Optimising 3D-CNN Design towards Human Pose Estimation on Low Power Devices

Manolis Vasileiadis[1,2]
m.vasileiadis16@imperial.ac.uk
Christos-Savvas Bouganis[1]
christos-savvas.bouganis@imperial.ac.uk
Georgios Stavropoulos[2]
stavrop@iti.gr
Dimitrios Tzovaras[2]
dimitrios.tzovaras@iti.gr

[1] Department of Electrical and Electronic Engineering
Imperial College London
London SW7 2AZ, UK

[2] Information Technologies Institute
Centre for Research and Technology Hellas
Thessaloniki, 57001, Greece

Abstract

3D CNN-based architectures have found application in a variety of 3D vision tasks, significantly outperforming earlier approaches. This increase in accuracy, however, has come at the cost of computational complexity, with deep learning models becoming more and more complex, requiring significant computational resources, especially in the case of 3D data. Meanwhile, the growing adoption of low power devices in various technology fields has shifted the research focus towards the implementation of deep learning on systems with limited resources. While plenty of approaches have achieved promising results in terms of reducing the computational complexity in 2D tasks, their applicability in 3D-CNN designs has not been thoroughly researched. The current work aims at filling this void, by investigating a series of efficient CNN design techniques within the scope of 3D-CNNs, in order to produce guidelines for 3D-CNN design that can be applied to already established architectures, reducing their computational complexity. Following these guidelines, a computationally efficient 3D-CNN architecture for human pose estimation from 3D data is proposed, achieving comparable accuracy to the state-of-the-art. The proposed design guidelines are further validated within the scope of 3D object classification, achieving high accuracy results at a low computational cost.

1 Introduction

Deep Learning has revolutionized, in recent years, multiple scientific fields, including Computer Vision [25]. Applied in a broad spectrum of vision applications, convolutional neural networks have achieved remarkable results in a variety of tasks [6, 22, 28, 40], significantly outperforming earlier approaches, and in some cases even surpassing human perception levels [10]. 3D Computer Vision has also benefited from these advances, with multiple works undertaking tasks such as object recognition [55] and reconstruction [4], semantic segmentation [38] and pose estimation[49] from 3D data.

This immense increase in accuracy, however, has come at the cost of computational complexity [21]. The complexity of deep learning models has been steadily growing, increasing the required computational resources during both training and inference.This increase in complexity becomes even more evident in the case of 3D vision tasks, as the addition of a third dimension makes the cost of every new layer even steeper, while also limiting the spatial resolution of the input data, in an attempt to bound this increase.

Meanwhile, the growing adoption of low power devices in various applications, has shifted the research focus towards the implementation of deep learning methods on such systems, making it necessary to reduce the computational complexity of the proposed networks while maintaining the level of accuracy. Many approaches have been proposed attempting to reduce the complexity of deep learning models by either shrinking the models [13] or minimizing the cost of computations [54]. While these methods have achieved promising results in 2D vision tasks, their applicability in 3D-CNN designs has not been thoroughly researched, hindering the utilization of such architectures on systems with limited resources.

The proposed work aims at filling this void, by investigating a series of efficient 3D-CNN design techniques, towards their implementation on low power devices. More specifically, the main contributions of this paper are:

- A series of network design guidelines, that can reduce the computational complexity of already established 3D-CNN architectures, while maintaining comparable accuracy
- A novel 3D-CNN architecture for multi-person 3D human pose estimation from 3D data, based on the above guidelines, which performs comparably to state-of-the-art methods, while requiring significantly fewer computational resources
- Experimental validation of the applicability of the design guidelines in other 3D tasks, through the optimisation of a 3D object classification network

The rest of the paper is organized as follows: Section 2 provides a summary of the state-of-the-art in the fields of 3D-CNNs in Computer Vision and efficient network design, Section 3 introduces the computational complexity metrics used throughout the paper and Section 4 describes the network optimisation pipeline. Section 5 presents the results from the comparative experimental evaluation and, finally, Section 6 concludes the paper.

