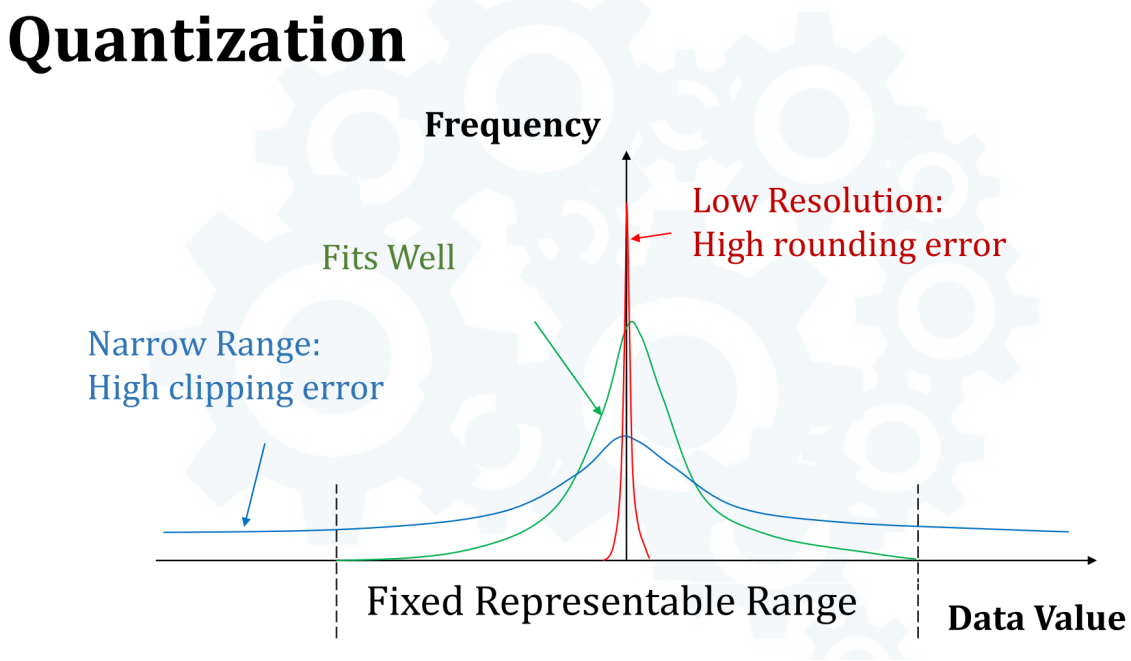
2022/1/5

Talk1 - by Zidong Du

- Dedicated NN processors
  - 提到GPU for Image Processing, DSP for Signal Processing, __ for Intelligence Processing的观点
  - *市场巨大
- 2 set of research work(DNN related architecture work):
  - DianNao
  - Cambricon Series
- Training is critical to AI application
  - time-consuming
- Low bit width
  - smaller size of memory accesses
  - faster computing
  - smaller area of hardware
  - 现有的硬件厂商已部分支持低比特推理（quantization is widely used in inference）
  - **有一张backward量化的图，没太懂（top_diff & bottom_diff）：**

