Imperial College London

F-E3D: FPGA-based Acceleration of an Efficient 3D Convolutional Neural Network for Human Action Recognition

Hongxiang Fan, Cheng Luo, Chenglong Zeng, Martin Ferianc, Xinyu Niu and Wayne Luk
Department of Computing, Imperial College London
h.fan17@imperial.ac.uk

Motivation

- Human action recognition (HAR)
  - required by demanding applications, e.g. autonomous driving, surveillance…

- Algorithms for HAR with best accuracy
  - 3-dimensional convolutional neural networks (3D CNNs)

- 3D CNN inference on ARM CPU: 0.25 frame per second (fps)
  - does not meet real-time requirements

Challenges

- High Computational Complexity
  - standard 3D-CNNs: at least 3x computations of 2D-CNNs

- Large Numbers of Parameters
  - 3D convolution: parameters in three different dimensions

- Limited Compression Rate
  - By Quantization and 3D Winograd algorithms

Contributions

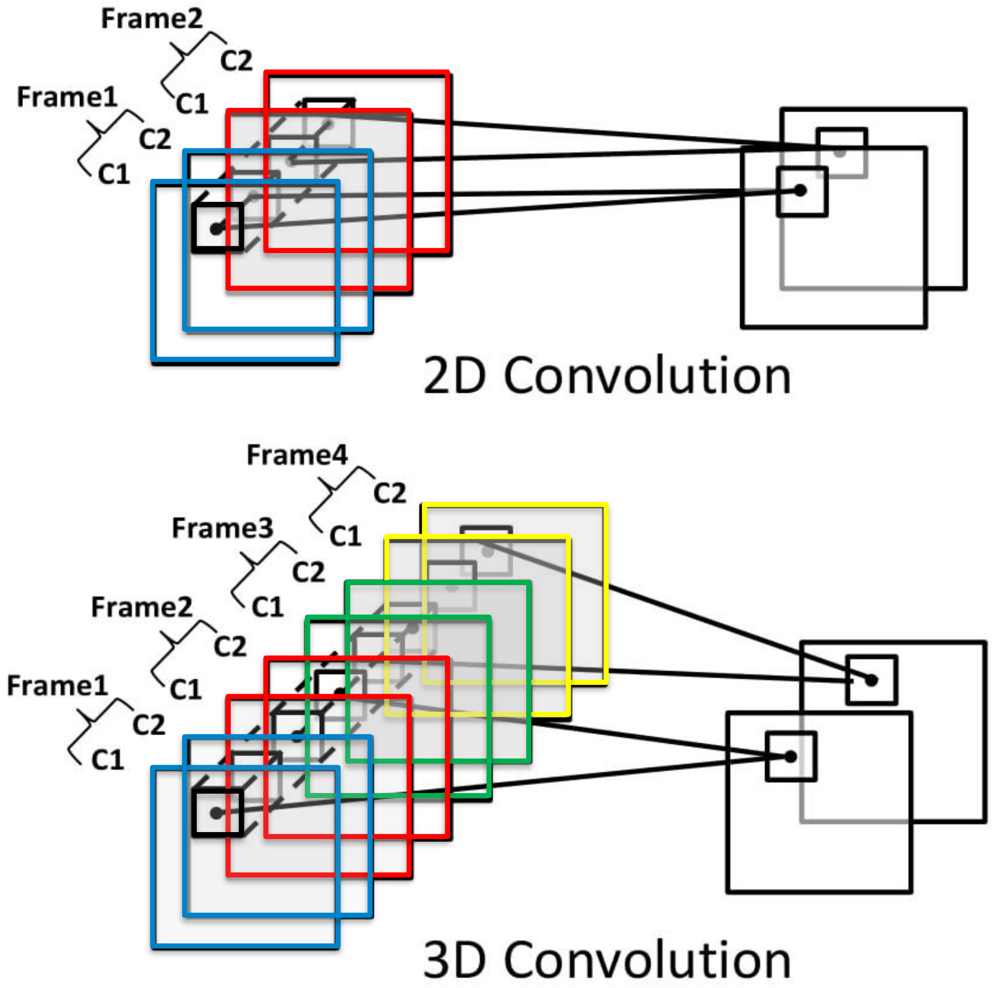
1. An efficient 3D CNN (E3DNet): better than standard 3D CNNs (C3D)
   - 37 times smaller
   - 5% more accurate on UCF101