2 Related Work

2.1 3D-CNNs in Computer Vision

3D-CNNs have been employed in Computer Vision mainly in the scope of 3D vision, where the three spatial dimensions correspond to the real world coordinates. Towards object detection, the Voxnet architecture [30] utilizes 3 different occupancy models, along with a 4-layer detection network. In [55] 3D geometric shapes are represented as a probabilistic distribution of binary variables and combined with a Convolutional Deep Belief Network, while Qi *et al.* [37] introduce auxiliary learning tasks to scrutinize details of 3D objects and anisotropic kernels to probe for long-distance interactions. Song and Xiao [46] propose the first 3D RPN to learn objectness from geometric shapes, utilizing the projective Directional Truncated Signed Distance Function representations. In [41] a hybrid Grid-Octree data structure is presented, allowing to focus memory allocation and computation to dense regions, while in [24, 62] random sampling is employed to generate multidimensional features from raw 3D points. In [5] the projective DTSDF representation is employed for 3D hand pose estimation, while in [33, 49] 3D extensions of the hourglass [35] and convolutional pose machines [52] architectures are introduced for 3D human pose estimation. Towards 3D shape reconstruction, 3D-R2N2 [4] employs a 3D-Convolutional LSTM along with a 3D Deconvolutional Neural Network, Wu *et al*. [53] extend GANs to 3D space, and Yi *et al*. [59] present the Densely Connected 3D Auto-encoder architecture.

Additionally, 3D-CNNs have also been employed for spatio-temporal learning, where the time axis acts as the third dimension. Multiple approaches use 3D-CNNs for action recognition from short video sequences [12, 18, 47]. Varol *et al*. [48] extend these approaches to longer temporal convolutions, also exploring the impact of optical flow. Wang *et al*. [51] introduce saliency-aware maps into the 3D-CNN architecture, while the I3D model [1] achieves improvements in action classification utilizing an Inception-like 3D-CNN network. Building upon the action recognition models, [36, 57] add LSTMs and visual-semantic Embedding for video description, while [27, 31, 32] utilize 3D CNNs for gesture recognition.

2.2 Computationally Efficient CNN Design

Multiple recent research efforts have focused in building small and efficient neural networks, suitable for systems with limited resources, such as mobile devices. A common approach is to reduce the number of parameters in the convolutions, with the MobileNets [13, 42], Shufflenet [29, 60] and Xception [3] models utilizing depth-wise separable convolutions [44]. Meanwhile, Wang *et al*. [50] introduce factorized convolutions and Jin et al. [19] propose the use of topological connections for further reducing computational requirements. Other small networks include the Squeezenet [16] which uses a bottleneck approach to design a very small network, structured transform networks [45] and deep fried convnets [56].

A different approach is to obtain a small network by shrinking a pre-trained network, with the most popular network compression techniques including: 1) quantisation [17, 54], in which filter weight matrices are quantised to lower bit depths, 2) hashing [2], which uses a low-cost hash function to randomly group connection weights into hash buckets, witl connections within the same bucket sharing a single parameter, and 3) Huffman coding [8] which reduces the size of the networks using Huffman coding on the weights of the network.

While these techniques have been thoroughly tested on 2D architectures, corresponding work towards efficient 3D-CNN designs has been rather limited. Ye *et al*. [58] present a preliminary investigation of the use of 3D depthwise convolutions in 3D classification and reconstruction. Zhi *et al*. [61] leverage multitask learning to improve the efficiency of Voxnet, while Kumawat *et al*. [23] propose the ReLPV block, a four-layer alternative efficient representation of the standard 3D convolutional layer. Additionally, efficient 3D-CNN architectures have been proposed within the scope of spatio-temporal learning [15, 39].

The current work, on the other hand, attempts a thorough investigation of 3D-CNN design optimisation techniques, using a state-of-the-art 3D-CNN human pose estimation network [49] as a baseline. All the stages of the architecture are optimised in terms of accuracy and computational cost, leading to the definition of a series of design guidelines for the generation of a 3D-CNN model of comparable accuracy but of lower complexity.

3 Computational Complexity Metrics

To evaluate the computational complexity of a network design, three framework agnostic metrics are established (Table 1):

- **MACs** describe the number of the required arithmetic operations. In the case of neural networks most of the computations are dot products (multiplication followed by addition), with 1 MAC corresponding to one multiplication-addition
- **Network Parameters** are the number of trainable variables at each layer of a network, that need to be learned and stored
- **MEMs** describe the required amount of memory access operations in order to read and write all the data during compute. Disregarding, for simplicity, advanced techniques such as caching, three MEM groups are processed at each layer: a) read the input, b) read the weights, c) write the output.

