Huawei and University of Alberta Collaboration Project
Model Compression

Presented by: Alexander W. Wong, Sara Elkerdawy
Under supervision: Nilanjan Ray, Hong Zhang

Agenda

- Model Compression
  - Problem definition
  - State of the art methods and techniques
- Joint End-to-end Pruning With learnable Binary Masks
  - Architecture design
  - Monocular depth estimation use case
  - Classification experiments
- Platform Aware Pruning
- Future work & Conclusion
  - Distillation
  - Quantization
  - All-in-one Framework

Motivation

Deep neural networks (DNN) are one of the state-of-the-art methods for a variety of prediction and supervised learning tasks.

Because DNN models can be large, inference becomes computationally expensive. Embedded and mobile devices that are resource constrained may not be able to effectively use DNNs trained for powerful high-end GPU environment.

Questions of interest

How much can we prune from a large model without hurting the accuracy?

How to automatically explore filters redundancy and prune in an efficient training setup?

How can we compress a model with respect to device aware constraints to respect a resource budget (e.g memory, energy or latency)?

How do we transfer these models among these varying devices?

How to achieve small footprint models?

Start small

Small footprint

- **Manually designed**, expert knowledge
- **Different** for each task
- Training from **scratch** small models results in **drop in accuracy**

PeleeNet [1]
CondenseNet [2]
ShuffleNet [3]
MobileNet [4]

[1] Robert J. Wang, Xiang Li, Shuang Ao, Charles X. Ling, “Pelee: A Real-Time Object Detection System on Mobile Devices,” ICLR 2018 Workshop, accessed at: https://arxiv.org/abs/1804.06882
[2] Huang, Gao, et al. "Condensenet: An efficient densenet using learned group convolutions." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.
[3] Ma, Ningning, et al. "Shufflenet v2: Practical guidelines for efficient cnn architecture design." Proceedings of the European Conference on Computer Vision (ECCV). 2018.
[4] 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.

How to achieve small footprint models?

From large model to small

Distillation Teacher-Student

- **Guided by** pre-trained large models
- **Small student is pre-defined** so don't utilize architecture exploration
- **How** to distill and **which layers**?

- KD (softmax probability)
- FitNet (feature maps mimic)
- FSP (Gramian transfer)
- Attention transfer (heat maps)

Quantization

- **Compress** pre-trained large models
- **Memory** reduction
- **Special support in devices** for full potential utilization.

- Binary-weight networks
- Mixed quantization
- Quantization with RL

Model pruning

- Form of **transfer learning** from large to small models
- Learning based: customized per task
- Requires **layer-by-layer rank selection** (hard to scale with large models as in decoder-encoder)

- Low rank factorization
- Criteria based (i.e filters norm, gradient)
- Sparsity regularization

How to achieve small footprint models?

From large model to small

Distillation Teacher-Student

- KD (softmax probability) [1]
- FitNet (feature maps mimic) [2]
- FSP (Gramian transfer) [3]
- Attention transfer (heat maps) [4]

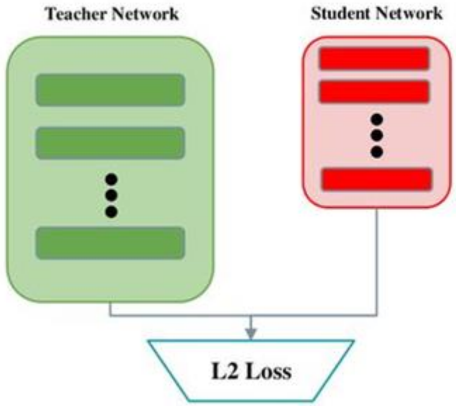
[1] Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network.
[2] Romero, Adriana, et al. "Fitnets: Hints for thin deep nets." *arXiv preprint arXiv:1412.6550* (2014).
[3] Yim, Junho, et al. "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning." The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Vol. 2. 2017.
[4] Zagoruyko, Sergey, and Nikos Komodakis. "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer." arXiv preprint arXiv:1612.03928 (2016).

How to achieve small footprint models?

From large model to small

Distillation Teacher-Student

- KD (softmax probability) [1]
- FitNet (feature maps mimic) [2]
- FSP (Gramian transfer) [3]
- Attention transfer (heat maps) [4]



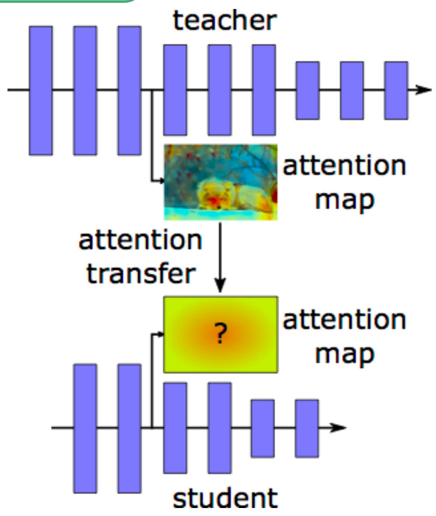
[1] Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network.
[2] Romero, Adriana, et al. "Fitnets: Hints for thin deep nets." *arXiv preprint arXiv:1412.6550* (2014).
[3] Yim, Junho, et al. "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning." The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Vol. 2. 2017.
[4] Zagoruyko, Sergey, and Nikos Komodakis. "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer." arXiv preprint arXiv:1612.03928 (2016).

How to achieve small footprint models?

From large model to small

Distillation Teacher-Student

- KD (softmax probability) [1]
- FitNet (feature maps mimic) [2]
- FSP (Gramian transfer) [3]
- Attention transfer (heat maps) [4]



[1] Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network.
[2] Romero, Adriana, et al. "Fitnets: Hints for thin deep nets." *arXiv preprint arXiv:1412.6550* (2014).
[3] Yim, Junho, et al. "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning." The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Vol. 2. 2017.
[4] Zagoruyko, Sergey, and Nikos Komodakis. "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer." arXiv preprint arXiv:1612.03928 (2016).

How to achieve small footprint models?

From large model to small

Distillation Teacher-Student

- **Guided by pre-trained** large models
- **Small** student **is pre-defined** so don't utilize architecture exploration
- **How** to distill and **which layers**?

- KD (softmax probability) [1]
- FitNet (feature maps mimic) [2]
- FSP (Gramian transfer) [3]
- Attention transfer (heat maps) [4]

[1] Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network.
[2] Romero, Adriana, et al. "Fitnets: Hints for thin deep nets." *arXiv preprint arXiv:1412.6550* (2014).
[3] Yim, Junho, et al. "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning." The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Vol. 2. 2017.
[4] Zagoruyko, Sergey, and Nikos Komodakis. "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer." arXiv preprint arXiv:1612.03928 (2016).

How to achieve small footprint models?

From large model to small

Distillation Teacher-Student

- **Guided by pre-trained** large models
- **Small student is pre-defined** so don't utilize architecture exploration
- **How** to distill and **which layers**?

Quantization

- KD (softmax probability)
- FitNet (feature maps mimic)
- FSP (Gramian transfer)
- Attention transfer (heat maps)

- Binary-weight networks [1]
- Mixed quantization [2]
- Quantization with RL [3]

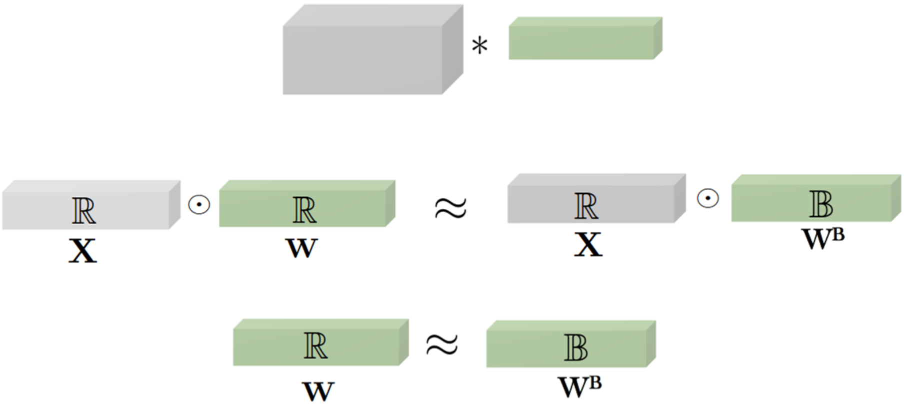
[1] Courbariaux, Matthieu, et al. "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1." arXiv preprint arXiv:1602.02830 (2016).
[2] Wu, Bichen, et al. "Mixed precision quantization of convnets via differentiable neural architecture search." arXiv preprint arXiv:1812.00090 (2018).
[3] Yazdanbakhsh, Amir, et al. "Releq: An automatic reinforcement learning approach for deep quantization of neural networks." arXiv preprint arXiv:1811.01704 (2018).