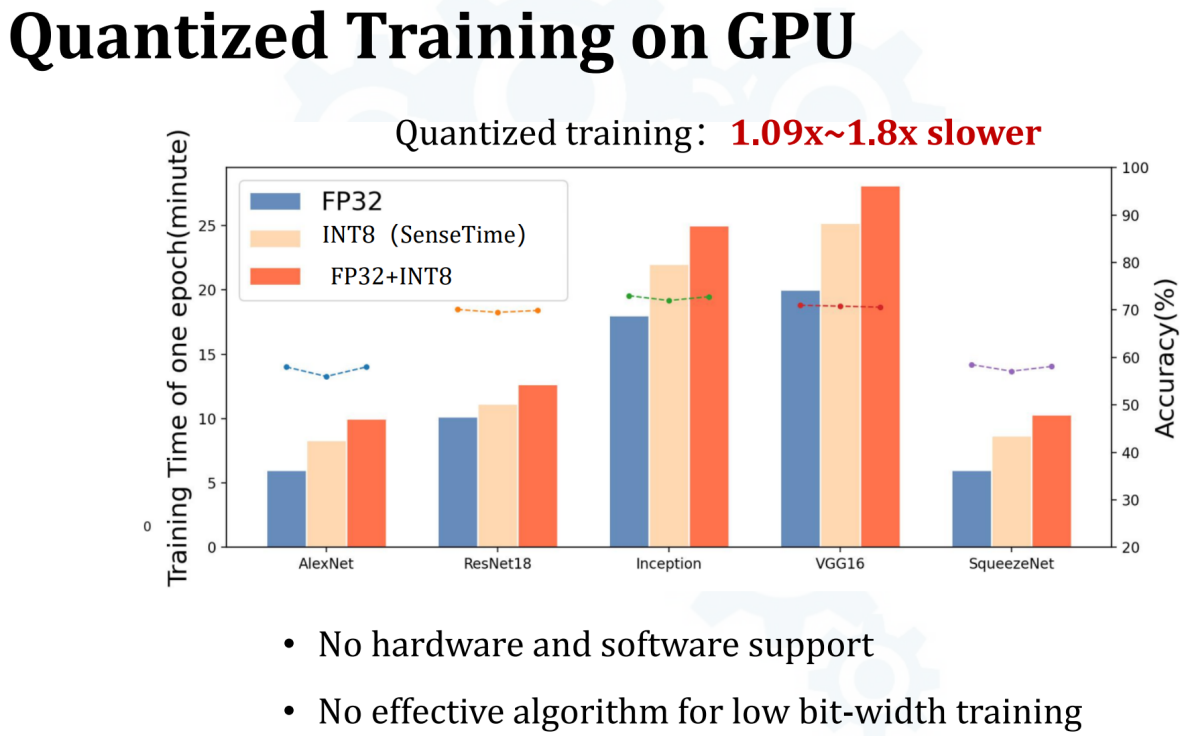


- 数据分布与量化区间的三种关系：



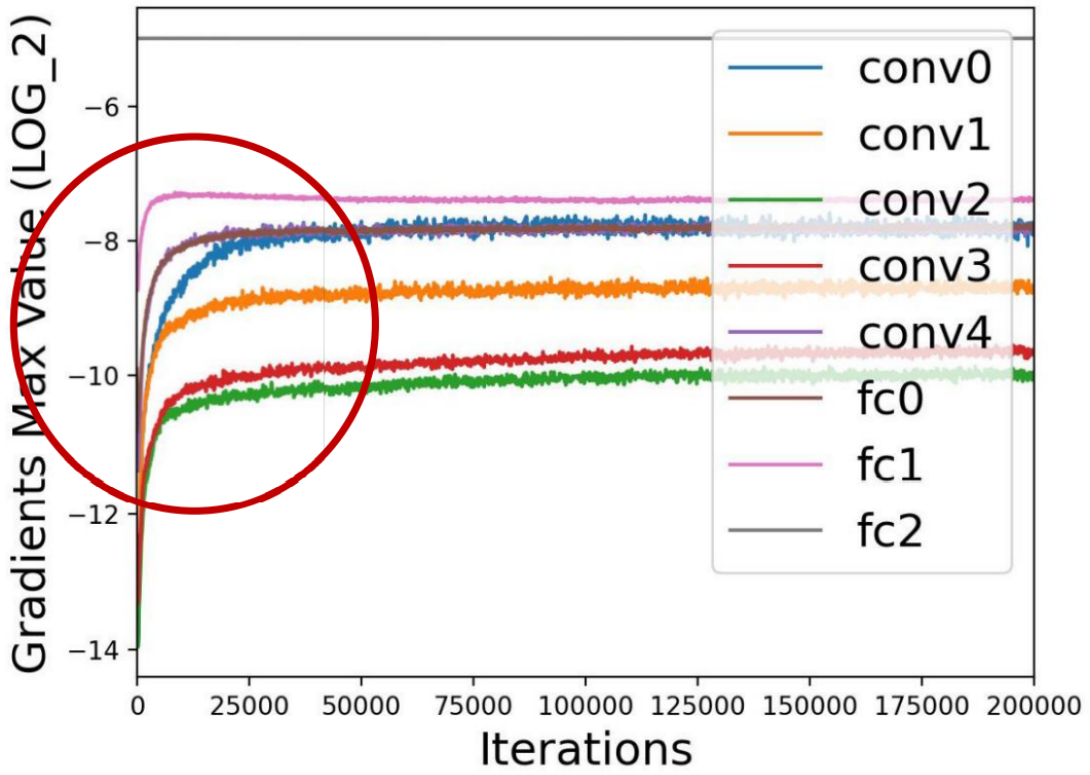
  - 绿色分布合理，rounding error与clipping error都比较小
  - 红色rounding error较大
  - 蓝色clipping error较大

- Quantization Algorithm
  - 现有的算法需要根据统计数字确定量化范围，需要额外的GPU/CPU access开销，因此没有实际的加速效果
  - Quantized Training on GPU - 因此在GPU上Int8训练比FP32训练还慢



  - 缺少软硬件支持（量化）
  - 缺少低比特训练的有效算法支持（G也被量化）

- **参数梯度差异显著**

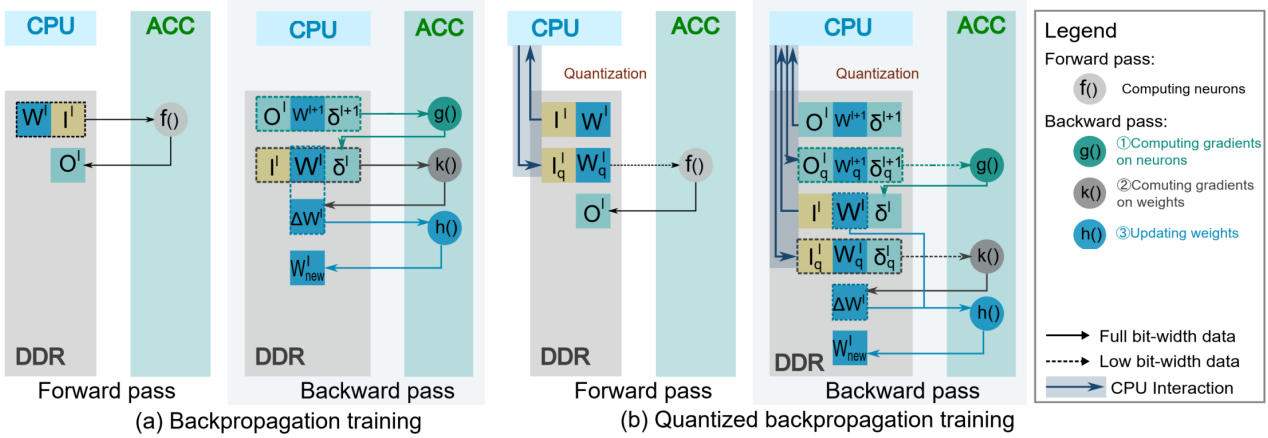


Require statistical information

  - 不同层间梯度可差2数量级，不同epoch间梯度可差3数量级 -> 需要**statistical info**来确定量化范围
  - epoch间梯度量级变化迅速 -> 需要**dynamic quantization**