Layer	Input	Filter	MACs	MEMs	Params
FC	D	n nodes	$D \cdot n$	$D + D \cdot n + n$	$D \cdot n$
Conv	$D^3 \times C_{in}$	$k \times k \times k \times C_{out}$ stride s groups g	$C_{in} \cdot C_{out} \cdot k^3 \cdot D^3 / g \cdot s^3$	$D^3 \cdot C_{in} +$ $k^3 \cdot C_{out} / g +$ $D^3 \cdot C_{out} / s^3$	$C_{in} \cdot k^3 \cdot C_{out} / g$
Pool	$D^3 \times C_{in}$	$k \times k \times k$ stride s	$k^3 \cdot D^3 \cdot C_{in} / s^3$	$D^3 \cdot C_{in} +$ $D^3 \cdot C_{in} / s^3$	n/a
ReLu	$D^3 \times C_{in}$	n/a	$D^3 \cdot C_{in}$	$2 \cdot D^3 \cdot C_{in}$	n/a

Table 1: Calculation of the computational complexity metrics for 3D-CNNs. D is the input's spatial dimension, C_{in}, C_{out} are the input and output channels and k is the kernel size.

4 Network Design Optimisation

The optimisation process involves the revision of the state-of-the-art 3D-CNN architecture for human pose estimation from Vasileiadis *et al.* [49]. The final goal is to reduce its overall computational complexity, while maintaining its accuracy, achieving an optimal trade-off.

The baseline network employs a fully-convolutional 3D-CNN architecture, that uses as input a 3D voxel grid and produces per-voxel likelihood maps of human joints and body parts, for multi-person 3D human pose estimation from 3D point cloud data [49] (Fig. 1).

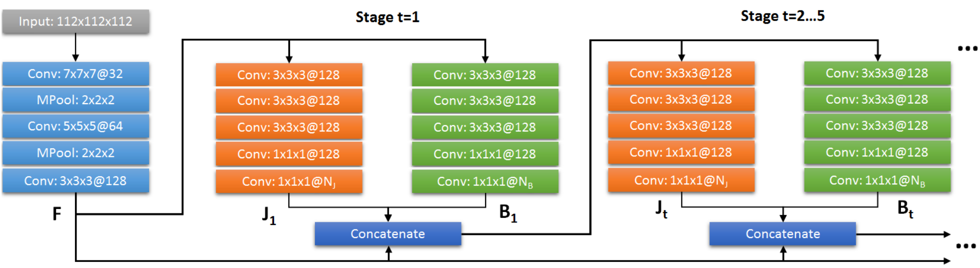

Figure 1: The baseline architecture from Vasileiadis *et al.* [49]. The 3D feature maps F are generated from a 112^3 input voxel grid, and passed to the sequential multistage structure, which produces per-voxel likelihood maps J_t, B_t for N_J joints and N_B body parts respectively. The feature maps and stage predictions are then concatenated and passed to the next stage

4.1 Optimisation Protocol

A protocol is established to consistently evaluate the effect in accuracy and computational complexity of each potential network design. Every alternative network architecture is trained from scratch and evaluated on the ITOP-front dataset [9], following the same training guidelines as in [49], with the *Mean Average Precision at 0.1m (mAP)* metric used for evaluation. Additionally, the three complexity metrics are estimated for each architecture. Overall, the goal at each optimisation step is to reduce the network's total computational complexity, while maintaining the accuracy of the previous step.

4.1.1 Efficient Convolutional Building Blocks

The first step in the design optimisation process is to find efficient alternatives to standard convolutions. Inspired by the use of depthwise separable convolutions [44], towards the same goal, in multiple proposed 2D-CNN architectures, five specific blocks are investigated, based on the Mobilenets [13, 42], Shufflenet [29, 60] and Bottleneck [11] models (Fig. 2).

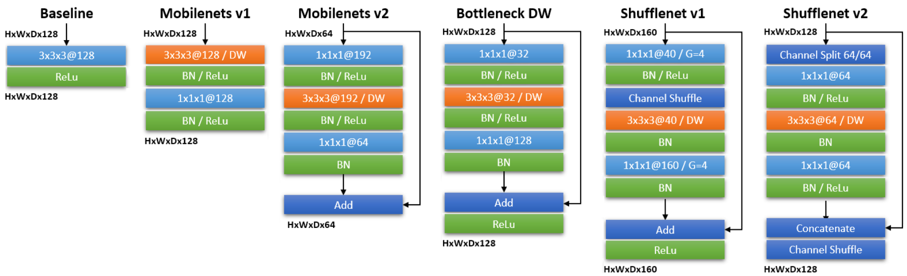

Figure 2: The basic convolutional building blocks ivestigated, based on the Mobilenets [13, 42], Shufflenet [29, 60] and Residual Bottleneck [11] architectures.

