



Universiteit  
Leiden  
The Netherlands

## System-level design for efficient execution of CNNs at the edge

Minakova, S.

### Citation

Minakova, S. (2022, November 24). *System-level design for efficient execution of CNNs at the edge*. Retrieved from <https://hdl.handle.net/1887/3487044>

Version: Publisher's Version

License: [Licence agreement concerning inclusion of doctoral thesis in the Institutional Repository of the University of Leiden](#)

Downloaded from: <https://hdl.handle.net/1887/3487044>

**Note:** To cite this publication please use the final published version (if applicable).

# **System-level Design For Efficient Execution of CNNs at the Edge**

Svetlana Minakova

This work has received funding from the European Unions Horizon 2020 Research and Innovation project under grant agreement No. 780788.

System-level Design For Efficient Execution of CNNs at the Edge. Svetlana Minakova. - Dissertation Universiteit Leiden.

Copyright © 2022 by Svetlana Minakova.

This dissertation was typeset using L<sup>A</sup>T<sub>E</sub>X.

Cover design: from images generated using DALL-E mini Deep Learning algorithm [20]. The images are combined and post-processed by Anna Minakova.

# **System-level Design For Efficient Execution of CNNs at the Edge**

## **Proefschrift**

ter verkrijging van  
de graad van doctor aan de Universiteit Leiden,  
op gezag van rector magnificus prof.dr.ir. H. Bijl,  
volgens besluit van het college voor promoties  
te verdedigen op donderdag 24 november 2022  
klokke 11.15 uur

door

Svetlana Minakova  
geboren te Ryazan, Rusland  
in 1993

**Promotores:**

Dr. T.P. Stefanov

Prof.dr. H.A.G. Wijshoff

**Promotiecommissie:**

Prof.dr. S. Ha (Seoul National University)

Prof.dr. J. Castrillon (Technical University of Dresden)

Prof.dr. H.E. Bal (Vrije Universiteit Amsterdam)

Prof.dr. A. Plaat

Prof.dr. N. Mentens

Prof.dr. M.S.K. Lew

*To my family and friends*



# Contents

<b>Table of Contents</b>	<b>vii</b>
<b>List of Figures</b>	<b>xi</b>
<b>List of Tables</b>	<b>xiii</b>
<b>List of Abbreviations</b>	<b>xv</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Accuracy and platform-aware characteristics of a CNN . . . . .	3
1.2 Requirements posed on a CNN executed at the Edge . . . . .	4
1.3 Current trends in the design of CNNs executed at the Edge . .	4
1.4 Limitations of the state-of-the-art design flow for CNNs executed at the Edge . . . . .	7
1.4.1 Limitation 1 . . . . .	7
1.4.2 Limitation 2 . . . . .	8
1.5 Research contributions . . . . .	9
1.5.1 RC1: Methodology for high-throughput CNN inference	12
1.5.2 RC2: Methodology for low-memory CNN inference . .	13
1.5.3 Methodology for run-time adaptive inference of CNN-based applications . . . . .	13
1.5.4 Methodology for joint memory optimization of multiple CNNs . . . . .	14
1.6 Thesis organization . . . . .	15
<b>2 Background</b>	<b>17</b>
2.1 CNN model . . . . .	18
2.1.1 Layer in the CNN model . . . . .	18
2.1.2 Edge in the CNN model . . . . .	22
2.2 CNN deployment and inference at the Edge . . . . .	22



