Machine Learning Challenges and Opportunities in Education, Industry, and Research

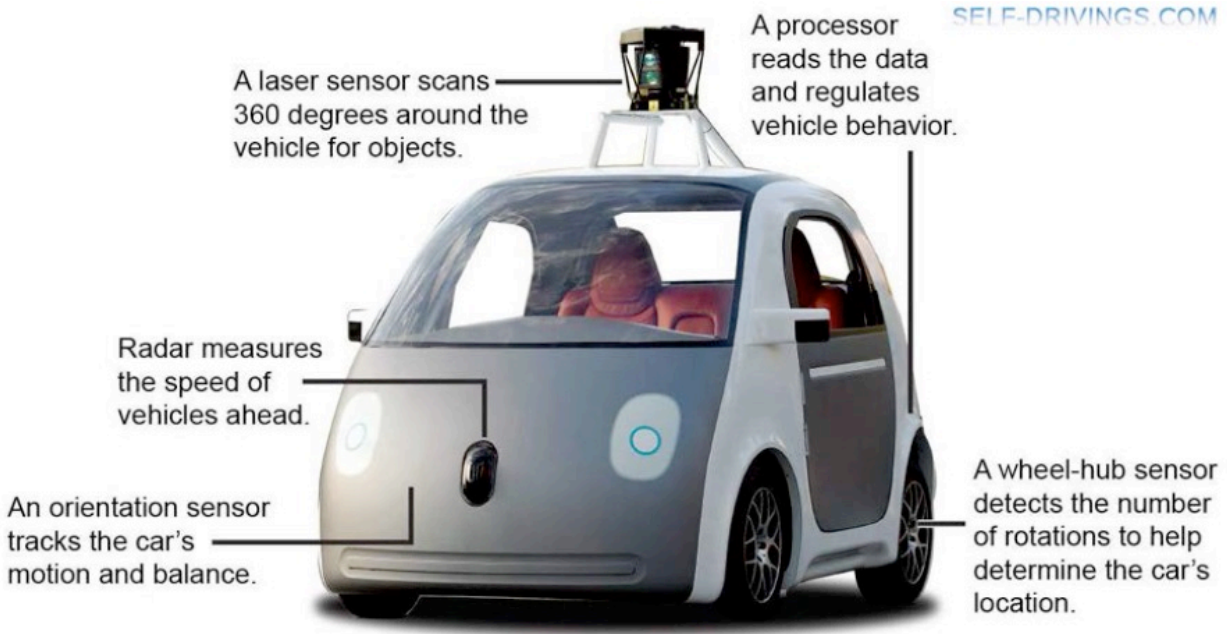
Nader Bagherzadeh
University of California, Irvine
EECS Dept.

AI, ML, and Brain-Like

- **Artificial Intelligence** (AI): The science and engineering of creating intelligent machines. (John McCarthy, 1956)
  - **Machine Learning** (ML): Field of study that gives computers the ability to learn without being explicitly programmed. (Arthur Samuel, 1959); requires large data sets
    - **Brain-Like**: A machine that its operation and design are strongly inspired by how the human brain functions.
      - **Neural Networks**
        - **Deep Learning:** many layers used for data processing
      - **Spiking**

Major Technologies Impacted by Machine Learning

- Data Centers
  - Heterogenous
  - Power
  - Thermal
  - Machine Learning
- Autonomous Vehicles
  - Reliability
  - Cost
  - Safety
  - Machine Learning



Global Market Impact of AI

- The global market for memory and processing semiconductors used in **artificial intelligence (AI) applications will soar to $128.9 billion** in 2025, **three times the $42.8 billion total in 2019**, according to IHS.
- The **AI hardware market will expand at a comparable rate, hitting $68.5 billion by the mid-2020s**, IHS said.

Intuition vs. Computation

- *"A **self-driving car** powered by one of the more popular **artificial intelligence** techniques may need to **crash** into a tree **50,000** times in virtual simulations before learning that it's a bad idea. But **baby wild goats** scrambling around on incredibly steep mountainsides **do not have the luxury of living and dying millions of times** before learning how to climb with sure footing without falling to their deaths."*

- *"Will the Future of AI Learning Depend More on Nature or Nurture?" IEEE Spectrum, October 2017*

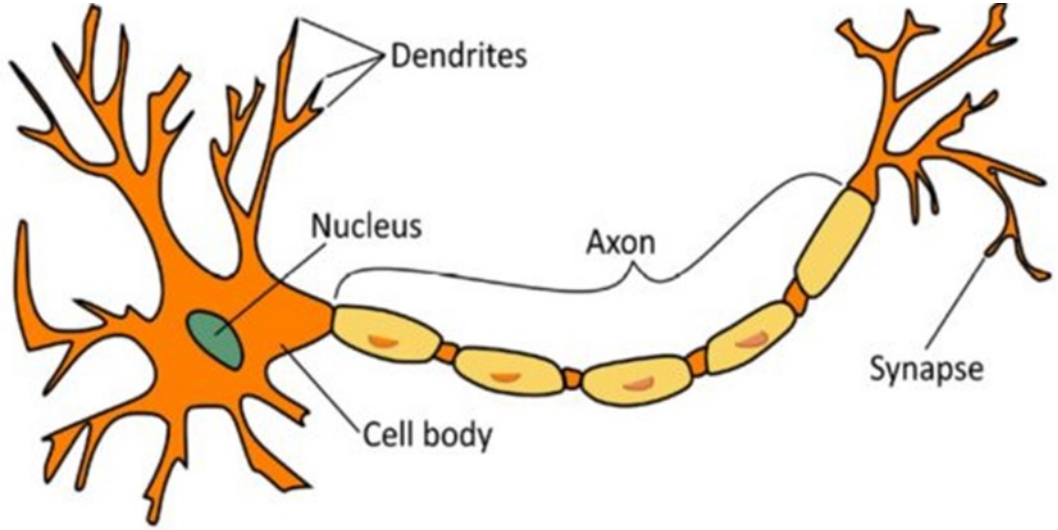
Data Quality Impacts ML Performance

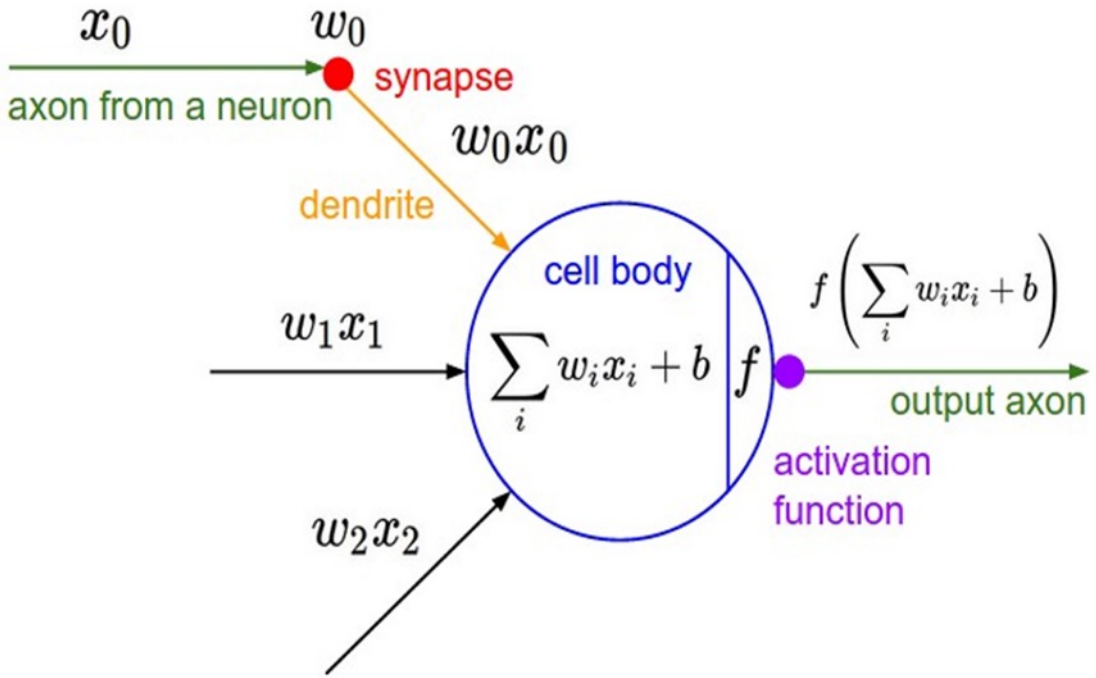
- Garbage-in Garbage-out
- Data must be correct and properly labeled
- Data must be the “right” one; unbiased over the input dynamic range
- Data is training a predictive model, and must meet certain requirements

Fun Facts about Your Brain

- **1.3 Kg neural tissue** that consumes **20% of your body metabolism**
- A supercomputer running at **20 Ws**, instead of **20 MW** for exascale
- Computation and storage is done together locally
- Network of **100 Billion neurons** and **100 Trillion synapses** (connections)
- Neurons accumulate charge like a capacitor (analog), but brain also uses spikes for communication (digital); **brain is mixed signal computing**
- There is **no centralized clock** for processing synchronization
- **Simulating the brain** is very **time consuming** and energy inefficient
- **Direct implementation** in electronics is **more plausible**:
  - 10 femtojules for brain; CMOS gate 0.5 femtojules; synaptic transmission = 20 transistors
  - PIM is closer to the neuron; coexistence of data and computing

Modeling Biological Neurons





Deep Learning Limitation and Advantages

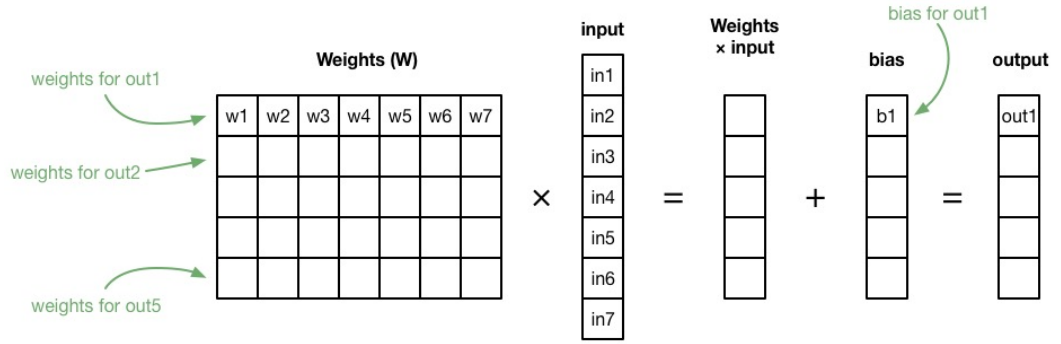
- Better than K-means, liner regression, and others, because it does not require data scientists to identify the features in the data they want to model.
- Related features are identified by the deep learning model itself.
- Deep learning is excellent for language translation but not good at the meaning of the translation.

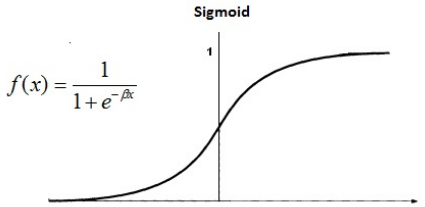
Training and Inference

- **Learning Step**: Weights are produced by training, initially random, using successive approximation that includes backpropagation with gradient descent. Mostly floating-point operations. Time consuming.
- **Inference Step**: Recognition and classifications. More frequently invoked step. Fixed point operation.
- Both steps are many dense matrix vector operations

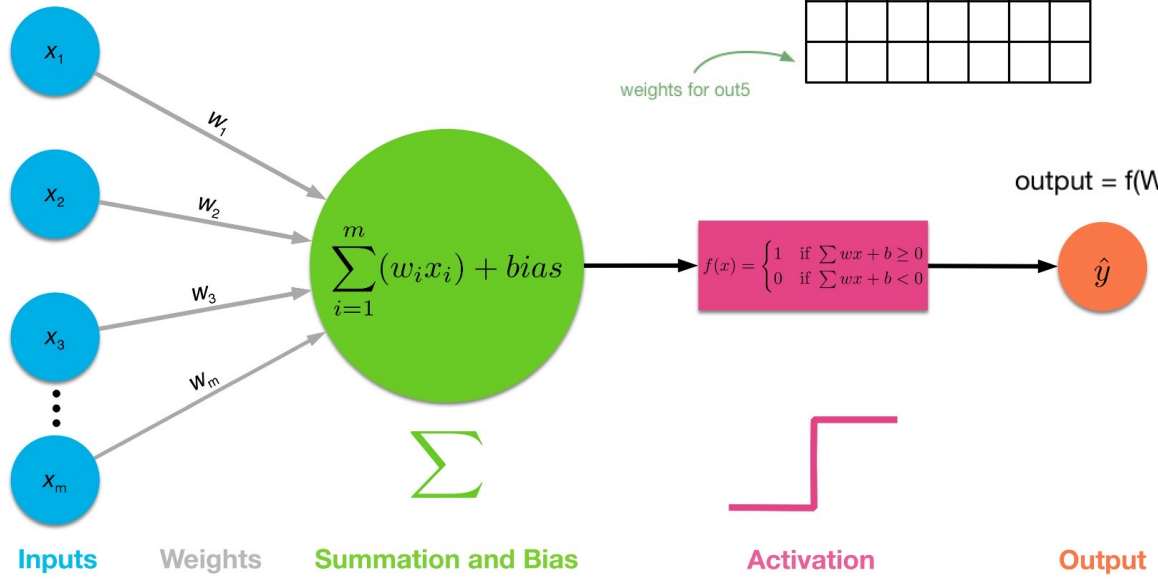
ML Computation is Mostly Matrix Multiplications

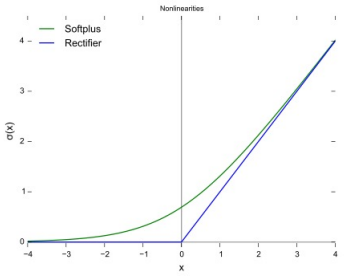
- M by N matrix of weights multiplied by N by 1 vector of inputs
- Need an activation function after this matrix operation: Rectifier, Sigmoid, and etc.
- Matrices are dense





output = f(Weights × input + bias)

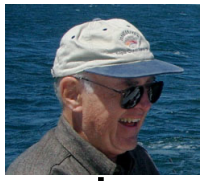




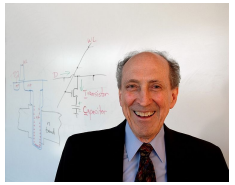
Biological Neurons are not Multiplying and Adding

- We can model them by performing multiply and add operations
- Efficient direct implementation is not impossible but still far away
- Computer architecture in terms of development of accelerators and approximate computation are some of the current solutions
- We are far below in capability what neurons can do in terms of connectivity and the number of active neurons
- VLSI scaling is a limiting factor
- Computing with an accelerators is making a come back

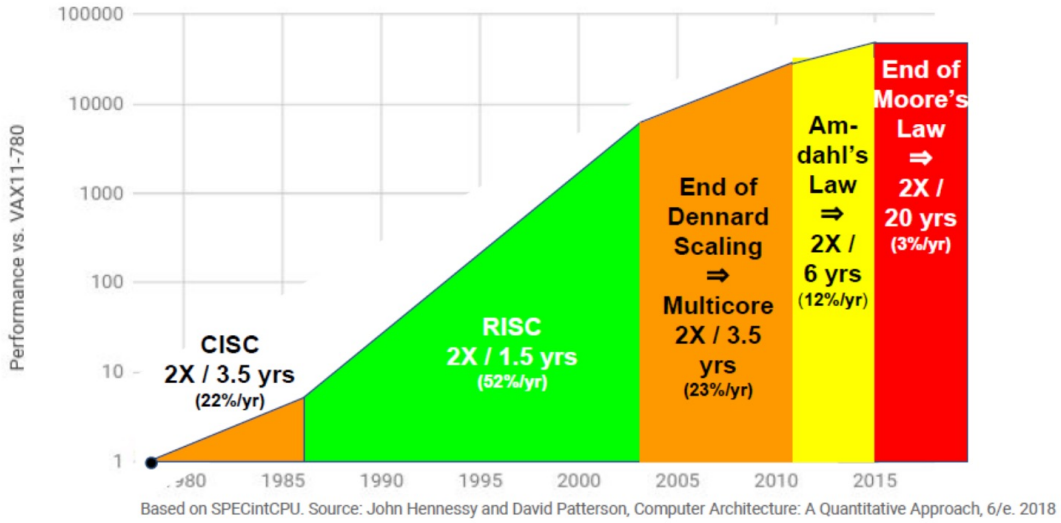
VLSI Laws are not Scaling Anymore



- **Moore's law:**
  - Doubling of the number of transistors every about 18 months is *slowing down*.



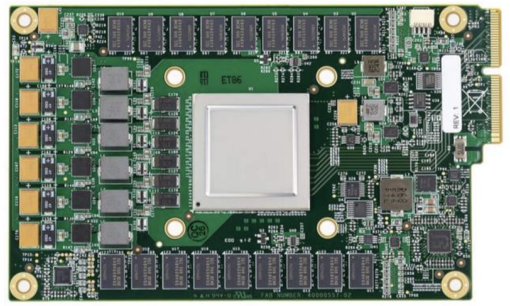
- **Dennard's law:**
  - Transistors shrink =>
    - *L, W* are reduced.
    - Delay is reduced (*T=RC*), frequency increased (*1/f*).
    - *I & V* reduced since they are proportional to *L, W*.
  - Power consumption ($P = C * V^2 * f$) stays the same is *no longer valid*.



