xperience / ai

Optimizing Neural Nets with Quantization and Pruning

Anna Petrovicheva

anna@xperience.ai

Xperience.ai

- Deep Learning for Computer Vision
  - Digital Surveillance
  - Retail
  - Automating routine work
- Optimized for Mobile and FPGA
  - Quantization and Pruning
- Full pipeline of model development
  - Getting the data right
  - Building the model
  - Porting to target hardware

Pedestrian tracking for Surveillance

- Popular scenario
- Target hardware
  - Low-power
    - FPGA
    - RISC-V
- Custom datasets

Image source: [COCO]

Pedestrian tracking for Surveillance

- How to solve
  - Pedestrian detection
  - Person re-identification
- Privacy concerns

KITTI Leaderboard

	Method	Setting	Code	Moderate	Easy	Hard	Runtime	Environment
1	iDST-VC			**90.55 %**	90.88 %	81.04 %	4 s	GPU @ 2.5 Ghz (Python + C/C++)
2	BM-NET			90.48 %	90.83 %	80.63 %	4.0 s	GPU @ 2.5 Ghz (C/C++)
3	TuSimple		code	90.33 %	90.77 %	82.86 %	1.6 s	GPU @ 2.5 Ghz (Python + C/C++)

F. Yang, W. Choi and Y. Lin: Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. Proceedings of the IEEE Conferer and Pattern Recognition 2016.
K. He, X. Zhang, S. Ren and J. Sun: Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition 2016.

	Method	Setting	Code	Moderate	Easy	Hard	Runtime	Environment
4	THU CV-AI			90.31 %	90.75 %	72.20 %	0.2 s	GPU @ 2.5 Ghz (Python + C/C++)
5	RRC		code	90.22 %	90.61 %	**87.44 %**	3.6 s	GPU @ 2.5 Ghz (Python + C/C++)

J. Ren, X. Chen, J. Liu, W. Sun, J. Pang, Q. Yan, Y. Tai and L. Xu: Accurate Single Stage Detector Using Recurrent Rolling Convolution. CVPR 2017.

	Method	Setting	Code	Moderate	Easy	Hard	Runtime	Environment
6	SJTU-HW			90.08 %	90.81 %	79.98 %	0.85 s	GPU @ 1.5 Ghz (Python + C/C++)
7	SWC			90.05 %	90.82 %	80.59 %	0.5 s	GPU @ >3.5 Ghz (Python + C/C++)
8	Deep MANTA			90.03 %	**97.25 %**	80.62 %	0.7 s	GPU @ 2.5 Ghz (Python + C/C++)

F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulière and T. Chateau: Deep MANTA: A Coarse-to-fine Many-Task Network for joint 2D and 3D vehicle analysis from monocular image. CVPR 2

	Method	Setting	Code	Moderate	Easy	Hard	Runtime	Environment
9	lpm			90.03 %	90.75 %	80.99 %	1 s	4 cores @ 3.5 Ghz (C/C++)
10	sensekitti		code	90.00 %	90.76 %	81.83 %	4.5 s	GPU @ 2.5 Ghz (Python + C/C++)

Image sources: KITTI leaderboard

KITTI Leaderboard

	Method	Setting	Code	Moderate	Easy	Hard	Runtime	Environment
1	iDST-VC			**90.55 %**	90.88 %	81.04 %	4 s	GPU @ 2.5 Ghz (Python + C/C++)
2	BM-NET			90.48 %	90.83 %	80.63 %	4.0 s	GPU @ 2.5 Ghz (C/C++)
3	TuSimple		code	90.33 %	90.77 %	82.86 %	1.6 s	GPU @ 2.5 Ghz (Python + C/C++)

F. Yang, W. Choi and Y. Lin: Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. Proceedings of the IEEE Conferer and Pattern Recognition 2016.
K. He, X. Zhang, S. Ren and J. Sun: Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition 2016.

4	THU CV-AI			90.31 %	90.75 %	72.20 %	0.2 s	GPU @ 2.5 Ghz (Python + C/C++)
5	RRC		code	90.22 %	90.61 %	**87.44 %**	3.6 s	GPU @ 2.5 Ghz (Python + C/C++)

J. Ren, X. Chen, J. Liu, W. Sun, J. Pang, Q. Yan, Y. Tai and L. Xu: Accurate Single Stage Detector Using Recurrent Rolling Convolution. CVPR 2017.

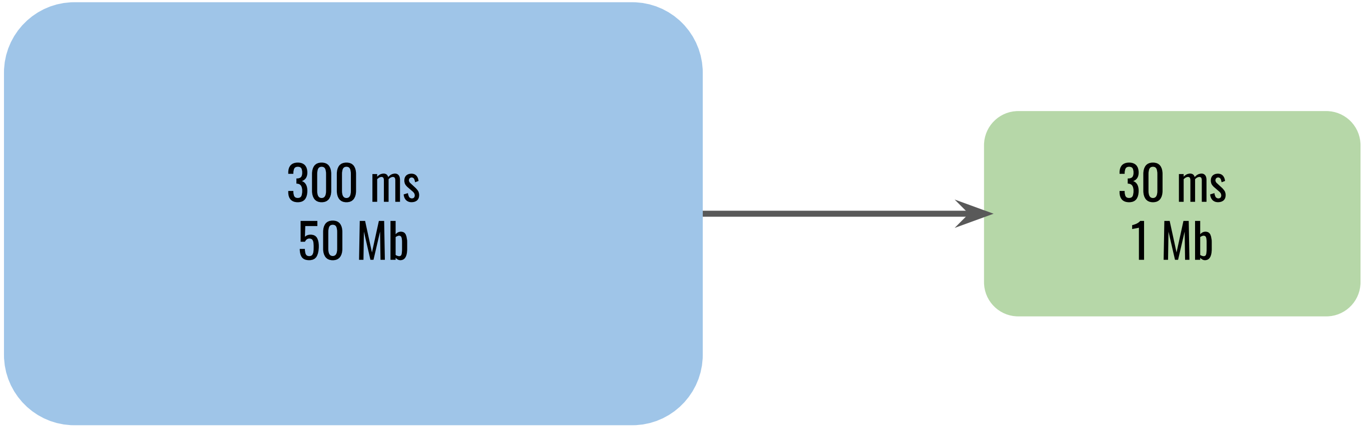
6	SJTU-HW			90.08 %	90.81 %	79.98 %	0.85 s	GPU @ 1.5 Ghz (Python + C/C++)
7	SWC			90.05 %	90.82 %	80.59 %	0.5 s	GPU @ >3.5 Ghz (Python + C/C++)
8	Deep MANTA			90.03 %	**97.25 %**	80.62 %	0.7 s	GPU @ 2.5 Ghz (Python + C/C++)

F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulière and T. Chateau: Deep MANTA: A Coarse-to-fine Many-Task Network for joint 2D and 3D vehicle analysis from monocular image. CVPR 2(

9	lpm			90.03 %	90.75 %	80.99 %	1 s	4 cores @ 3.5 Ghz (C/C++)
10	sensekitti		code	90.00 %	90.76 %	81.83 %	4.5 s	GPU @ 2.5 Ghz (Python + C/C++)

Image sources: KITTI leaderboard

Porting to Mobile / FPGA



Ways to optimize a model

- Small backbone
- Pruning
- Quantization
  - Untrainable
  - Trainable

Small backbone

- State-of-the-art in compression: compress VGG
  - Too big
- Good choices:
  - MobileNet v1 & v2
    - V2: not that small
  - NASNet
  - ESPNet