2. An FPGA-based architecture (F-E3D)
   - high performance and enhanced hardware efficiency

3. Comprehensive comparison
   - with other 3D CNN models on various platforms

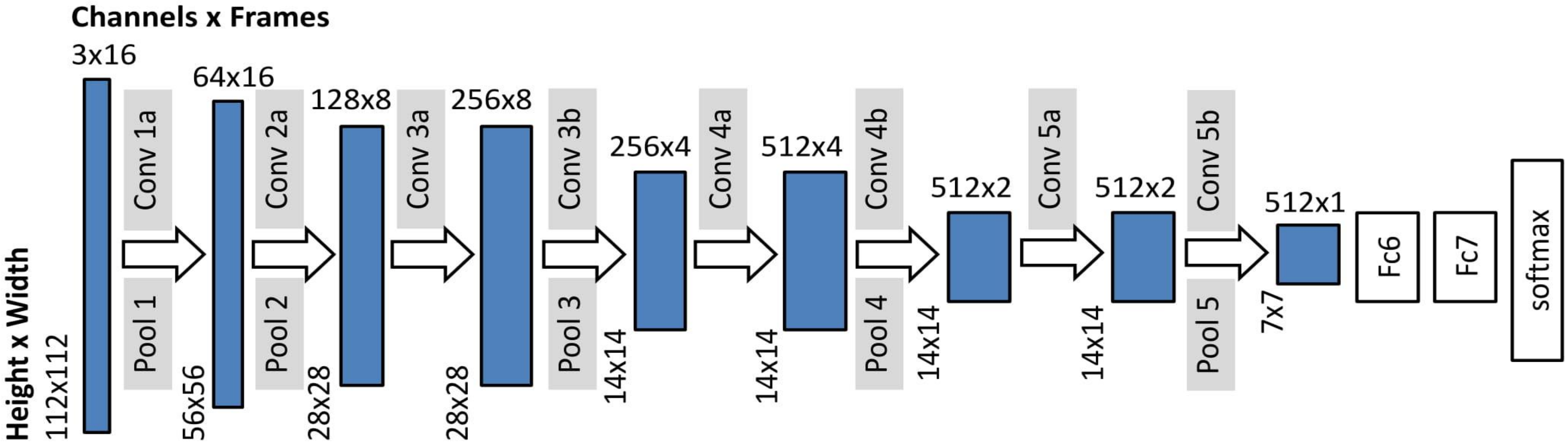
Background: 3D CNNs

- 3D convolution: accumulates results from different frames to generate output feature maps



Background: 3D CNNs

- C3D is one of the most commonly used 3D CNNs for HAR.

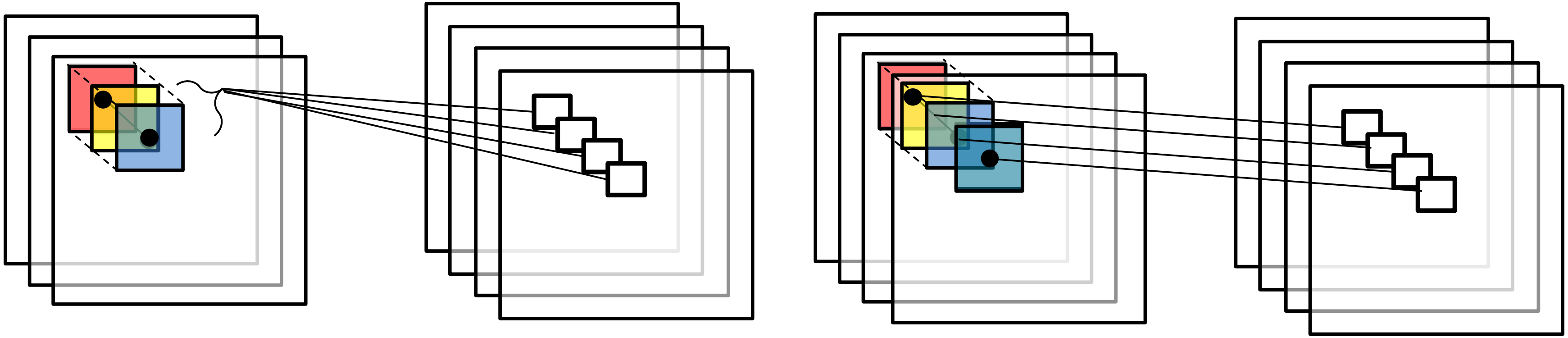


Background: Model Compression

- Quantization
  - Linear integer quantization, Binary and Ternary quantization
- Weight Pruning and Approximation
  - Low-Rank Factorization and Structural Matrix
- Efficient Building Blocks
  - Depth-wise convolution and Bottleneck residual block