How to achieve small footprint models?

From large model to small

Quantization

- Binary-weight networks [1]
- Mixed quantization [2]
- Quantization with RL [3]



$$\mathbf{W}^{\mathbb{B}} = \mathrm{sign}(\mathbf{W})$$

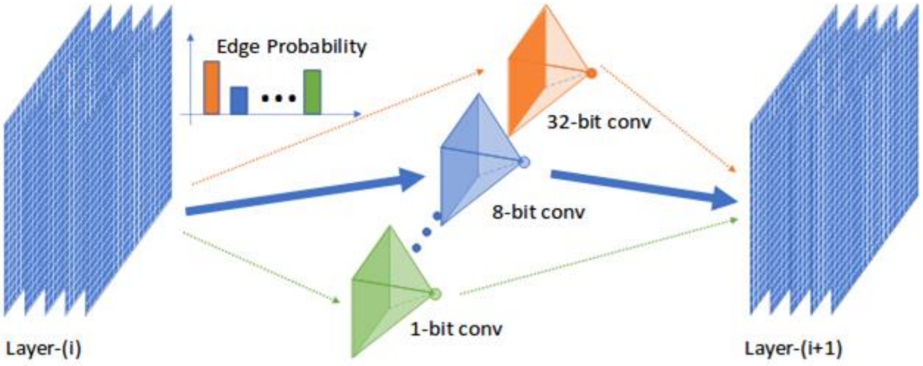
[1] Courbariaux, Matthieu, et al. "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1." arXiv preprint arXiv:1602.02830 (2016).
[2] Wu, Bichen, et al. "Mixed precision quantization of convnets via differentiable neural architecture search." arXiv preprint arXiv:1812.00090 (2018).
[3] Yazdanbakhsh, Amir, et al. "Releq: An automatic reinforcement learning approach for deep quantization of neural networks." arXiv preprint arXiv:1811.01704 (2018).

How to achieve small footprint models?

From large model to small

Quantization

- Binary-weight networks [1]
- Mixed quantization [2]
- Quantization with RL [3]



[1] Courbariaux, Matthieu, et al. "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1." arXiv preprint arXiv:1602.02830 (2016).
[2] Wu, Bichen, et al. "Mixed precision quantization of convnets via differentiable neural architecture search." arXiv preprint arXiv:1812.00090 (2018).
[3] Yazdanbakhsh, Amir, et al. "Releq: An automatic reinforcement learning approach for deep quantization of neural networks." arXiv preprint arXiv:1811.01704 (2018).

How to achieve small footprint models?

From large model to small

Distillation Teacher-Student

- **Guided by pre-trained** large models
- **Small student is pre-defined** so don't utilize architecture exploration
- **How** to distill and **which layers**?

Quantization

- **Compress** pre-trained large models
- Careful **quantization level per layer**
- **Special support in devices** for full potential utilization.

- KD (softmax probability)
- FitNet (feature maps mimic)
- FSP (Gramian transfer)
- Attention transfer (heat maps)

- Binary-weight networks [1]
- Mixed quantization [2]
- Quantization with RL [3]

[1] Courbariaux, Matthieu, et al. "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1." arXiv preprint arXiv:1602.02830 (2016).
[2] Wu, Bichen, et al. "Mixed precision quantization of convnets via differentiable neural architecture search." arXiv preprint arXiv:1812.00090 (2018).
[3] Yazdanbakhsh, Amir, et al. "Releq: An automatic reinforcement learning approach for deep quantization of neural networks." arXiv preprint arXiv:1811.01704 (2018).

How to achieve small footprint models?

From large model to small

Distillation Teacher-Student
- **Guided by pre-trained** large models
- **Small student is pre-defined** so don’t utilize architecture exploration
- **How** to distill and **which layers**?

Quantization
- **Compress** pre-trained large models
- Careful **quantization level per layer**
- **Special support in devices** for full potential utilization.

Model pruning

- KD (softmax probability)
- FitNet (feature maps mimic)
- FSP (Gramian transfer)
- Attention transfer (heat maps)

- Binary-weight networks
- Mixed quantization
- Quantization with RL

- Low rank factorization [1,2]
- Criteria based (i.e filters norm, gradient) [3,4]
- Sparsity regularization [5]

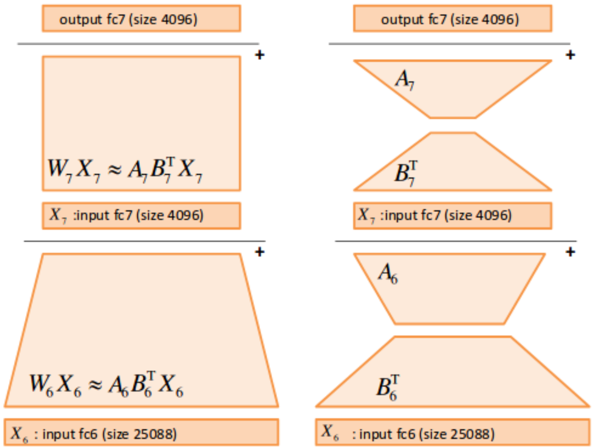
[1] Masana, Marc, Joost van de Weijer, Luis Herranz, Andrew D. Bagdanov, and Jose MAlvarez. "Domain-adaptive deep network compression." ICCV 2017.
[2] Kim, Yong-Deok, et al. "Compression of deep convolutional neural networks for fast and low power mobile applications." arXiv preprint arXiv:1511.06530 (2015).
[3] Li, Hao, et al. "Pruning filters for efficient convnets." arXiv preprint arXiv:1608.08710 (2016).
[4] Molchanov, Pavlo, et al. "Pruning convolutional neural networks for resource efficient transfer learning." ICLR (2017)
[5] Liu, Zhuang, et al. "Learning efficient convolutional networks through network slimming." Proceedings of the IEEE International Conference on Computer Vision. 2017.

How to achieve small footprint models?

From large model to small

Model pruning

- Low rank factorization [1,2]
- Criteria based (i.e filters norm, gradient) [3,4]
- Sparsity regularization [5]



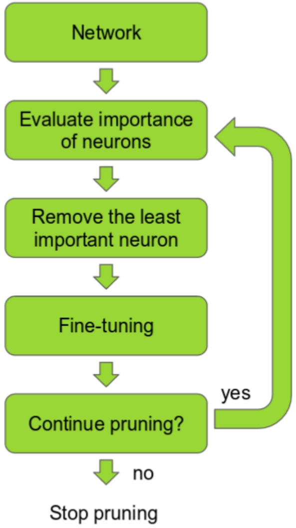
[1] Masana, Marc, Joost van de Weijer, Luis Herranz, Andrew D. Bagdanov, and Jose MAlvarez. "Domain-adaptive deep network compression." ICCV 2017.
[2] Kim, Yong-Deok, et al. "Compression of deep convolutional neural networks for fast and low power mobile applications." arXiv preprint arXiv:1511.06530 (2015).
[3] Li, Hao, et al. "Pruning filters for efficient convnets." arXiv preprint arXiv:1608.08710 (2016).
[4] Molchanov, Pavlo, et al. "Pruning convolutional neural networks for resource efficient transfer learning." ICLR (2017)
[5] Liu, Zhuang, et al. "Learning efficient convolutional networks through network slimming." Proceedings of the IEEE International Conference on Computer Vision. 2017.