Pruning

Neuroplasticity

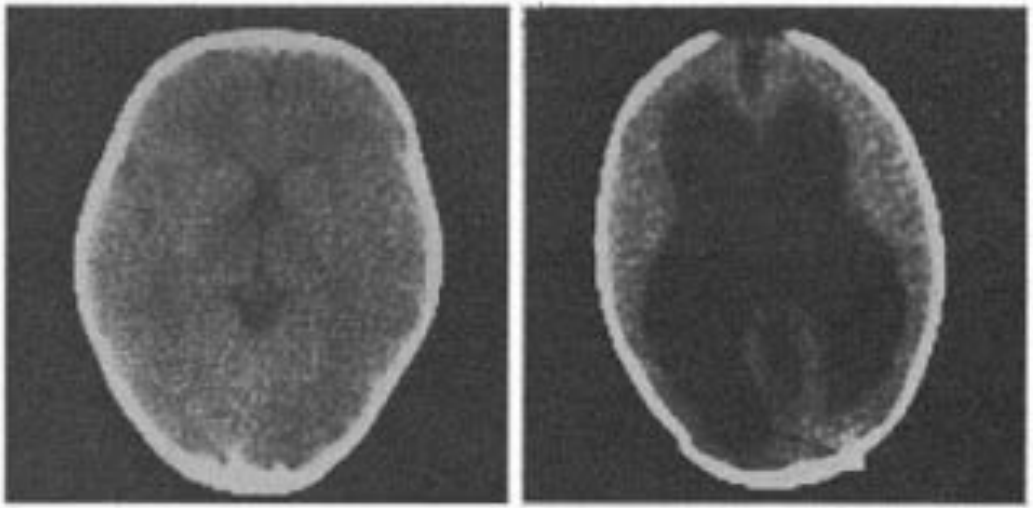
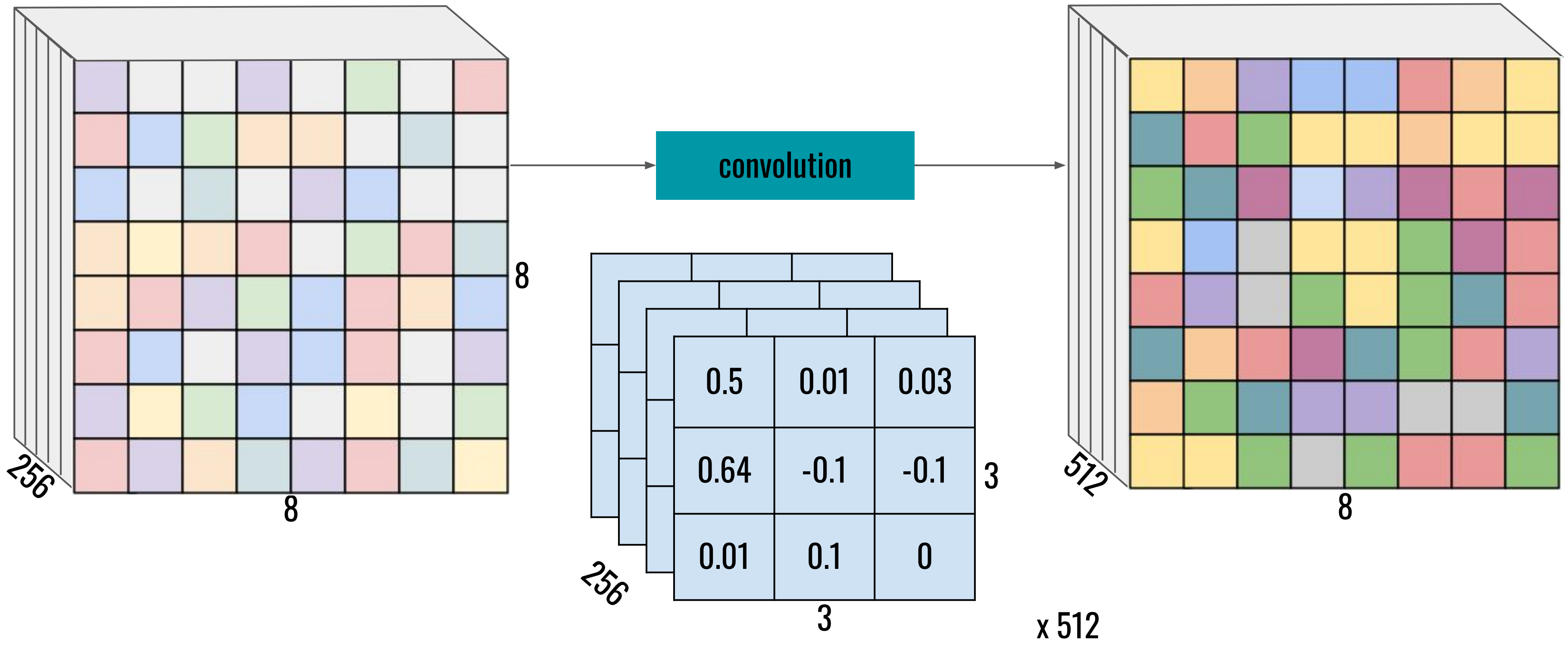
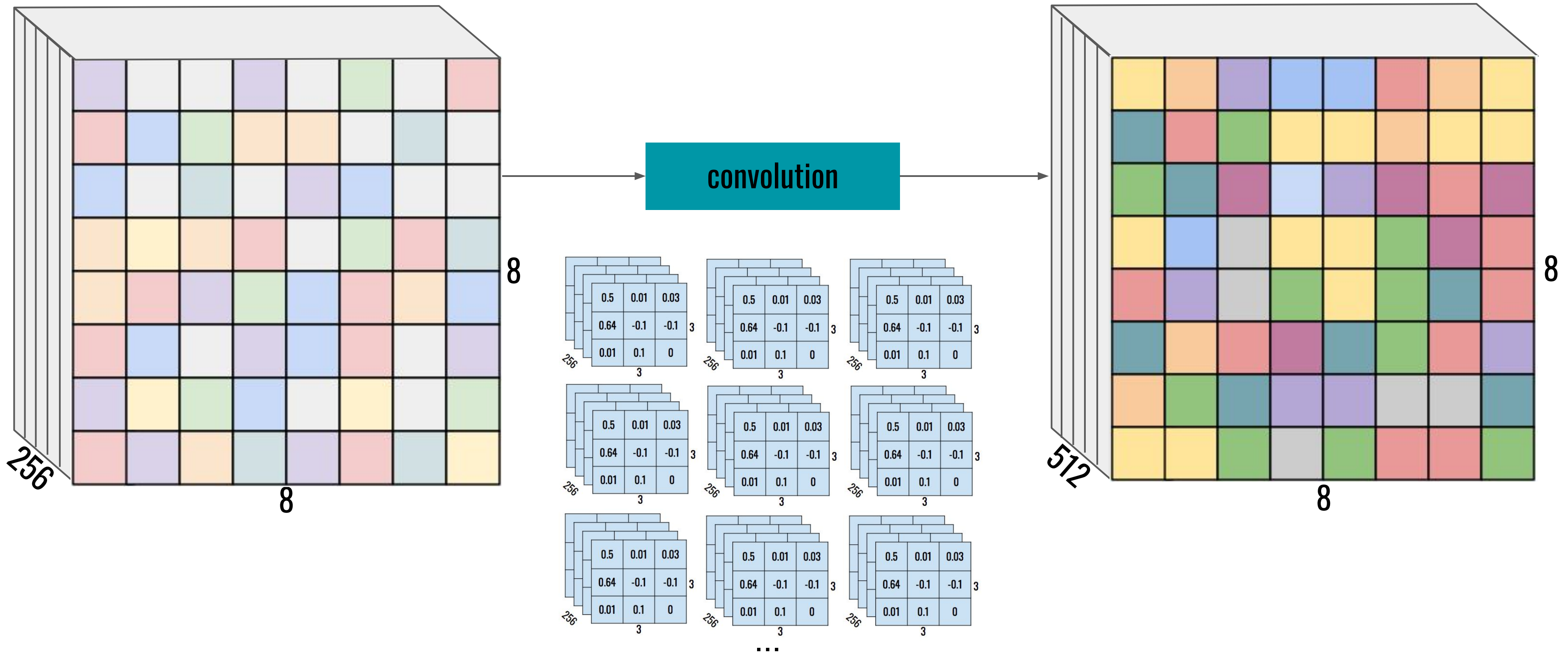