Background: Depth-wise Convolution

- Without channel accumulation
- Channel number is equal to filter number

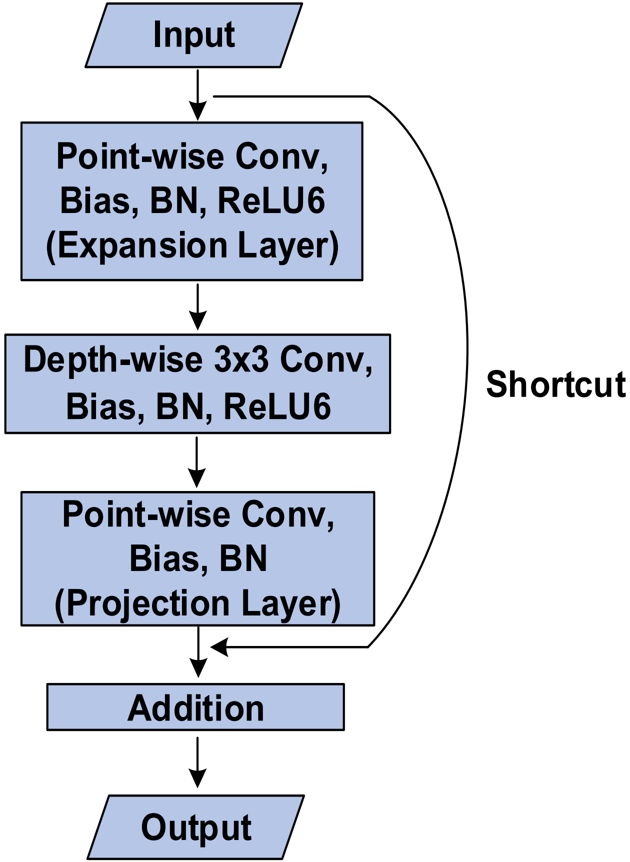


Standard Convolution   Depth-wise Convolution

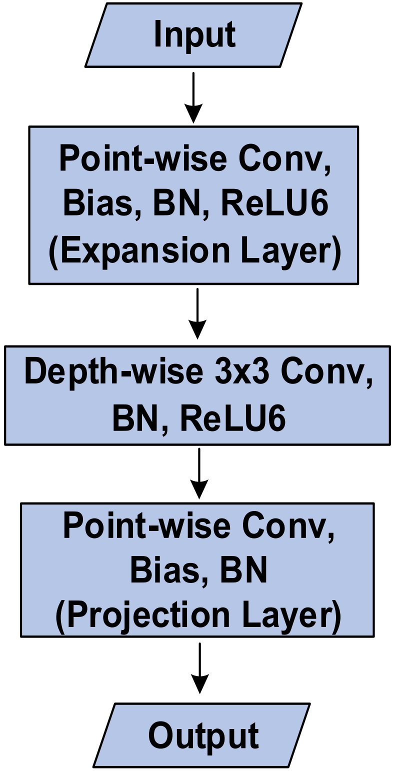
Background: Bottleneck Residual Block

- Fewer parameters
- Fewer operations

* BN: Batch Normalization



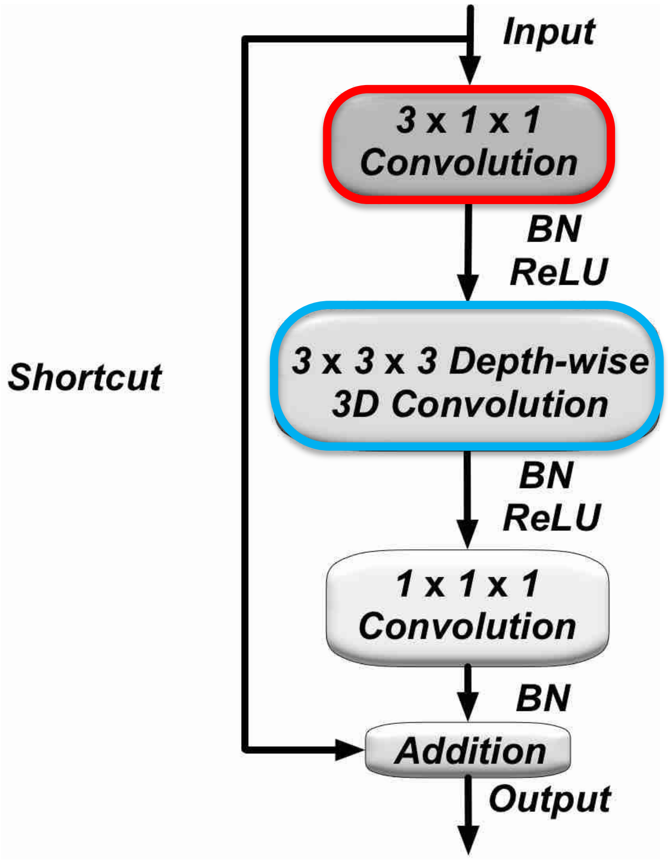
(a) depth-wise stride = 1



(b) depth-wise stride = 2

1. Efficient 3D-CNNs: (a) 3D-1 BRB

- Generalize the BRB to 3D-CNNs
- Expand all 2D convolutions to 3D convolutions
- Temporal kernel size of 3 added to:
  - the first 3D convolution
  - the second 3D convolution