How to achieve small footprint models?

From large model to small

Model pruning

- Low rank factorization [1,2]
- Criteria based (i.e filters norm, gradient) [3,4]
- Sparsity regularization [5]



[1] Masana, Marc, Joost van de Weijer, Luis Herranz, Andrew D. Bagdanov, and Jose MAlvarez. "Domain-adaptive deep network compression." ICCV 2017.
[2] Kim, Yong-Deok, et al. "Compression of deep convolutional neural networks for fast and low power mobile applications." arXiv preprint arXiv:1511.06530 (2015).
[3] Li, Hao, et al. "Pruning filters for efficient convnets." arXiv preprint arXiv:1608.08710 (2016).
[4] Molchanov, Pavlo, et al. "Pruning convolutional neural networks for resource efficient transfer learning." ICLR (2017)
[5] Liu, Zhuang, et al. "Learning efficient convolutional networks through network slimming." Proceedings of the IEEE International Conference on Computer Vision. 2017.

How to achieve small footprint models?

From large model to small

Distillation Teacher-Student

- **Guided by pre-trained** large models
- **Small student is pre-defined** so don't utilize architecture exploration
- **How** to distill and **which layers**?

- KD (softmax probability)
- FitNet (feature maps mimic)
- FSP (Gramian transfer)
- Attention transfer (heat maps)

Quantization

- **Compress** pre-trained large models
- Careful **quantization level per layer**
- **Special support in devices** for full potential utilization.

- Binary-weight networks
- Mixed quantization
- Quantization with RL

Model pruning

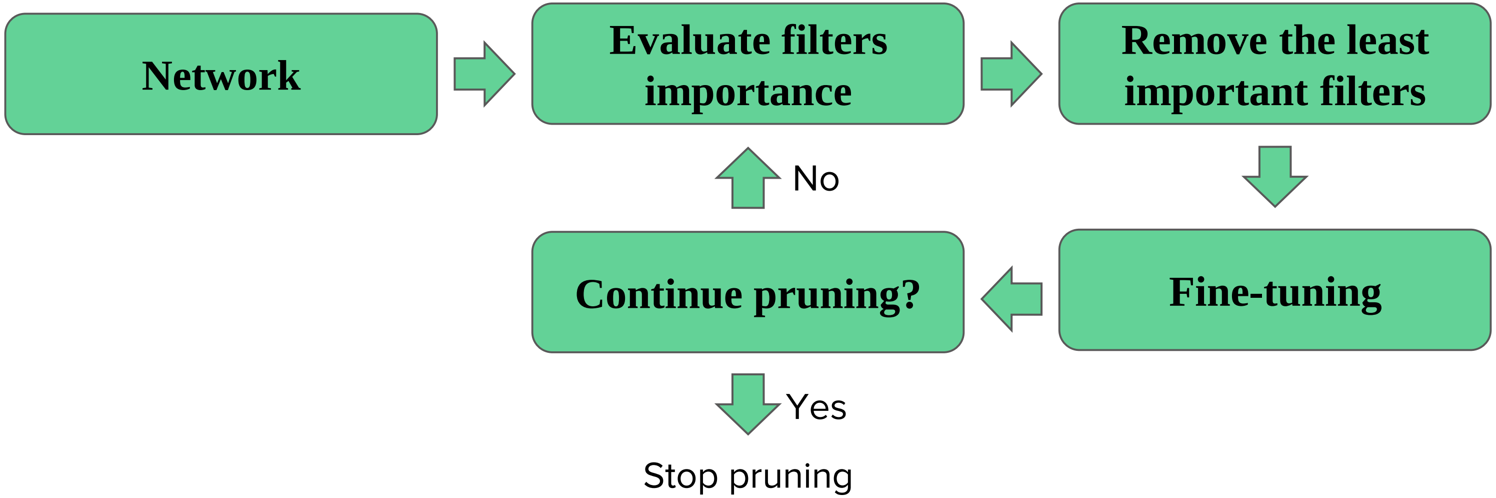
- **Bounded** architecture search
- Learning based: customized per task
- Some requires **layer-by-layer rank selection**
- **Thinning** only

- Low rank factorization [1,2]
- Criteria based (i.e filters norm, gradient) [3,4]
- Sparsity regularization [5]

[1] Masana, Marc, Joost van de Weijer, Luis Herranz, Andrew D. Bagdanov, and Jose MAlvarez. "Domain-adaptive deep network compression." ICCV 2017.
[2] Kim, Yong-Deok, et al. "Compression of deep convolutional neural networks for fast and low power mobile applications." arXiv preprint arXiv:1511.06530 (2015).
[3] Li, Hao, et al. "Pruning filters for efficient convnets." arXiv preprint arXiv:1608.08710 (2016).
[4] Molchanov, Pavlo, et al. "Pruning convolutional neural networks for resource efficient transfer learning." ICLR (2017)
[5] Liu, Zhuang, et al. "Learning efficient convolutional networks through network slimming." Proceedings of the IEEE International Conference on Computer Vision. 2017.