Image source: [Oscillatory Thoughts]

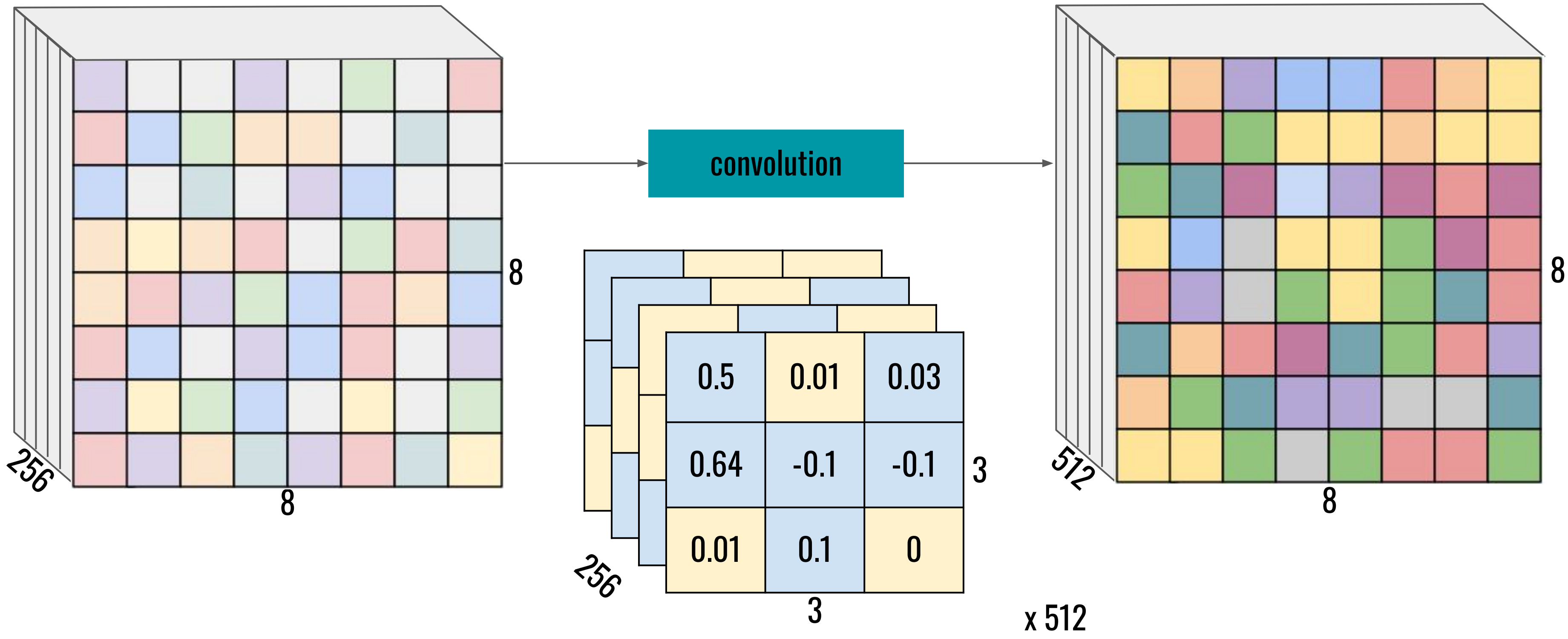
Convolution



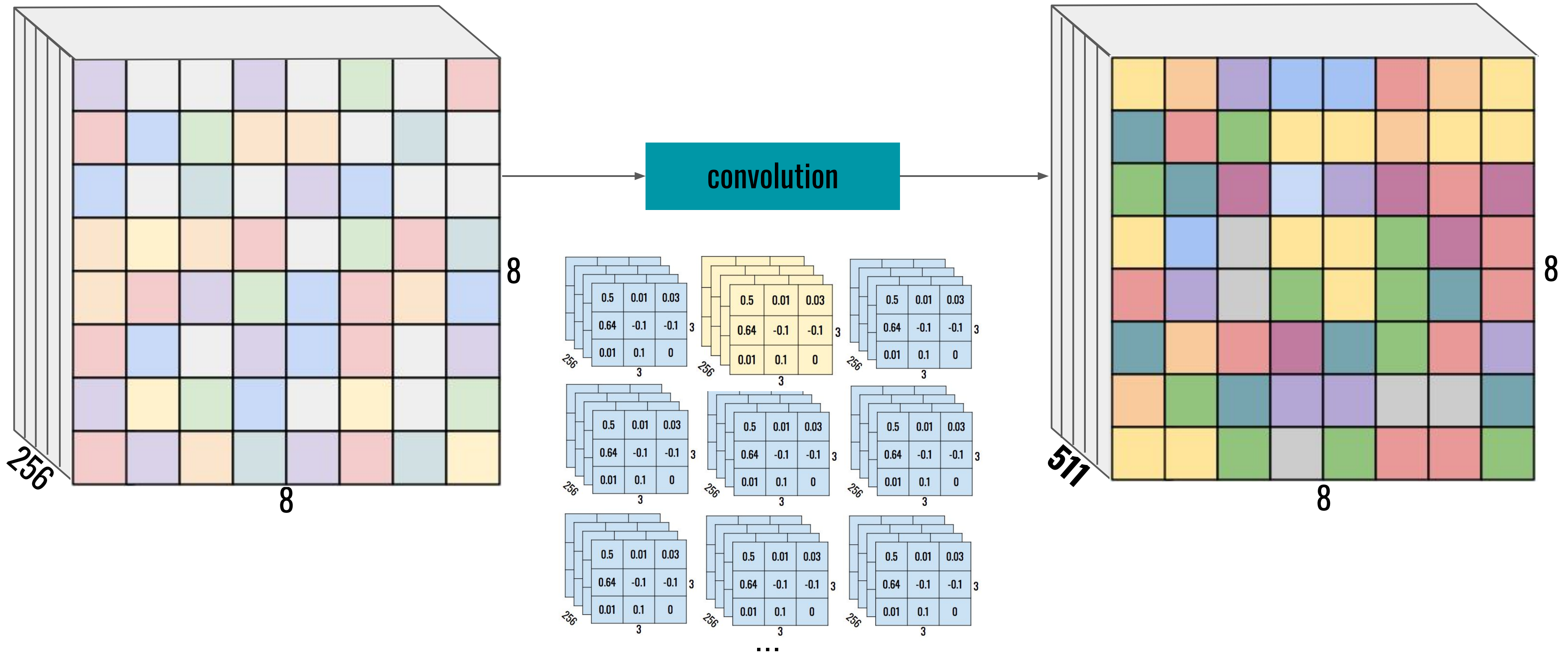
Convolution



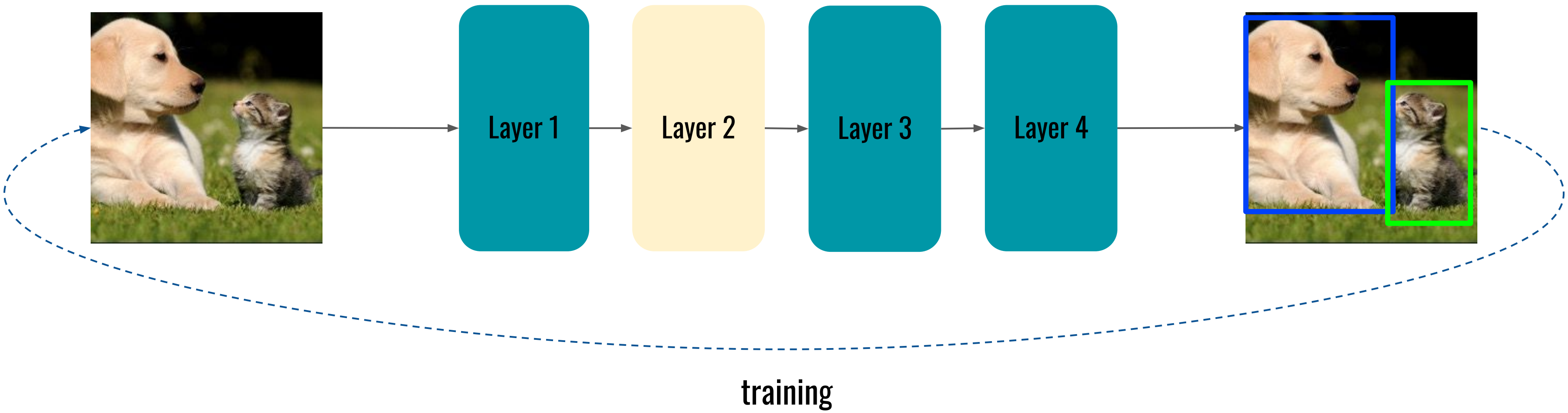
Pruning per element



Pruning per channel



Finetuning

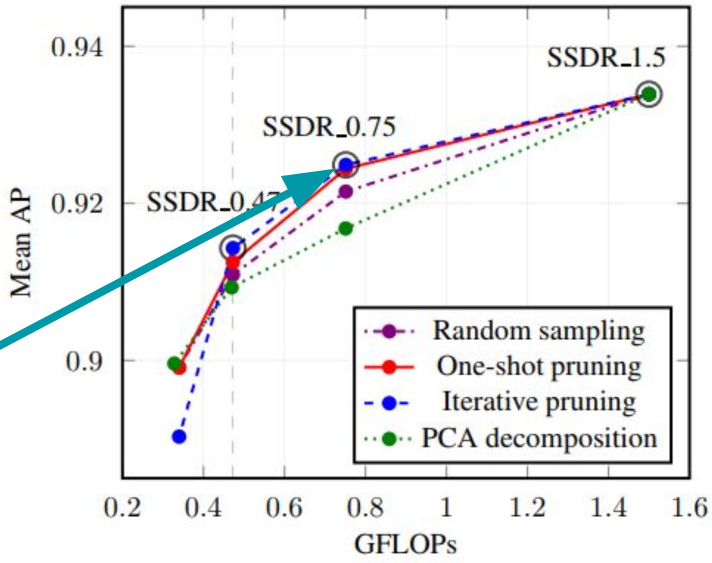


How to select convolutions to prune

- Randomly
    - Not that bad
- By lowest mean absolute weights