- CPU + ACC（hardware） - **Dynamic on-the-fly Quantization Training**

CPU+ACC

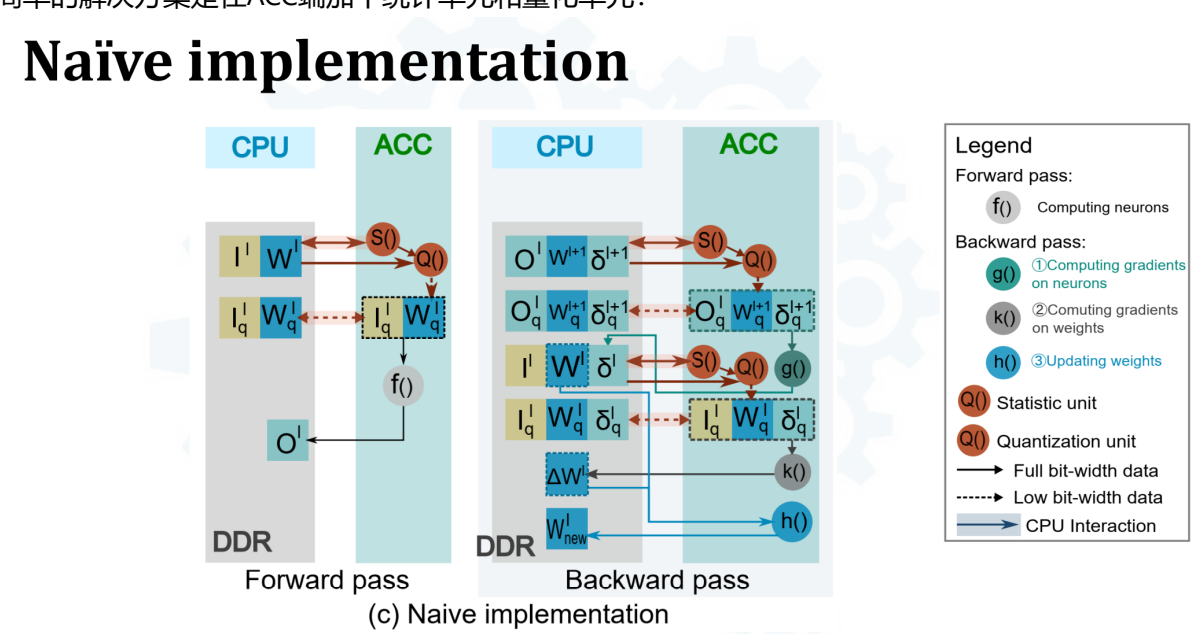


(a) Backpropagation training (b) Quantized backpropagation training

- CPU interactions, e.g., **2.55x** more data
- Slower, e.g., **1.09x~1.8x** on V100

  - FP前向与反向比较简单
    - 前向f()很好理解
    - *g() -> computing gradients on neurons是指?* 关于act的梯度?
    - k() -> computing gradients on weights
    - h() -> update weights

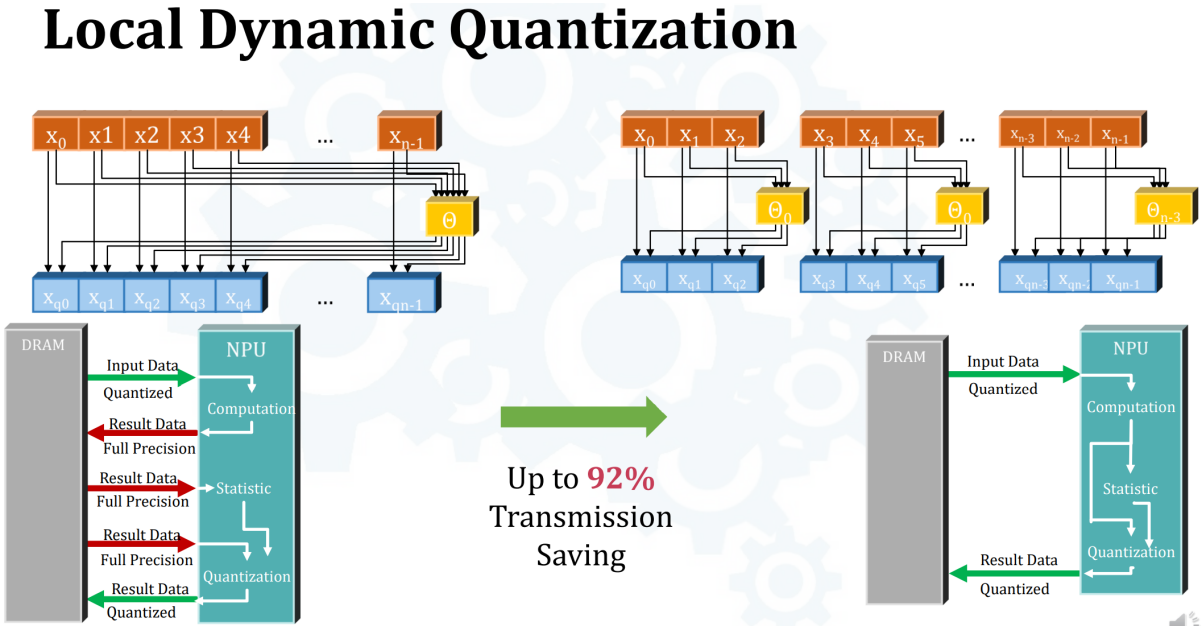
- 量化前向与反向更加复杂
  - 前向时需要由CPU量化W和I，会拖慢速度，而且消耗更多数据获取资源
- 简单的解决方案是在ACC端加个统计单元和量化单元：