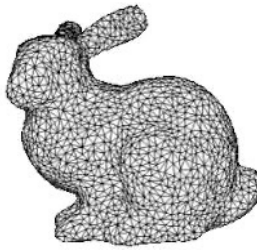
Computer Architecture is Making a Come Back

- **Past success stories**:
  - Clock speed
  - Instruction Level Parallelism (ILP); spatial and temporal; branch prediction
  - Memory hierarchy; cache optimizations
  - Multicores
  - Reconfigurable computing
- **Current efforts**:
  - Accelerators; domain specific ASICs
  - Exotic memories; STT-RAM, RRAM
  - In memory processing
  - Systolic resurrected; non-von Neuman memory access
  - ML is not just algorithms; HW/SW innovations

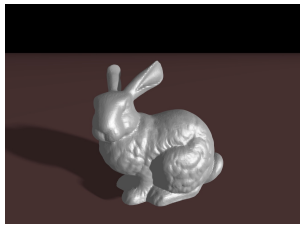
TPU-1



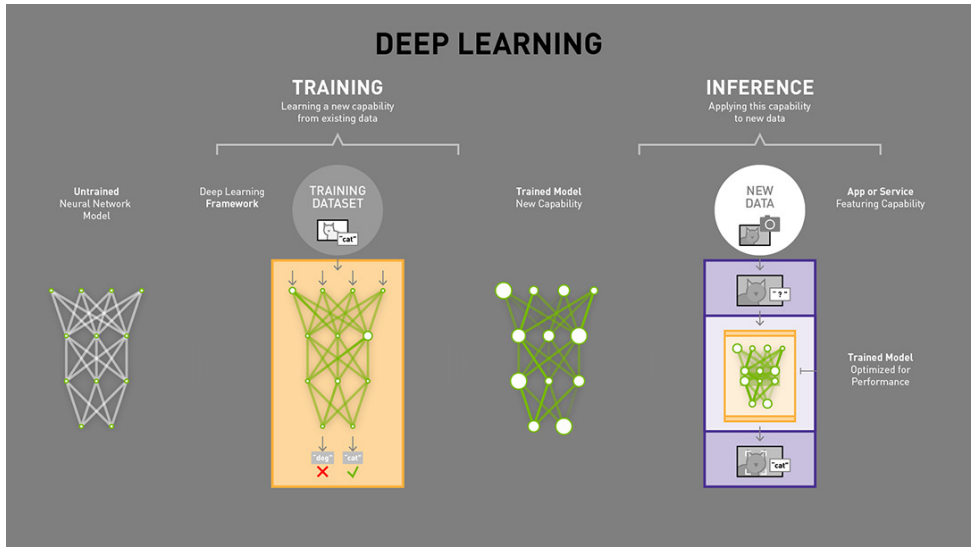
Computation Variety



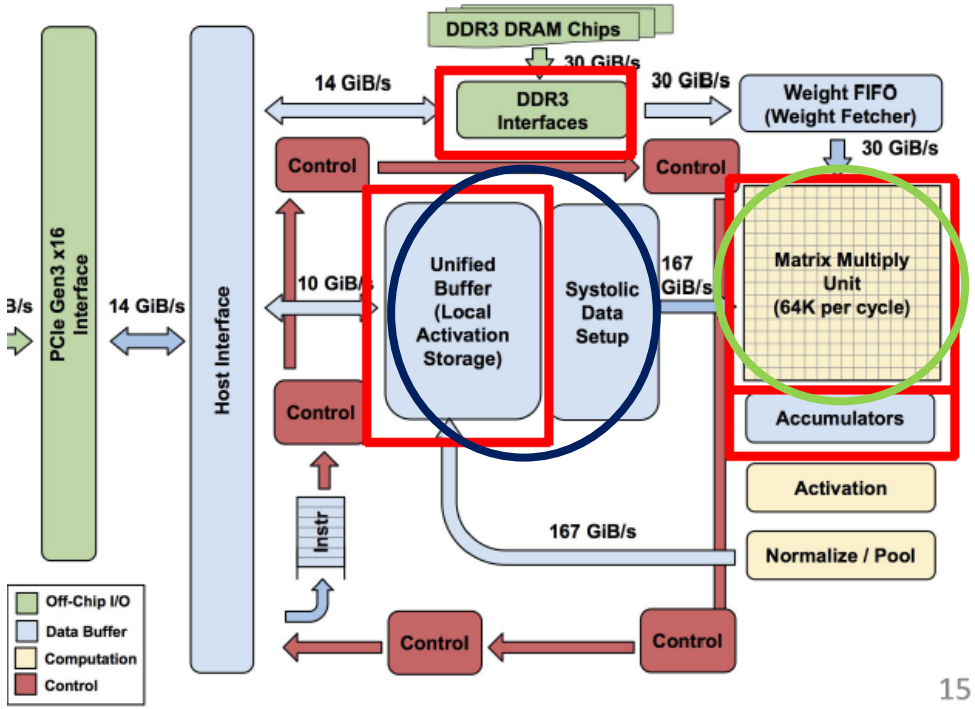
- Graphics
  - Vertex Processing; floating point;
  - Pixel processing and rasterization; integer



- ML
  - Training; floating point
  - Inference; integer



TPU: High-level Chip Architecture



Coarse Grain Reconfigurable

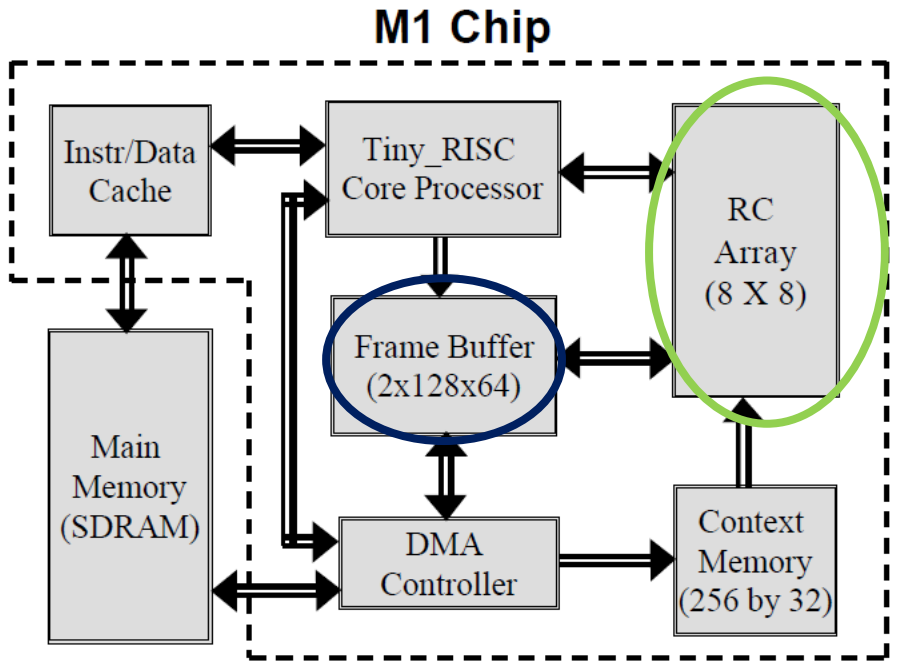
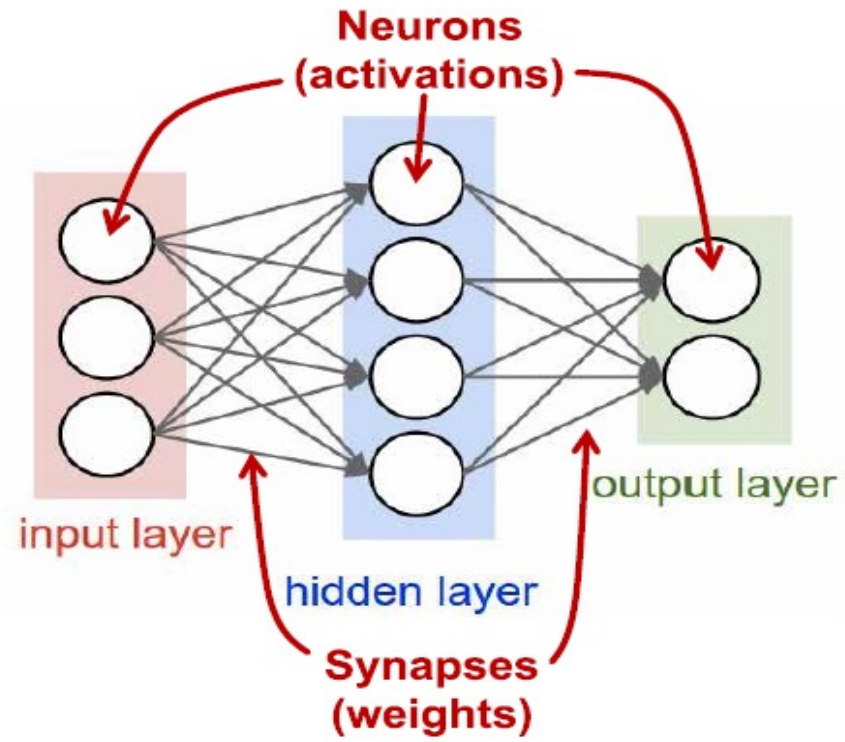
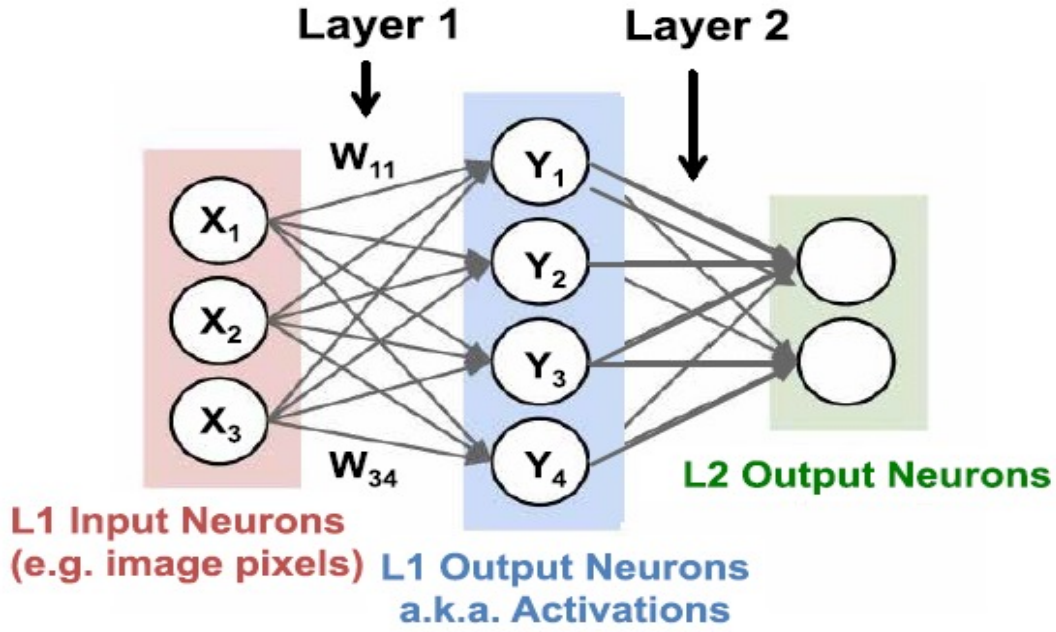


Figure 1: Block diagram of MorphoSys (M1 chip)

Basic of neural network



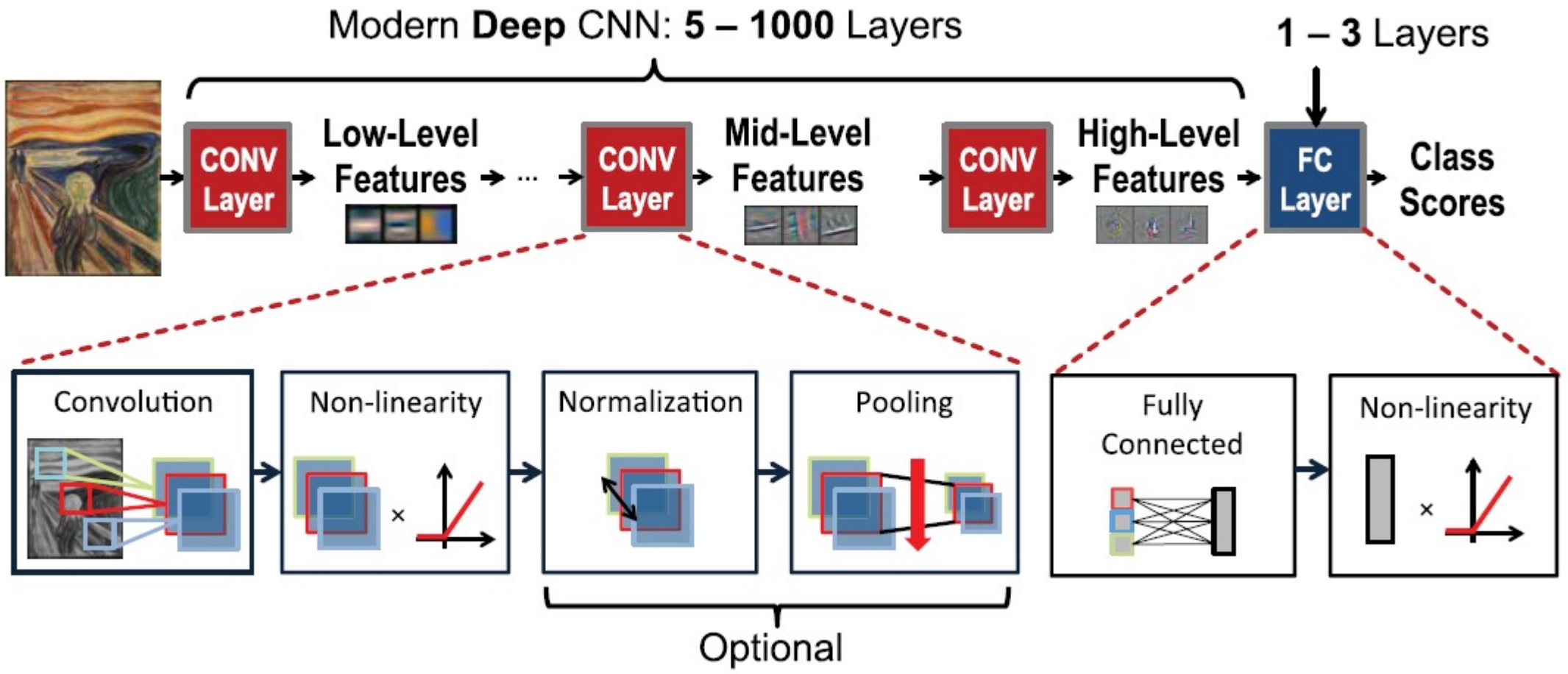


Computation at each layer:

$$y_j = f\left(\sum_{i=1}^{3} W_{ij} \times x_i + b\right)$$

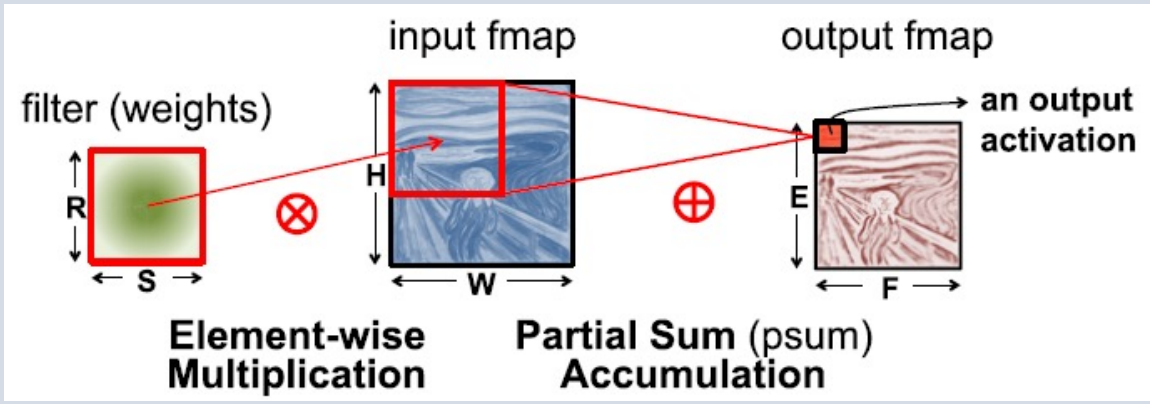
The fundamental elements of neural network's computation are multiply-and-accumulation (MAC) operations which can be easily parallelized.

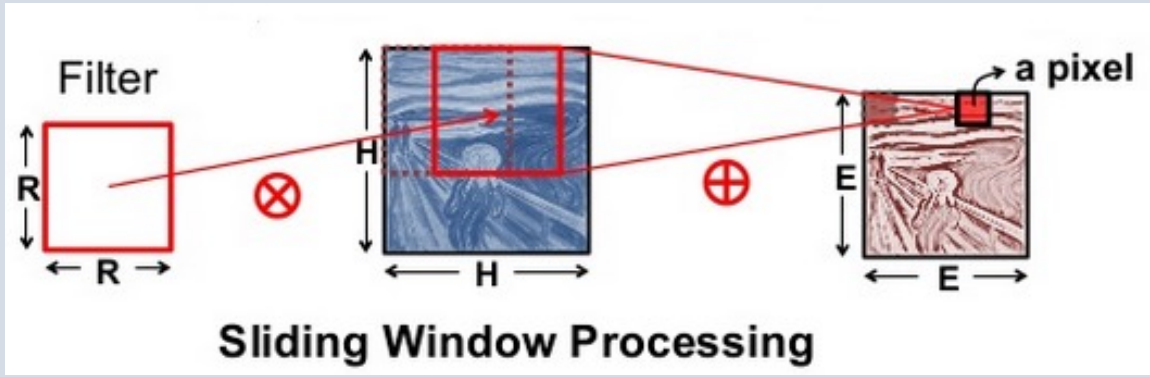
Convolutional neural network



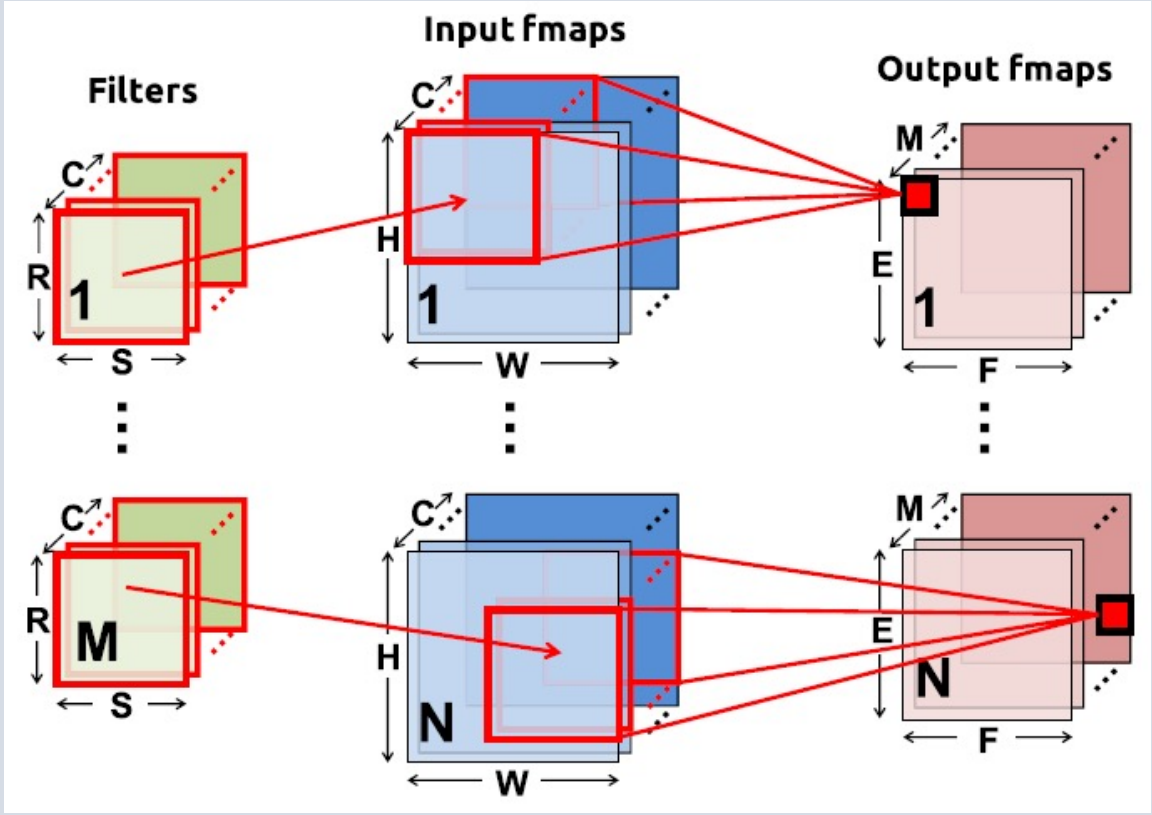
Convolution alone accounts for more than 90% of CNNs' computations.

What is convolution?