While in the corresponding papers, the authors propose specific architectures based on these blocks, herein standard convolutions are directly substituted by the corresponding efficient blocks, maintaining the kernel sizes and channels widths of the baseline model. For the Mobilenets v2 and Shufflenet v1 blocks, the channel width is slightly altered in order to produce an architecture with a similar computational cost to the other three (Fig. 2, Table 2(a)). When a layer is followed by strided Max Pooling, the strided versions of the blocks are used, to maintain the spatial resolution of the baseline model. Moreover, for blocks that include residual connections, an identity layer is added, if the input and output do not have the same number of channels. Finally, the first input layer, and the final two pointwise layers at the end of each stage, are not substituted, as they are investigated on their own below.

From the evaluation of the building blocks (Table 2(a)), it becomes evident that all potential architectures achieve massive reductions, over 90%, in terms of MACs and parameters, while approaching the accuracy of the baseline model, with an average decrease in mAP of just 1.5%. Of the five architectures, the Shufflenet v2 [29] building block not only achieves the highest accuracy score, but also presents the lowest complexity across all three metrics.

4.1.2 Input Layer

The first layer in the baseline model is comprised of a $7 \times 7 \times 7 \times 32$ convolution followed by strided Max pooling. Since it deals with the full resolution input grid, it accounts for approximately half of the total MACs of the model, thus making it necessary to optimise it. Two potential blocks are investigated: a) standard strided convolution and b) Mobilenets v1 strided block, where the stride is employed on the depthwise convolution. The other four blocks from Section 4.1.1 are not employed as they would require an extra expensive pointwise convolution at the full resolution of the input.

Both alternative first layers (Table 2(b)), manage to outperform the baseline layer, while reducing the total MACs in half. The Mobilenets v1 block achieves larger complexity and accuracy gains, even though it may seem counter-intuitive to perform a depthwise convolution on a single channel input.

4.1.3 Kernel Size

In the previous steps the kernel dimensions of the baseline layers are maintained in the depthwise convolutions. Meanwhile, in the original Mobilenets and Shufflenet architectures, the authors opt-out to use only kernel size $k = 3$ for the depthwise convolutions, along with the pointwise convolutions, as most deep learning frameworks have custom, highly optimised implementations for these kernel types. The same principle is applied to the optimised architecture, specifically to the first two blocks as the rest of the network uses only kernel sizes $k = \{1, 3\}$. Additionally, the use of dilated convolutions is investigated, attempting to maintain the receptive field of the original kernels. The $3 \times 3 \times 3$ kernels present the same performance with the baseline kernels, slightly reducing the MACs and network parameters, while the dilated convolutional kernels result in a significant drop in accuracy (Table 2(c)).

4.1.4 End-of-stage Pointwise Layers

At the end of each prediction stage two pointwise convolutional layers are employed, following [28] where fully connected layers are substituted with pointwise convolutions in order to generate fully convolutional architectures. While pointwise convolutions are generally considered computationally efficient, in the current optimised architecture the second to last end-of-stage pointwise layers account for approximately a quarter of the total MACs and parameters, as they are applied on the full channel width (128 channels). Removing those layers results in a small decrease in accuracy of less than 0.7%, leading, however, to significant gains in computational complexity, thus justifying this decision (Table 2(d))

4.1.5 Squeeze-and-Excitation Blocks

Moving in the opposite direction to the previous steps, the utilization of *Squeeze - and - Excitation (SE) blocks* [14] is investigated, as a means of increasing the accuracy at a minor computational cost. SE blocks adaptively recalibrate the feature responses of each channel by modelling interdependencies between them, and have demonstrated significant improvements in performance for existing state-of-the-art CNN architectures.

SE blocks with squeeze ratio $r = 4$ are added after the last BN/ReLu block in every Shufflenet v2 block, leading to an impressive 1.3% increase in accuracy (Table 2(e)).

4.1.6 Input Data Representation

The baseline model [49] employs the computationally inexpensive Hitgrid [30] volumetric representation to model the input data to a dense, fixed size 3D occupancy grid. Using, however, slightly more complex representations, could potentially increase the overall accuracy, leading to a better accuracy/complexity trade-off.