2.3	Edge platform used for CNN inference . . . . .	24
2.4	Task- and data-level parallelism available in a CNN . . . . .	25
2.5	CSDF and SDF models of computation . . . . .	28
2.6	Genetic Algorithm (GA) . . . . .	30
<b>3</b>	<b>Methodology for high-throughput CNN inference</b>	<b>33</b>
3.1	Problem statement . . . . .	34
3.2	Contributions . . . . .	35
3.3	Related work . . . . .	36
3.4	Edge platform model . . . . .	37
3.5	Methodology . . . . .	38
3.5.1	CNN-to-SDF conversion . . . . .	41
3.5.2	GA-based mapping . . . . .	42
3.5.3	CNN-to-CSDF model conversion . . . . .	44
3.6	Experimental results . . . . .	47
3.7	Conclusion . . . . .	49
<b>4</b>	<b>Methodology for low-memory CNN inference</b>	<b>51</b>
4.1	Problem statement . . . . .	51
4.2	Contributions . . . . .	52
4.3	Related Work . . . . .	54
4.4	Motivational Example . . . . .	55
4.5	Methodology . . . . .	60
4.5.1	Phases derivation . . . . .	61
4.5.2	CNN-to-CSDF model conversion . . . . .	62
4.6	Experimental Results . . . . .	66
4.7	Conclusion . . . . .	69
<b>5</b>	<b>Methodology for run-time adaptive inference of CNN-based applica-</b>	<b>71</b>
	<b>tions</b>	
5.1	Problem statement . . . . .	71
5.2	Contributions . . . . .	72
5.3	Related Work . . . . .	73
5.4	Motivational Example . . . . .	74
5.5	SBRS methodology . . . . .	78
5.6	Automated scenarios derivation . . . . .	79
5.7	SBRS application model . . . . .	81
5.7.1	Scenarios supergraph . . . . .	82
5.7.2	Control node . . . . .	85
5.7.3	Control edges . . . . .	85

---

5.7.4	Deployment and inference . . . . .	86
5.8	SBRS MoC automated derivation . . . . .	86
5.9	Transition protocol . . . . .	89
5.10	Experimental Study . . . . .	94
5.10.1	Automated scenarios derivation . . . . .	95
5.10.2	SBRS MoC memory reuse efficiency . . . . .	99
5.10.3	SBRS-TP efficiency . . . . .	101
5.10.4	Comparative study . . . . .	103
5.11	Conclusion . . . . .	105
<b>6</b>	<b>Methodology for joint memory optimization of multiple CNNs</b>	<b>107</b>
6.1	Problem statement . . . . .	107
6.2	Contributions . . . . .	108
6.3	Related Work . . . . .	109
6.4	CNN-based application . . . . .	111
6.5	Methodology . . . . .	113
6.5.1	Buffers Reuse Algorithm . . . . .	115
6.5.2	Buffers Reduction Algorithm . . . . .	118
6.5.3	Final application derivation . . . . .	123
6.6	Experimental Results . . . . .	124
6.6.1	Comparison to existing memory reuse methodologies .	124
6.6.2	Joint use of quantization and our proposed methodology	128
6.7	Conclusions . . . . .	132
<b>7</b>	<b>Summary and concluding remarks</b>	<b>133</b>
	<b>Bibliography</b>	<b>137</b>
	<b>Summary</b>	<b>149</b>
	<b>Samenvatting</b>	<b>151</b>
	<b>List of Publications</b>	<b>153</b>
	<b>Curriculum Vitae</b>	<b>155</b>
	<b>Acknowledgments</b>	<b>157</b>



