Intelligent Architectures

5LIL0

Convolutional Neural Network Design

Berk Ulker

b.ulker@tue.nl

TUEindhoven

2019

Overview

Network design :

- Design space
- Tradeoffs
- Common and practical methods

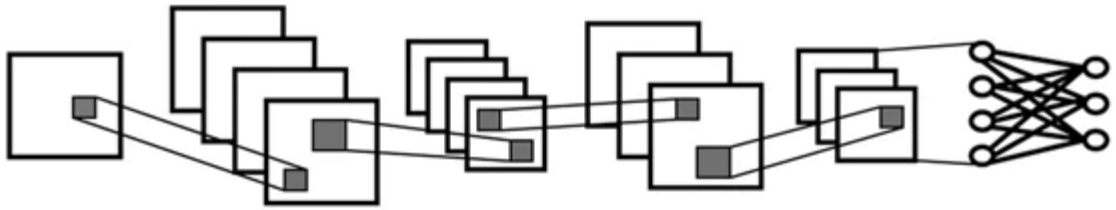
Network Design

Designing deep neural networks is still more art than science

- Large design space
- Many architecture solutions for a single problem

Recent research focuses on

- Automated design
- Guided optimization



Network Design

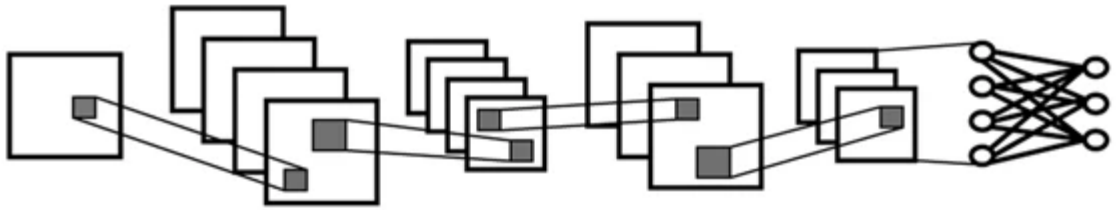
Network design procedure

- Understand problem
- Evaluate application requirements and resource limitations
- Design the architecture
- Training, validation and reiteration

Design space

Network design space has many dimensions

- Network size, depth and width
- Operator composition
- Specialized building blocks
- Optimizations



Common Trade-offs in NN Design

We can move in design space with tradeoffs on:

- Accuracy / memory use
- Accuracy / latency
- Accuracy / energy consumption
- Energy consumption / speed

Speed-Accuracy Tradeoff

Network Optimization Techniques

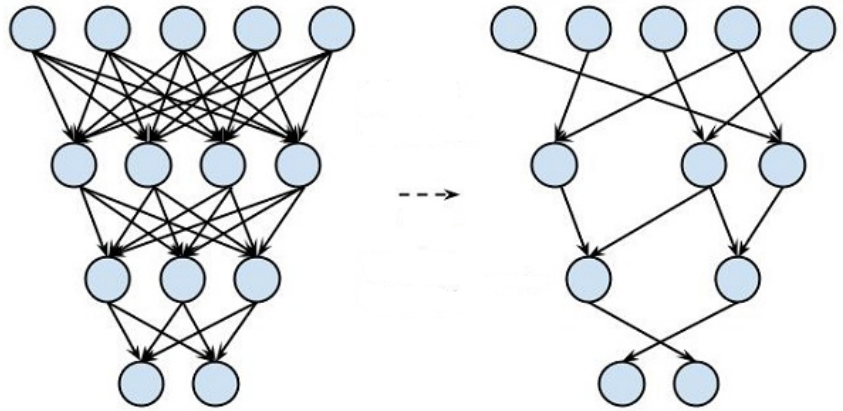
Several techniques exist to optimize network execution efficiency