Model Pruning

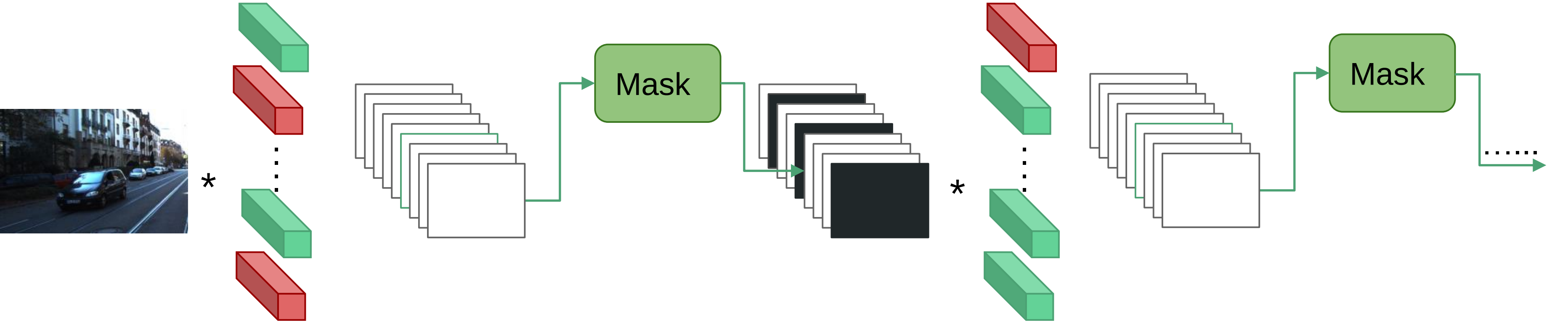
Model Pruning



Model Pruning

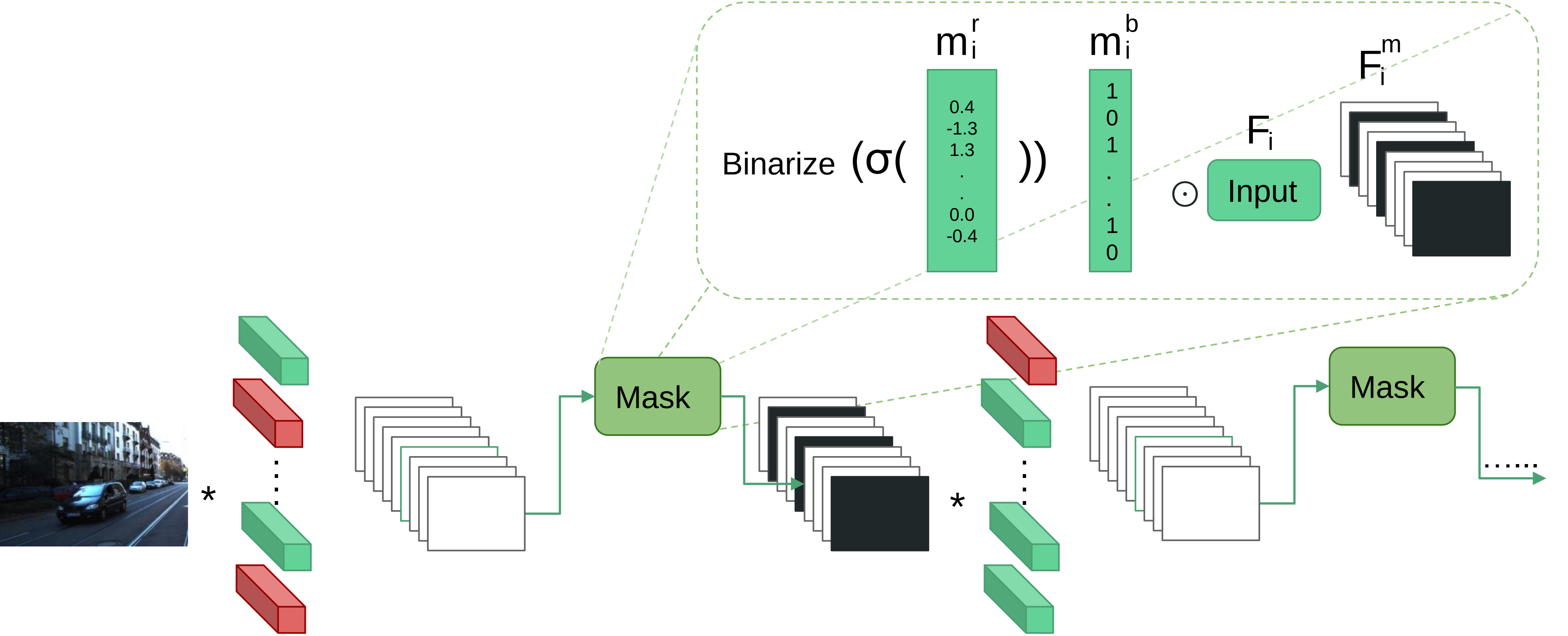
- **Exhaustive:**
  - Requires multi-stage training not suitable for large datasets and deeper models.
- **Sub-optimal:**
  - Once a filter is removed there is no turning back.
- **Pre-defined layer-wise compression rate prior:**
  - Each layer has different sensitivity to filter removal in which with a non-joint solution requires exhaustive analysis to define layer-wise compression rates.

Proposed Method[1]



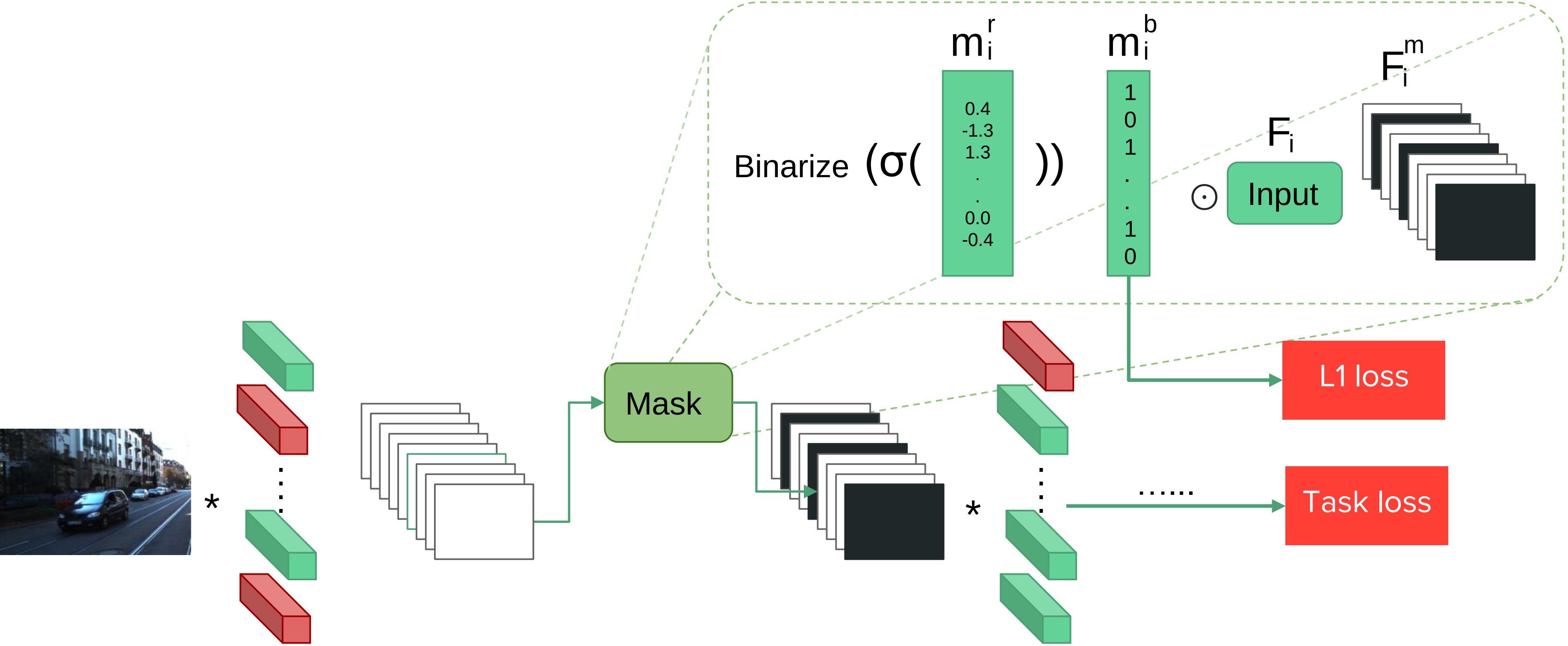
[1] S. Elkerdawy, H. Zhang and N. Ray, "Lightweight Monocular Depth Estimation Model by Joint End-to-End Filter Pruning," 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 2019, pp. 4290-4294.

Proposed Method[1]



[1] S. Elkerdawy, H. Zhang and N. Ray, "Lightweight Monocular Depth Estimation Model by Joint End-to-End Filter Pruning," 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 2019, pp. 4290-4294.

Proposed Method[1]



[1] S. Elkerdawy, H. Zhang and N. Ray, "Lightweight Monocular Depth Estimation Model by Joint End-to-End Filter Pruning," 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 2019, pp. 4290-4294.

Use cases

- Monocular depth estimation (Encoder-Decoder)
- Classification

Monocular Depth Estimation

Case study

Monocular depth estimation

- Can we estimate the depth of an object with only one camera?

Image from MegaDepth

Monocular depth estimation

- We follow monodepth [1] by learning disparity from stereo input at training and monocular at testing.

FPS/Board (CUDA cores)	1080 Ti (3584 @ 1600MHz)	TX2 (256 @ 1300MHz)	TX1 (256 @ 998MHz)
Baseline	33.7	5.6	3.0

[1] Clement Godard, Oisin Mac Aodha, and Gabriel J Brostow, “Unsupervised monocular depth estimation with left-right consistency,” in CVPR, 2017, vol. 2, p. 7.

Monocular depth estimation

- We follow monodepth [1] by learning disparity from stereo input at training and monocular at testing.

FPS/Board (CUDA cores)	1080 Ti (3584 @ 1600MHz)	TX2 (256 @ 1300MHz)	TX1 (256 @ 998MHz)
Baseline	33.7	5.6	3.0
Ours pruned (80% #Params reduction)	58.8	14.4 2.5x↑	8.5 2.8x↑

[1] Clement Godard, Oisin Mac Aodha, and Gabriel J Brostow, “Unsupervised monocular depth estimation with left-right consistency,” in CVPR, 2017, vol. 2, p. 7.