(c) Naive implementation

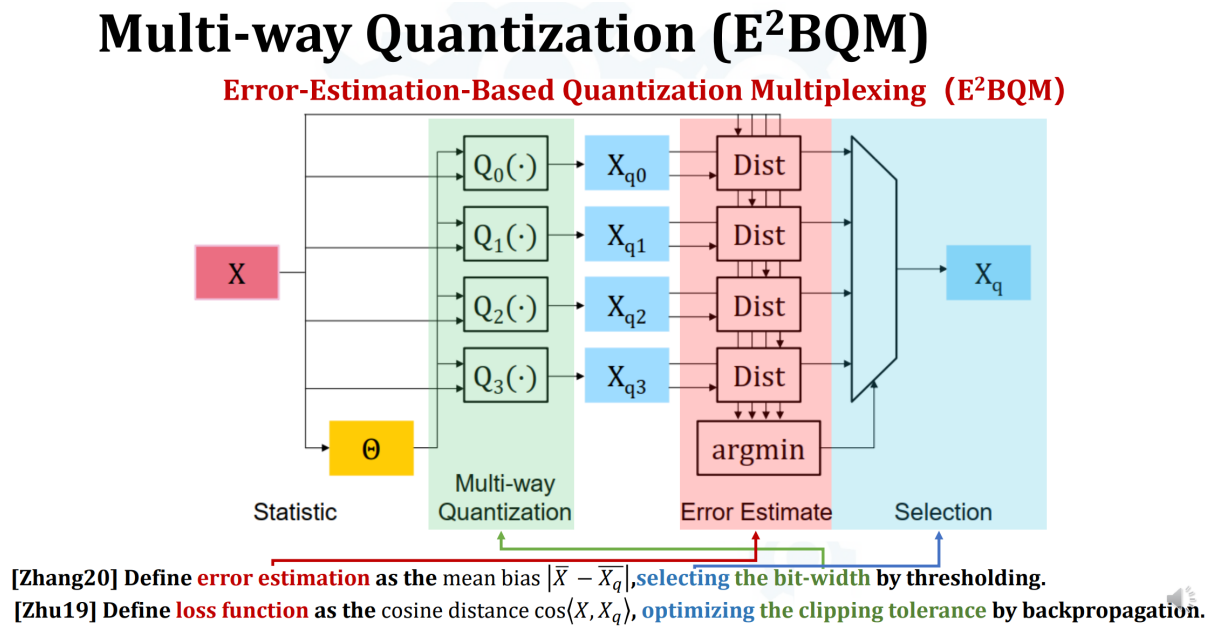
Adding Quantization Units: **two-pass data accesses** (statistic+quantization)

  - 不是最佳方案，因为基于统计数据的量化方案要求统计数据信息，涉及额外的搬运开销 (two-pass data accesses statistic + quantization)
    - 除了往statistic unit和quantization unit搬，还有把量化中间结果搬回DDR的开销
- Local Dynamic Quantization



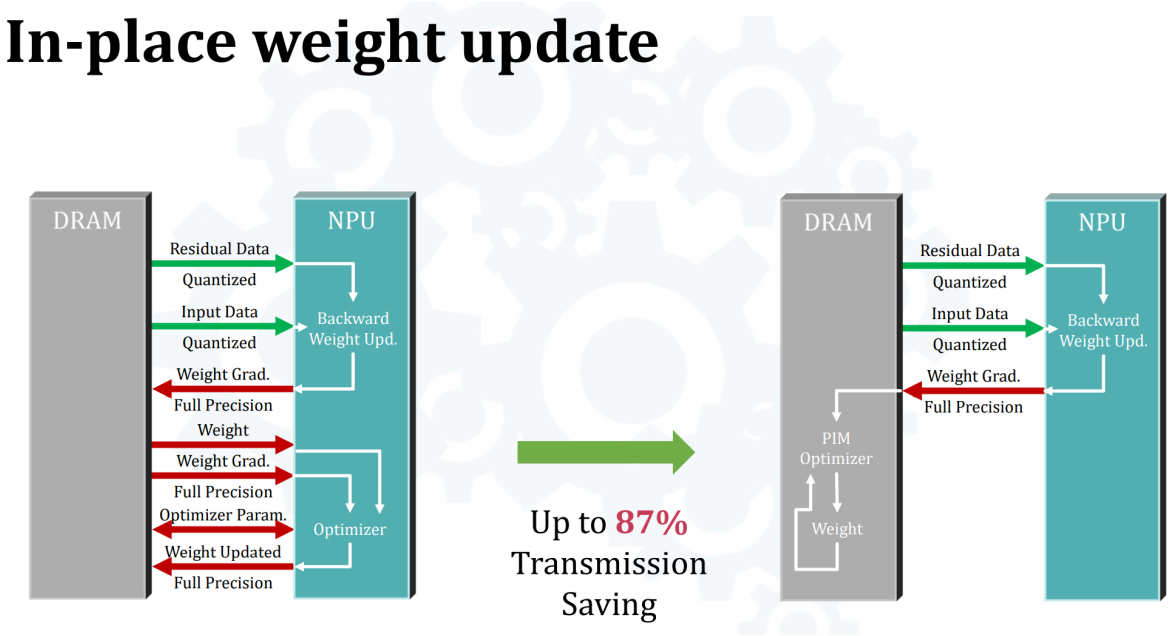
  - 普通基于统计数据的量化需要算一个全局统计特性θ出来，bottleneck
  - 解决方法是locally（把数据分block）计算θ，rounding error会更小 -> **这是把scale粒度变得更小了，scale不就更多了?**

- Multi-way Quantization(E^2BQM)