# List of Figures

1.1	CNN . . . . .	2
1.2	Execution of CNNs as cloud services and at the Edge . . . . .	2
1.3	Current trends in the design and inference of CNN-based applications executed at the Edge . . . . .	5
1.4	CNNs associated with alternative manners of execution . . . . .	7
1.5	Extended CNN design flow . . . . .	10
2.1	CNN model . . . . .	18
2.2	Processing of input data $X_2$ by layer $l_2$ . . . . .	21
2.3	Padding . . . . .	22
2.4	Jetson TX2 edge platform . . . . .	24
2.5	data-level parallelism . . . . .	26
2.6	task-level (pipeline) parallelism . . . . .	27
2.7	CSDF model of computation . . . . .	29
2.8	SDF model of computation . . . . .	29
2.9	Chromosome . . . . .	30
2.10	Single-gene mutation . . . . .	30
2.11	Simple two-parent recombination (cross-over) . . . . .	30
3.1	Methodology for high-throughput CNN inference . . . . .	38
3.2	CNN (input) model . . . . .	39
3.3	SDF (analysis) model . . . . .	39
3.4	CSDF (executable CNN inference) model . . . . .	40
3.5	Mapping chromosome example . . . . .	43
4.1	Example CNN . . . . .	57
4.2	Input data processing by layer $l_2$ . . . . .	58
4.3	Input data processing by layer $l_3$ , Ex4 . . . . .	59
4.4	Methodology for low-memory CNN inference . . . . .	60
4.5	CSDF model, derived from the CNN model shown in Figure 4.1 . . . . .	62

---

5.1	Execution of a CNN-based application, affected by the application environment and designed using different methodologies	76
5.2	SBRS methodology	79
5.3	Application scenarios	80
5.4	SBRS MoC	82
5.5	Switching from scenario $CNN^1$ to scenario $CNN^2$	90
5.6	Pareto-fronts based on 3 evaluation parameters, namely, accuracy (F1-score for Pascal VOC), throughput and energy	98
5.7	SBRS-TP efficiency evaluation	102
5.8	Comparison between SBRS MoC and MSDNet CNN [39], performing classification on the CIFAR-10 dataset with throughput-driven adaptive mechanism	104
6.1	Example CNN-based application <i>APP</i>	111
6.2	Our methodology design flow	114
6.3	Experimental results	130

# List of Tables

2.1	Attributes of layer $l_i$ . . . . .	19
2.2	Most common CNN layer types and operators . . . . .	19
3.1	Mapping example . . . . .	43
3.2	Experimental results, average over 100 runs . . . . .	48
4.1	Execution of CNN inference with phases . . . . .	56
4.2	Evaluation of our memory reduction methodology . . . . .	66
4.3	CNN characteristics affecting CNN memory reduction and throughput decrease . . . . .	67
5.1	Capturing of scenarios' components (layers and edges) in the scenarios supergraph . . . . .	83
5.2	CNN-based applications . . . . .	96
5.3	Algorithm parameters for platform-aware NAS [82] . . . . .	97
5.4	Scenarios derived from pareto-fronts shown in Figure 5.6 for three applications shown in Table 5.2 . . . . .	98
5.5	SBRS MoC memory reuse efficiency evaluation . . . . .	100
6.1	Naive CNN buffers allocation . . . . .	112
6.2	Reused CNN buffers . . . . .	115
6.3	reduced CNN buffers . . . . .	120
6.4	Chromosome . . . . .	121
6.5	Comparison of the memory reduction principles and features associated with the memory reuse methodologies in [76], [65], and our proposed methodology . . . . .	125
6.6	Experimental Results . . . . .	126
6.7	Applications . . . . .	128
6.8	Quantization in the TensorFlow DL framework [1] . . . . .	129



# List of Abbreviations

BFS	Breadth-First Search
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CSDF	Cyclo-Static Data Flow
DL	Deep Learning
DSE	Design Space Exploration
FC	Fully Connected
FLOP	floating-point operation
FPGA	Field-programmable Gate Array
GA	Genetic Algorithm
GPU	Graphics Processing Unit
HAR	Human Activity Recognition
IoT	Internet-of-Things
KD	Knowledge Distillation
MB	MegaBytes
MoC	Model of Computation
MPSoC	Multi-Processor System-on-Chip
NAS	Neural Architecture Search



ONNX	Open Neural Network Exchange Format
SBRS	Scenario-Based Run-time Switching
SDF	Synchronous Data Flow
SOTA	State Of The Art
SSR	Scenario Switch Request
TPU	Tensor Processing Unit
UAV	Unmanned Aerial Vehicle