- Pruning
- Quantization
- Weight scaling
- Tensor decomposition

Network Pruning

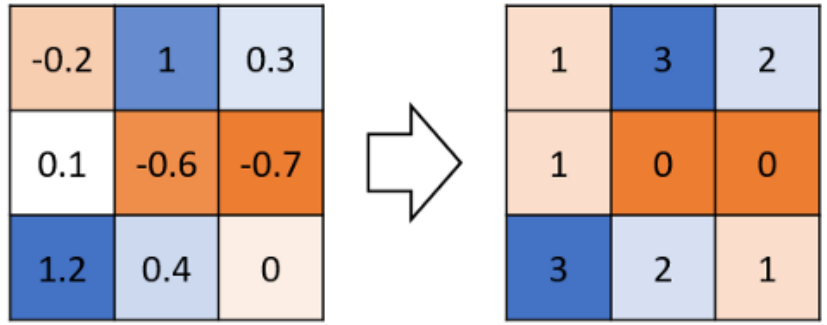
Network pruning is removal of nodes, connections or kernels

- Can be part of the training – learning both weights and connections
- Can be adaptively/selectively applied
- Benefits may be limited for non-structured pruning



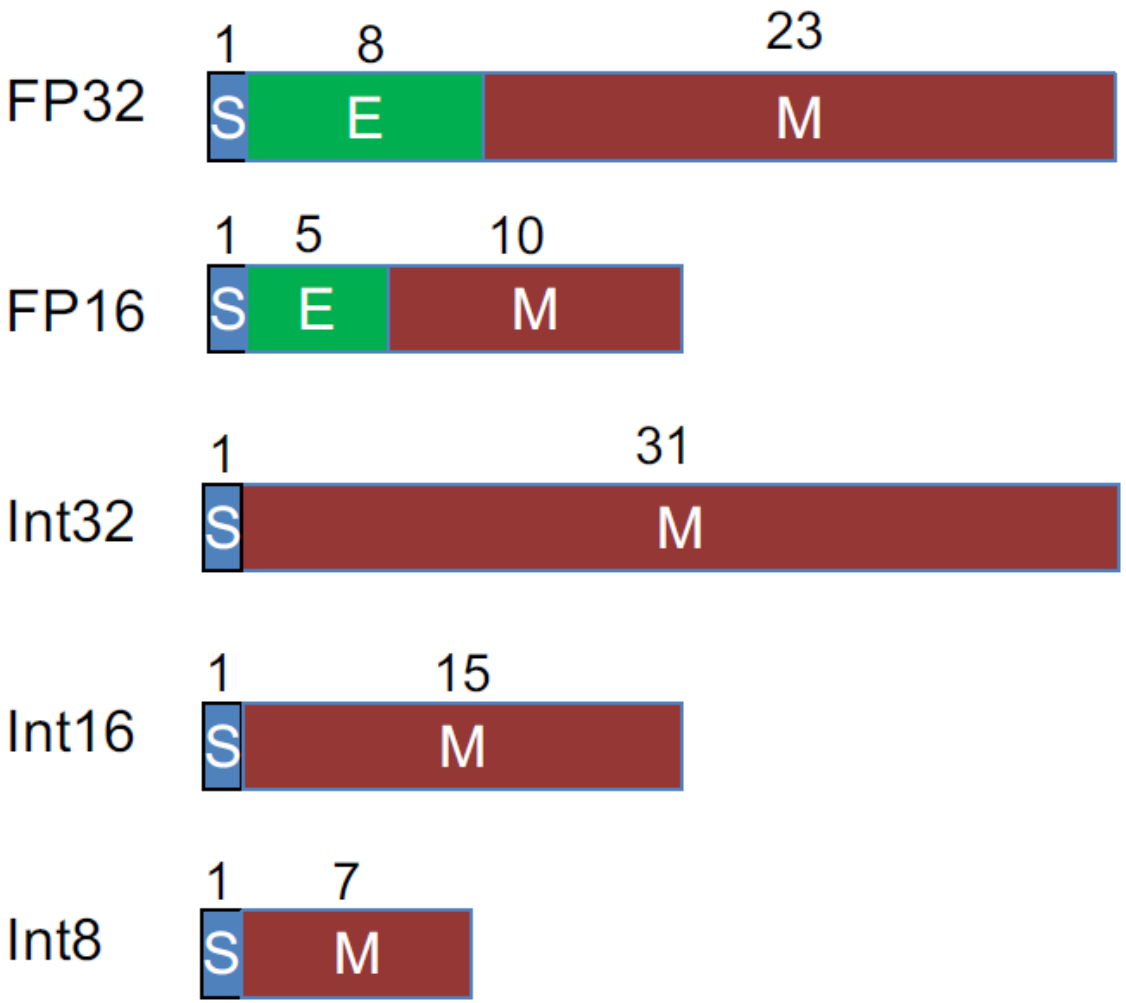
Quantization

Quantization reduces precision of stored data and operators



- Reduce overall memory use
- Compress network, exploiting redundancy
- Supported for several HW platforms with different precision levels
- FP16, INT16, INT8 are most common
- Training may require full precision