  - 多路量化，根据不同准则（bit-width, loss）选择用什么方式量化

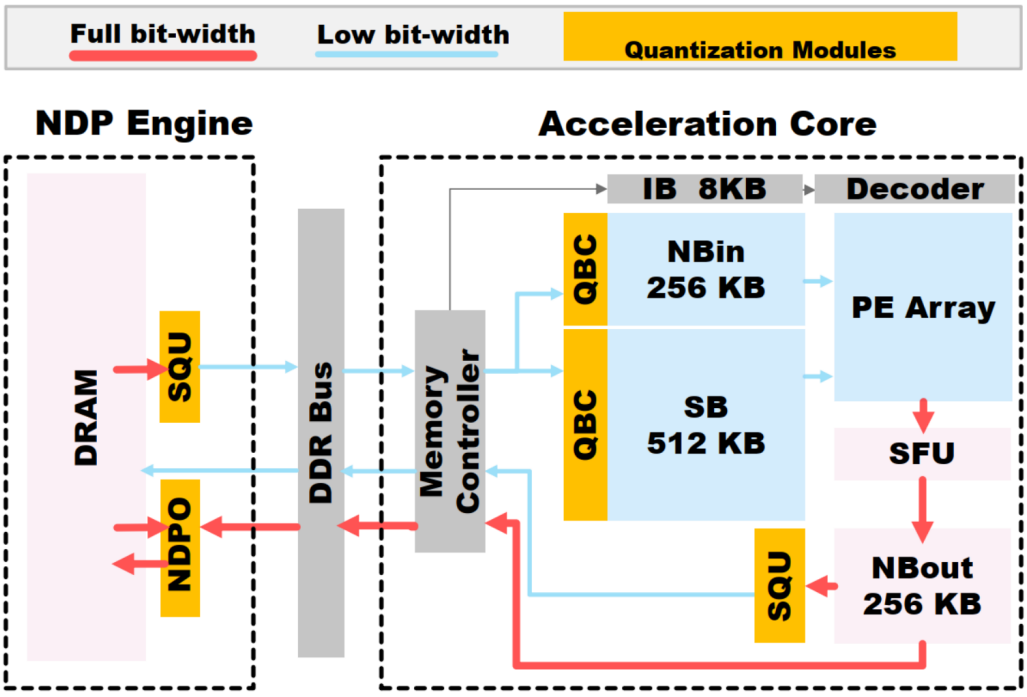
- In-place Weight Update



  - 在DRAM端做参数更新

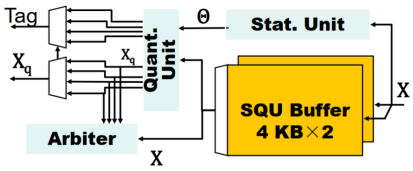
- Cambricon-Q - DNN training arch

Cambricon-Q



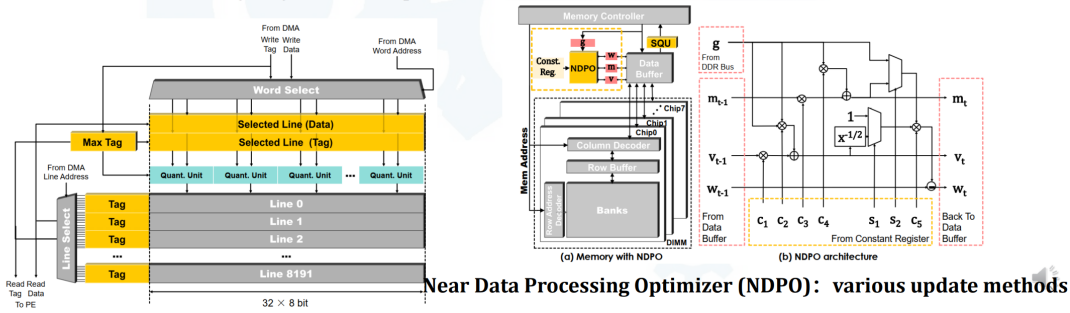
- (**没细听，听了也不懂**)黄框是和上面三个trick结合的结果，三个components(SQU, QBC, NDPO)具体是:

SQU，QBC，NDPO



Statistic&Quantization Unit (SQU)

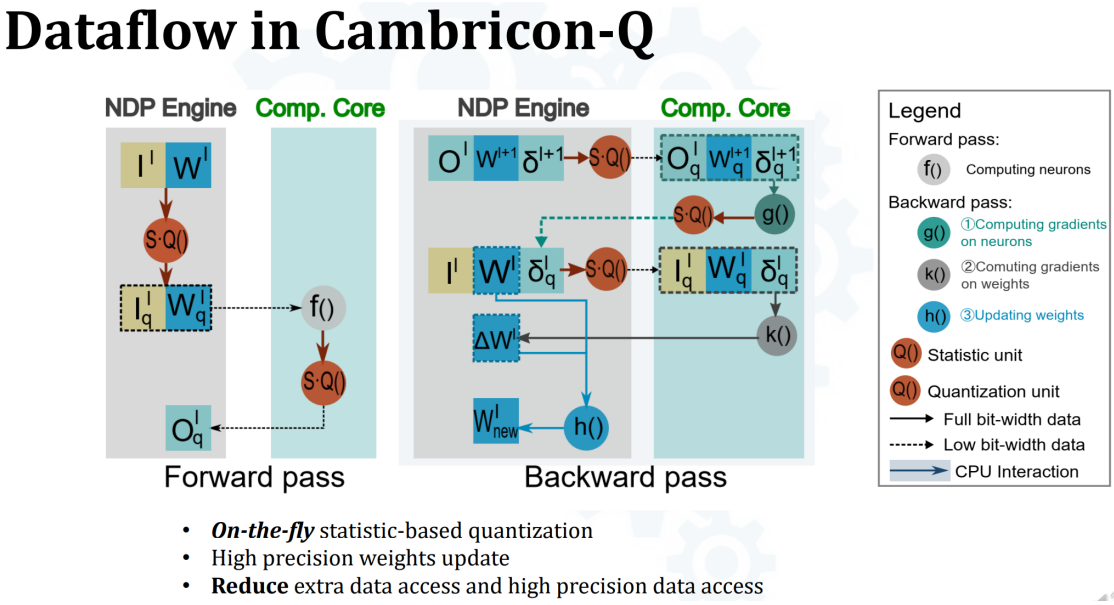
Quantized Buffer Controller (QBC)：different parameters