Two low cost data modelling approaches are investigated: a) projective D-TSDF [46] which offers a fast approximation of the TSDF model [34] and b) PointGrid [24] which combines the dense volumetric representation with raw point cloud features. For the former, the truncation limit is set to $l = 5$ voxels, while for the later each voxel is represented by $K = 6$ points and the grid resolution is halved, with stride set to $s = 1$. All three representation models require less than $10ms$ to be generated from 25K 3D points on a single CPU core.

However, neither of the two representations manages to outperform the Hitgrid model (Table 2(f)). In the case of the projective D-TSDF, it can be partially attributed to the presence of obstacles that may block the projection beam, while for PointGrid, the low number of 3D points around smaller body parts (hands, feet) results in randomly duplicating data which can make it harder for the convolution to extract good features.

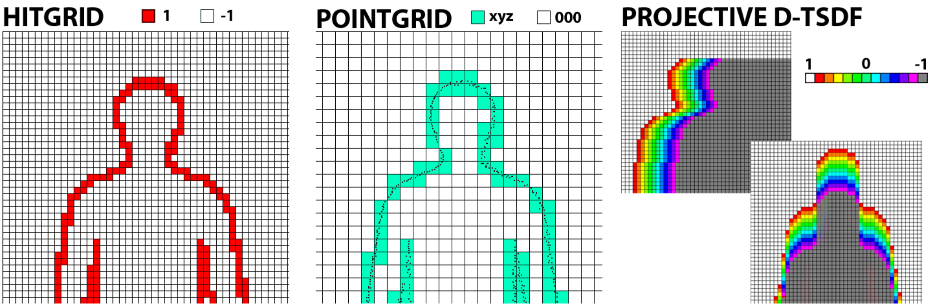

Figure 3: 2D illustrations of the *Hitgrid*, *PointGrid* and *projective D-TSDF* representations

4.2 Network Dimensionality Parameterization

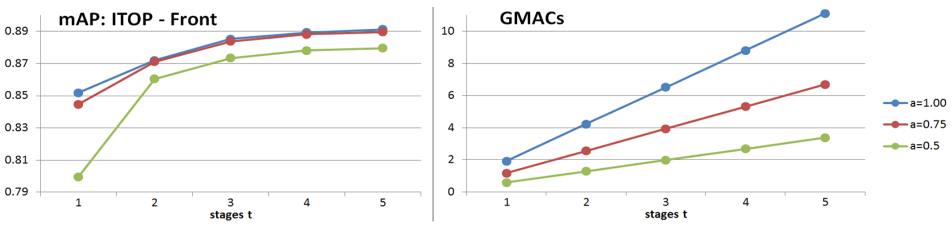

Figure 4: mAP on the ITOP-front dataset and computational complexity in giga-MACs, for different width multiplier values a and number of sequential stages t

In order to further optimise the accuracy/complexity trade-off, two network dimensionality hyperparameters are introduced: a) the width multiplier hyperparameter $a \in (0, 1]$, which uniformly reduces the width of the network at each layer and b) the sequential stages hyperparameter $t > 0$, which defines the number of repetitive prediction stages in the sequential multi-stage structure (the baseline configuration described above corresponds to $\{a = 1, t = 5\}$).

By evaluating all potential combinations (Fig. 4), the $\{a = 0.75, t = 4\}$ configuration achieves the best accuracy/complexity trade-off, as it presents minimal loss in accuracy (

-0.3% compared to $\{a=1, t=5\}$), while requiring approximately half of the computational resources.

4.3 Design Guidelines Overview

Based on the study presented above, the design guidelines for improved accuracy/complexity trade-off of established 3D-CNN architectures, are defined as follows:

a. Hitgrid [30] volumetric data representation model
b. Mobilenets v1 [13] convolutional block for the first layer
c. Shufflenet v2 [29] convolutional blocks for the rest non-pointwise layers
d. Kernel size $k=3$ for all non-pointwise layers
e. Strided convolution instead of pooling
f. Remove extra pointwise layers, besides last one
g. Squeeze and Excitation blocks [14]
h. Uniform reduction in network width and depth for best accuracy/complexity trade-off