- By lowest mean absolute results
- More complicated ways

Pruning results

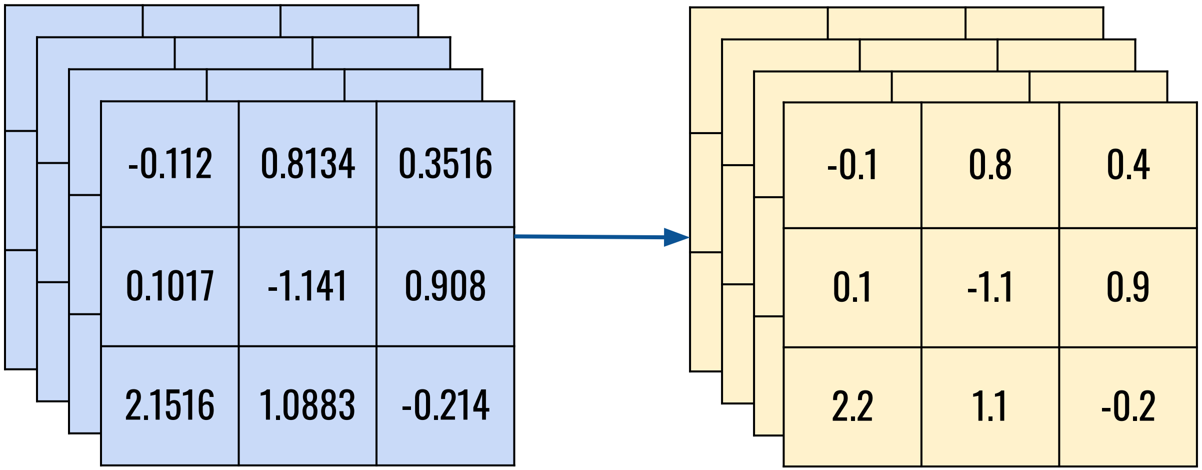
Figure 1: Comparison of channel reduction techniques on DETRAC validation split. Real-time performance on CPU is labeled with vertical line.



-50% computations,
-1% quality

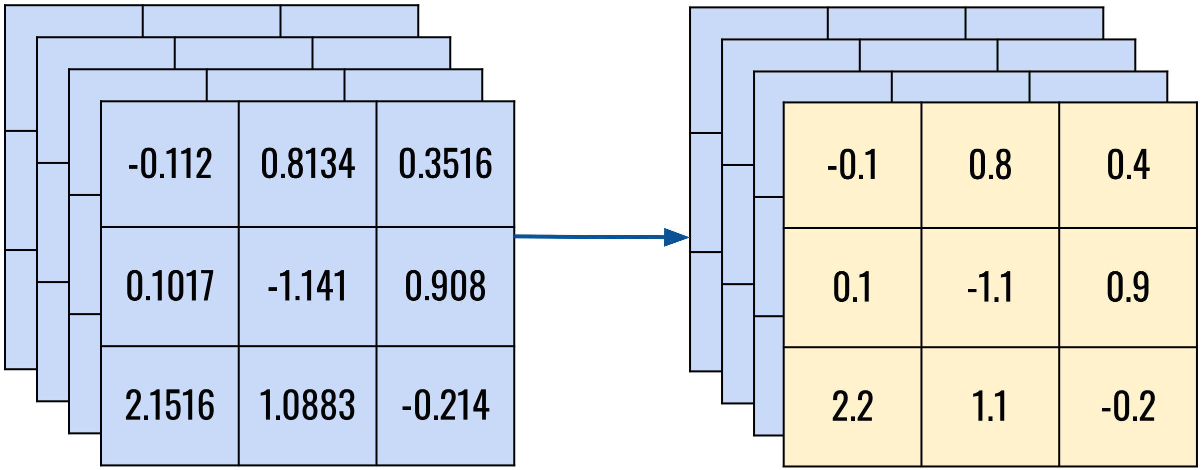
Image source: [Towards lightweight convolutional neural networks for object detection]