Near Data Processing Optimizer (NDPO)：various update methods

- SQU - on-the-fly statistics counting & quantization
- QBC - tackling neighboring data that may be split into 2 independent quantization processor with different parameters
- NDPO - weight update

- Cambricon-Q数据通路：



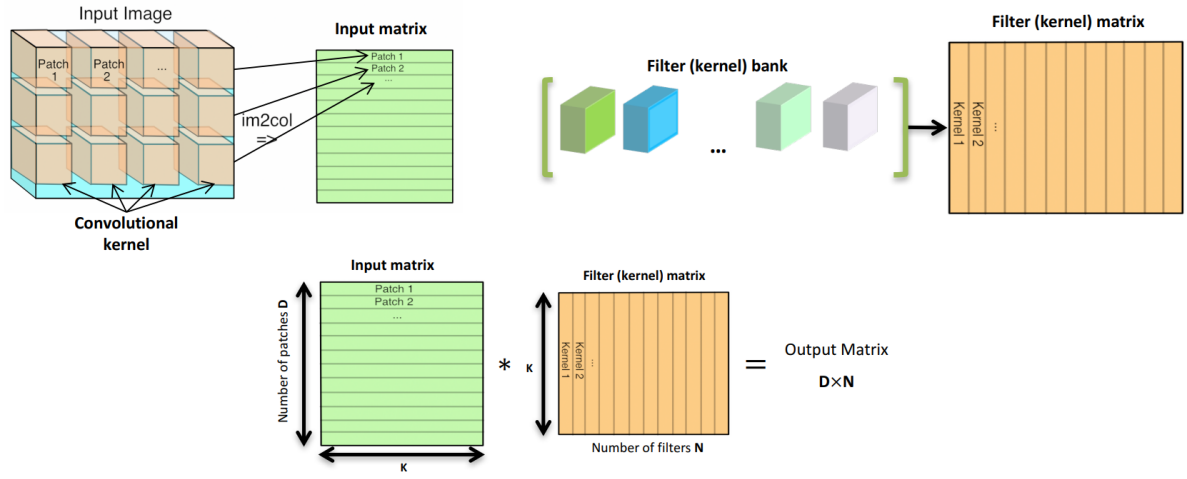
- ***On-the-fly*** statistic-based quantization
- High precision weights update
- **Reduce** extra data access and high precision data access

- **Conclusion**
  - **Cambricon-Q**: A Hybrid Architecture for Efficient Training
    - incorporate 3 units
    - targeting @ on-the-fly statistics quantization DNN training
    - Quantized Training
      - Statistic info
      - Dynamic quant
      - High-precision weight update
    - Existing platform
      - fake quant
      - 2-pass data accesses

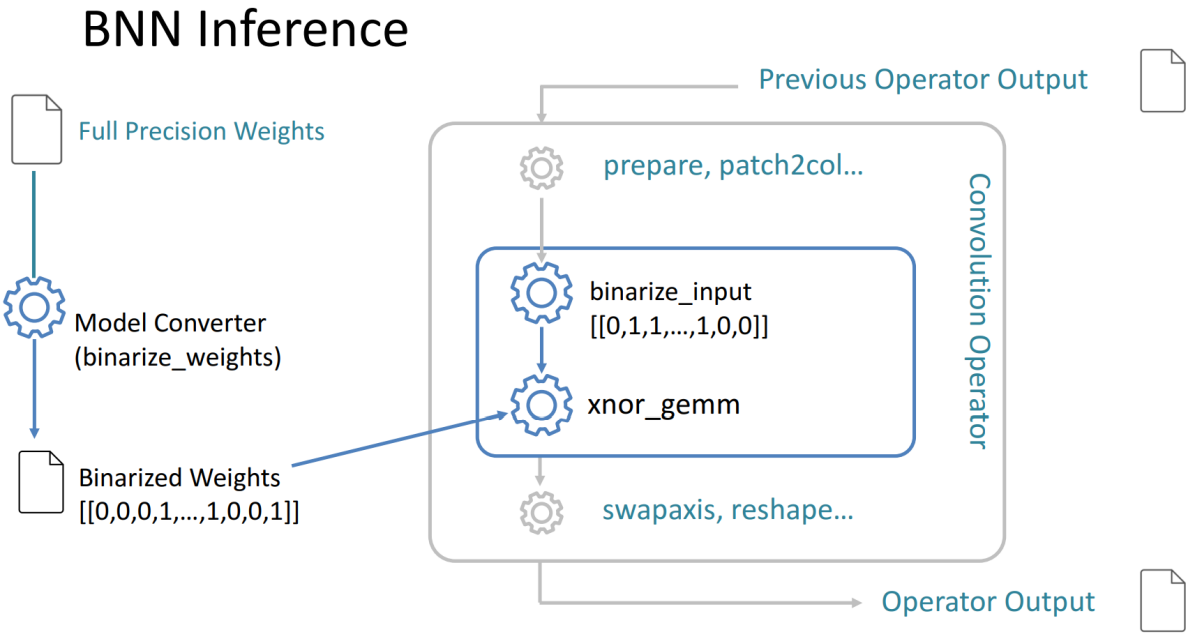
Talk2 - by Haojin Yang

- 经典的BNN/QNN故事：
  - Deep Learning Models is expensive
    - model is large, computation is extensive
    - model training is not environment-friendly
  - DL on Mobile Devices
- QAT & BNN基础与收益 - 无新意