Layer	mAP	MACs	Params	MEMs
(a) Building Blocks				
Baseline [49]	0.8923	377.65 G	14720 K	3.11 G
Mobilenets v1 [13]	0.8772	33.74 G	840 K	4.34 G
Mobilenets v2 [42]	0.8769	33.88 G	1184 K	6.49 G
Bottleneck DW [11]	0.8765	30.91 G	707 K	4.93 G
Shufflenet v1 [60]	0.8744	34.20 G	685 K	5.91 G
Shufflenet v2 [29]	0.8782	30.17 G	674 K	4.25 G
(b) Input Layer				
Conv + Max Pooling	0.8782	30.17 G	674 K	4.25 G
Conv / s=2	0.8830	16.64 G	674 K	3.57 G
Mobilenets v1 / s=2	0.8841	14.78 G	663 K	3.58 G
(c) Kernel Size				
7x / 5x / 3x	0.8841	14.78 G	663 K	3.58 G
3x / 3x / 3x	0.8841	14.58 G	656 K	3.58 G
3x / 3x / 3x dilated	0.8454	14.58 G	656 K	3.58 G
(d) Pointwise Layers				
Keep layers	0.8841	14.58 G	656 K	3.58 G
Remove layers	0.8782	10.97 G	491 K	3.12 G
(e) SE Blocks [14]				
No SE blocks	0.8782	10.97 G	491 K	3.12 G
SE blocks / r=4	0.8913	11.11 G	558 K	3.83 G
(f) Data Representation Model				
HitGrid [30]	0.8913	11.11 G	558 K	3.83 G
PointGrid / K=6 [24]	0.8801	11.28 G	558 K	3.84 G
Projective D-TSDF / l=5 [46]	0.8853	11.13 G	558 K	3.83 G

Table 2: mAP on the ITOP-front dataset and overall computational complexity for different network architectures. At each step, the last optimised architecture (gray) is revised

5 Experimental Evaluation

5.1 3D Human Pose Estimation

Following the evaluation protocol in [49], the final optimised architecture[1] is evaluated on the single-person ITOP [9] and multi-person CMU PanopticStudio (Band and Haggling sequences) [20] datasets, matching the accuracy of the Baseline [49] on the ITOP-front and CMU Haggling subsets. On the other hand, a larger drop in performance is observed on the more challenging ITOP-top and CMU Band subsets (2.5% and 1.5% respectively), with the overall accuracy, however, being comparable to the state-of-the-art [7, 33, 49].

	ITOP - front	ITOP - top	CMU Haggling	CMU Band
Head	0.983	0.981	0.983	0.981
Neck	0.986	0.983	0.995	0.981
Shoulders	0.967	0.957	0.991	0.975
Elbows	0.819	0.772	0.982	0.967
Hands	0.700	0.627	0.965	0.955
Torso	0.983	0.973	n/a	n/a
Hips	0.955	0.844	0.961	0.827
Knees	0.910	0.776	0.980	0.963
Feet	0.870	0.708	0.978	0.947
Ours MEAN	0.888	0.820	0.978	0.945
Baseline [49]	0.893	0.845	0.988	0.960
Moon 2018 [33]	0.887	0.834	n/a	n/a
Guo 2017 [7]	0.849	0.755	n/a	n/a

Table 3: Comparison (mAP@0.1m) of the proposed optimised architecture to the state-of-the-art on the ITOP and CMU PanopticStudio datasets

	MACs	Params	MEMs	CPU / GPU	Model Size
Ours	5.32 G	264 K	2.35 G	5.45 / 0.17 s	1.06 MB
Baseline [49]	377.65 G	14720 K	3.10 G	15.21 / 0.32 s	58.88 MB
Ours Single	1.34 G	135 K	0.64 G	1.38 / 0.05 s	0.54 MB
Baseline Single	105.71 G	7440 K	1.03 G	4.69 / 0.13 s	29.76 MB
Moon 2018 [33]	36.39 G	3398 K	1.28 G	3.49 / 0.11 s	13.59 MB

Table 4: Computational complexity and inference runtime comparison of the proposed optimised architecture and state-of-the-art 3D-CNN human pose estimation architectures

Moreover, the computational complexity of the optimised architecture is estimated and compared against the Baseline [49] and Moon *et al.* [33]. A "single person" configuration is also presented, using only the joints detection branch and an 88^3 input grid, as in [33] (Table 4). Additionally, all architectures are benchmarked on a CPU-only and a GPU-based hardware configurations[2], in order to provide an indication about their actual performance.

The proposed network achieves massive gains in terms of MACs and Network parameters, with smaller improvements in MEMs, mainly due to the optimisation process not

[1] a detailed diagram of the optimised architecture is available in the supplemental material

[2] Intel Core i5-4670K, 2 cores activated / Nvidia GTX970

affecting the dimensions of the feature maps. Meanwhile, the speedup in runtime is in the range of 3x on both configurations, with the single model reaching 20fps on the GPU.