2-D convolution in traditional image processing



High dimensional convolutions in convolutional neural network

Convolution (cont.)

a_{11}	a_{12}	a_{13}	a_{14}
a_{21}	a_{22}	a_{23}	a_{24}
a_{31}	a_{32}	a_{33}	a_{34}
a_{41}	a_{42}	a_{43}	a_{44}

$*$

$\times b_{11}$	$\times b_{12}$	$\times b_{13}$
$\times b_{21}$	$\times b_{22}$	$\times b_{23}$
$\times b_{31}$	$\times b_{32}$	$\times b_{33}$

$=$

c_{11}	c_{12}
c_{21}	c_{22}

Output dimension= $\left\lfloor \frac{n+2p-f}{s} \right\rfloor + 1$

$\times b_{11}$ a_{11}	$\times b_{12}$ a_{12}	$\times b_{13}$ a_{13}	$\times b_{13}$ a_{14}
$\times b_{21}$ a_{21}	$\times b_{22}$ a_{22}	$\times b_{23}$ a_{23}	$\times b_{23}$ a_{24}
$\times b_{31}$ a_{31}	$\times b_{32}$ a_{32}	$\times b_{33}$ a_{33}	$\times b_{33}$ a_{34}
$\times b_{31}$ a_{41}	$\times b_{32}$ a_{42}	$\times b_{33}$ a_{43}	$\times b_{33}$ a_{44}

$$c_{11}=b_{11}\times a_{11} + b_{12}\times a_{12}+ b_{13}\times a_{13}+ b_{21}\times a_{21}+ b_{22}\times a_{22}+ b_{23}\times a_{23}+ b_{31}\times a_{31}+ b_{32}\times a_{32}+ b_{33}\times a_{33}$$

$$c_{12}=b_{11}\times a_{12} + b_{12}\times a_{13}+ b_{13}\times a_{14}+ b_{21}\times a_{22}+ b_{22}\times a_{23}+ b_{23}\times a_{24}+ b_{31}\times a_{32}+ b_{32}\times a_{33}+ b_{33}\times a_{34}$$

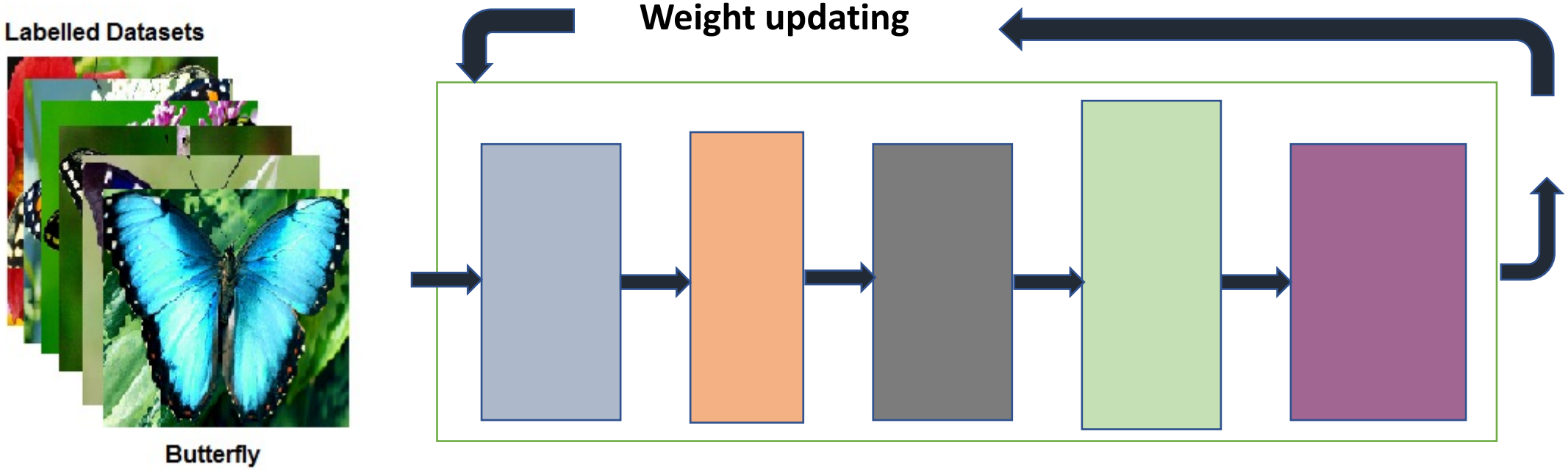
$$c_{21}=b_{11}\times a_{21} + b_{12}\times a_{22}+ b_{13}\times a_{23}+ b_{21}\times a_{31}+ b_{22}\times a_{32}+ b_{23}\times a_{33}+ b_{31}\times a_{41}+ b_{32}\times a_{42}+ b_{33}\times a_{43}$$

$$c_{22}=b_{11}\times a_{22} + b_{12}\times a_{23}+ b_{13}\times a_{24}+ b_{21}\times a_{32}+ b_{22}\times a_{33}+ b_{23}\times a_{34}+ b_{31}\times a_{42}+ b_{32}\times a_{43}+ b_{33}\times a_{44}$$

Training(learning)

❖ A common machine learning algorithm or deep neural networks (DNNs) have two phases:

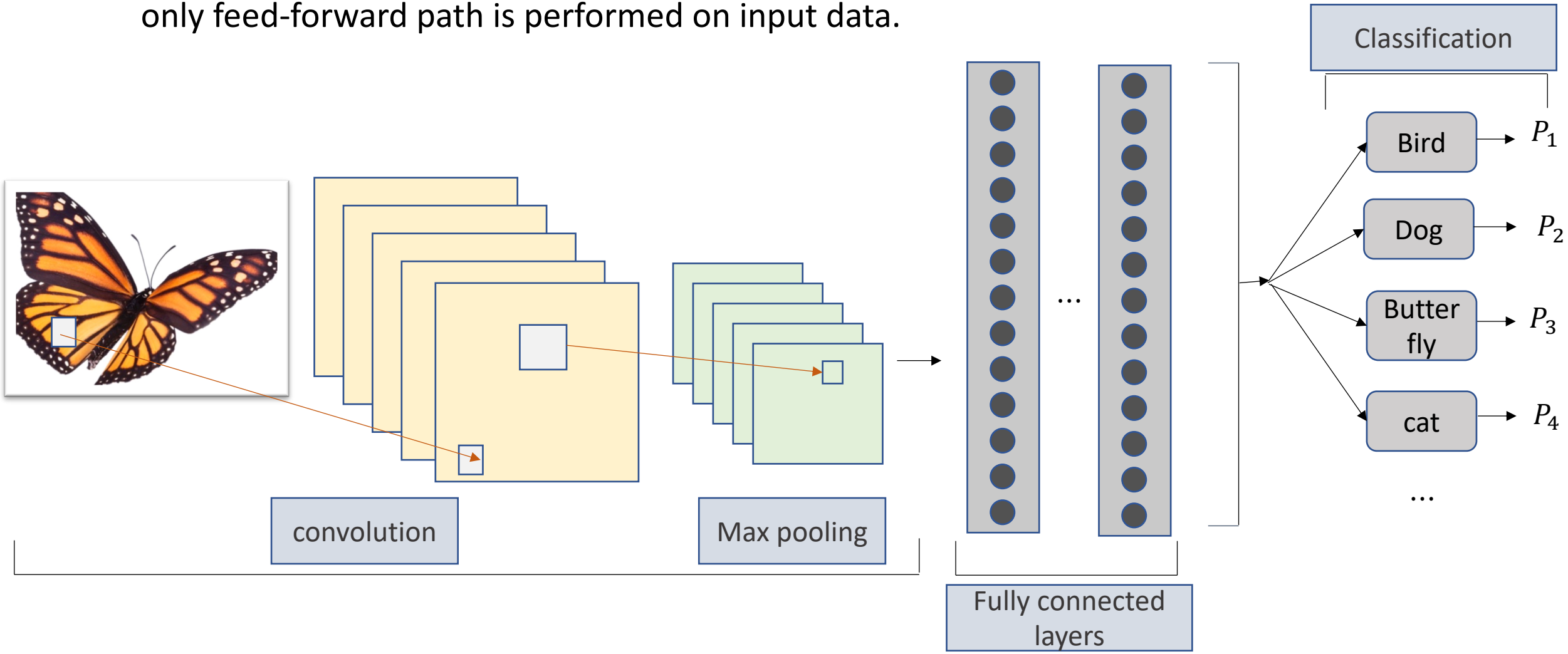
- Training phase in neural networks is a one-time process based on two main functions: feedforward and back-propagation. The network goes through all instances of the training set and iteratively update the weights. At the end, this procedure yields a trained model.



Inference(prediction)

P_1

- Inference phase uses the learned model to classify new data samples. In this phase only feed-forward path is performed on input data.



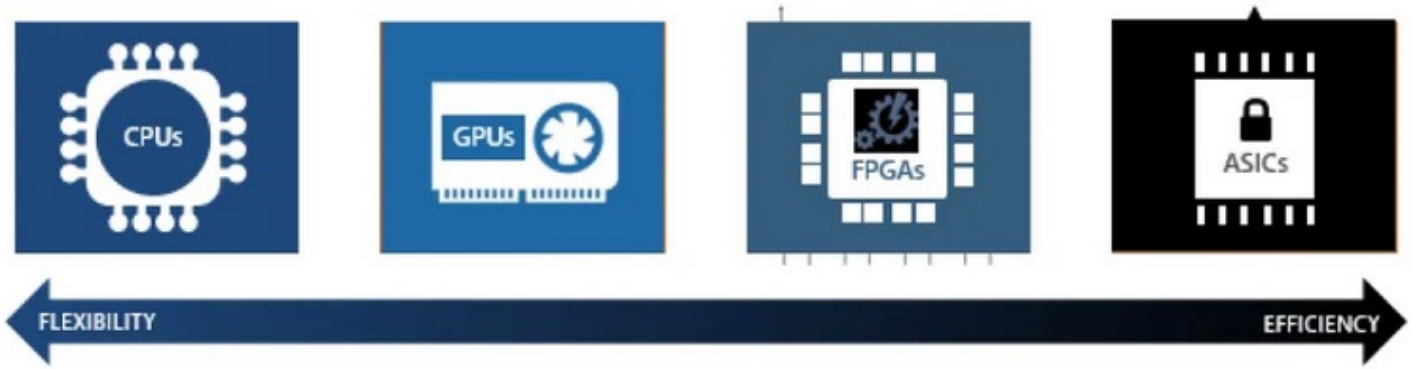
Workload of CNNs' inference

Model	AlexNet	GoogleNet	VGG16	VGG19	ResNet50	ResNet101	ResNet-152
Top1 err	42.9 %	31.3 %	28.1 %	27.3 %	24.7%	23.6% %	23.0%
Top5 err	19.80 %	10.07 %	9.90 %	9.00 %	7.8 %	7.1 %	6.7 %
conv layers	5	57	13	16	53	104	155
conv workload (MACs)	666 M	1.58 G	15.3 G	19.5 G	3.86 G	7.57 G	11.3 G
conv parameters	2.33 M	5.97 M	14.7 M	20 M	23.5 M	42.4 M	58 M
Activation layers	ReLU						
pool layers	3	14	5	5	2	2	2
FC layers	3	1	3	3	1	1	1
FC workload (MACs)	58.6 M	1.02 M	124 M	124 M	2.05 M	2.05 M	2.05 M
FC parametrs	58.6 M	1.02 M	124 M	124 M	2.05 M	2.05 M	2.05 M
Total workload (MACs)	724 M	1.58 G	15.5 G	19.6 G	3.86 G	7.57 G	11.3 G
Total parameters	61 M	6.99 M	138 M	144 M	25.5 M	44.4 M	60 M

The accuracy of CNN models have been increased at the price of high computational cost, because in these networks, there are a huge number of parameters and computational operations (MACs).

In AlexNet: $\frac{conv\ workload(MAC)}{conv\ workload(MAC)+FC\ workload(MAC)} = \frac{666M}{666M+58.6M} \times 100 = 91\%$

Devices for Deep neural network



CPU

- CPUs have a few but complex cores. They are fast on sequential processing.
- CPUs are good at fetching small amount of data quickly, but they are not suitable for big chunk of data.

Deep learning involves lots of matrix multiplications and convolutions, so it would take a long time to apply sequential computational approach on them.

We need to utilize architectures with high data bandwidth which can takes advantage of parallelism in DNNs.

Thus the trend is toward other three devices rather than CPUs to accelerate the training and inferencing.

CPU vs. GPU

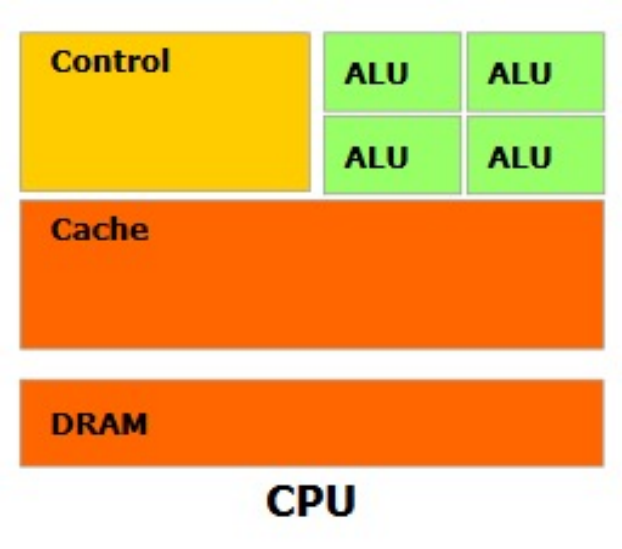
GPUs are designed for high parallel computations. They contain hundreds of cores that can handle many threads simultaneously. These threads execute in SIMD manner.

They have high memory bandwidth, and they can fetch a large amount of data.

They can fetch high dimensional matrices in DNNs and perform the calculations in parallel.

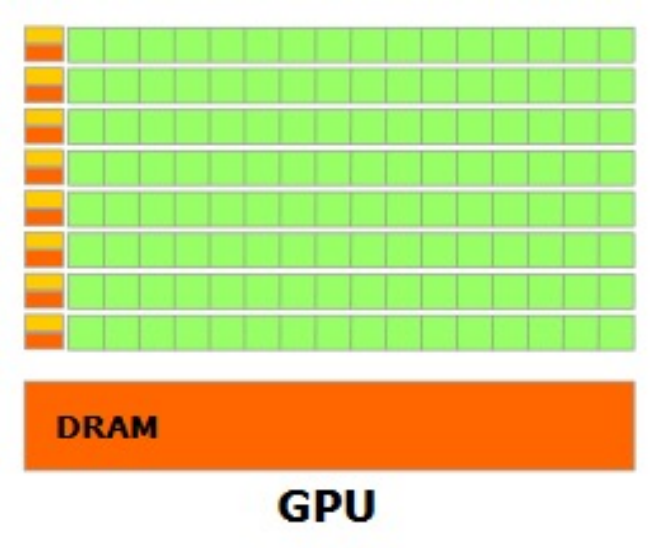
Designed for low-latency operation:

- Large caches
- Sophisticated control
- Powerful ALUs



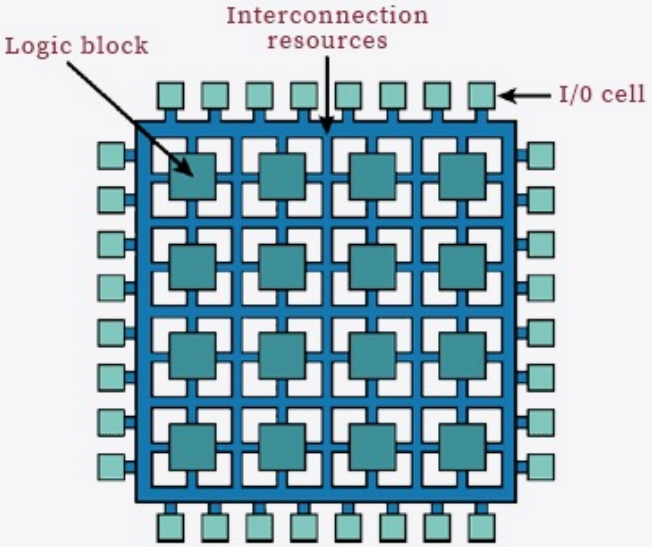
Designed for high-throughput:

- Small caches
- Simple control
- Energy efficient ALUs
- Latencies compensated by large number of threads



FPGAs for neural network

Field Programmable Gate Arrays (FPGAs) are semiconductor devices consist of **configurable logic blocks** connected via **programmable interconnects**. Because of their **high energy-efficiency**, **computing capabilities and reconfigurability** they are becoming the platform of choice for DNNs accelerators.

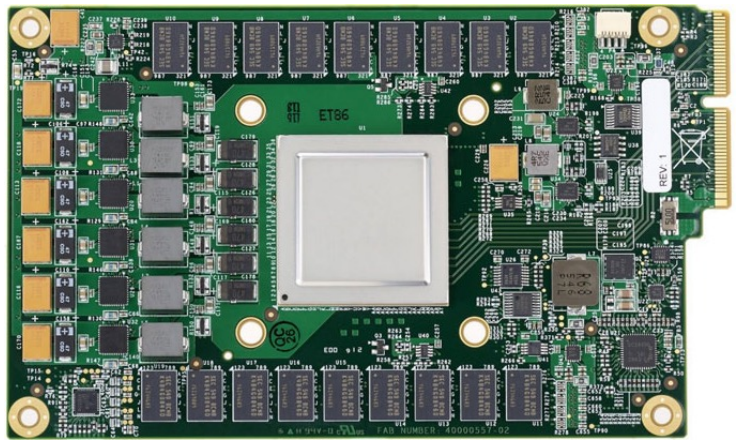


GPU vs. FPGA :

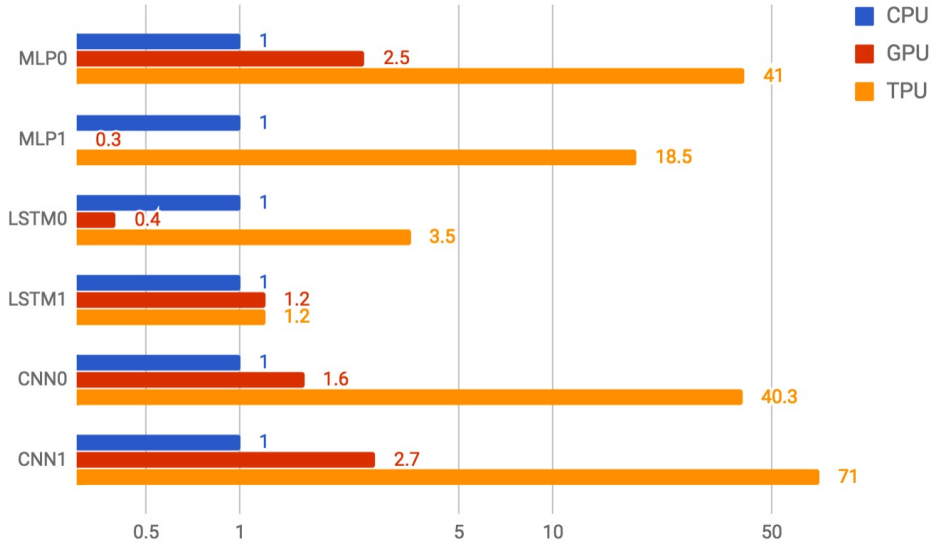
- FPGA are more power-efficient than GPUs. GPUs' computing resources are more complicated than FPGAs' to facilitate software programming. (programming a GPU is usually easier than developing a FPGA accelerator)
- According to flexibility of FPGAs, they can support various data type like binary or ternary data type.
- Datapath in GPUs is SIMD while in FPGA user can configure a specific data path.

ASICs-based neural network devices

- GPUs and FPGAs perform better than CPUs for DNNs' applications, but more efficiency can still be gained via an Application-Specific Integrated Circuit (ASIC).
- ASICs are the least flexible but the most high-performance options. They can be designed for either training or inference.
- They are most efficient in terms of performance/dollar and performance/watt but require huge investment costs that make them cost-effective only in very high-volume designs.
- The first generation of Google's **Tensor Processing Unit (TPU)** is a machine learning device which focuses on 8-bit integers for inference workloads. Tensor computations in TPU take advantage of systolic array.