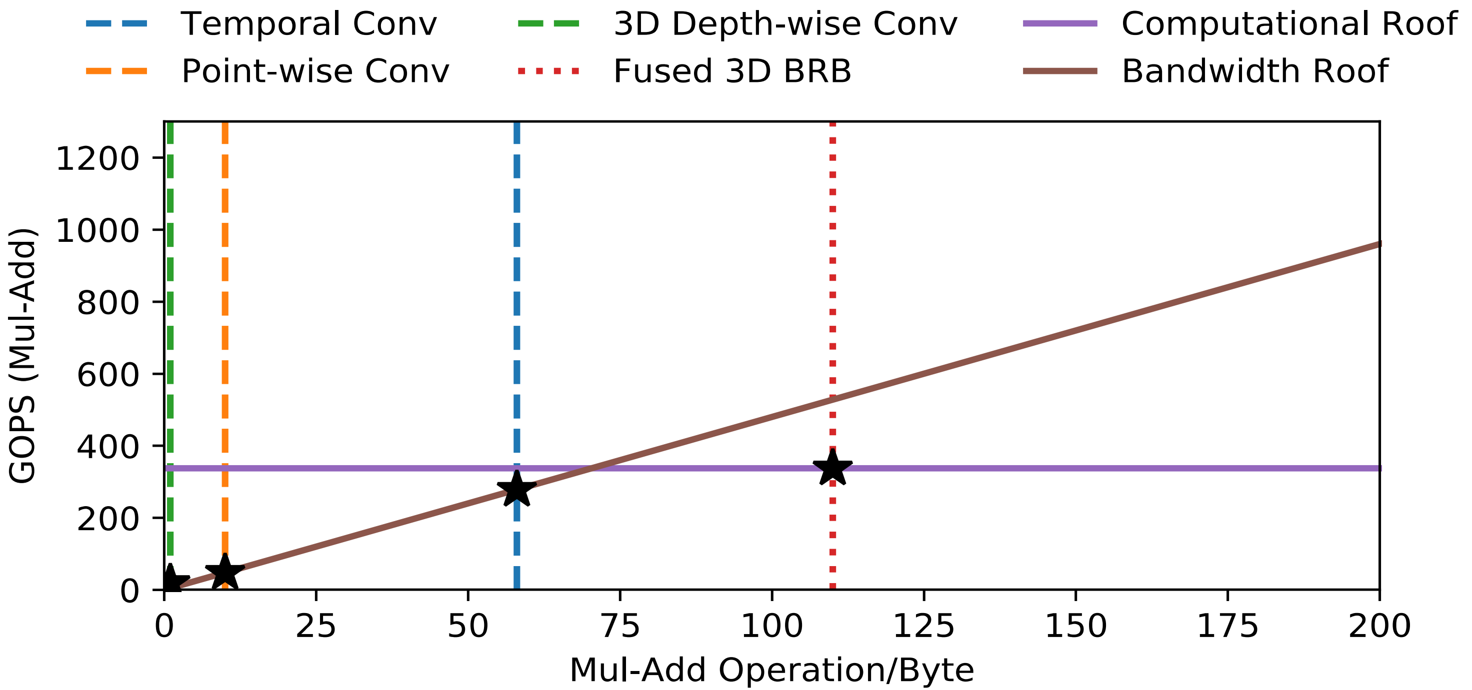
1. Efficient 3D-CNNs: (b) E3DNet

- Similar network structure to MobileNetV2
- 17 3D-1 BRBs
- Input size: 16 x 112 x 112 x 3

Input	**Operation**	t	N_f	n
$16 \times 112^2 \times 3$	Conv $1 \times 3 \times 3$	-	45	1
$16 \times 56^2 \times 45$	Conv $3 \times 1 \times 1$	-	64	1
$16 \times 56^2 \times 64$	3D-1 BRB	1	24	1
$16 \times 56^2 \times 24$	3D-1 BRB	6	24	2
$16 \times 56^2 \times 24$	3D-1 BRB	6	48	4