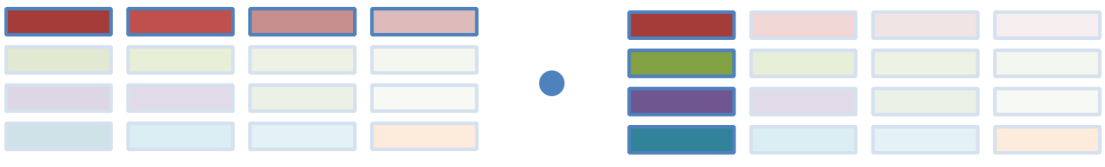
- Im2col for GEMM: 将输入图片映射到colume

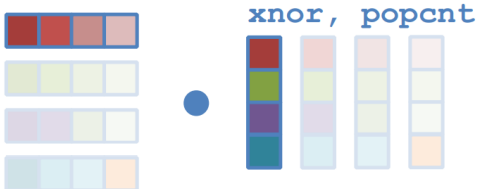


- BNN inference



  - Binarizing **activations row-wise, weights column-wise**
  - 这张图不错，FP infer和Bi infer：





  - 由于硬件的限制，还是需要少量加法

- Challenges of Binary Neural Networks
  - Loss of accuracy
  - BNN's tailor-made optimizer
  - Balancing accuracy and energy consumption
  - Lack of support for solid inference acceleration on heterogeneous hardware
- Classic BNNs（对应于上述第一个问题，减少精度损失）
  - XNOR-Net
    - 一些**over-claim**：
      - 加速58x的claim言过其实（似乎是对比的某种实现在实际中不常用）
      - 并不能在CPU上实时运行
    - 一些没说明的重要细节：
      - 用了1x1 FP downsampling layers
  - ABC-Net & GroupNet
    - 用很多binary bases近似FP value
    - 计算复杂，且不一定有实际加速效果
  - Bi-real Net
    - Binary-real valued information flow design(密集跳连、BN)
    - approx sign func
    - 2-stage training
  - BinaryDenseNet

    Golden rules for training accurate BNNs:
    - Core theory: keep **rich information flow** of the network can effectively compensate the precision loss caused by quantization.
    - Not all the well-known real-valued network architectures can be seamlessly applied for BNNs.
    - Bottleneck design should be eliminated in BNNs.
    - Seriously consider using **full-precision downsampling layer** in your BNNs to preserve the information flow.
    - To overcome bottlenecks of information flow, we should appropriately increase the network width (the dimension of feature maps) while going deeper.
    - Using **shortcut connections** is a straightforward way to **avoid bottlenecks of information flow**, which is particularly essential for BNNs. It has been proved by proposed ***BinaryDenseNet.***
    - Commonly used techniques such as scaling factor, customized gradient, fp initialization are not crucial for well performed BNNs.
  - ReActNet
    - based on MobileNet V1
    - channel-wise reshaping & shifting
    - training tricks - KD,etc.
- BNN Optimizer
  - 依赖latent weights
  - 问题
    - Mismatching of optimization objective

        - unnecessary computation
    - Progressive Binarization
        - （**这篇文章似乎不错，可看？**）TPAMI21 - Gradient Matters: Designing Binarized Neural Networks via Enhanced Information-Flow
        - 逐渐从32bit过渡到1bit

- Balancing Accuracy and Energy Consumption
    - BoolNet

- BNN frameworks

BNN Frameworks

Opensource Framework	Support for		APIs		Demo Tasks				Demo Systems		Implemented SOTA
	1 bit	2-8 bit	Python	C++	Classification	Detection	DLRM	BERT	Android	Raspberry Pi	
dabnn	✘			✘	✘				✘		
Larq	✘		✘		✘				✘	✘	✘
DoReFa	✘	✘	✘								
Bolt	✘			✘	✘						
Riptide	✘	✘	✘		✘					✘	
BMXNet 2	✘	✘	✘	✘	✘	✘			✘	✘	✘
BITorch	✘	✘	✘	✘	✘	✘	✘	✘		✘	✘

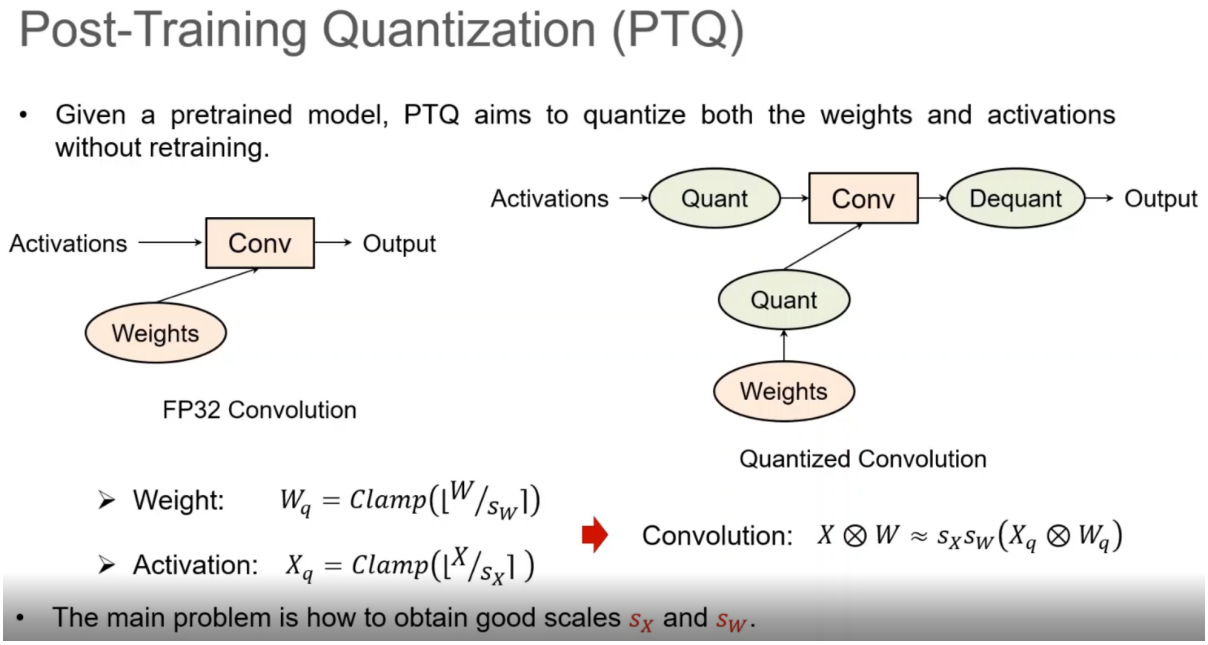
- BMXNet 2: https://github.com/hpi-xnor/BMXNet-v2
- Larq: https://github.com/larq/larq
- Riptide (tf+tvm): https://github.com/jwfromm/Riptide
- Bolt: https://github.com/huawei-noah/bolt
- Dabnn: https://github.com/JDAI-CV/dabnn
- BITorch: https://github.com/hpi-xnor/BITorch