5.2 3D Object Classification

In order to investigate the effectiveness of the design guidelines in other 3D tasks, they are applied in 3D object classification. A naive 3D extension of VGG13 [26] is employed as a baseline and optimised following the proposed guidelines. Both the baseline and optimised architectures are evaluated on the ModelNet10 subset of the ShapeNet dataset [55], which includes 55k 3D CAD models, sampled as $32 \times 32 \times 32$ grids, split into 10 object categories.

The optimised VGG13 3D architecture matches the accuracy of the baseline model (Table 5), while performing comparably to the state-of-the-art [23, 30, 43, 61]. Moreover, a reduction across all three computational complexity metrics is observed (Table 6), similar to the human pose estimation task, with a speedup of over 2x on the CPU-only configuration.

VGG13 3D Ours	**VGG13 3D**	**Maturana 2015 [30]**	**Zhi 2017 [61]**	**Sedaghat 2017 [43]**	**Kumawat 2019 [23]**
0.910	0.916	0.920	0.934	0.938	0.944

Table 5: Accuracy comparison of the baseline and optimised VGG13 3D architectures, to state-of-the-art volumetric 3D-CNN models on the ModelNet10 dataset

	MACs	**Params**	**MEMs**	**CPU / GPU**	**Model Size**
VGG13 3D Ours	0.26 G	19 M	243 M	0.19 / 0.018 s	76 MB
VGG13 3D	8.59 G	61 M	340 M	0.42 / 0.019 s	244 MB

Table 6: Computational complexity and inference runtime comparison of the baseline and proposed optimised VGG13 3D architectures

6 Conclusions

This paper presented a series of CNN design guidelines, towards the computationally efficient redesign of established 3D-CNN architectures, in order to make them suitable for deployment on low power devices. Following these guidelines, a novel 3D-CNN human pose estimation architecture was proposed, achieving comparable results to the state-of-the-art, at a significantly lower computational cost, demonstrating the effectiveness of the introduced guidelines. Moreover, the proposed design guidelines were further validated through their application in 3D object classification.

Future work could include evaluating the guidelines in more 3D-CNN architectures, and investigating techniques for further complexity reduction, such as more memory-efficient data representation, non-depthwise 3D convolutions [23] and lower bit-width compute [17].

Acknowledgements

This work was supported by the EU Horizon 2020 funded project “SMart MobILilty at the European land borders” (SMILE) under grant agreement number 740931.

References

[1] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[2] Wenlin Chen, James Wilson, Stephen Tyree, Kilian Weinberger, and Yixin Chen. Compressing neural networks with the hashing trick. In *International Conference on Machine Learning*, pages 2285–2294, 2015.

[3] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[4] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016.

[5] Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann. 3D convolutional neural networks for efficient and robust hand pose estimation from single depth images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[7] Hengkai Guo, Guijin Wang, Xinghao Chen, and Cairong Zhang. Towards good practices for deep 3D hand pose estimation. *arXiv preprint arXiv:1707.07248*, 2017.

[8] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[9] Albert Haque, Boya Peng, Zelun Luo, Alexandre Alahi, Serena Yeung, and Li Fei-Fei. Towards viewpoint invariant 3D human pose estimation. In *European Conference on Computer Vision*, pages 160–177. Springer, 2016.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Rui Hou, Chen Chen, and Mubarak Shah. Tube convolutional neural network (T-CNN) for action detection in videos. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[13] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[14] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[15] Zhongxu Hu, Youmin Hu, Jie Liu, Bo Wu, Dongmin Han, and Thomas Kurfess. 3D separable convolutional neural network for dynamic hand gesture recognition. *Neurocomputing*, 318:151–161, 2018.

[16] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[17] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2018.

[18] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:221–231, 2010.

[19] Jonghoon Jin, Aysegul Dundar, and Eugenio Culurciello. Flattened convolutional neural networks for feedforward acceleration. *arXiv preprint arXiv:1412.5474*, 2014.

[20] Hanbyul Joo, Tomas Simon, Xulong Li, Hao Liu, Lei Tan, Lin Gui, Sean Banerjee, Timothy Godisart, Bart Nabbe, Iain Matthews, et al. Panoptic studio: A massively multiview system for social interaction capture. *IEEE transactions on pattern analysis and machine intelligence*, 41(1):190–204, 2019.