$8 \times 28^2 \times 48$	3D-1 BRB	6	64	6
$4 \times 14^2 \times 64$	3D-1 BRB	6	96	3
$2 \times 7^2 \times 96$	3D-1 BRB	6	512	1
$2 \times 7^2 \times 512$	GAP	-	-	1
$1 \times 1^2 \times 512$	Conv $1 \times 1 \times 1$	-	k	1

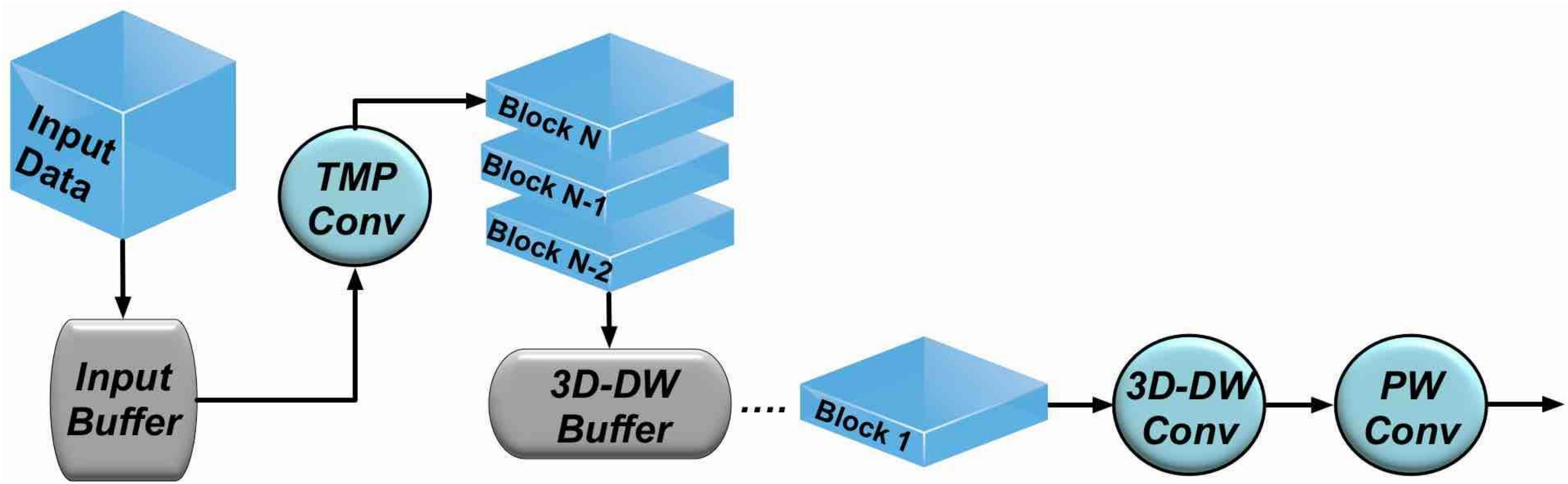
2. Design Methodology: (a) Fused 3D BRB

- Memory-bound if accelerate each layer separately
- Cache the intermedia results within 3D-1 BRB on chip