Quantization

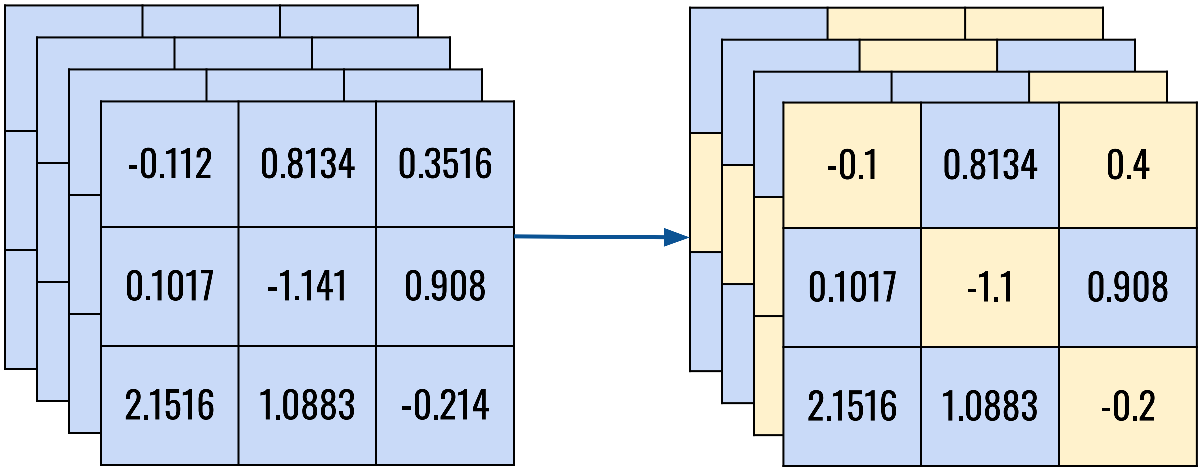


Fixed point computations

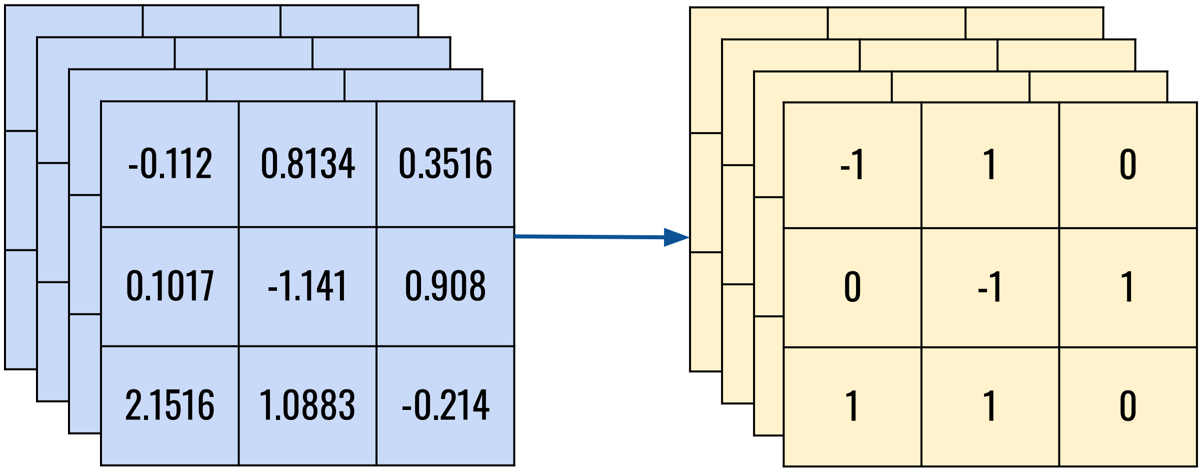
Quantization: trainable



Quantization: trainable



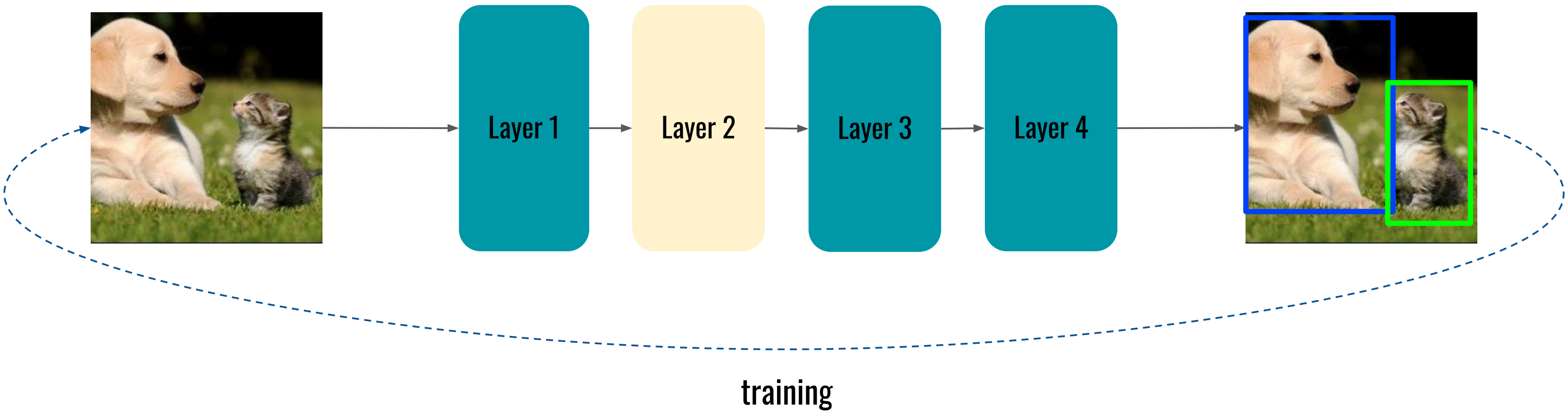
Ternary weight quantization: trainable



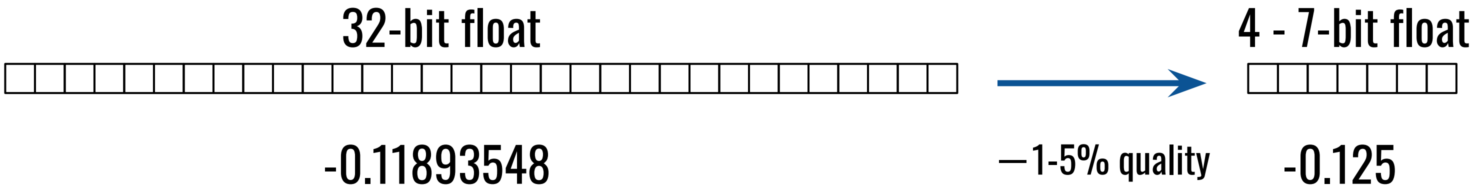
a = 0.835

b = 1.12

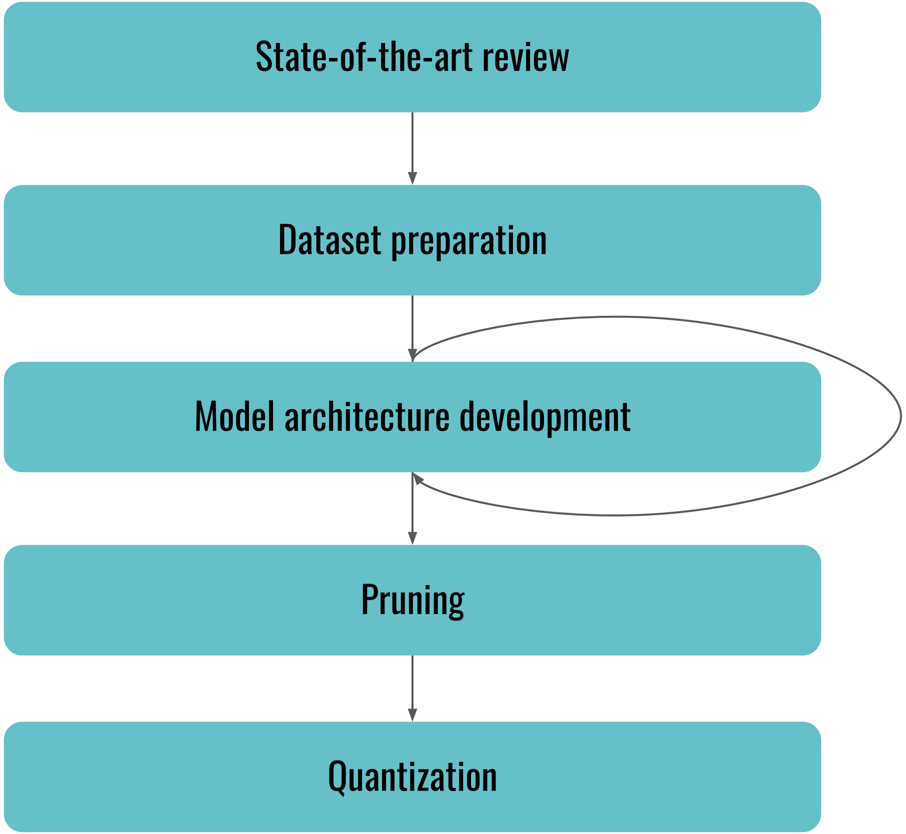
Finetuning



Quantization



Process


State-of-the-art review
Dataset preparation
Model architecture development
Pruning
Quantization

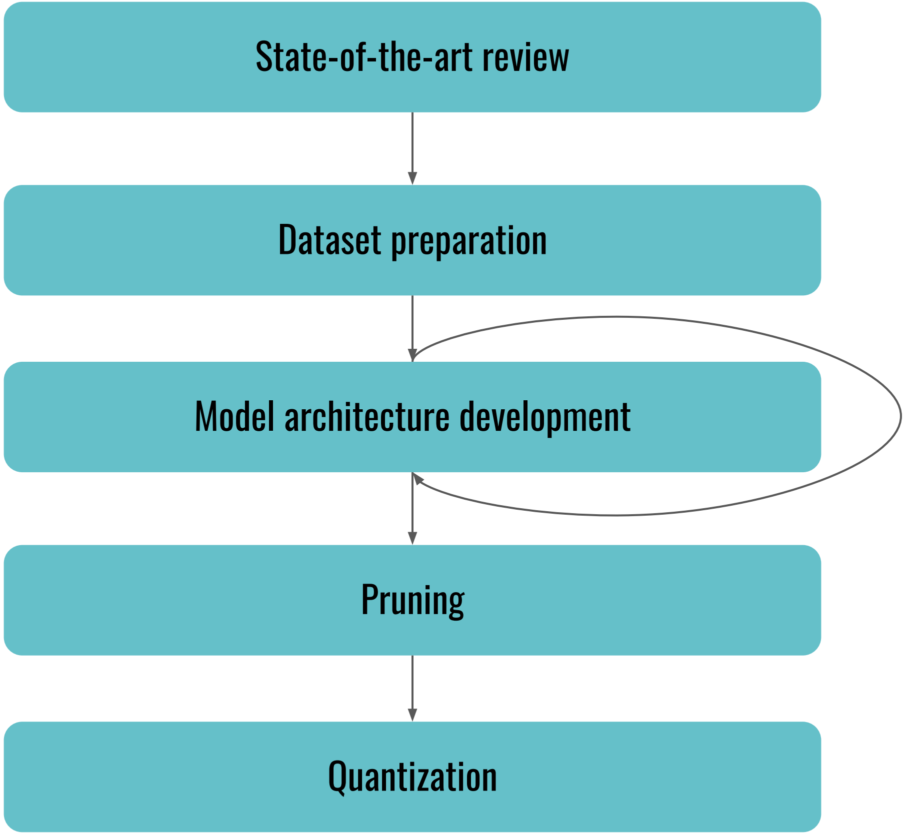
- SSD: Single Shot Multibox Detector. Generic object detector applicable for AABB.
  - \+ Runs fast on mobile devices.
  - \- Does not output rotated boxes.

- Single Shot Text Detector with Regional Attention. Based on SSD detector, generalized for rotated boxes.
  - \+ Can detect very small text.
  - \+ Has special inception-like modules to handle multi-scale text.