- Future Directions
    - Narrow the gap to full precision counterparts
        - Dedicated architecture design
        - Optimizer for BNN
            - The low-dimensional space is mapped to the high-dimensional space for optimization
            - Adjusts the expression space of binary network
    - Algorithm-Hardware co-design
        - Open platform for BNN performance evaluation on accelerators

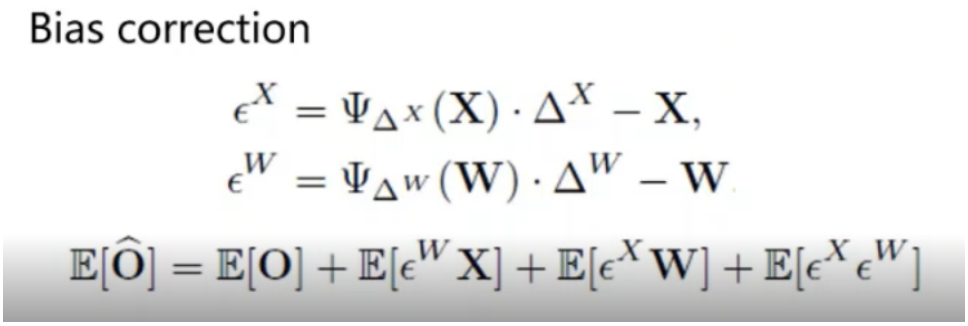
Talk3 - by Kai Han

- PTQ

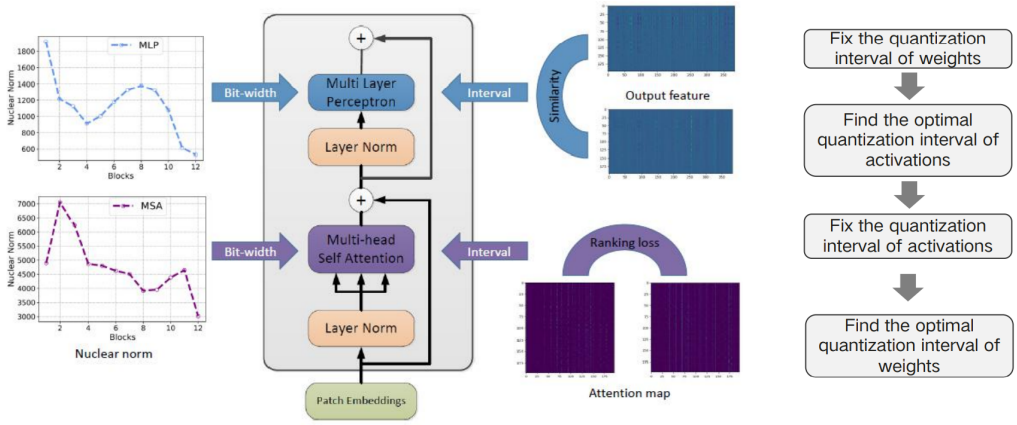
- 主要问题还是确定s_x和s_w：



- ACIQ
- OCS
- PTQ for ViT
  - ViT的量化问题：量化后self-attention信息丢失
  - 解决方案：
    - Ranking-aware quantization (To preserve the functionality)
    - Person Coefficient(Maximize the similarity of the features)
    - **Bias Correction(这里简单介绍了bias correction的作用，23：45-25：00)**
      - 目的是为了减少bias error，如果输出error的期望不是0，则输出的均值会发生改变，这种分布的偏移会导致后续层表现异常（**Q:这种quant error非零的情况应该不能通过BN校正吧？**）
      - 有一说一这里bias correction写得挺好的，同时考虑了W和X的量化误差：



- Mixed-precision



- QAT
  - 将4bit-QAT性能不好归因于the weights are unmatched to the low-bit setting
  - DoReFa-Net
  - PACT
  - LSQ
- BNN
  - FDA-BNN
- Arch for Low-bit
  - Wider Channel
  - NAS + Quant
  - NAS + BNN - BATS
- Conclusion

Conclusion

- PTQ
  - No need for retraining
  - Almost lossless for 8-bit quantization
- QAT
  - Need retraining
  - Almost lossless for 4-bit quantization
- Low-bit Network Architecture
  - Joint NAS and quantization
  - Much higher accuracy for 1-bit quantization