2. Design Methodology: (b) Online Blocking

- Large on-chip memory requirement
- Online Blocking: Controlling the computational flow



2. Design Methodology: (c) Kernel Reuse

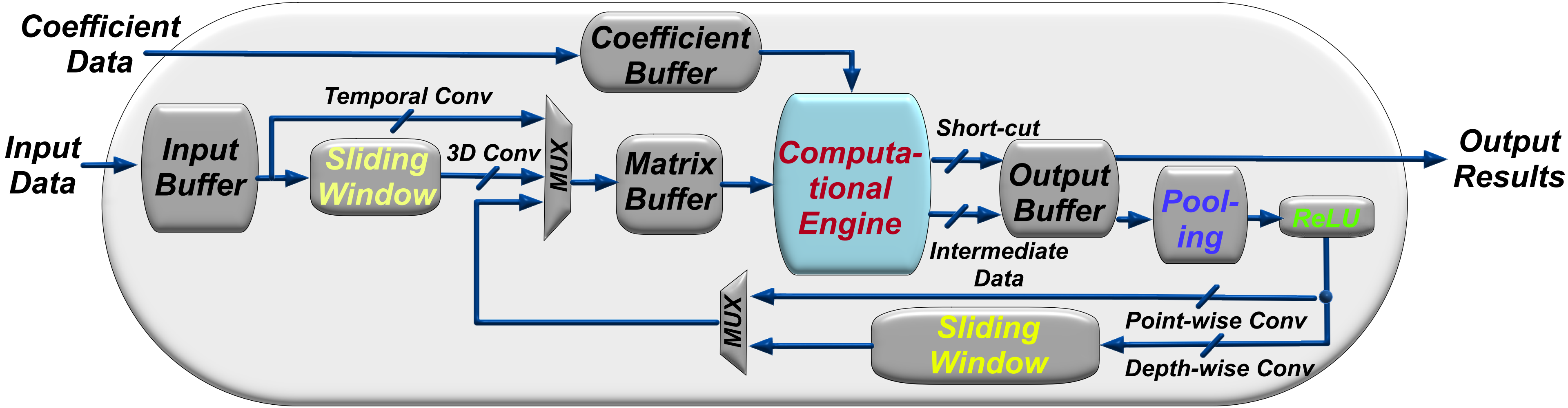
- Map 1x1x1 and 3x1x1 convolution into the computational kernel of 3D depth-wise convolution

Algorithm 2 Kernel Reuse with Temporal Convolution.

for $frm = 0$ to N_l **do**
    ~~**for** $f\_th = frm$ to $frm + K_t$ **do**~~ ▷ Loop Interchange
    for $fltr = 0$ to N_f **do**
        ~~**for** $c = 0$ to N_c **do**~~ ▷ Loop Unrolling
        for $c = 0$ to $\frac{N_c}{K_{dw} \times K_{dw}}$ **do**
            for $h = 0$ to H **do**
                for $w = 0$ to W **do**
                    for $c\_unrol = c \times K_{dw}^2$ to $(c+1) \times K_{dw}^2$ **do**
                        for $f\_th = frm$ to $frm + K_t$ **do**
                            outpt$[frm][fltr][h][w]$+=
                            coef$[f_th][fltr][c_unrol]$ $\times$
                            inpt$[f_th][c_unrol][h][w]$;

3. Hardware Design: (a) Architecture

- mainly consists of a computational engine, sliding window, ReLU, pooling modules and several buffers.