  - \- Worse quality compared to other methods from overview.
  - \- Has redundant functionality for this task.

- TextBoxes++: A Single-Shot Oriented Scene Text Detector. Based on SSD detector, generalized for rotated boxes.
  - \+ Good quality on various public datasets (such as ICDAR 2015 and COCO-Text).
  - \+ Can be optimized for mobile devices.
  - \- Has redundant functionality for this task.

- R2CNN: Rotational Region CNN for Orientation Robust Scene Text Detection. Based on Faster-RCNN object detector with generalization for rotated bounding boxes.
  - \+ Good quality on some public datasets.
  - \- Bad performance on mobile devices.

Process


State-of-the-art review
Dataset preparation
Model architecture development
Pruning
Quantization

A. Karpathy on data in real-life projects

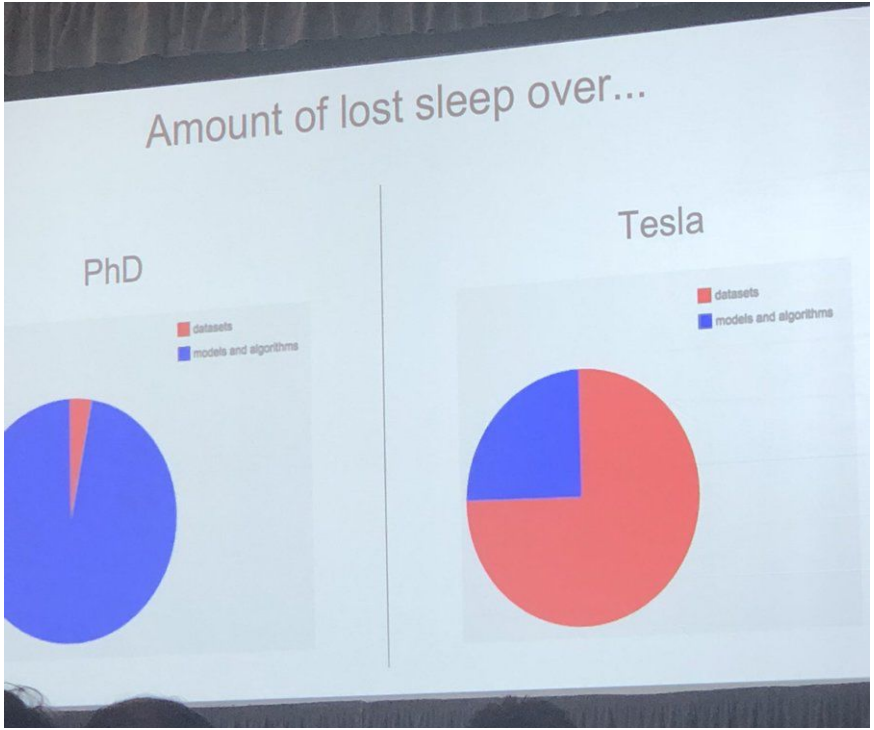


Image source: [Twitter]