[21] Daniel Justus, John Brennan, Stephen Bonner, and Andrew Stephen McGough. Predicting the computational cost of deep learning models. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 3873–3882. IEEE, 2018.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[23] Sudhakar Kumawat and Shanmuganathan Raman. LP-3DCNN: Unveiling local phase in 3D convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[24] Truc Le and Ye Duan. Pointgrid: A deep network for 3D shape understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[25] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553): 436, 2015.

[26] S. Liu and W. Deng. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 730–734, Nov 2015.

[27] Zhi Liu, Chenyang Zhang, and Yingli Tian. 3D-based deep convolutional neural network for action recognition with depth sequences. *Image and Vision Computing*, 55: 93–100, 2016.

[28] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[29] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 116–131, 2018.

[30] Daniel Maturana and Sebastian Scherer. Voxnet: A 3D convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015.

[31] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Jan Kautz. Hand gesture recognition with 3D convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2015.

[32] Pavlo Molchanov, Xiaodong Yang, Shalini Gupta, Kihwan Kim, Stephen Tyree, and Jan Kautz. Online detection and classification of dynamic hand gestures with recurrent 3D convolutional neural network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[33] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. V2v-posenet: Voxel-to-voxel prediction network for accurate 3D hand and human pose estimation from a single depth map. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5079–5088, 2018.

[34] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew W Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *ISMAR*, volume 11, pages 127–136, 2011.

[35] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016.

[36] Yingwei Pan, Tao Mei, Ting Yao, Houqiang Li, and Yong Rui. Jointly modeling embedding and translation to bridge video and language. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[37] Charles R. Qi, Hao Su, Matthias Niessner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view cnns for object classification on 3D data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[38] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[39] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3D residual networks. In *proceedings of the IEEE International Conference on Computer Vision*, pages 5533–5541, 2017.

[40] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[41] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3D representations at high resolutions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[42] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[43] Nima Sedaghat, Mohammadreza Zolfaghari, E Amiri, and Thomas Brox. Orientation-boosted voxel nets for 3D object recognition. In *British Machine Vision Conference (BMVC)*, 2017.

[44] Laurent Sifre and Stéphane Mallat. Rigid-motion scattering for image classification. *PhD thesis, Ph. D. thesis*, 1:3, 2014.

[45] Vikas Sindhwani, Tara Sainath, and Sanjiv Kumar. Structured transforms for small-footprint deep learning. In *Advances in Neural Information Processing Systems*, pages 3088–3096, 2015.

[46] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3D object detection in rgb-d images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[47] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3D convolutional networks. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[48] Gül Varol, Ivan Laptev, and Cordelia Schmid. Long-term temporal convolutions for action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1510–1517, 2018.

[49] Manolis Vasileiadis, Christos-Savvas Bouganis, and Dimitrios Tzovaras. Multi-person 3D pose estimation from 3D cloud data using 3D convolutional neural networks. *Computer Vision and Image Understanding*, 185:12 – 23, 2019.

[50] Min Wang, Baoyuan Liu, and Hassan Foroosh. Factorized convolutional neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 545–553, 2017.

[51] Xuanhan Wang, Lianli Gao, Jingkuan Song, and Heng Shen. Beyond frame-level CNN: Saliency-aware 3D CNN with LSTM for video action recognition. *IEEE Signal Processing Letters*, PP:1–1, 09 2016.

[52] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[53] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In *Advances in neural information processing systems*, pages 82–90, 2016.

[54] Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, and Jian Cheng. Quantized convolutional neural networks for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4820–4828, 2016.

[55] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3D shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[56] Zichao Yang, Marcin Moczulski, Misha Denil, Nando de Freitas, Alex Smola, Le Song, and Ziyu Wang. Deep fried convnets. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1476–1483, 2015.

[57] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[58] Rongtian Ye, Fangyu Liu, and Liqiang Zhang. 3D depthwise convolution: Reducing model parameters in 3D vision tasks. In *Canadian Conference on Artificial Intelligence*, pages 186–199. Springer, 2019.

[59] Li Yi, Lin Shao, Manolis Savva, Haibin Huang, Yang Zhou, Qirui Wang, Benjamin Graham, Martin Engelcke, Roman Klokov, Victor Lempitsky, et al. Large-scale 3D shape reconstruction and segmentation from shapenet core55. *arXiv preprint arXiv:1710.06104*, 2017.

[60] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018.

[61] Shuaifeng Zhi, Yongxiang Liu, Xiang Li, and Yulan Guo. Lightnet: A lightweight 3D convolutional neural network for real-time 3D object recognition. In *3DOR*, 2017.

[62] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3D object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.