Tensor Processing Unit (TPU)

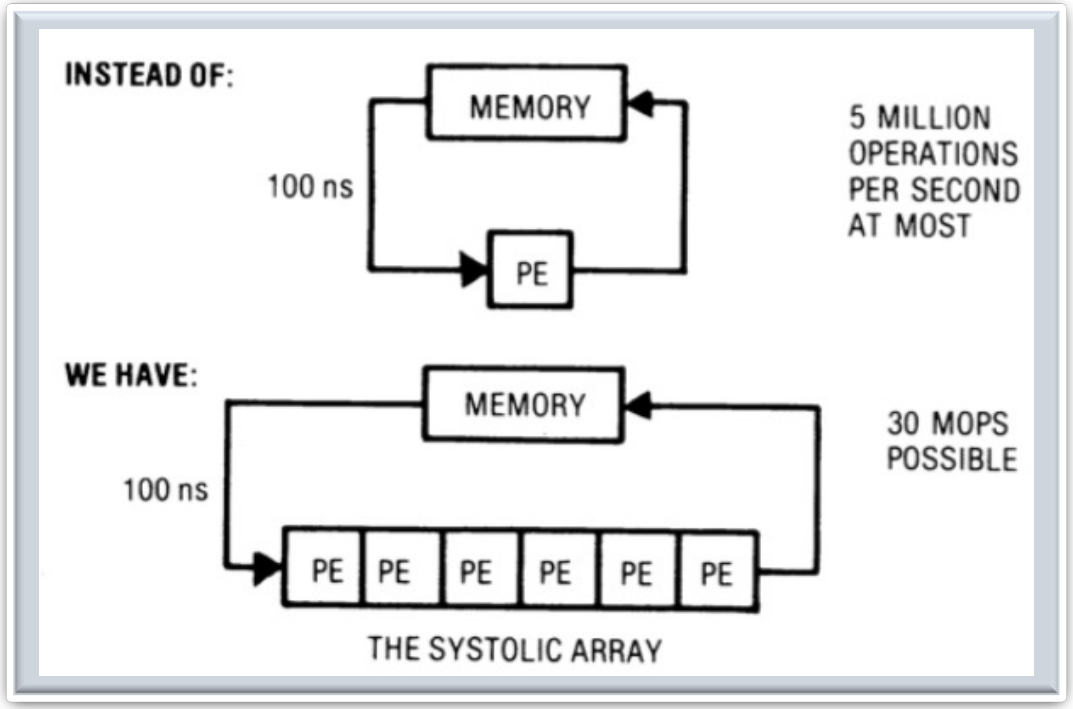


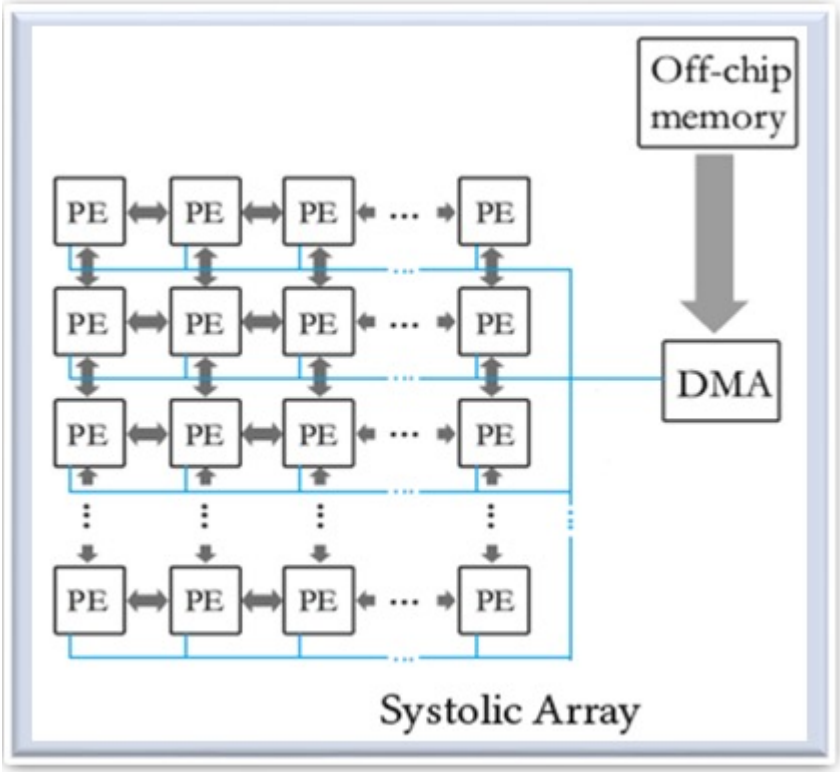
CPU, GPU and TPU performance on six reference workloads

Systolic array

- Idea: Data flows from the computer memory, passing through many processing elements before it returns to memory.

The principal for systolic array:



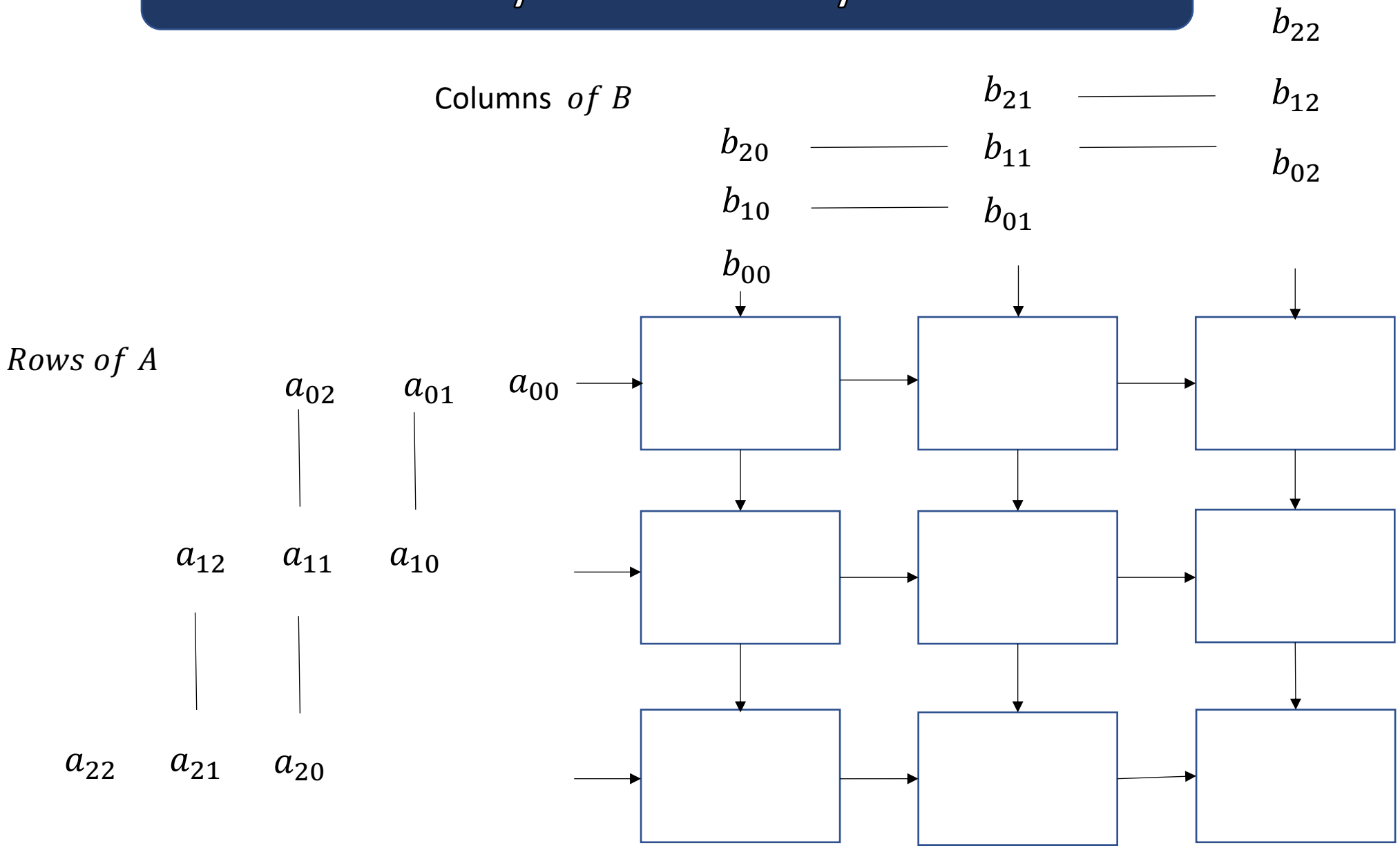


Systolic array

Assume we want to perform the matrix multiplication by a Systolic array:

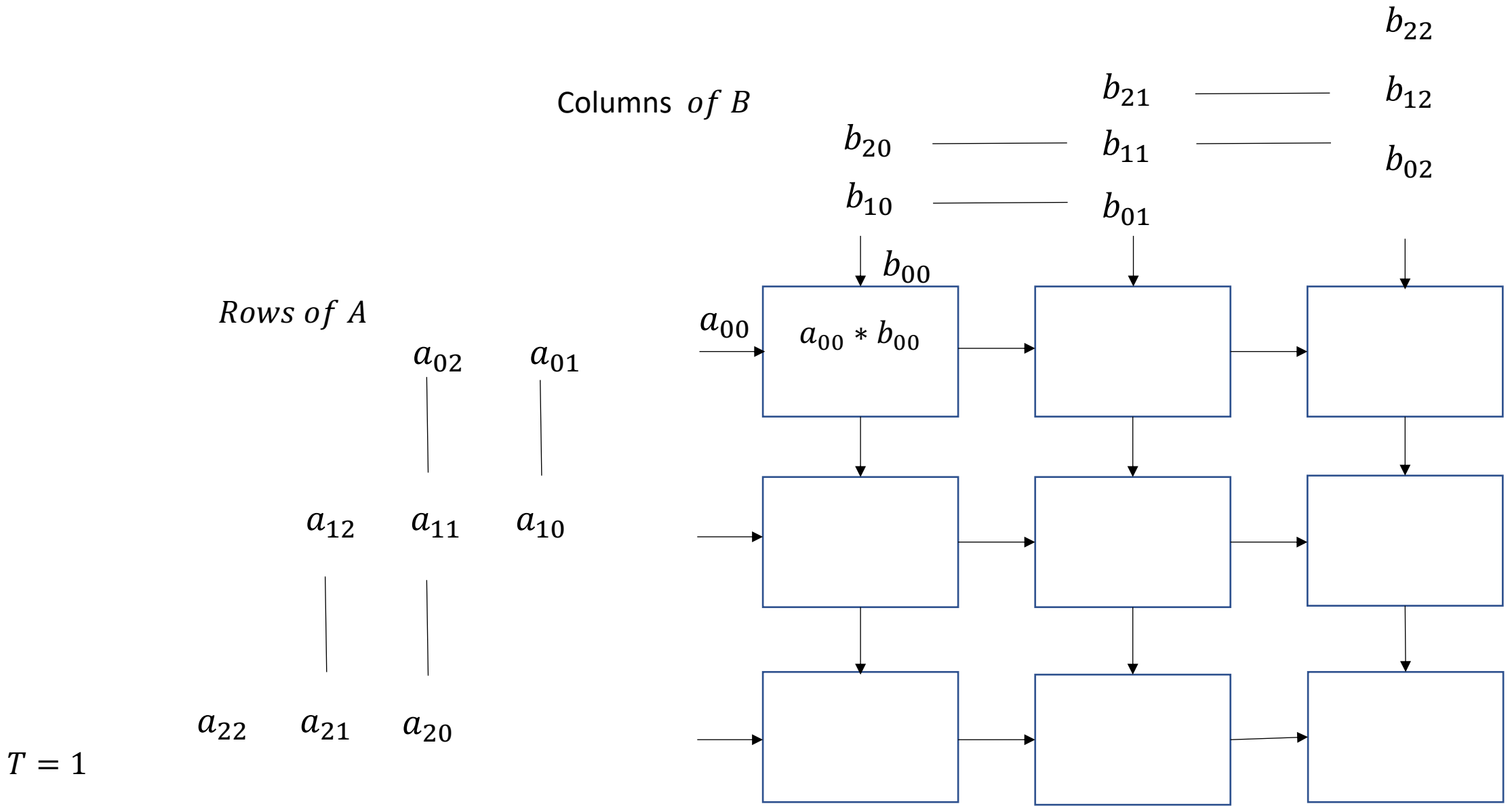
$$\underset{Matrix\ A}{\begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix}} \times \underset{Matrix\ B}{\begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ b_{20} & b_{21} & b_{22} \end{bmatrix}}$$