Dataset

- Biggest portion of time
- Public + custom data
  - Versatile
  - Big enough
    - Synthetic data generation
- Fixed data splits
  - Versioning
  - Data honesty
    - Splits never intersect
    - Different splits -> different videos

CVPR18 Orals/Spotlights: Multistage Adversarial Losses for Pose-Based Human Image Synthesis

- Human 3.6M dataset
- 11 people
- 17 poses
  - Discussion
  - Smoking
  - Talking on the phone

Image source: [Human 3.6M]

CVPR18 Orals/Spotlights: Multistage Adversarial Losses for Pose-Based Human Image Synthesis

	Person 1	Person 2	Person 3	Person 4
Pose 1				
Pose 2				
Pose 3				
Pose 4				

Image source: [Human 3.6M]

CVPR18 Orals/Spotlights: Multistage Adversarial Losses for Pose-Based Human Image Synthesis

		Person 1	Person 2	Person 3	Person 4
train	Pose 1				
	Pose 2				
val	Pose 3				
	Pose 4				

Image source: [Human 3.6M]

CVPR18 Orals/Spotlights: Multistage Adversarial Losses for Pose-Based Human Image Synthesis

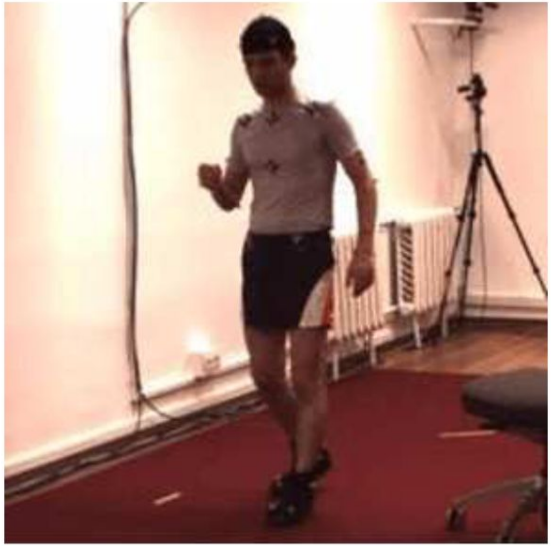


Image source: [Multistage Adversarial Losses for Pose-Based Human Image Synthesis]()

CVPR18 Orals/Spotlights: Multistage Adversarial Losses for Pose-Based Human Image Synthesis

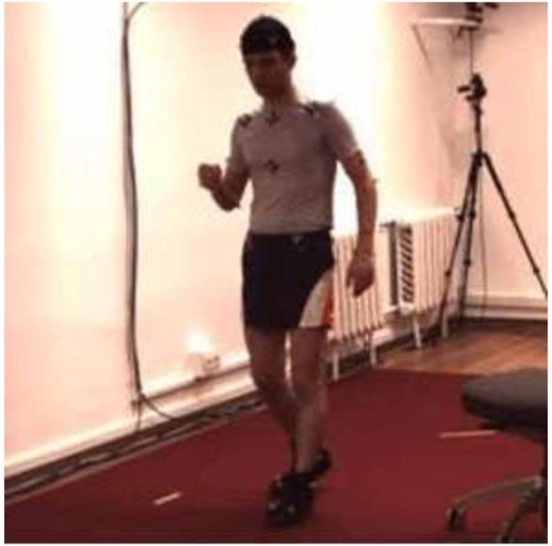


Image source: [Multistage Adversarial Losses for Pose-Based Human Image Synthesis]

CVPR18 Orals/Spotlights: Multistage Adversarial Losses for Pose-Based Human Image Synthesis