Monocular depth estimation

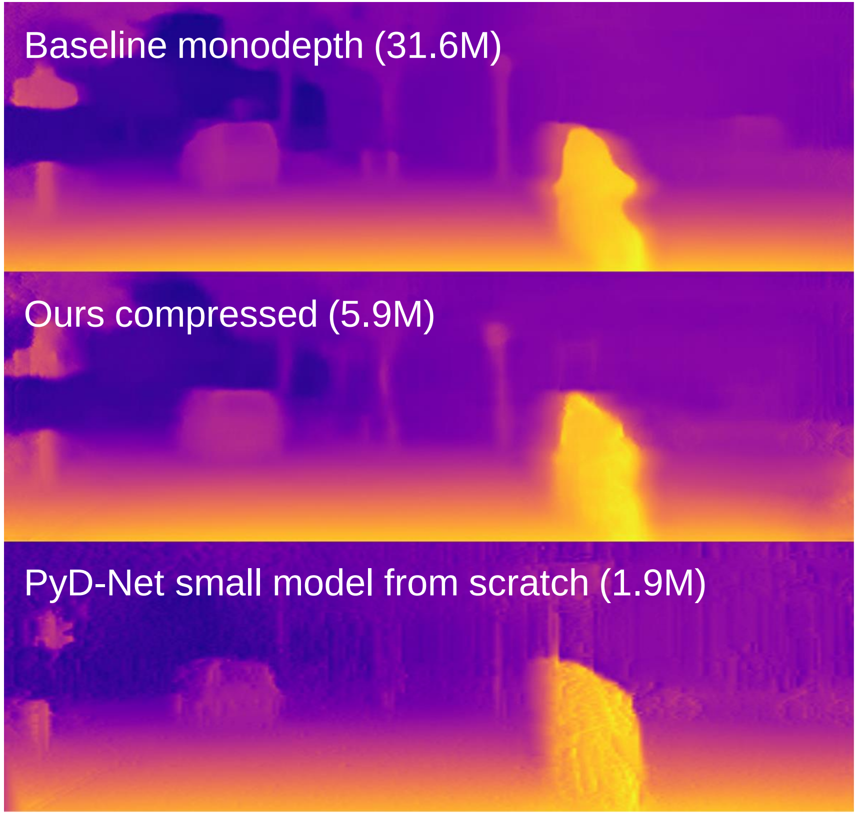
- We follow monodepth [1] by learning disparity from stereo input at training and monocular at testing.

FPS/Board (CUDA cores)	1080 Ti (3584 @ 1600MHz)	TX2 (256 @ 1300MHz)	TX1 (256 @ 998MHz)
Baseline	33.7	5.6	3.0
Ours pruned (80% #Params reduction)	58.8	14.4 2.5x↑	8.5 2.8x↑

Note how same pruned model achieves different speed up on different devices ➔ This highlights the fact that platforme aware pruning with operational metric like FPS/latency or energy is important

[1] Clement Godard, Oisin Mac Aodha, and Gabriel J Brostow, “Unsupervised monocular depth estimation with left-right consistency,” in CVPR, 2017, vol. 2, p. 7.

Qualitative Results



[1] S. Elkerdawy, H. Zhang and N. Ray, "Lightweight Monocular Depth Estimation Model by Joint End-to-End Filter Pruning," 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 2019, pp. 4290-4294.

Classification

Case study

VGG19 - CIFAR100

Model	Layer wise compression rate hyperparameter?	Needs Fine-tune?	Accuracy	Model size (No. param)
VGG_19 baseline	-	-	73.11	20.09M
VGG_19 (ours)	No	No	**73.45** **0.34%↑**	4.29M **78.65%↓**
Same architecture as **pruned** but trained from **scratch**	-	-	71.63 **1.48%↓**	4.29M **78.65%↓**
Same architecture as **pruned** but trained from **scratch** + **softmax distillation** [1]	-	-	72.39 **0.72%↓**	4.29M **78.65%↓**
Slimming (bn reg) [2] - ICCV17	Yes	Yes	**73.48** **0.5%↑***	5.00M **75.1%↓**
Variational CNN pruning [3] - CVPR19	No	No	73.33 **0.07%↑***	9.14 **37.87%↓**

* Accuracy increase is calculated from their baseline accuracy

[1] Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network.
[2] Liu, Zhuang, et al. "Learning efficient convolutional networks through network slimming." Proceedings of the IEEE International Conference on Computer Vision. 2017.
[3] Zhao, Chenglong, et al. "Variational Convolutional Neural Network Pruning." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

VGG - ImageNet

Model	Layer wise compression rate hyperparameter?	Needs Fine-tune?	Accuracy	#Param
vgg11	-	-	70.84	132.9M
Vgg11_pruned (Ours)	No	No	64.07	**23.79M**
Slimming ICCV17 [1]	Yes	Yes	63.34	23.2M
Taylor CVPR19 [2]	No	Yes	**70.65**	31.8M

- Our pruning achieves a more compressed network, so direct comparison with [2] would be unfair.

[1] Liu, Zhuang, et al. "Learning efficient convolutional networks through network slimming." Proceedings of the IEEE International Conference on Computer Vision. 2017.
[2] Molchanov, Pavlo, et al. "Importance Estimation for Neural Network Pruning." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

So far ...

- **One stage (fine-tune + prune) training**
- **Joint End-to-End pruning** → No greedy hard pruning
- **No need for layer wise compression rate** or pre-calculated sensitivity analysis.
- Scalability of the method with very large models.

But ..

- **No Control** over the size or operational properties of the model
  - e.g.) compress to reach ***x*% memory usage**, or ***x* runtime latency** on Huawei Mate10 phone.
- Filter **pruning** only, bounded with the smallest possible size of a network.

```
VGG( Conv2d(3, 64), BatchNorm2d, ReLU,             64,
     Conv2d(64, 64), BatchNorm2d, ReLU,            64,
     MaxPool2d,                                     M,
     Conv2d(64, 128), BatchNorm2d, ReLU,          128,
     Conv2d(128, 128), BatchNorm2d, ReLU,         128,
     MaxPool2d,                                     M,
     Conv2d(128, 256), BatchNorm2d, ReLU,         256,
     Conv2d(256, 256), BatchNorm2d, ReLU,         256,
     Conv2d(256, 256), BatchNorm2d, ReLU,         256,
     Conv2d(256, 256), BatchNorm2d, ReLU,         256,
     MaxPool2d,                                     M,
     Conv2d(256, 512), BatchNorm2d, ReLU,         512,
     Conv2d(512, 512), BatchNorm2d, ReLU,         512,
     Conv2d(512, 512), BatchNorm2d, ReLU,         512,
     Conv2d(512, 512), BatchNorm2d, ReLU,         512,
     MaxPool2d,                                     M,
     Conv2d(512, 512), BatchNorm2d, ReLU,         512,
     Conv2d(512, 512), BatchNorm2d, ReLU,         512,
     Conv2d(512, 512), BatchNorm2d, ReLU,         512,
     Conv2d(512, 512), BatchNorm2d, ReLU,         512,
     MaxPool2d, Linear(512, 100)                    M
)
```

```
VGG( Conv2d(3, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     MaxPool2d,                                     M,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     MaxPool2d,                                     M,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     MaxPool2d,                                     M,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     MaxPool2d,                                     M,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     Conv2d(1, 1), BatchNorm2d, ReLU,               1,
     MaxPool2d, Linear(1, 100)                      M
)
```