Systolic array

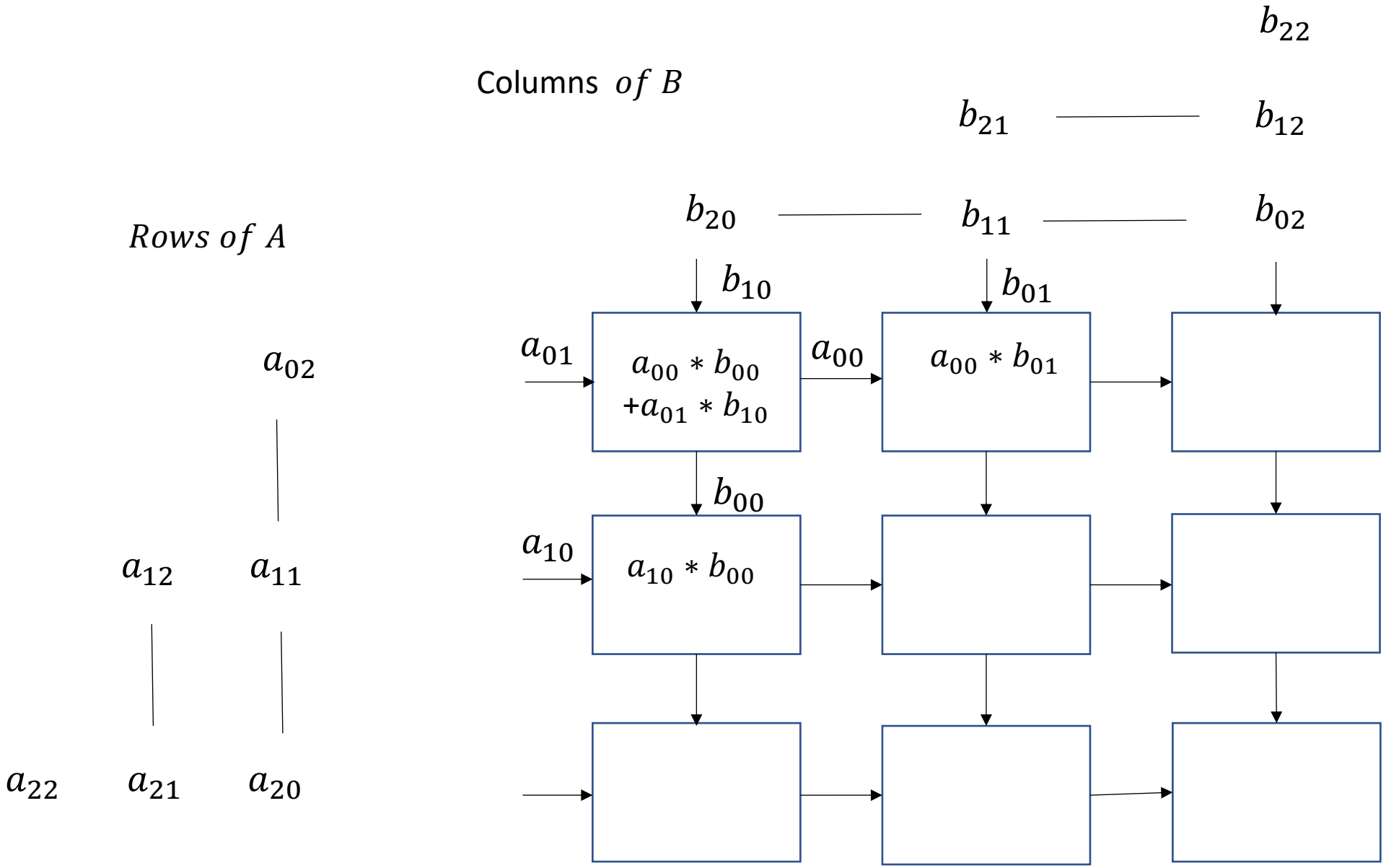


$T = 0$

Systolic array

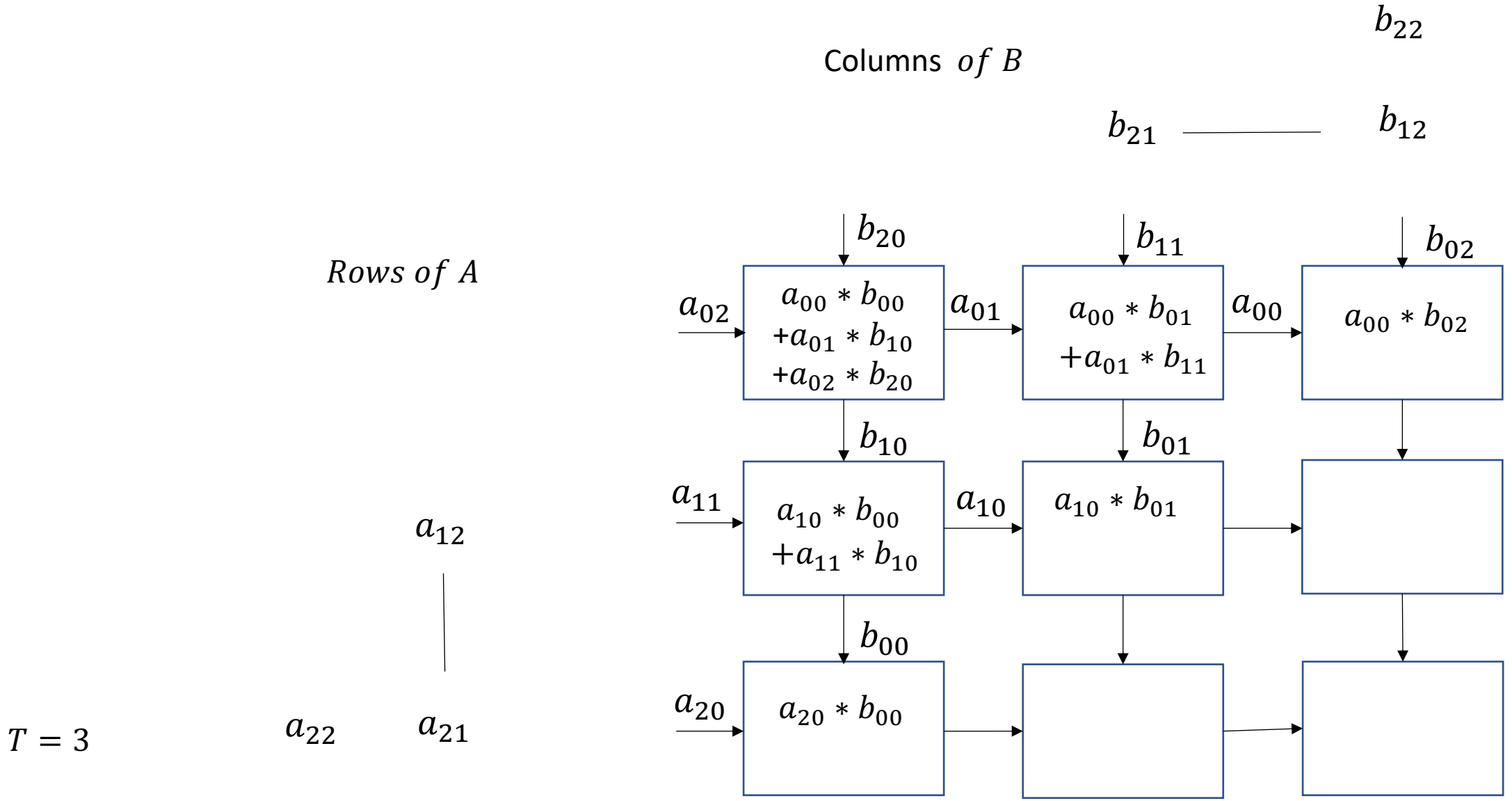


Systolic array



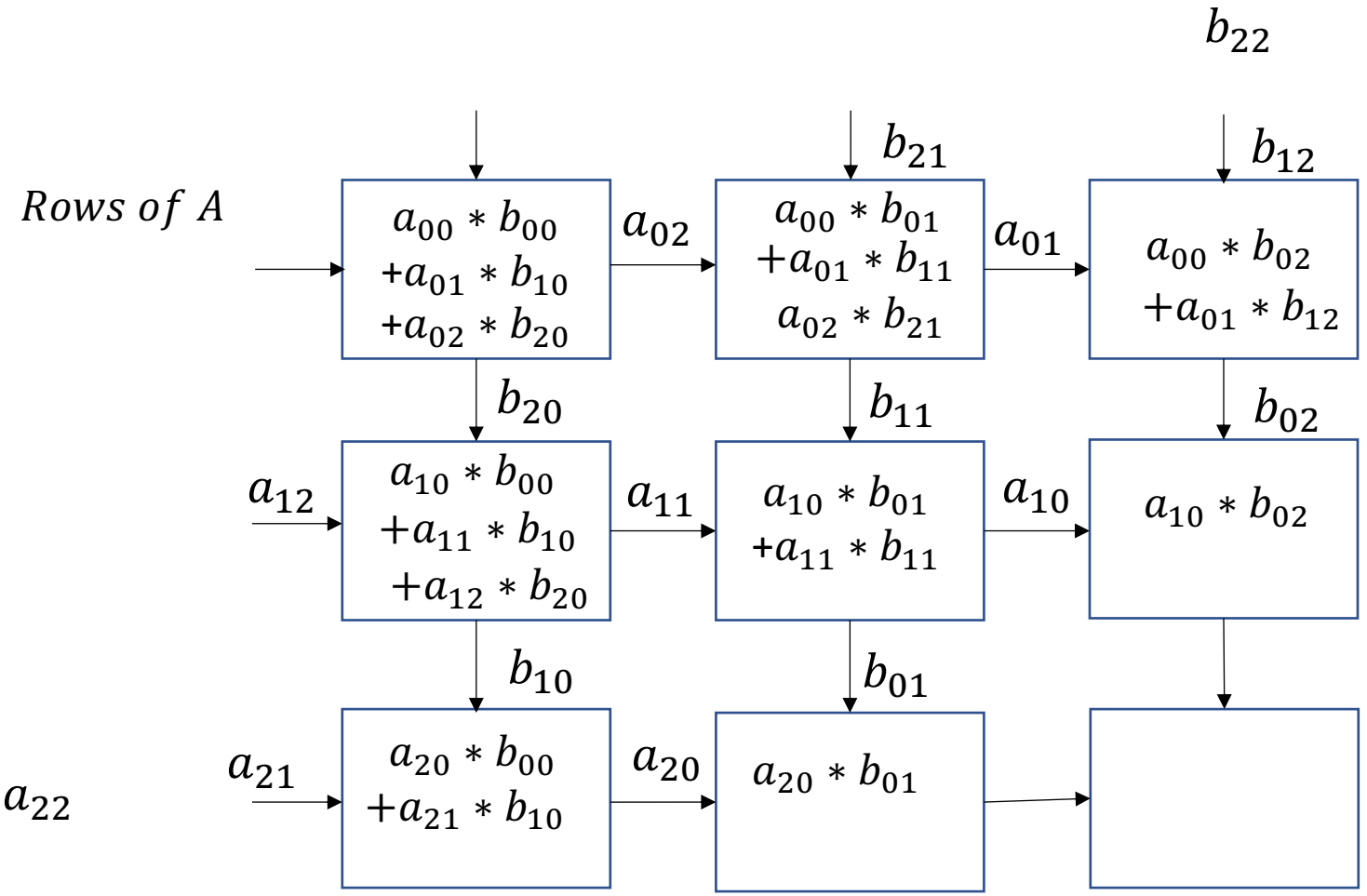
$T = 2$

Systolic array



Systolic array

Columns $of\ B$

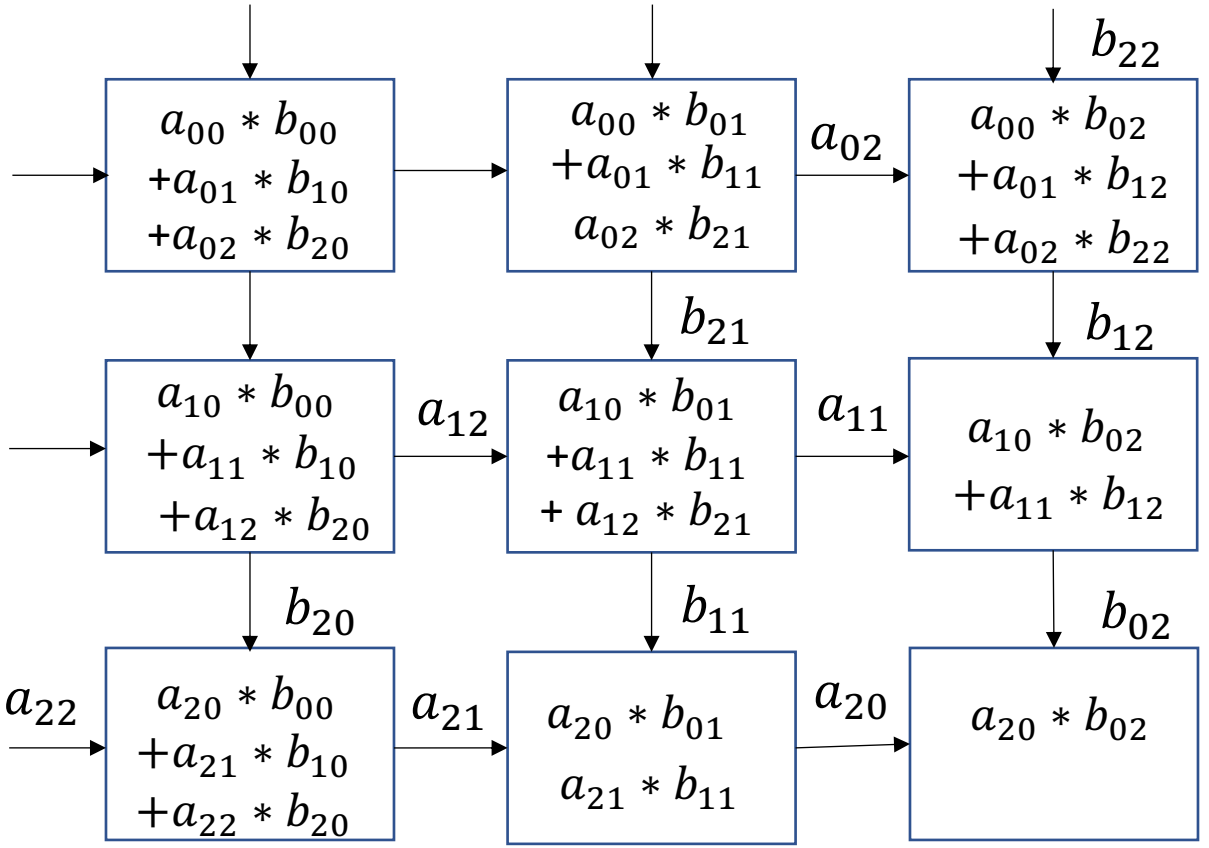


$T = 4$

Systolic array

Columns $of\ B$

$Rows\ of\ A$

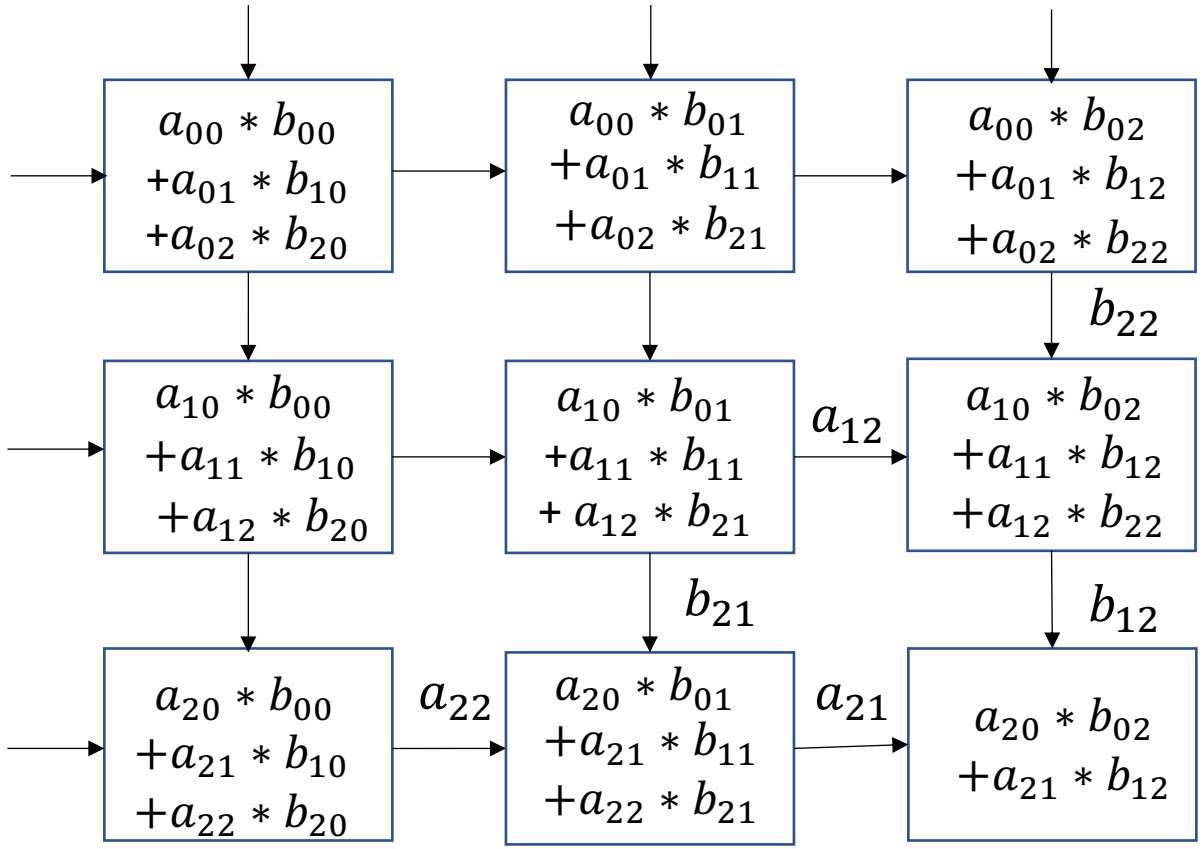


$T = 5$

Systolic array

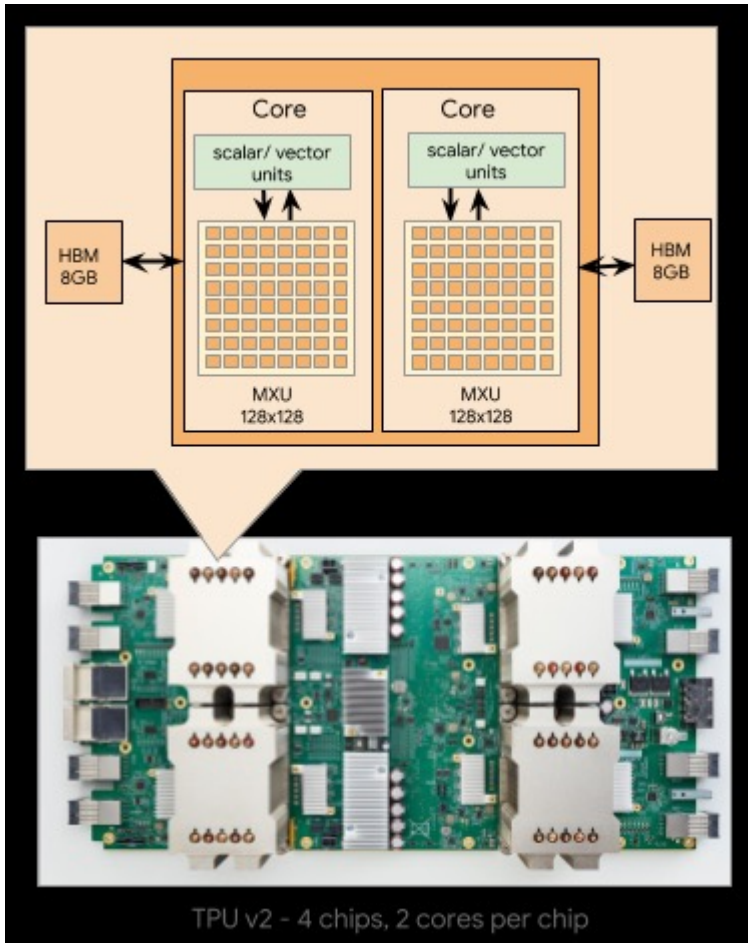
Columns $of\ B$

$Rows\ of\ A$



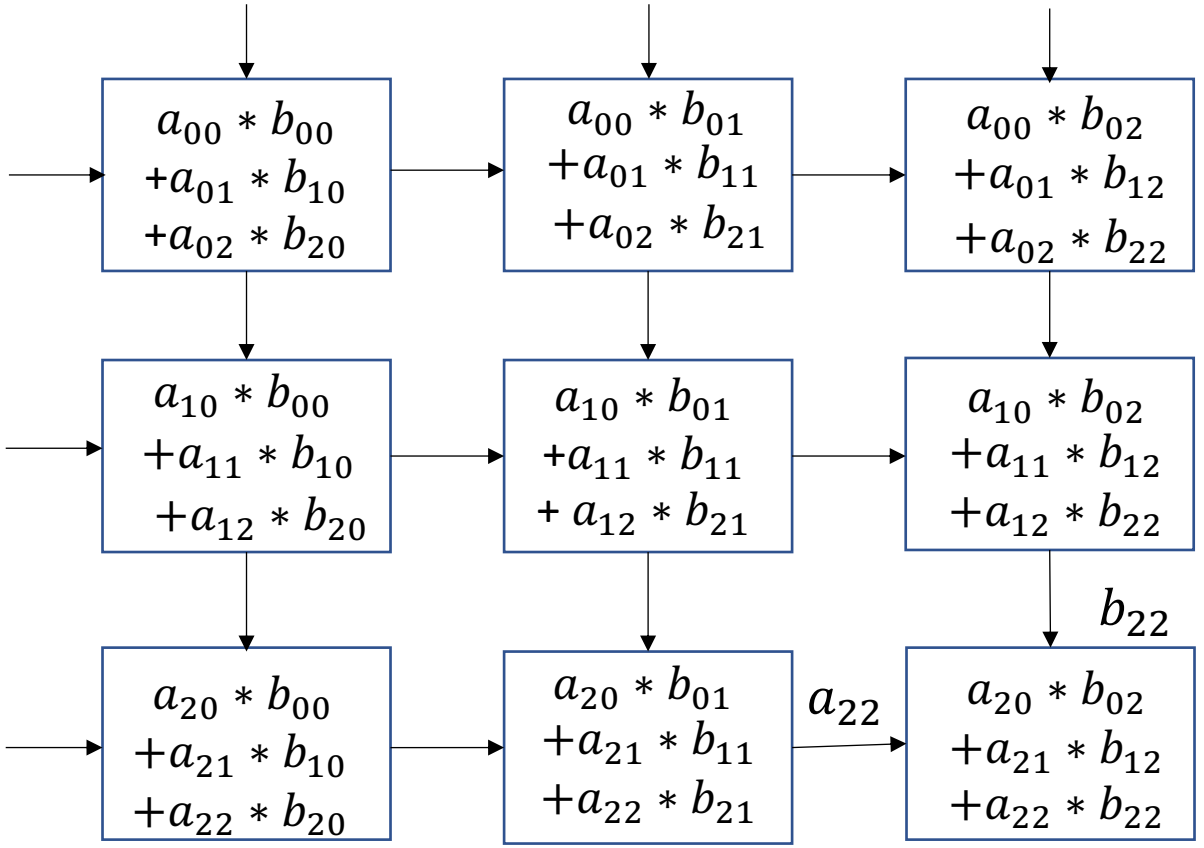
$T = 6$

Systolic array



Columns $of\ B$

$Rows\ of\ A$

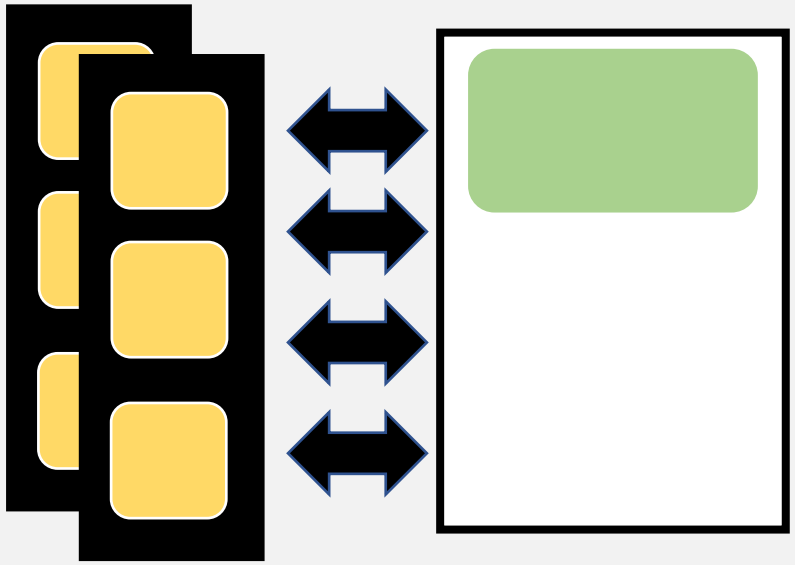


$T = 7$

Common memory designs

We need to feed these floating-point units from memory, and we have four choices for the memory architecture.

DDR4 modules

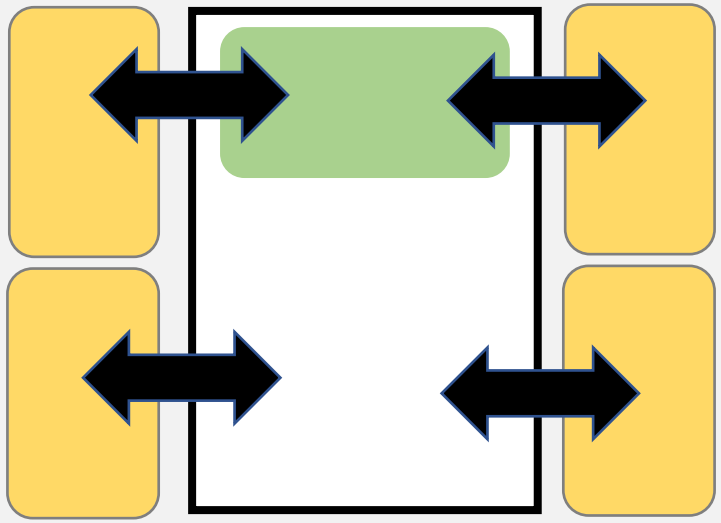


320pJ/B

256GB @ 64GB/s

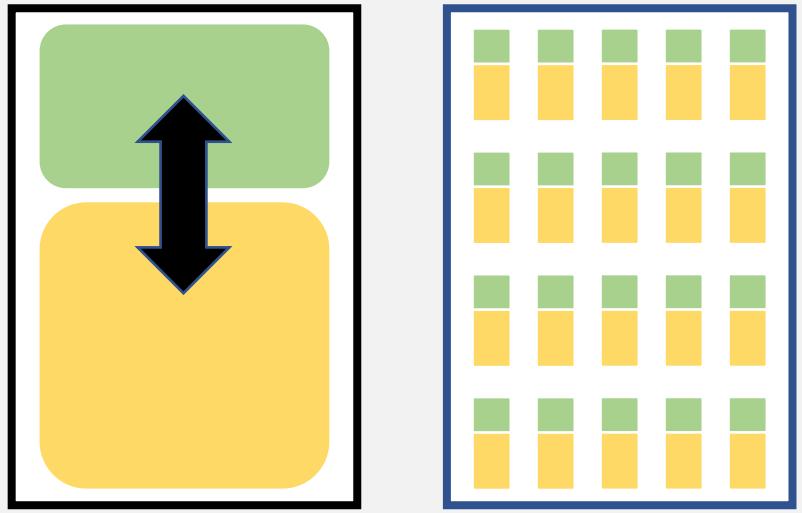
20W

High bandwidth memory on interposer



64pJ/B
16GB @ 900GB/s
60W

SRAM on chip

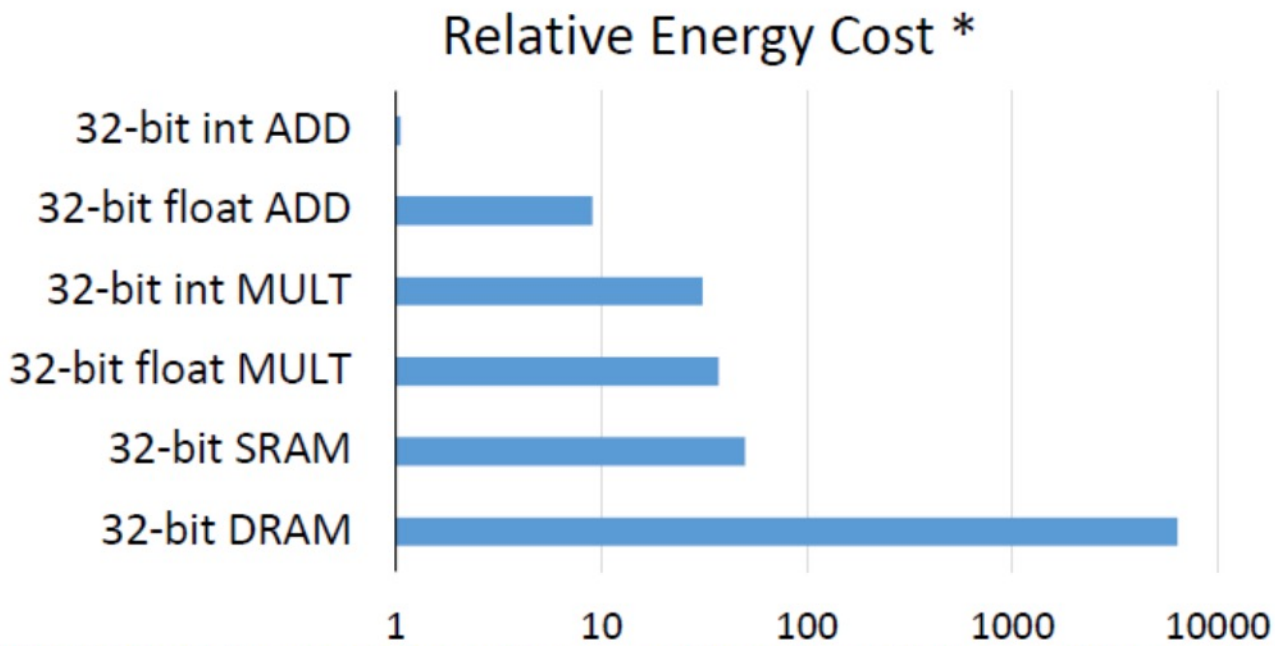


10pJ/B
256MB @ 6TB/s
60W

1pJ/B
1000×256kB @60TB/s
60W

Memory power density is ~25% of logic power density

Memory Access is Energy Hog



* Han, et al. "EIE: Efficient Inference Engine on Compressed Deep Neural Network." arXiv preprint arXiv:1602.01528 (2016).