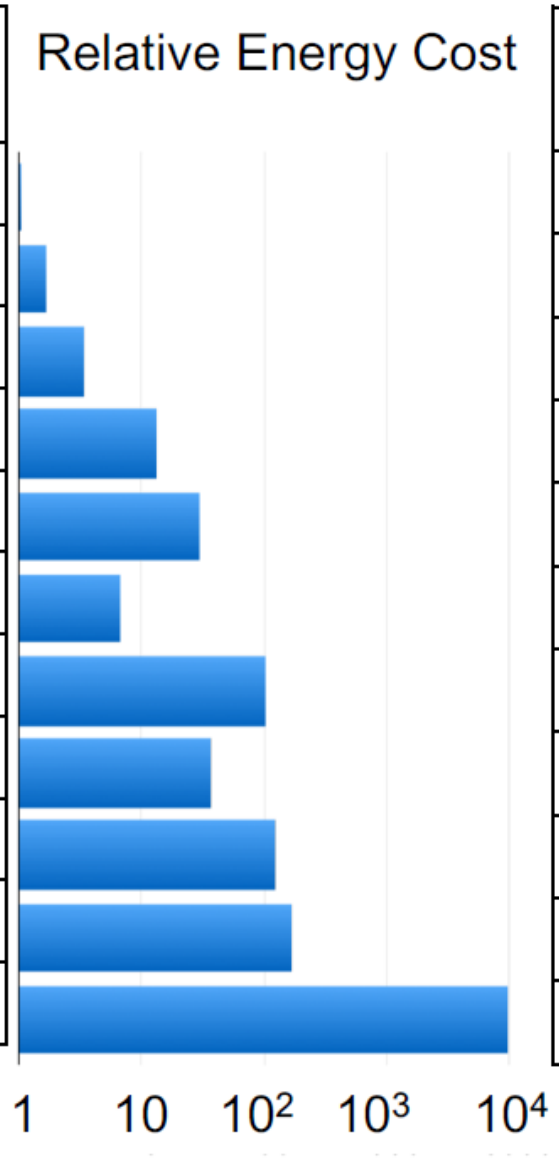
Quantization



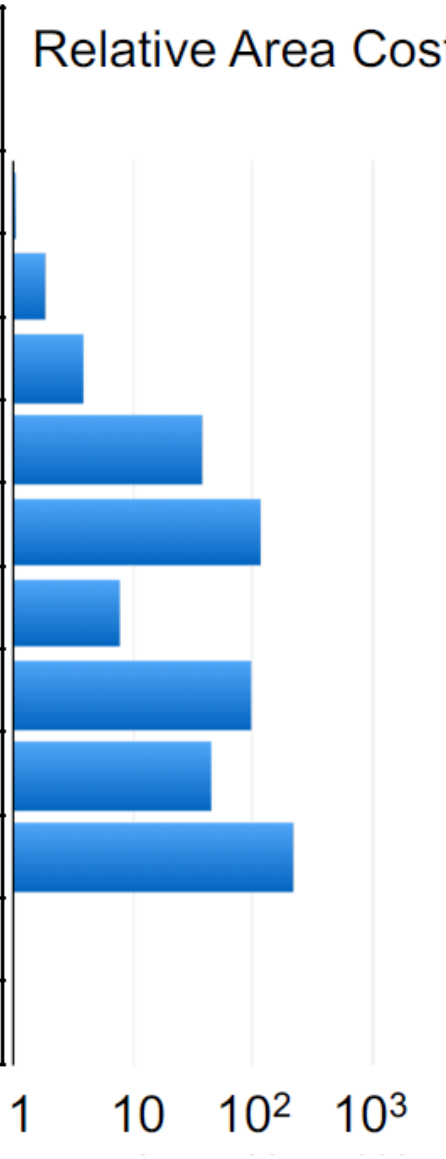
	Range	Accuracy
FP32	$10^{-38} – 10^{38}$	.000006%
FP16	$6x10^{-5}$ - $6x10^{4}$	.05%
Int32	0 – $2x10^{9}$	½
Int16	0 – $6x10^{4}$	½
Int8	0 – 127	½

Quantization

Operation:	Energy (pJ)
8b Add	0.03
16b Add	0.05
32b Add	0.1
16b FP Add	0.4
32b FP Add	0.9
8b Mult	0.2
32b Mult	3.1
16b FP Mult	1.1
32b FP Mult	3.7
32b SRAM Read (8KB)	5
32b DRAM Read	640