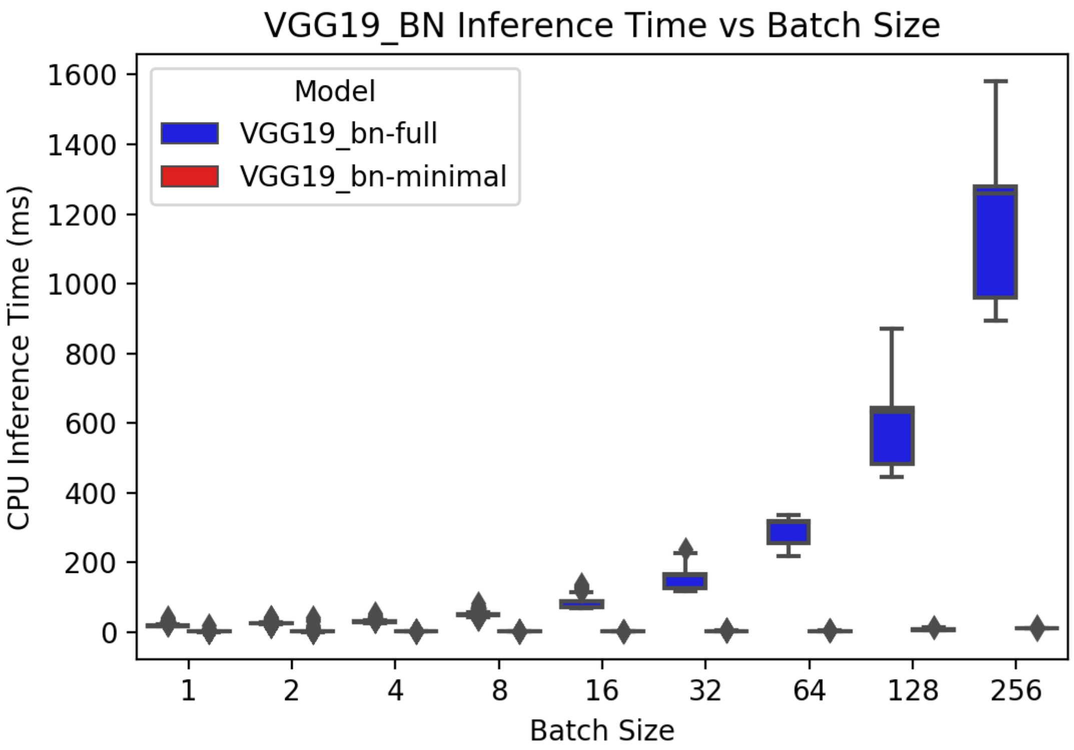
Latency Upper & Lower Bounds

How long does a single forward pass take?

Experiments run on my Dell XPS 9360, CPU inference only [Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz].

Averaged over the entire CIFAR-100 test dataset (10,000 images). All times are reported as milliseconds (ms).

Metrics derived using awwong1/torchprof.



Constrained Model Pruning

Common Constraints

Indirect Measurements (Model):

- Multiply-Accumulate Operations (MACs)
- Floating-Point Operations (FLOPs)
- Number of Parameters

Direct Measurements (Model + Device):

- Run Time/Latency
- Power (Watts)
- Energy (Run Time * Power)

Indirect measurements provide insight into a model's performance, but may not be good approximations of direct measurements of target device resources [1-3].

1. Not all ops are created equal! In: SysML (2018)
2. Designing energy-efficient convolutional neural networks using energy-aware pruning. In: CVPR (2017)
3. Scalpel: Customizing DNN pruning to the underlying hardware parallelism. In: ISCA (2017)

Motivation

- Previously in monocular depth estimation

  Same model achieves different speed up on different devices

FPS/Board (CUDA cores)	1080 Ti (3584 @ 1600MHz)	TX2 (256 @ 1300MHz)	TX1 (256 @ 998MHz)
Baseline	33.7	5.6	3.0
Ours pruned	58.8	14.4 2.5x ↑	8.5 2.8x ↑

Motivation

- Classification setup
- We sampled randomly multiple ‘signature’ models that can be obtained from pruning methods from VGG19 and trained each from scratch.

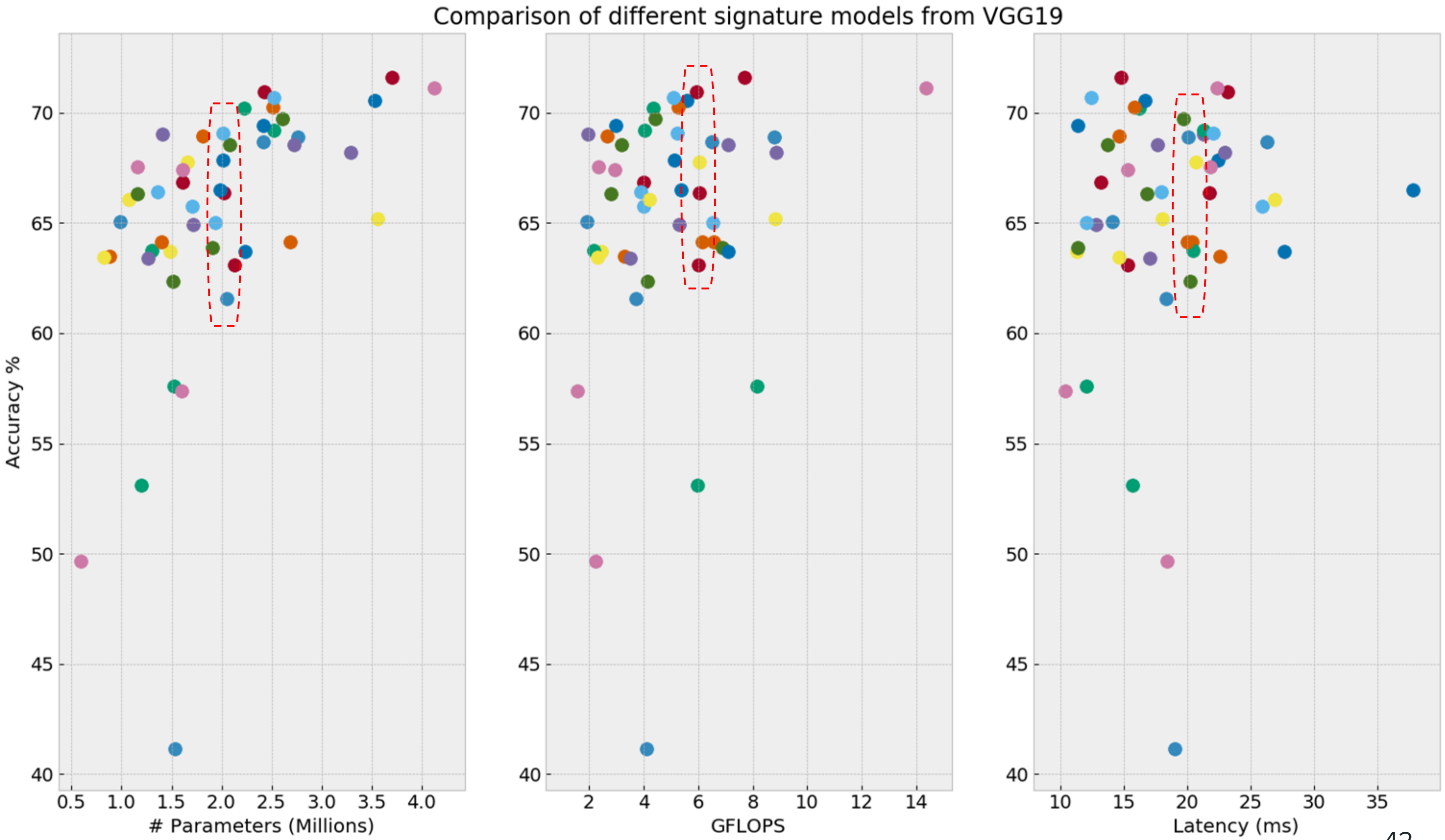
Example:

VGG19 : [64, 64, 128, 128, 256, 256, 256, 256, 512, 512, 512, 512, 512, 512, 512, 512]
Sample 1 : [39, 17, 52, 27, 255, 78, 104, 23, 72, 118, 124, 490, 67, 83, 77, 480]
Sample 2: [1, 18, 100, 96, 102, 47, 201, 236, 58, 407, 208, 254, 233, 479, 496, 242]
Sample 3: [36, 51, 33, 96, 227, 175, 126, 233, 435, 433, 465, 230, 16, 258, 444, 34]
….

Motivation

- Models with the same number of Params/FLOPs/latency can vary in accuracy by large margin.