Loop optimization

➢ **Loop unrolling:**

- Loop unrolling exploit parallelism between loop iterations by utilizing FPGA resources. (multiple iteration can be executed simultaneously)

Without loop unrolling

```
For (int i=0; i<N; i++){
    a[i]=b[i]+c[i];
}
```

Latency= N cycles (N iterations)

With loop unrolling

```
For (int i=0; i<N; i+=2){
    a[i]=b[i]+c[i];
    a[i+1]=b[i+1]+c[i+1];
}
```

Unrolled factor =2
Latency= N/2 cycles (N/2 iterations)

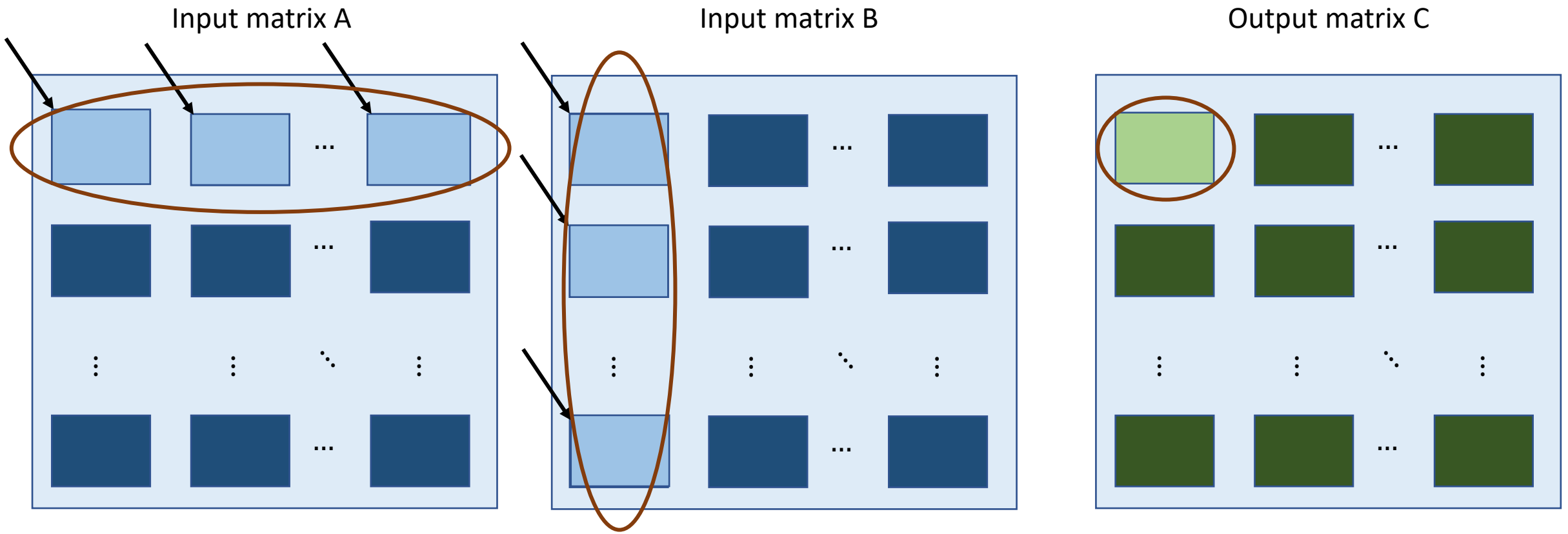
- **If we unroll a loop in a convolutional layer, we can accelerate the execution time at the expense of resource utilization (PEs)**.

➢ **Loop Tiling:**

- **loop tiling is used to divide the input data into multiple blocks, which can be accommodated in the on-chip buffers**. It exploits the data locality which results in reducing DRAM accesses, latency and power consumption.

Loop optimization (cont.)

Tiled Matrix Multiplication: ı:



Loop optimization (cont.)

```
int i, j, k;

for (i = 0; i < N; ++i)
{
   for (j = 0; j < N; ++j)
   {
      C[i][j] = 0;

      for (k = 0; k < N; ++k)
         C[i][j] += A[i][k] * B[k][j];
   }
}
```

With tiling

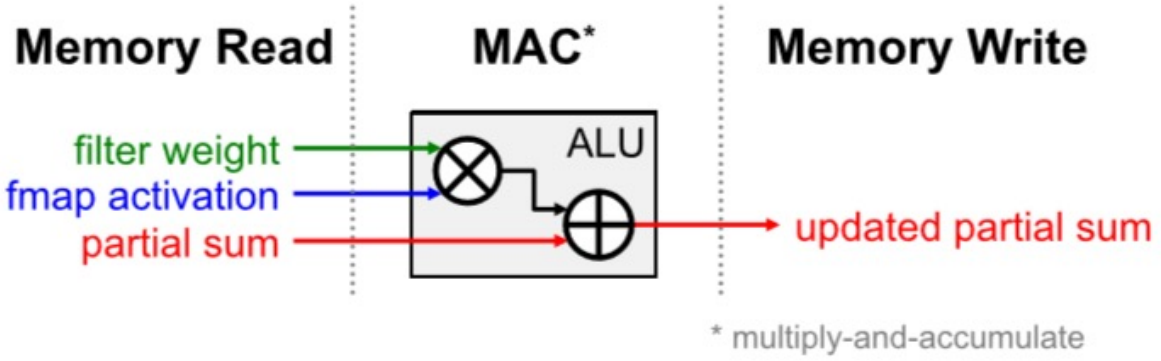
C = A . B

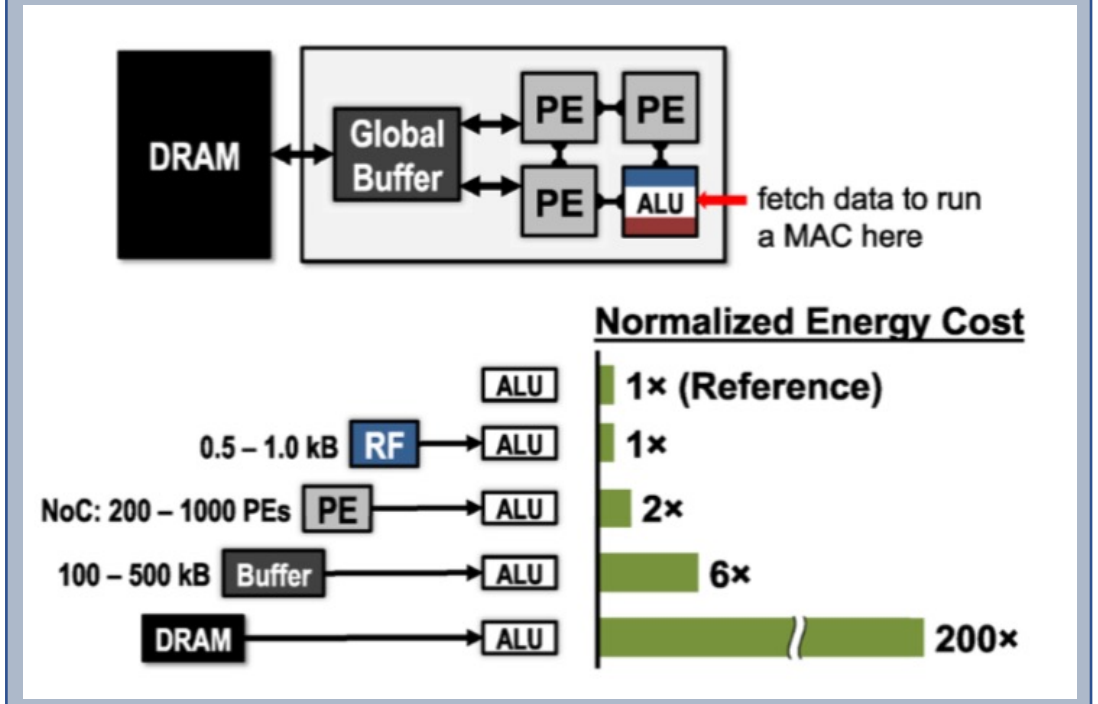
Without tiling

```
for (i = 0; i < N; i += 2)
{
   for (j = 0; j < N; j += 2)
   {
      acc00 = acc01 = acc10 = acc11 = 0;
      for (k = 0; k < N; k++)
      {
         acc00 += B[k][j + 0] * A[i + 0][k];
         acc01 += B[k][j + 1] * A[i + 0][k];
         acc10 += B[k][j + 0] * A[i + 1][k];
         acc11 += B[k][j + 1] * A[i + 1][k];
      }
      C[i + 0][j + 0] = acc00;
      C[i + 0][j + 1] = acc01;
      C[i + 1][j + 0] = acc10;
      C[i + 1][j + 1] = acc11;
   }
}
```

Data access in DNN

- **How can we have energy efficient device**?
- DNNs have lots of parameters and MAC operations. The parameters have to be stored in external memory (DRAM).
- **Each MAC operation requires three memory accesses to be performed**.
- DRAM accesses require up to several orders of magnitude higher energy consumption than MAC computation.
- Thus, If all accesses go to the DRAM, the latency and energy consumption of the data transfer may exceed the computation itself.





energy cost of data movement in different level of memory hierarchy.

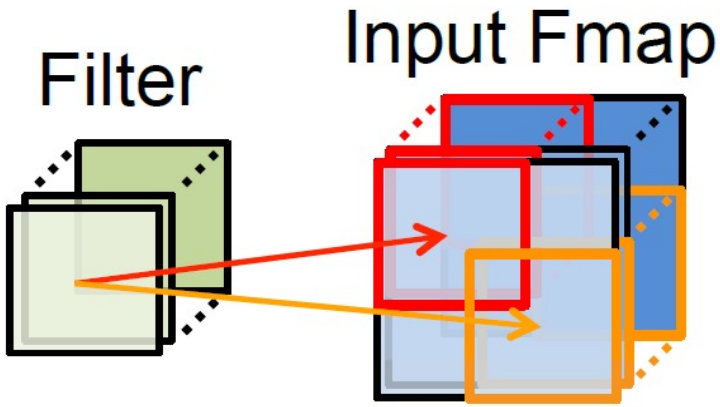
Data reuse opportunities in CNN

- To **reduce energy consumption of data movement**, every time a piece of data is moved from an expensive level to a lower cost memory level , the system should reuse the data as much as possible.

- In convolutional neural network, we can consider **three forms of data reuse**:

1.Convolutional reuse:

The **same input feature map** activations and **filter weights** are used within a given channel, just in **different combinations** for different weighted sums.

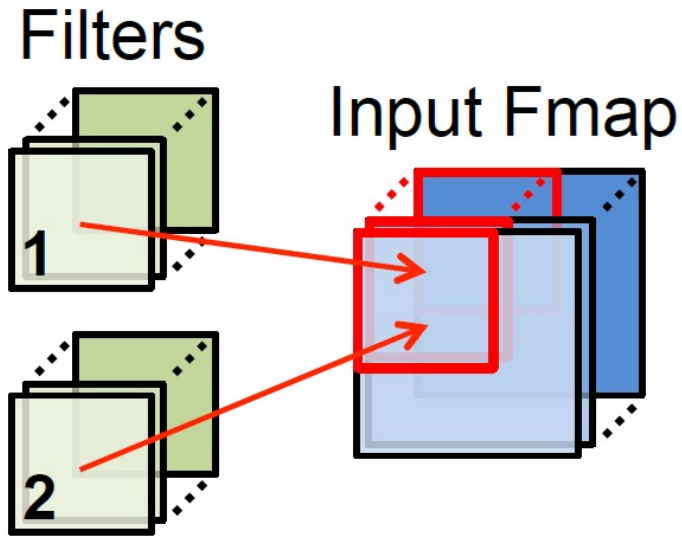


Reuse: { *Activations* / *Filter weights* }

Data reuse opportunities in CNN (cont.)

2.Feature map reuse

When **multiple filters are applied to the same feature map**, the input feature map activations are used multiple times across filters.

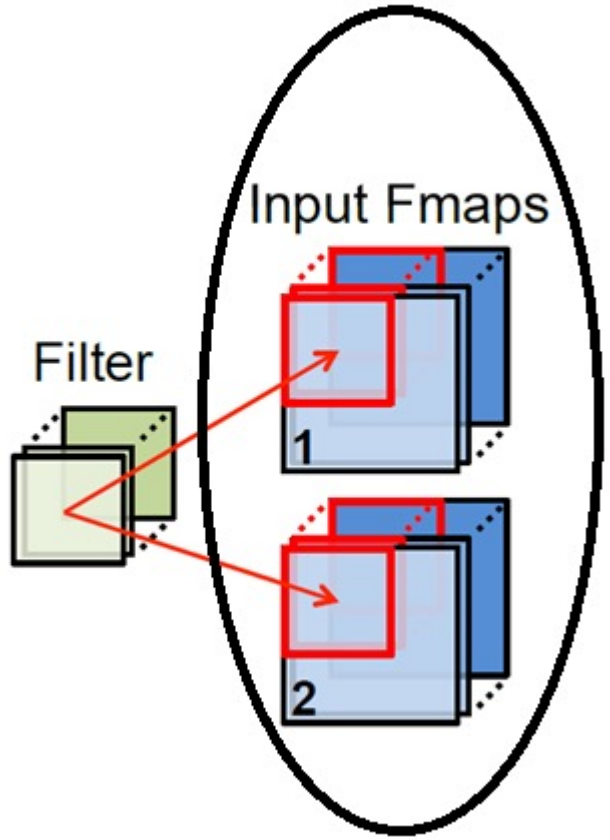


Reuse: *Activations*

3.Filter reuse

When **the same filter weights are used multiple times across input features maps.**

Multiple input frames/images can be simultaneously processed.



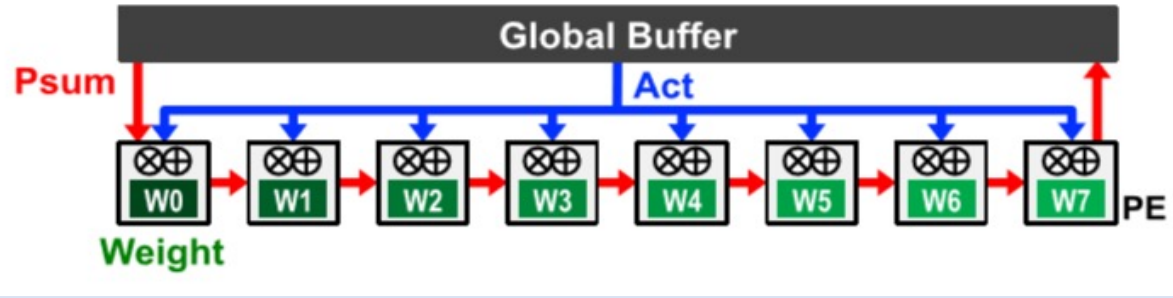
Reuse: *Filter weights*

Dataflow models

- **There are various related works in the literature that take advantage of different data reuse and dataflow approaches.**

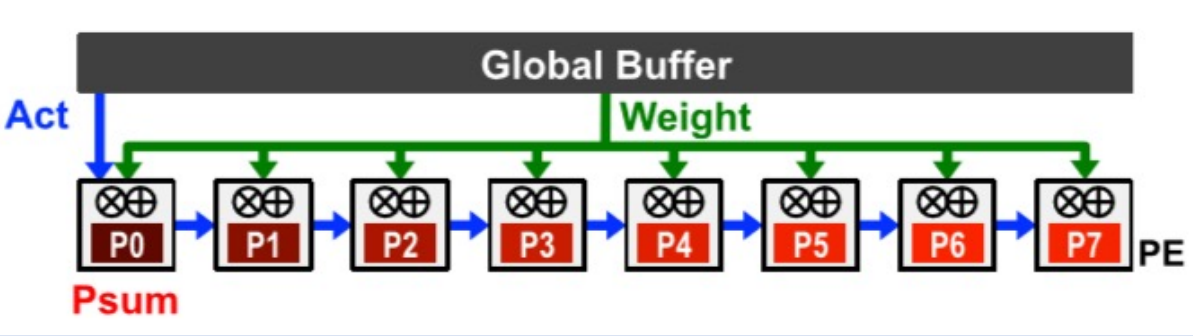
Weight Stationary dataflow (WS):

- The main idea is to minimize the energy consumption of reading weights.
- The weights store in register file, input pixels and partial sum move through network.



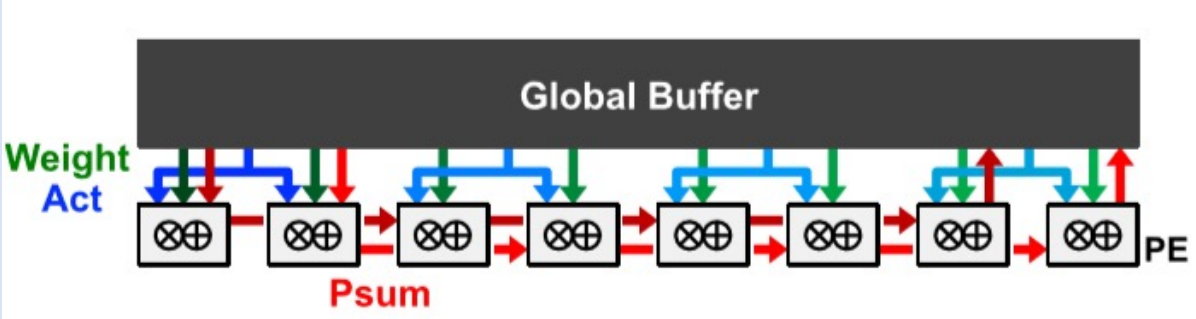
Output Stationary (OS):

- It keeps the partial sum locally in PE register file and access input pixels and weights through global buffer.
- For every partial sum, we need two memory accesses (R/W).