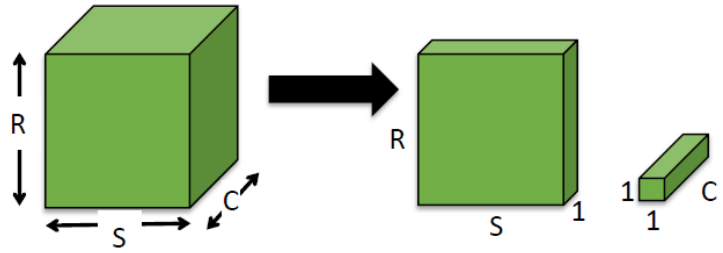


Area (μm^2)
36
67
137
1360
4184
282
3495
1640
7700
N/A
N/A



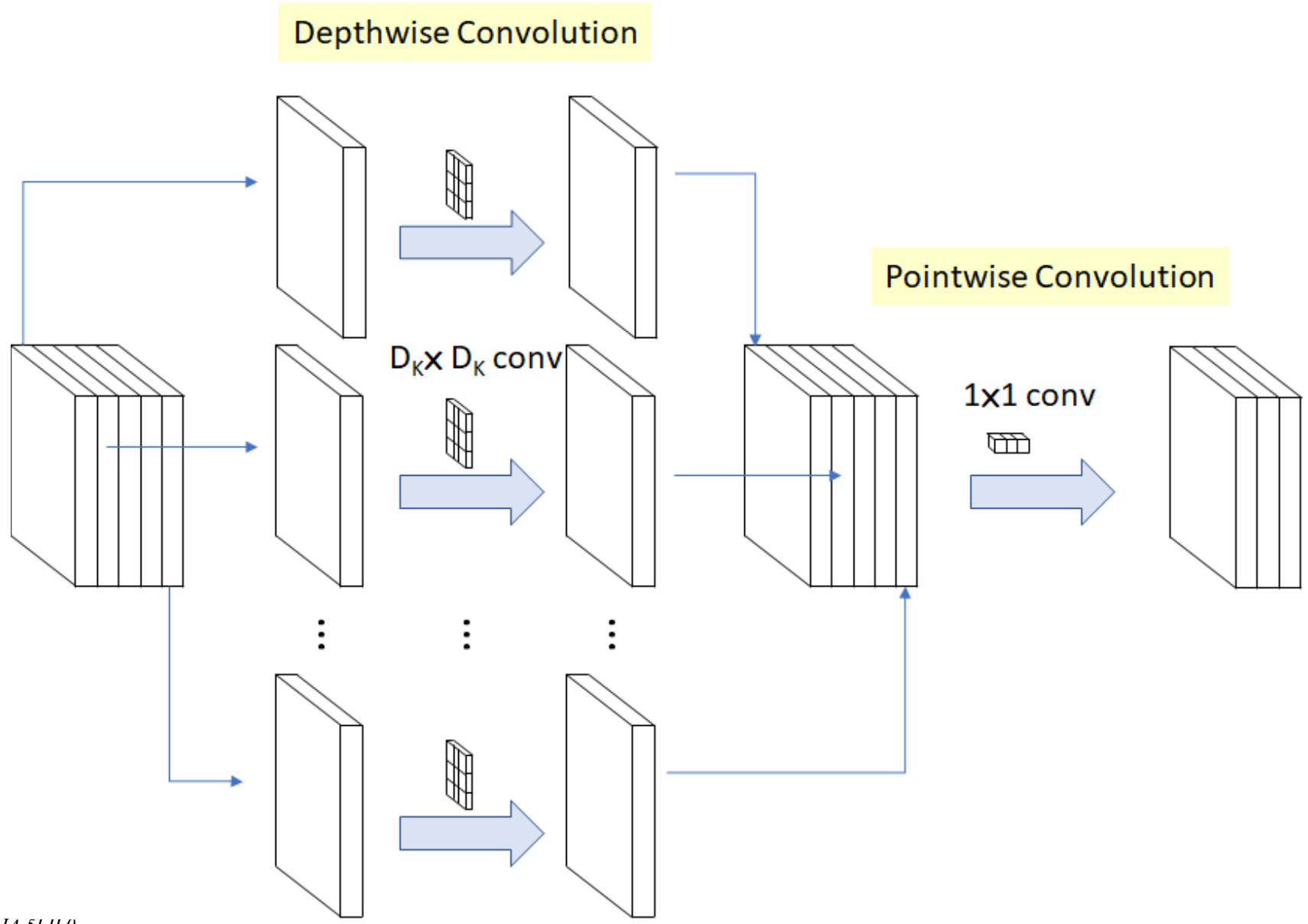
Kernel Decomposition

Decomposing a kernel into smaller kernels



- Assumes layers are over-parameterized
- CP Decomposition and Tucker Decomposition for convolutional layers
- Operates on a weights of a linear layer
- Applied per layer

Mobilenet



Standard Conv Cost:

$$D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F$$

Mobilenet Conv Cost:

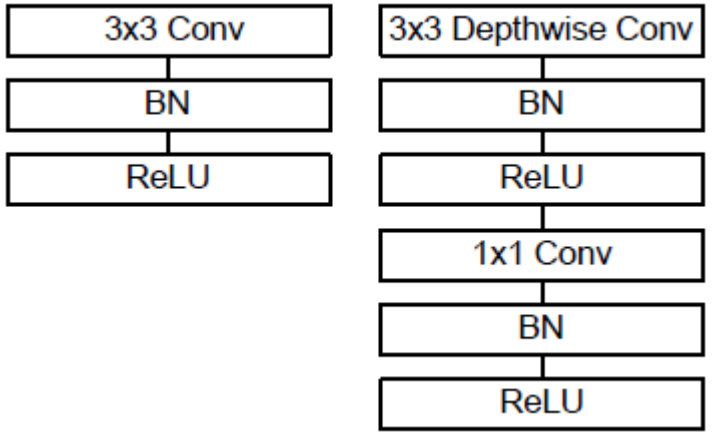
$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F$$