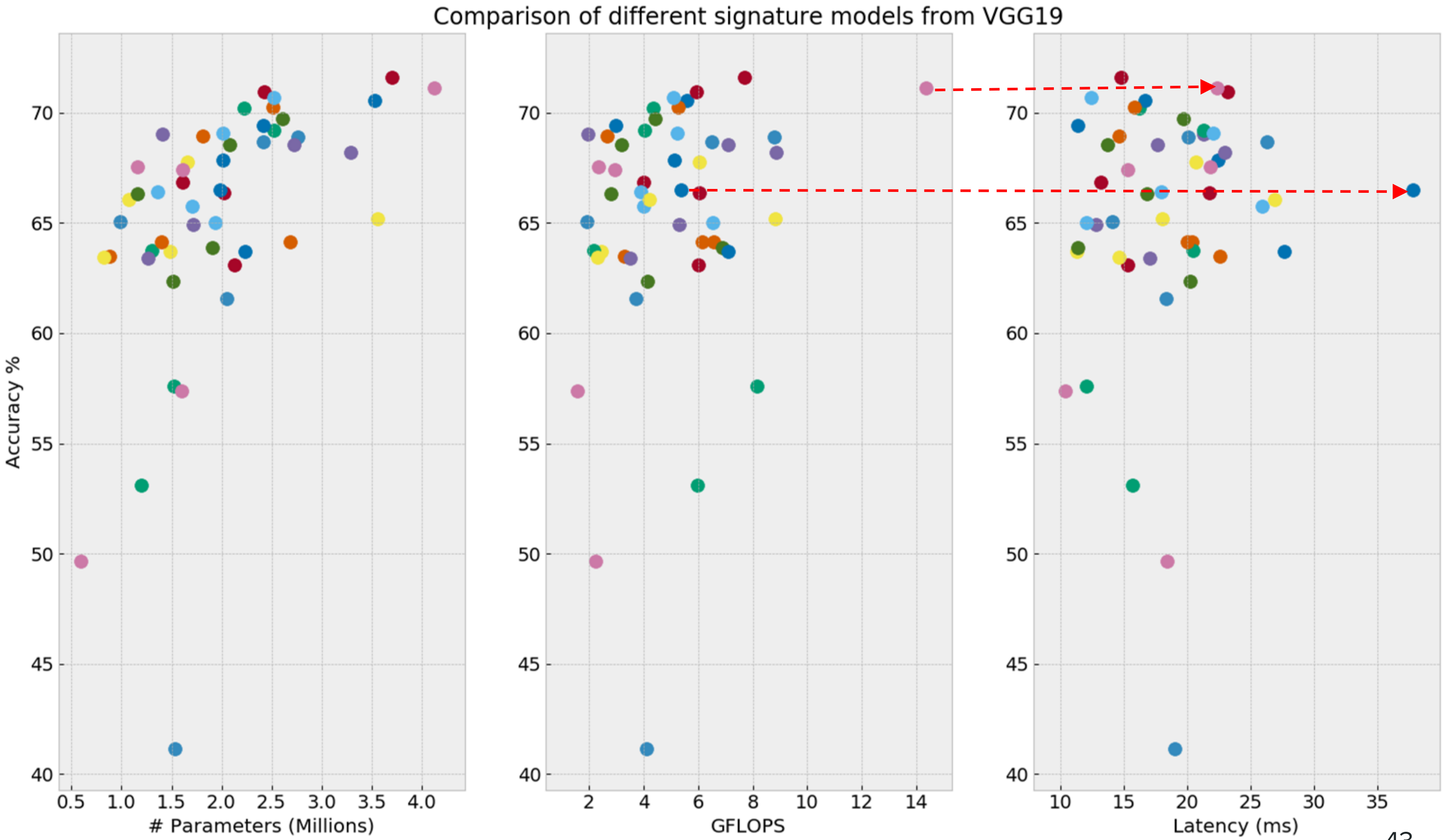
Joint task training and constraint optimization is important for optimal signature search.



Motivation

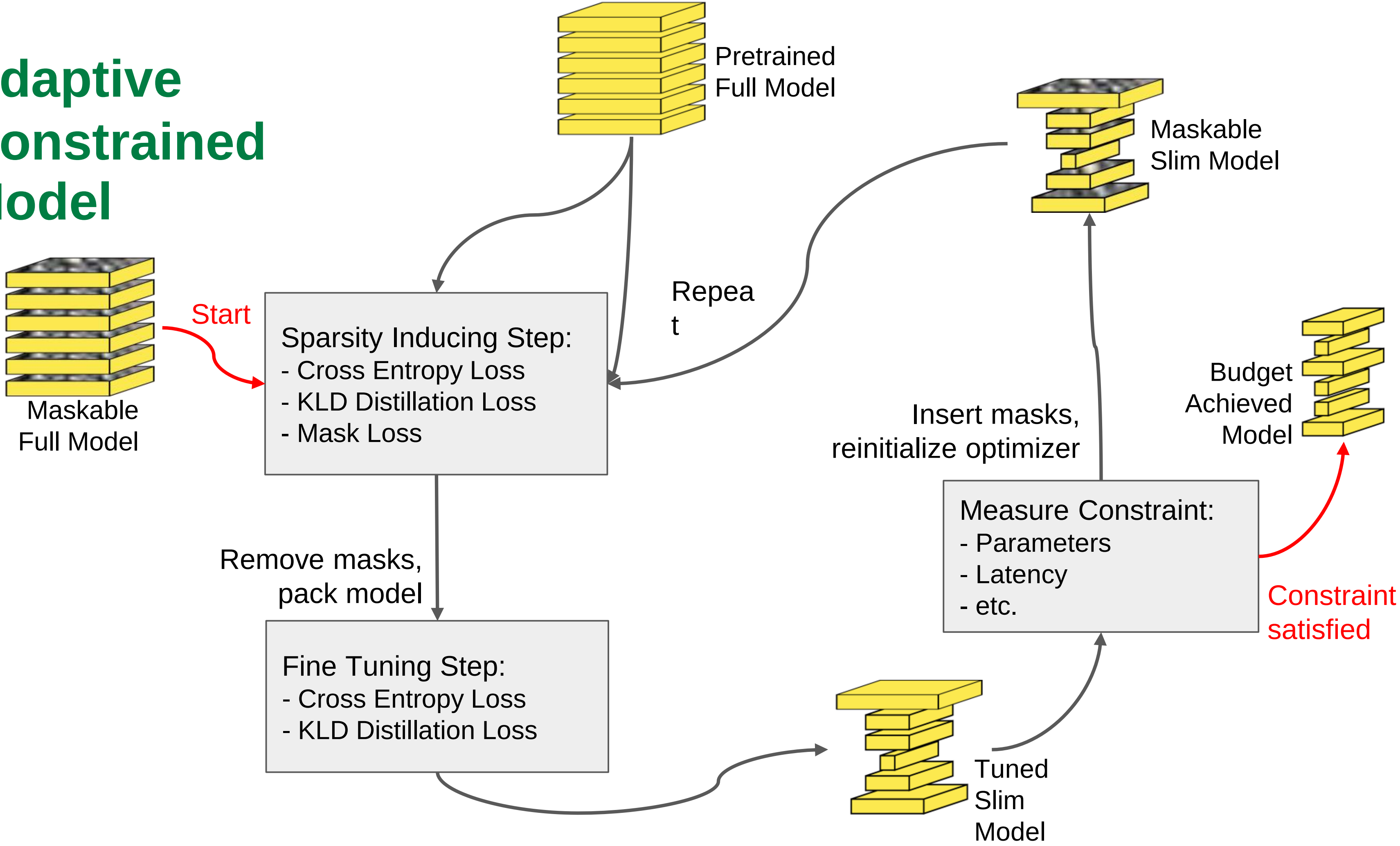
- No direct relation between FLOPs and latency.

Indirect measures can't guarantee optimization over direct measures.