4. Experiment: (a) Setting

- Intel Arria 10SX 660 platform:
  using Verilog toolchain

- Human action recognition on UCF101:
  13320 videos of 101 human action categories

- Input shape:
  16 X112 X 112 X 3

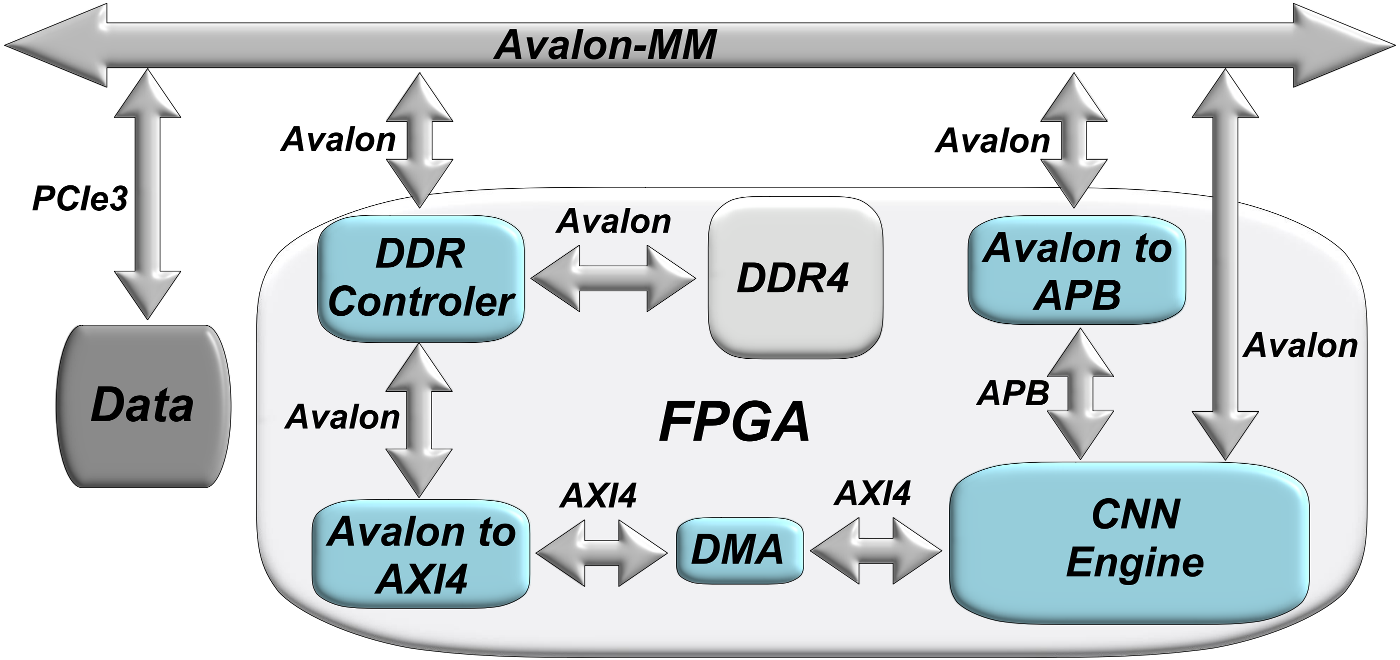
4. Experiment: (b) Model Size and Accuracy

- 37 times smaller and 5% more accurate than C3D

	E3DNet	ResNeXt-101	**P3D**	**C3D**
Clip@1 Accuracy	85.17%	87.7%	84.2%	79.87%
Model Size	8.6MB	365MB	261MB	321MB
Compression Rate	Baseline	42.3	30.3	37.3
MAdds	6.1G	9.8G	19.2G	38.2G
Operation Reduction	Baseline	1.6	3.1	6.2

4. Experiment: (c) FPGA Design

- Avalon memory mapped interface (Avalon-MM)



4. Experiment: (c) FPGA Design

- Resource consumption of FPGA design

	ALMs	DSPs	M20K
Available	251680	1687	2133
Utilization	113828	1584	1578
Percentage Used	45.2%	93.3%	74%