Reduction :

$$\frac{D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F} = \frac{1}{N} + \frac{1}{D_K^2}$$

Mobilenet

Activation needs to be used after each convolution



Standard Convolution vs Depthwise Separable Convolution for ImageNet :

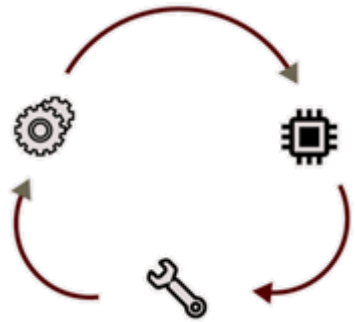
Table 4. Depthwise Separable vs Full Convolution MobileNet

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
Conv MobileNet	71.7%	4866	29.3
MobileNet	70.6%	569	4.2

Optimization Problem

Optimization methods have also “hyperparameters”

- Pruning threshold or percentage
- Quantization level



– How to find optimal values?

Optimization Problem – Automatizing design

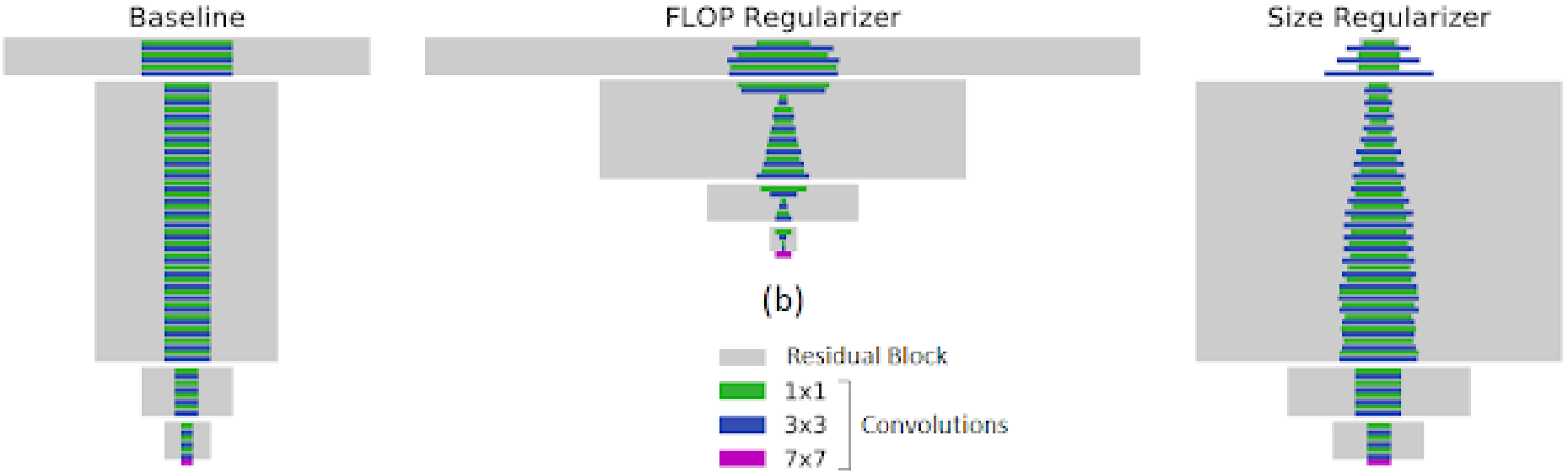
Search for network architecture can be automatized.