No Local Reuse (NLR):

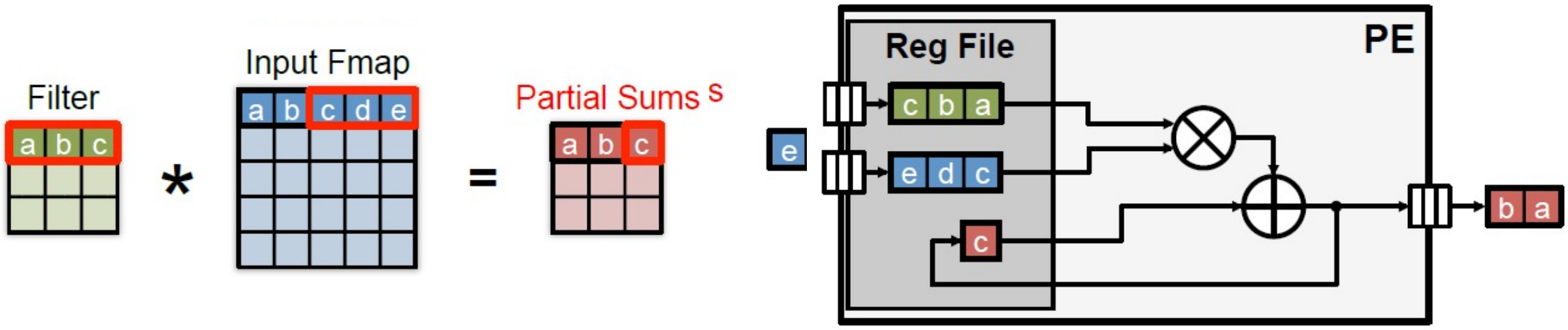
Instead of register file it uses a large global buffer, so it does not keep data locally in RF and access them through global buffer.



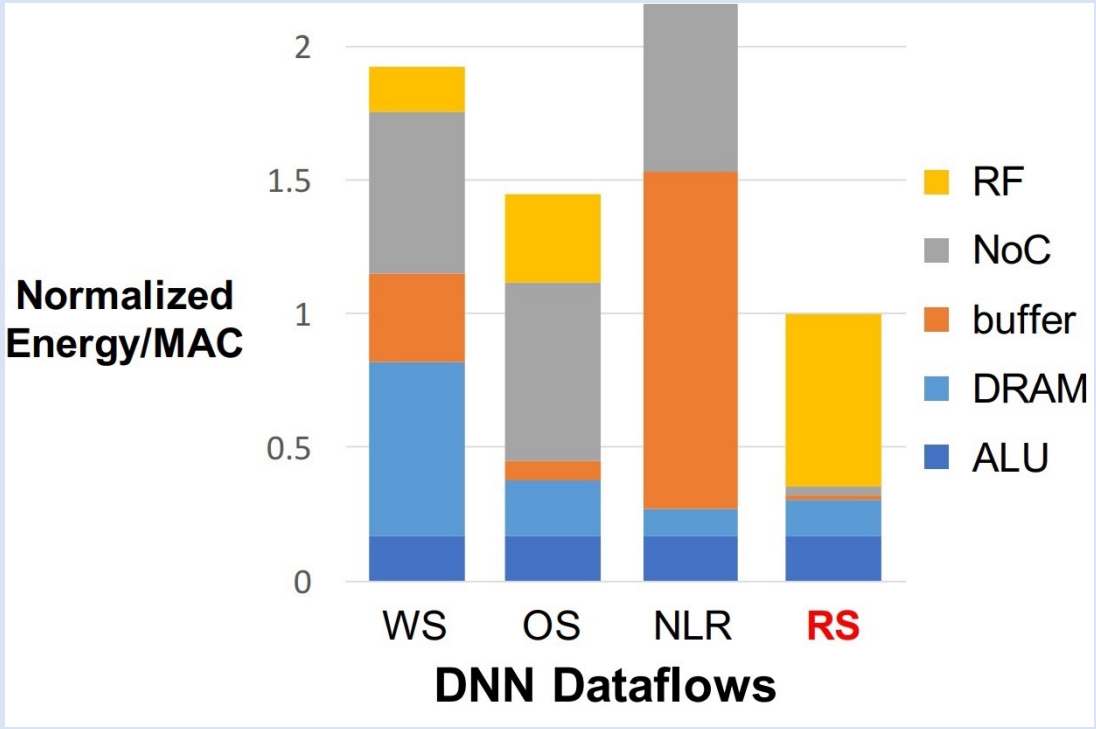
Dataflow models (cont.)

Row stationary data flow:

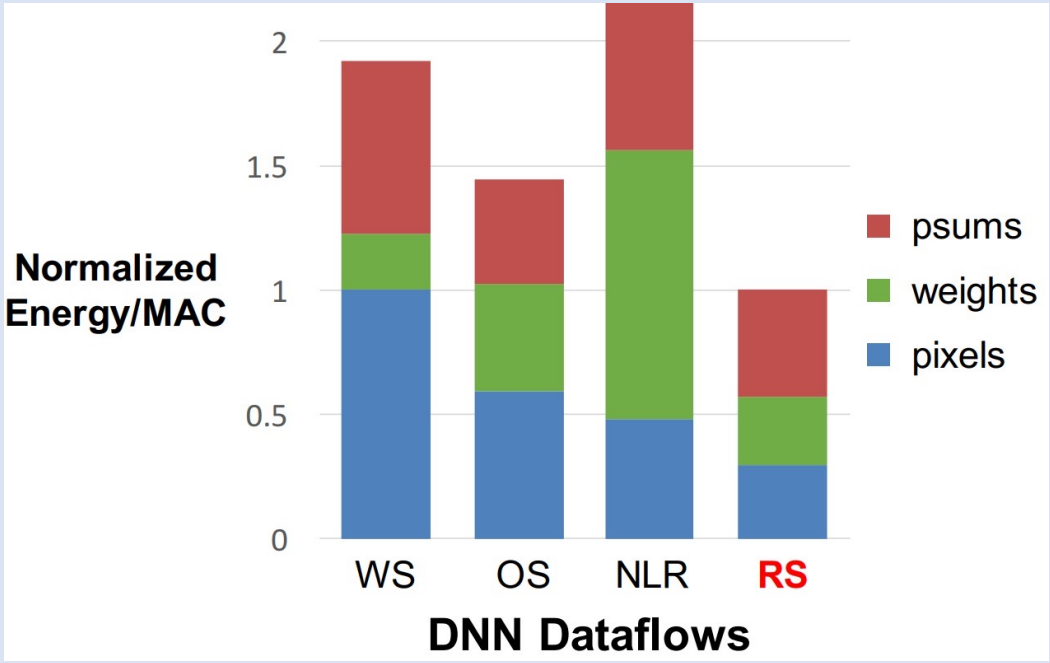
- Row stationary dataflow maximizes the reuse and accumulation at the RF level for all types of data for the overall energy efficiency.
- It keeps a row of filter weights stationary inside the RF of a PE and then streams the input activations into the PE. Since there are overlaps of input activations between different sliding windows the input activations can then be kept in the RF and get reused.



Energy comparison of different dataflows



Energy consumption across memory hierarchy



Energy comparison for different data types

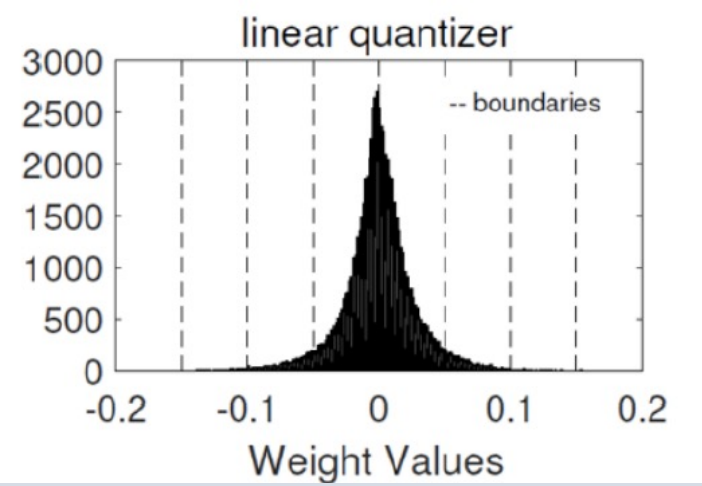
Compression

❖ **Compression** methods try to reduce the number of weights or reduce the number of bits used for each activation or weight. This technique lowers down the computation and storage requirements.

Quantization

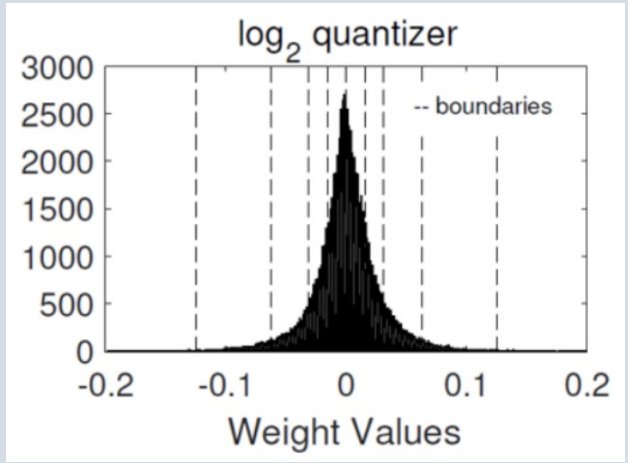
Quantization or reduced precision approach **allocates the smaller number of bits for representing weight and activations**.

➢ **Uniformed quantization**:
It uses a mapping function with uniform distance between each quantization level.



➢ **Nonuniformed quantization**: The distribution of the weights and activations are not uniform so nonuniform quantization, where the spacing between the quantization levels vary, can improve accuracy in comparison to uniformed quantization.

1. **Log domain quantization** : quantization levels are assigned based on logarithmic distribution.
2. **Learned quantization or weight sharing**



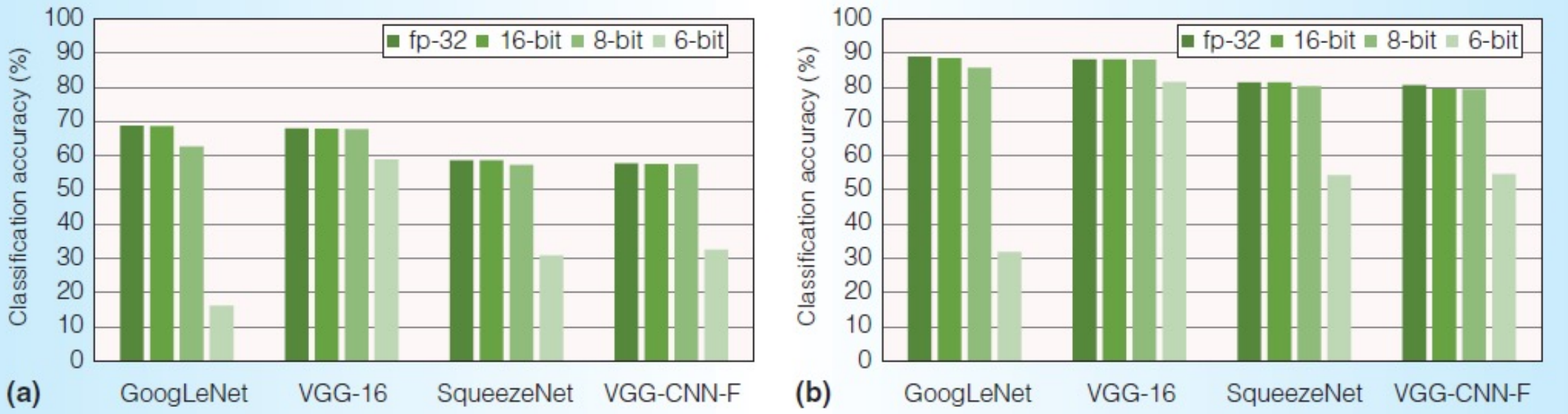
Quantization (cont.)

- **Learned quantization or weight sharing**: weight sharing forces several weights to share a single value **so it will reduce the number of unique weights that we need to store**. Weights can be grouped together using a k-means algorithm, and one value assigned to each group.

Note: the quantization can be fixed or variable.

Bitwidth of the group index = $log_2\ (the\ number\ of\ unique\ weights)$

Quantization



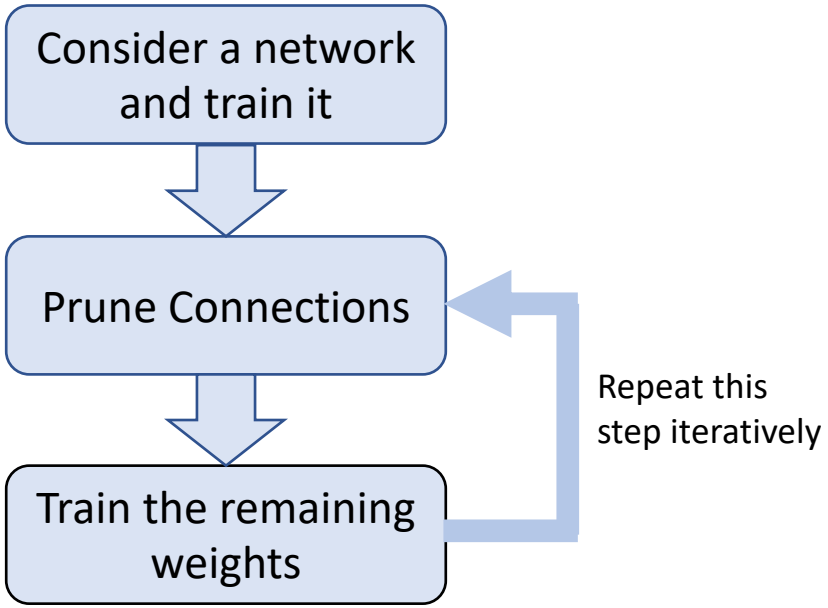
Quantization results for different CNN models. (a) Top-1 classification accuracy. (b) Top-5 classification accuracy.

Pruning

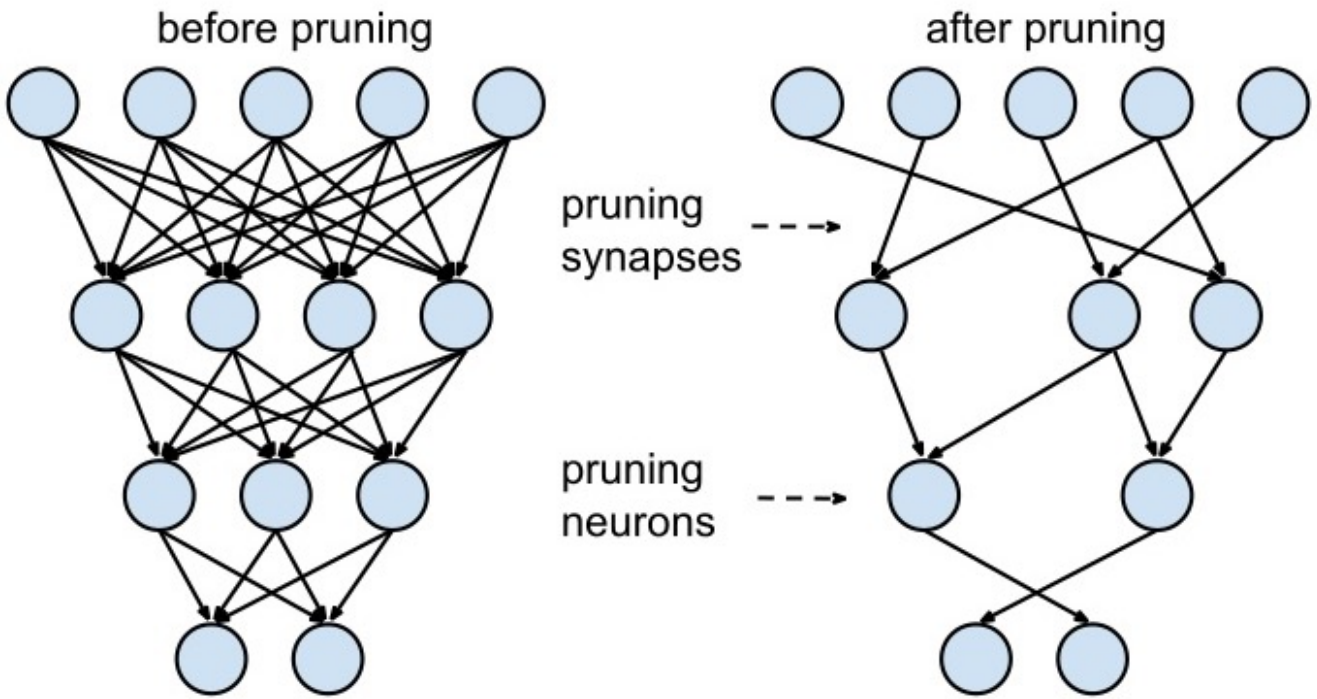
What is neural network pruning?

- Pruning algorithms reduce the size of the neural networks by removing unnecessary weights and activations.
- It can make neural network more **power and memory efficient**, **and faster at inference** with minimum loss in accuracy.

Procedure of pruning

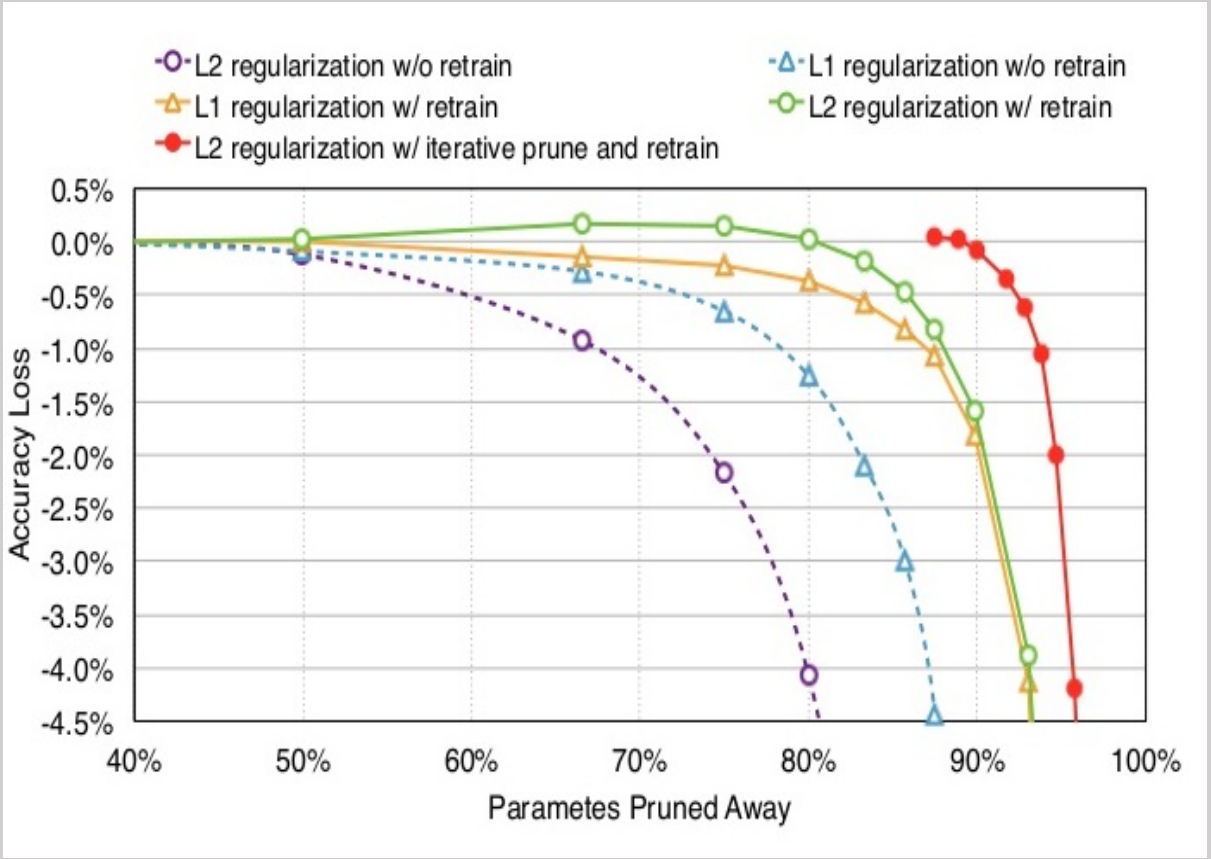


Synapses and neurons before and after pruning



It gives smaller model without losing accuracy

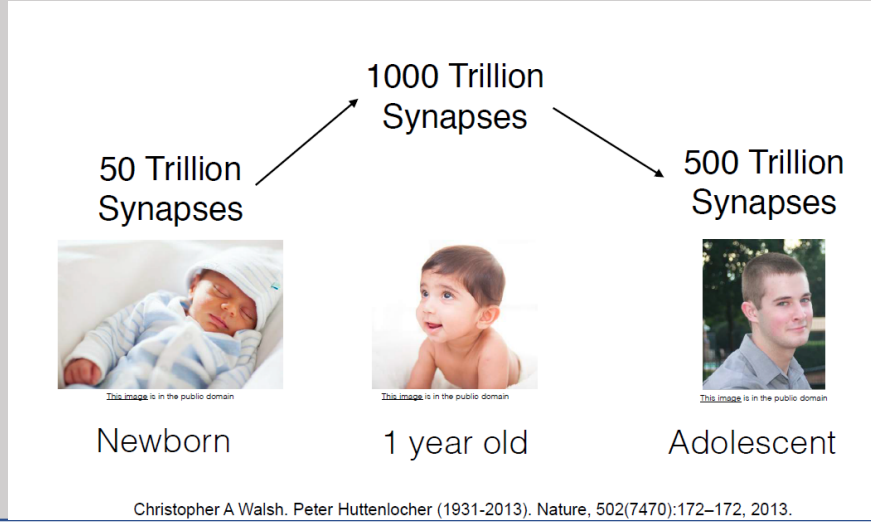
Pruning (cont.)