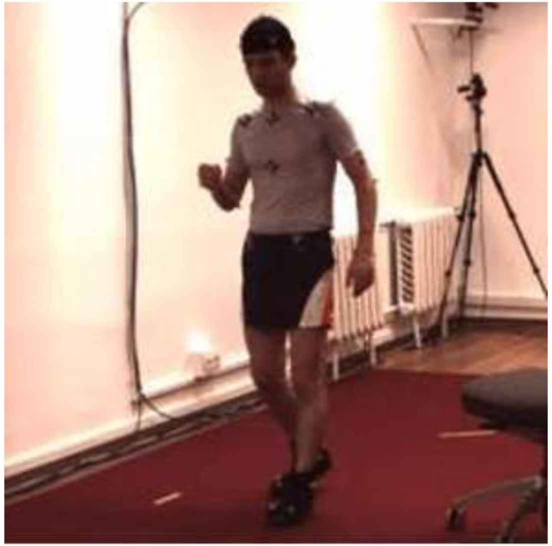
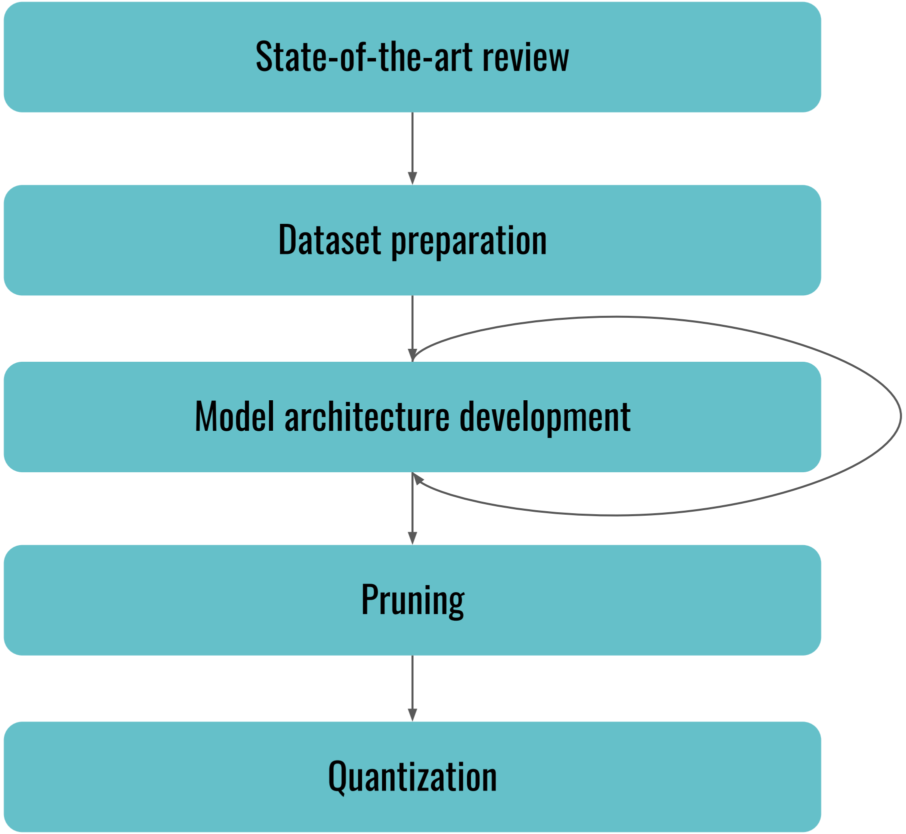
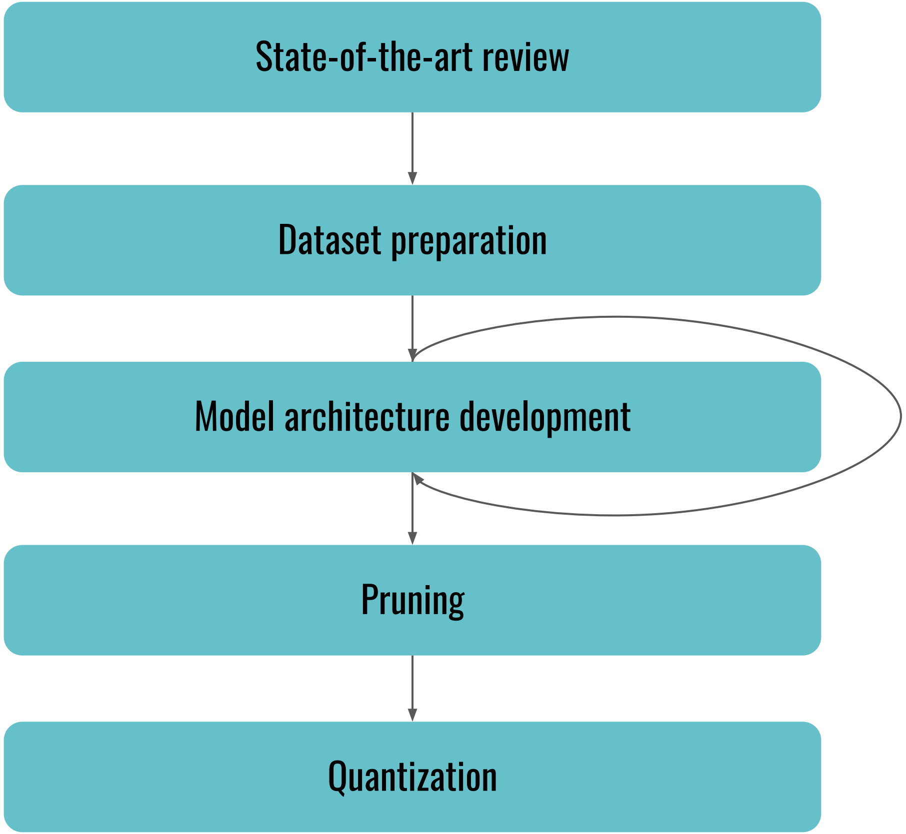


Image source: [Multistage Adversarial Losses for Pose-Based Human Image Synthesis]

Process


State-of-the-art review
Dataset preparation
Model architecture development
Pruning
Quantization

Process


State-of-the-art review
Dataset preparation
Model architecture development
Pruning
Quantization

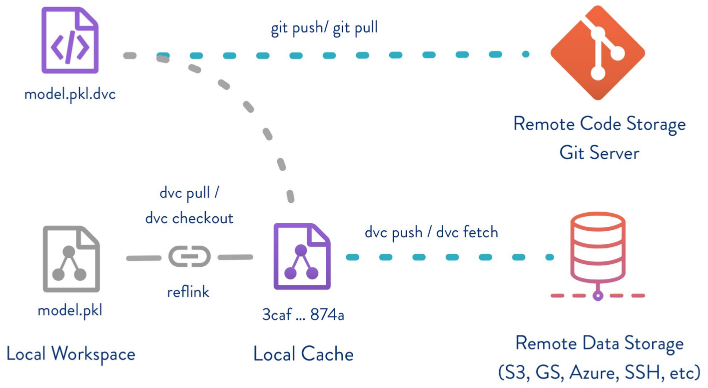
Requirements for good model development

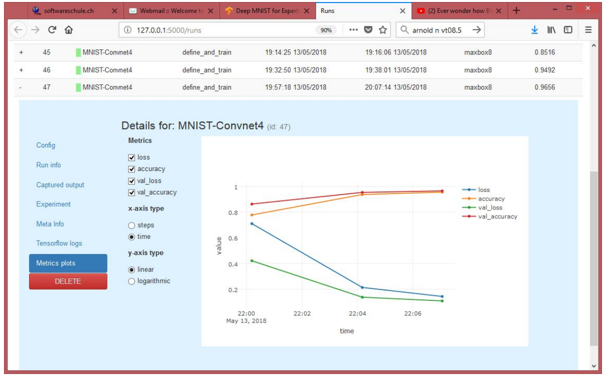
- Metric computation procedure
- Table with experiments
  - Data
  - Architecture
  - Metrics
- Inference speed measurement procedure
  - FLOPs / MACs computation script

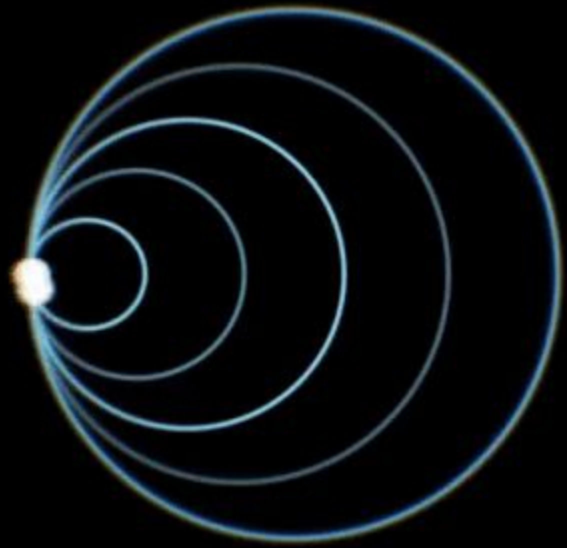
Experiment	Backbone	Training dataset	Validation dataset	Image width	Metric
0001	resnet_50	v1	v1	300	0.790
0003	resnet_10	v1	v1	300	0.72
0005	resnet_10	v1	v1	300	0.785
0006	nasnet	v1	v1	300	0.79
0008	nasnet	v1	v1	250	0.76
0009	mobilenet_v1	v1	v1	300	0.81
0011	mobilenet_v1	v1	v1	300	0.84
0012	mobilenet_v1	v1	v1	250	0.79
0013	mobilenet_v1_reduced	v1	v1	300	0.81
0015	mobilenet_v1_reduced	v1	v1	300	0.82
0016	mobilenet_v2	v1	v1	300	0.84
0018	mobilenet_v2	v1	v1	300	0.86
0019	mobilenet_v2_dilated	v1	v1	300	0.83
0021	mobilenet_v2_dilated	v1	v1	300	0.84
0022	mobilenet_v2_dilated_reduced	v1	v1	300	0.86
0024	mobilenet_v2_dilated_reduced	v1	v1	300	0.88

Comparability and reproducibility

- Dataset + Metric = **Comparability**
- Dataset + Metric + Table = **Reproducibility**







anna@xperience.ai