- Need heuristics to narrow down search space
- Need translation of functional requirements or limits to design parameters
  - Accuracy
  - Execution time
  - Memory size
  - Energy consumption

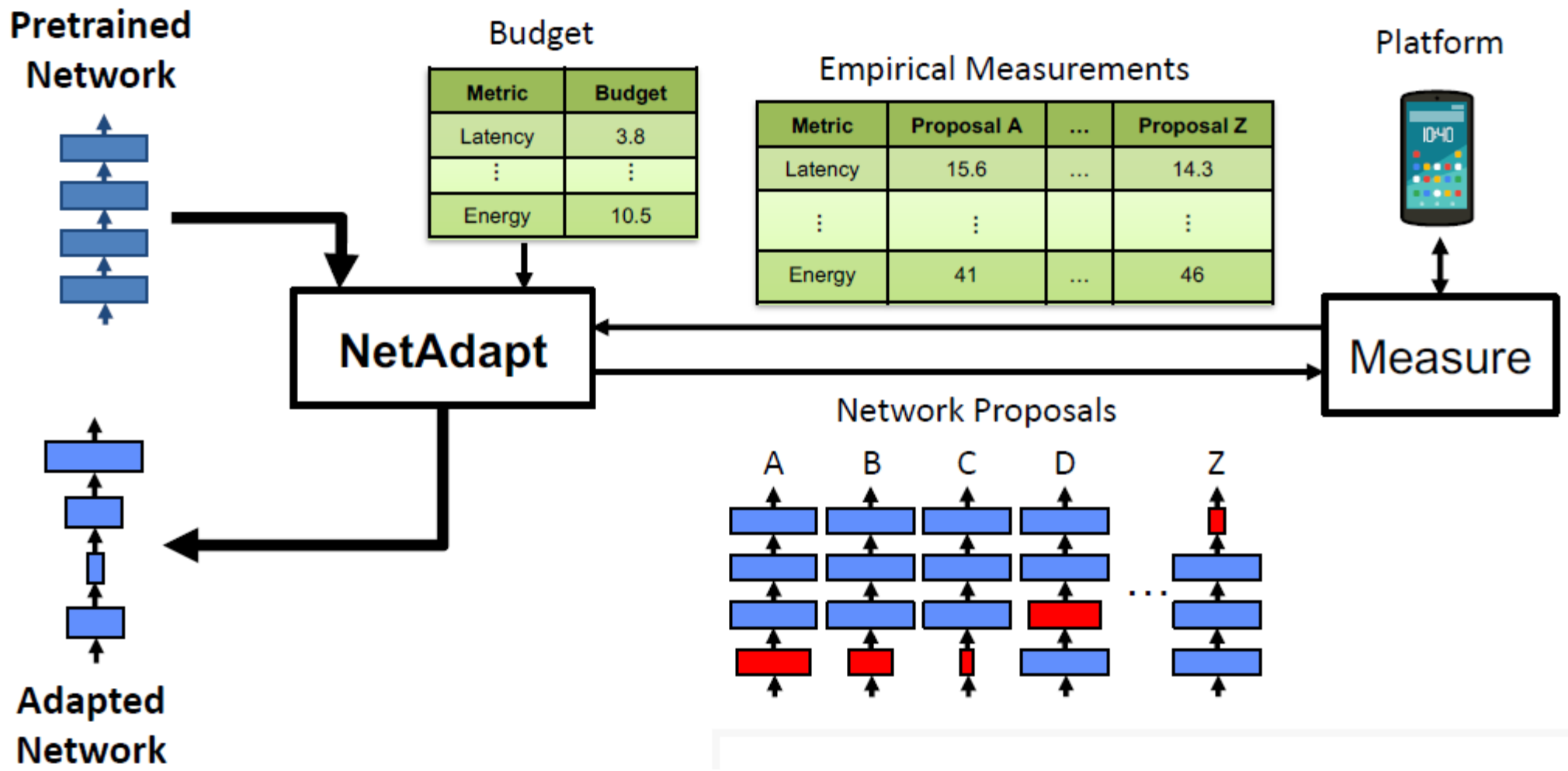
Case study : MorphNet

MorphNet optimizes an existing network architecture in one shot

- Based on training accuracy and required ops, re-scales layer widths
- Entire filters\layers can be removed
- Guided optimization



Case study : NetAdapt



Case study : NetAdapt

Automatically adapts DNN architecture to fit into budget