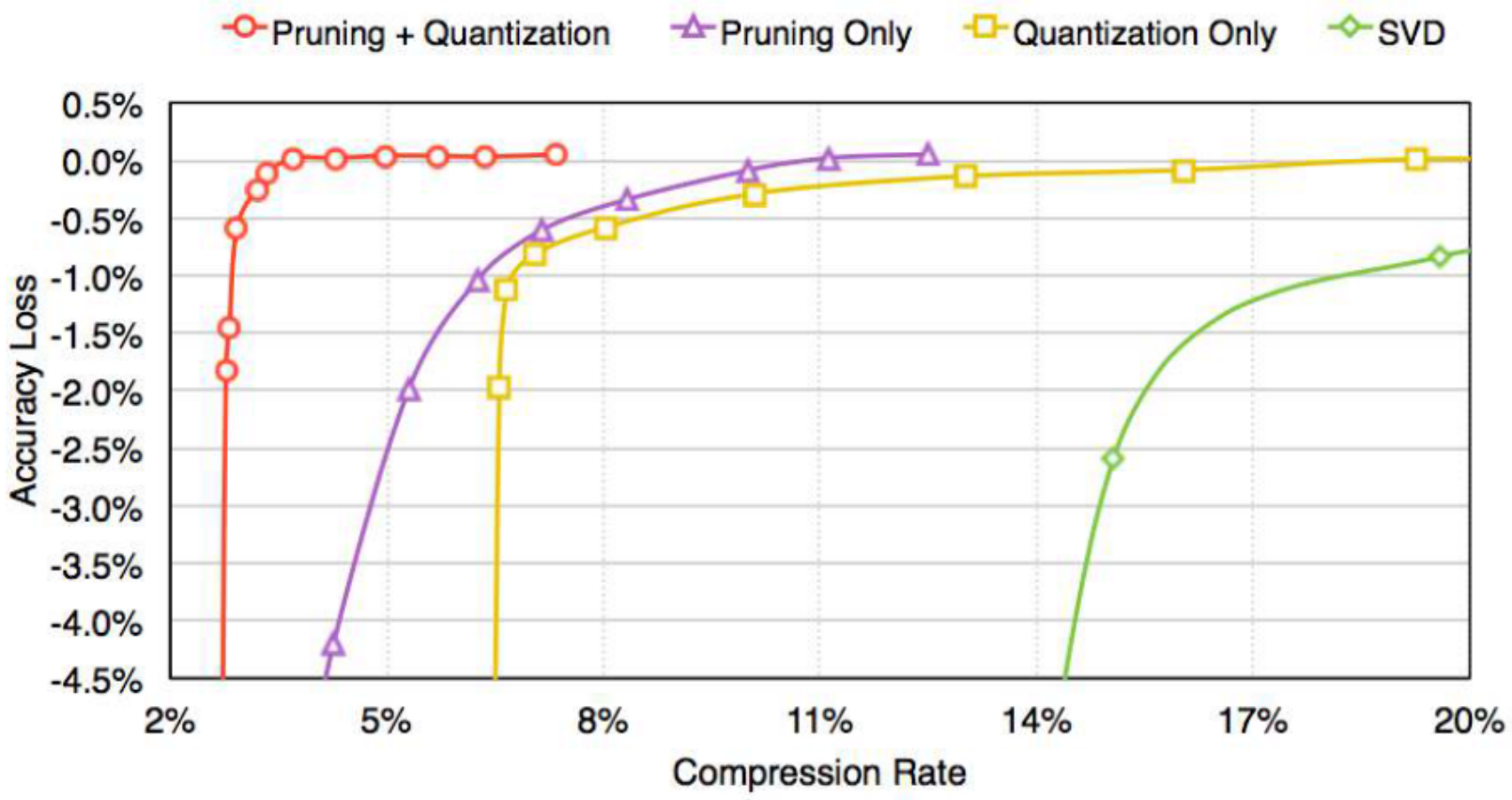
Comparing l1 and l2 regularization with and without retraining.

Network	Top-1 Error	Top-5 Error	Parameters	Compression Rate
LeNet-300-100 Ref	1.64%	-	267K	
LeNet-300-100 Pruned	1.59%	-	**22K**	**12×**
LeNet-5 Ref	0.80%	-	431K	
LeNet-5 Pruned	0.77%	-	**36K**	**12×**
AlexNet Ref	42.78%	19.73%	61M	
AlexNet Pruned	42.77%	19.67%	**6.7M**	**9×**
VGG-16 Ref	31.50%	11.32%	138M	
VGG-16 Pruned	31.34%	10.88%	**10.3M**	**13×**

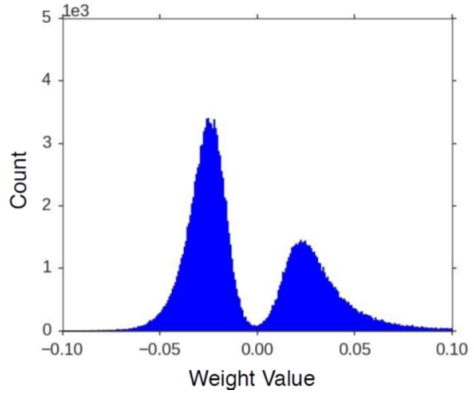
Network pruning can reduce parameters without drop in performance.

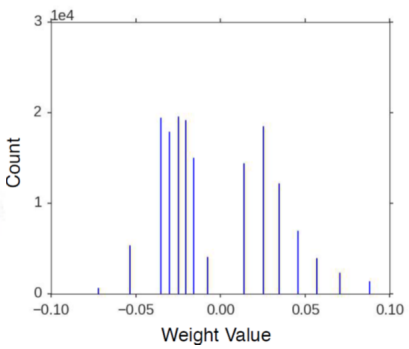


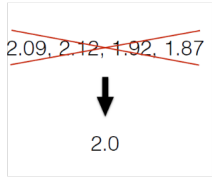
Pruning and Trained Quantization



AlexNet on ImageNet



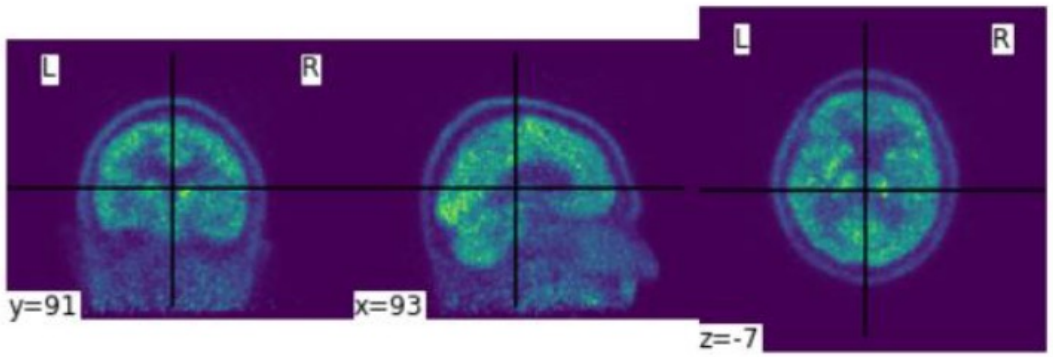




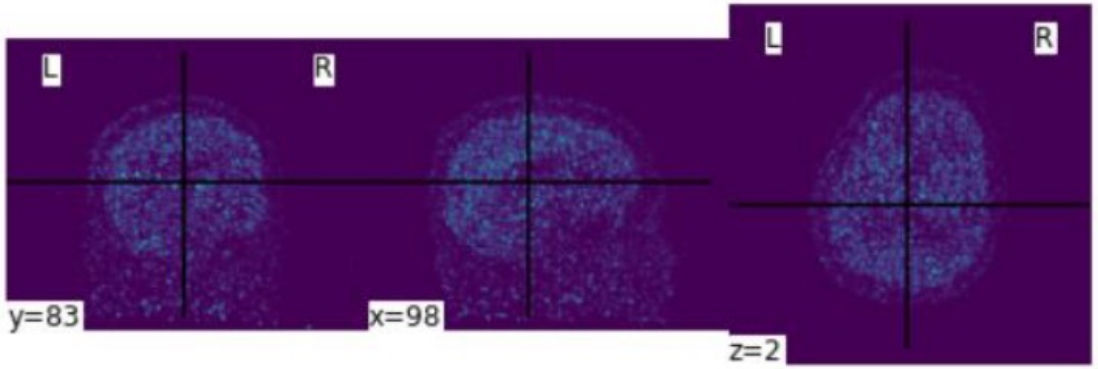
[Han et al. ICLR'16]

PET Images

- Positron Emission Tomography (PET) is an emerging imaging technology that can reveal metabolic activities of a tissue or an organ. Unlike other imaging technologies like CT and MRI that capture anatomical changes. PET scans detect biochemical and physiological changes.
- PET has a wide range of clinical applications, such as cancer diagnosis, tumor detection and early diagnosis of neuro diseases.



(a)



(b)

3D PET Image from ADNI dataset, (a) standard dose (b) low dose

PET Denoising

- The **noise** in PET images is caused by the low coincident photon counts detected during a given scan time and various physical degradation factors.

- In order to acquire high quality PET image for diagnostic purpose, a standard dose of radioactive tracer should be injected to the subject which will lead to higher risk of radiation exposure damage. So, to address this problem, many DL algorithms and networks were proposed to improve the image quality.

- Some of the denoising conventional methods like Gaussian filter smooths out important image structures during the denoising process.

PET Denoising Methods

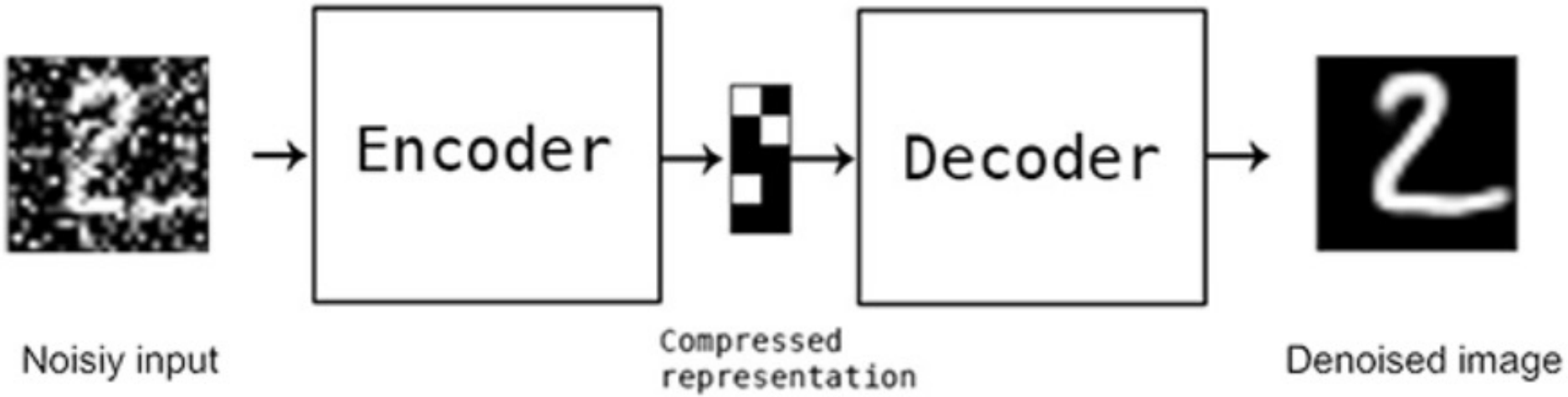
- **Supervised:** Machine learning methods utilize paired low-dose and standard-dose images to train models that can predicts standard-dose images from low-dose inputs.

- **Unsupervised:** DNNs can learn intrinsic structures from corrupted images without pre-training. No prior training pairs are needed, and random noise can be employed as the network input to generate clean images.

- **Deep Image Prior (DIP):** This is an **unsupervised** learning approach, which has no requirement for large data sets and high-quality label images. The original DIP approach learns using a single pair of random-noise input and noisy image.

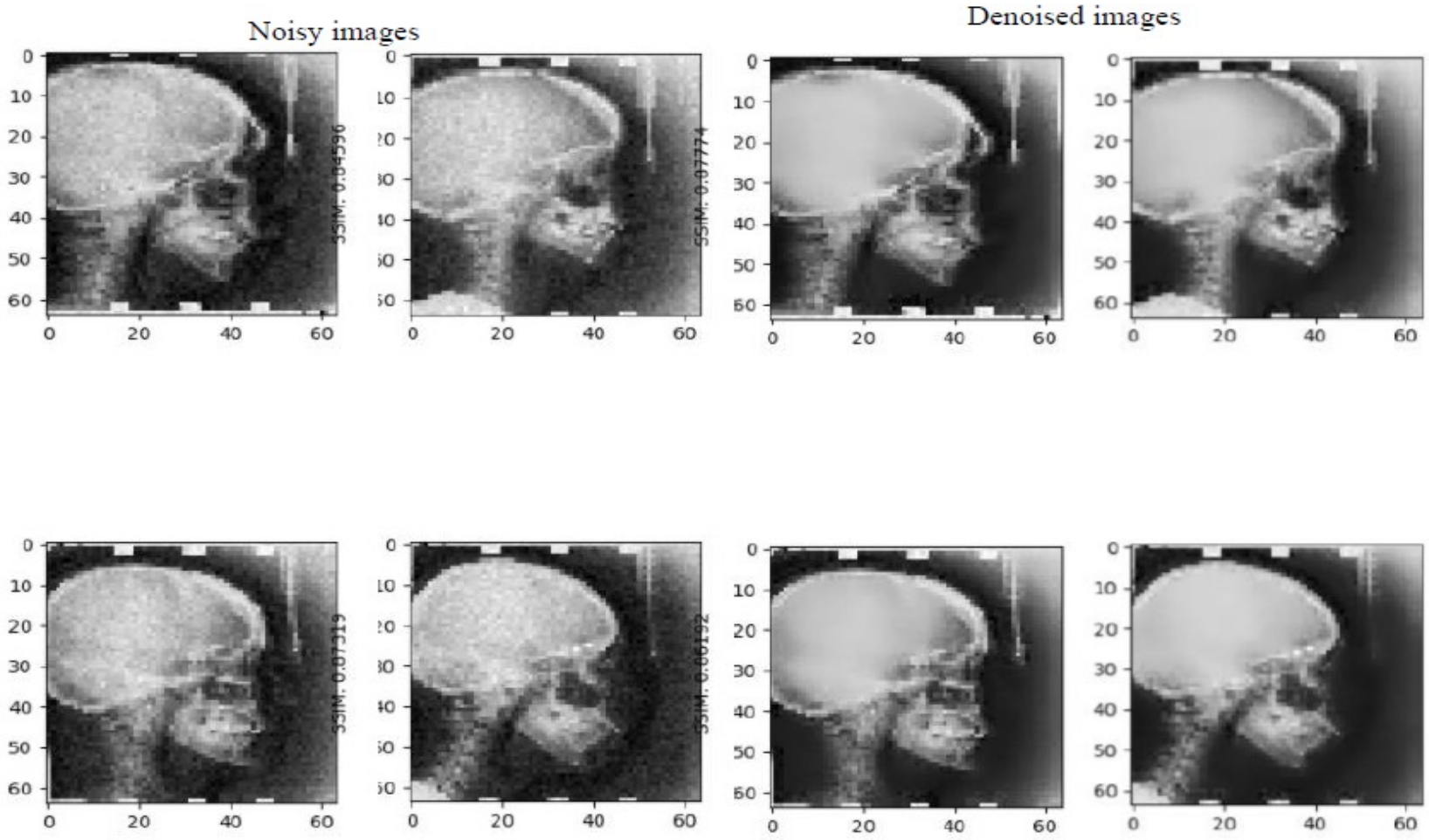
Denoising Autoencoder

- One of the important denoising architecture is autoencoder. Autoencoders are Neural Networks which are commonly used for feature selection and extraction. It has two steps: **encode** for extracting most **important image features** and **decoder** for **constructing denoised image based on those features**.



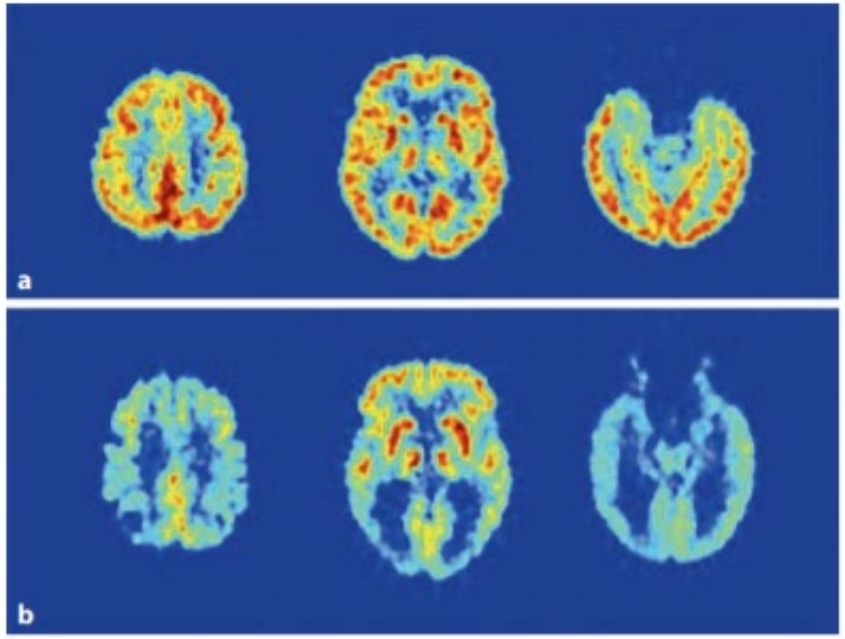
Simulation Results

Results of applying denoising autoencoder to PET images.



PET Classification

PET imaging together with Convolutional Neural Networks helps in the early detection and automated classification of Alzheimer's disease.



PET scans from two representative subjects:
a) normal subject, and
b) AD subject.

Academia Plays a Key Role to Address AI

- Revisit degree required courses, not just offering AI related electives
- Introduce specializations in AI for undergraduates. A sequence of courses that cover all subject areas: relevant mathematics, hardware techniques, tools and modeling environments, and capstone projects in collaboration with local industry.
- At the research and graduate studies level establish centers focused on specific topics of AI research:
  - Medical Imaging
  - Tools and modeling of low power high performance AI platforms
  - AI broad domain project development (social sciences, humanities, Arts, etc.)

Conclusions

- New applications related to Machine Learning are having a major impact on the design of future computer systems
- Industry is heavily invested in ALL aspects of ML, prominently in medical applications. MSF acquired Nuance ($20B)
- Universities have integrated ML into curriculum and continue do so; including specializations in ML
- It is a very crowded field with many players: startups, major corporations, government agencies, and academia.
- Government agencies are actively seeking ideas beyond Deep Neural Networks, such as the DARPA Next AI initiative.
- QC is following ML but has far more challenges and as yet to make it to the main stream, but that is potential next horizon for novel and exotic computing that is totally different from classical computers based on binary switches