Proposed Adaptive Constrained Pruning

Adaptive Constrained Model

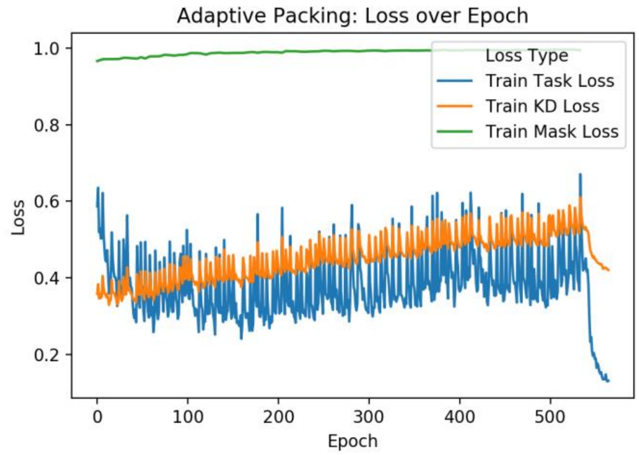


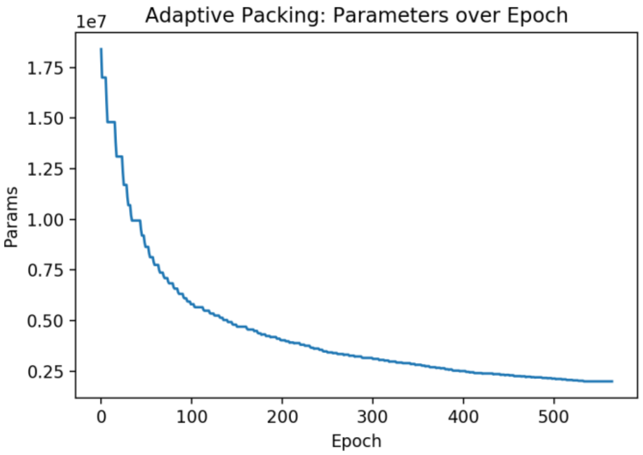
Results

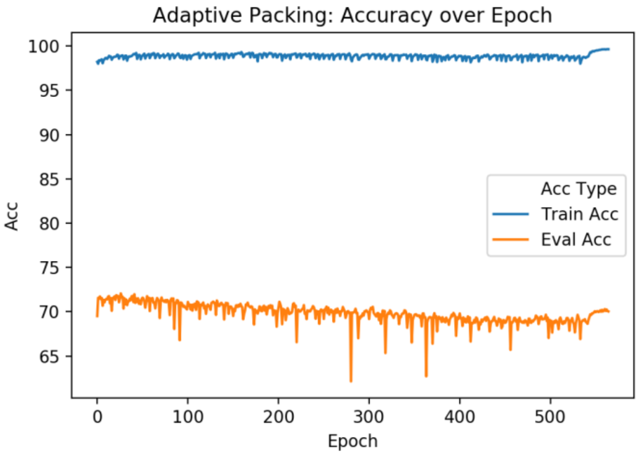
CIFAR100, VGG19_BN Constrained Pruning

Final evaluation accuracy of **70.3%** with 2,004,846 parameters (**>90%** parameter reduction).

Latency **3.9 times faster** than full model, batch sizes 1 to 256.







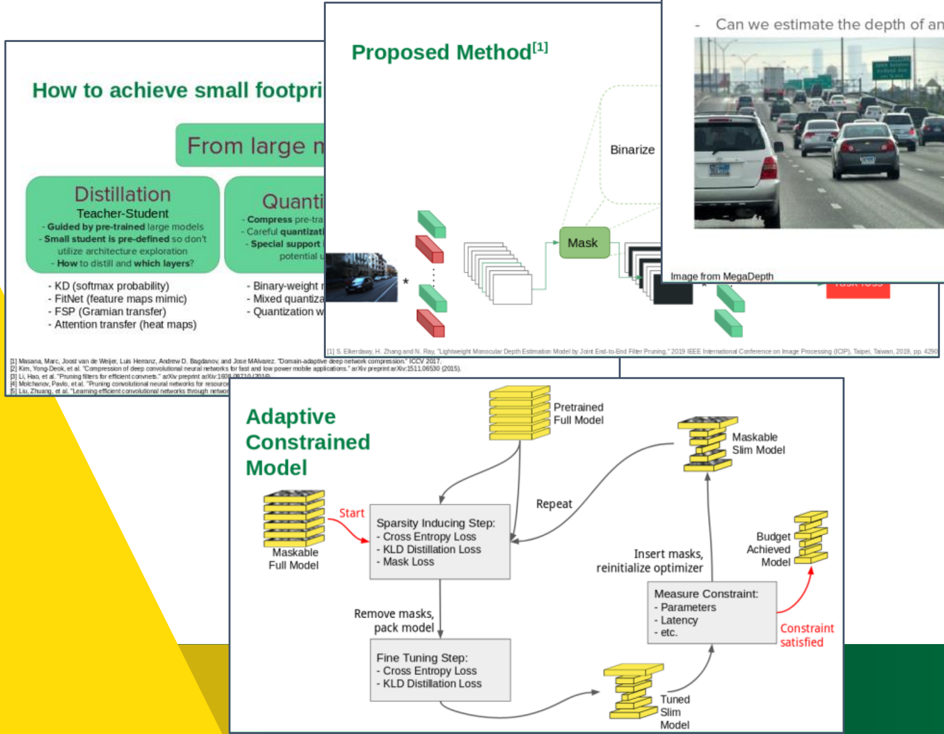
Practical benefits

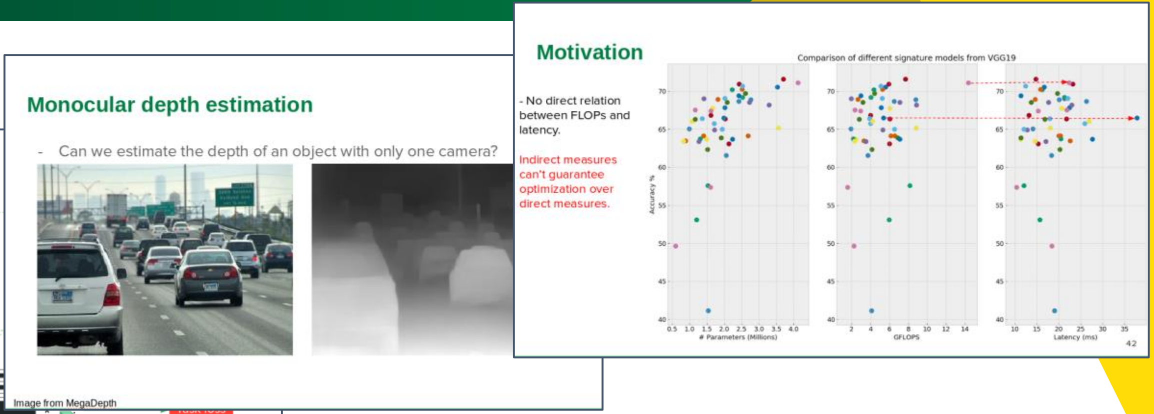
- Standalone differentiable binary mask module that can easily be inserted after any convolutional layer.
- Scalable to any large model
    - As shown on Encoder-Decoder models
- End-to-End training with:
    - No human intervention
    - No layerwise hyperparameter tuning.

Conclusion and Future Work

- **So far ..**
  - **One stage (fine-tune + prune) training**
    - No human intervention and minimal hyperparameter tuning
  - **Joint End-to-End pruning**
- **Work in progress ..**
  - **Platform aware adaptive** constrained optimization
  - **Scale up evaluation** on multiple **networks** and large **datasets** such as ImageNet
- **Future work ..**
  - **Incorporate quantization** to allow for further speed up and memory saving beyond the minimal possible model signature from pruning.

Thank You! Q&A





Huawei and University of Alberta Collaboration Project Model Compression

Presented by: Alexander W. Wong, Sara Elkerdawy
Under supervision: Nilanjan Ray, Hong Zhang