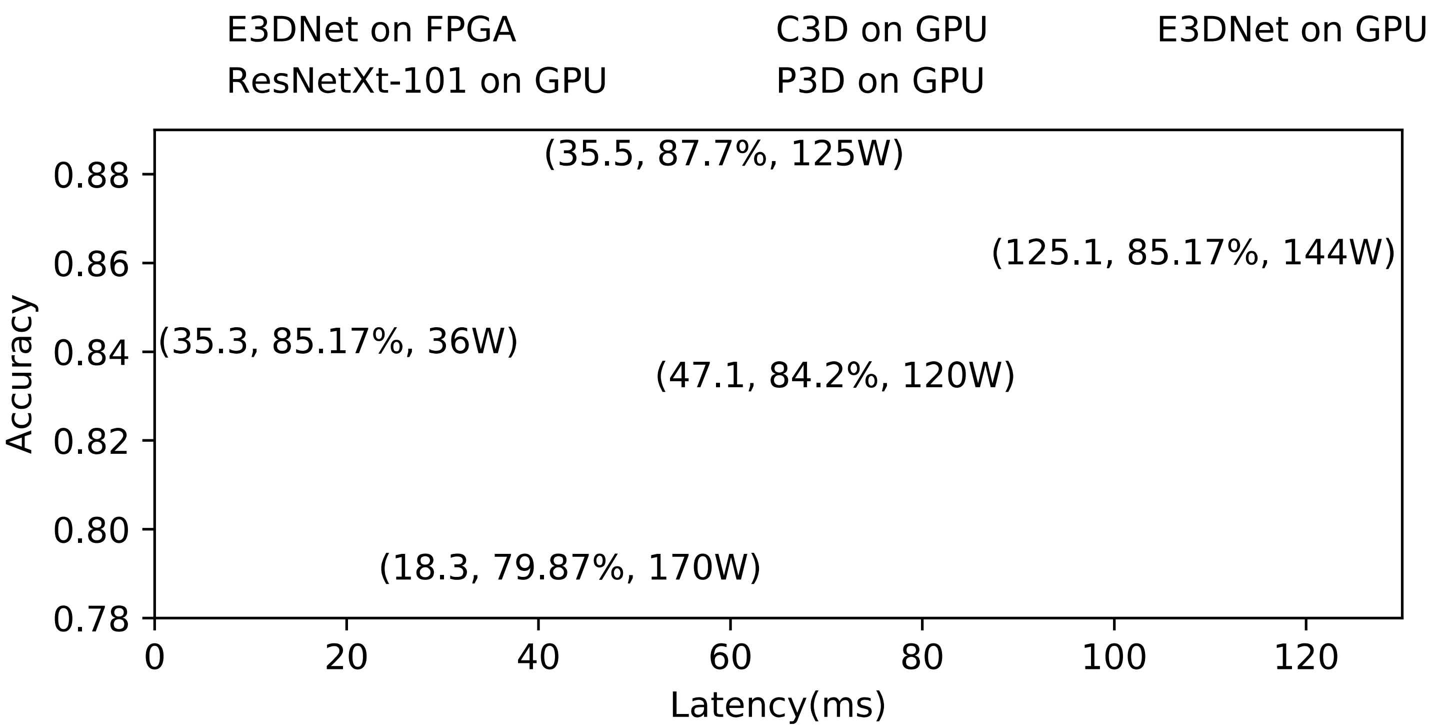
4. Experiment: (d) FPGA Performance Comparison

- Nearly the same performance with GPU with less energy
- 13 times faster previous FPGA design

	CPU	GPU	FPGA	Our Work
Platform	Intel Xeon E5-2680 v2	TITAN X Pascal	Xilinx ZC706	Intel Arria 10 SX660
Frequency	2.8 GHz	1.53 GHz	200 MHz	150 MHz
Model	E3DNet	E3DNet	C3D+SVM	E3DNet
Precision	32bit-float	32bit-float	block-float	32bit-float
Accuracy	85.17%	85.17%	< 81.99%	85.17%
Power (W)	135	240	9.9	36
Latency (ms)	6921.3	41.1	476.8	35.3

4. Experiment: (e) Comparison with Other 3D-CNNs

- The second place in accuracy and speed with the least power consumption



Dot size is proportional to power consumption

Future Work

- Further Improve E3DNet accuracy
  - for human action recognition

- Explore 3D-1 BRB
  - for other 3D computer vision tasks such as medical image diagnosis

- Optimize performance of 3D-1 BRB
  - for other technologies, e.g. CPU, GPU, ASIC

Summary

1. An efficient 3D CNN (E3DNet): better than standard 3D CNNs
   - 37 times smaller
   - 5% more accurate

2. An FPGA-based architecture (F-E3D)
   - high performance and enhanced hardware efficiency

3. Comprehensive comparison
   - with other 3D CNN models on various platforms

Code available at: https://github.com/os-hxfan/E3DNet.git