# 推理引擎-模型压缩

# 模型剪枝


ZOMI

# Talk Overview

1. **推理系统介绍**
   - 推理系统与推理引擎区别
   - 推理工作流程
   - 推理系统介绍
   - 推理引擎介绍
2. **模型小型化**
   - 基础参数概念
   - CNN小型化结构
   - Transform小型化结构
3. **离线优化压缩**
   - 低比特量化
   - 模型剪枝
   - 模型蒸馏
   - 二值化网络
4. **部署和运行优化**
   - 图转换优化（算子融合/重排/替换）
   - 并发执行与内存分配
   - 动态batch与bin Packing

# Talk Overview

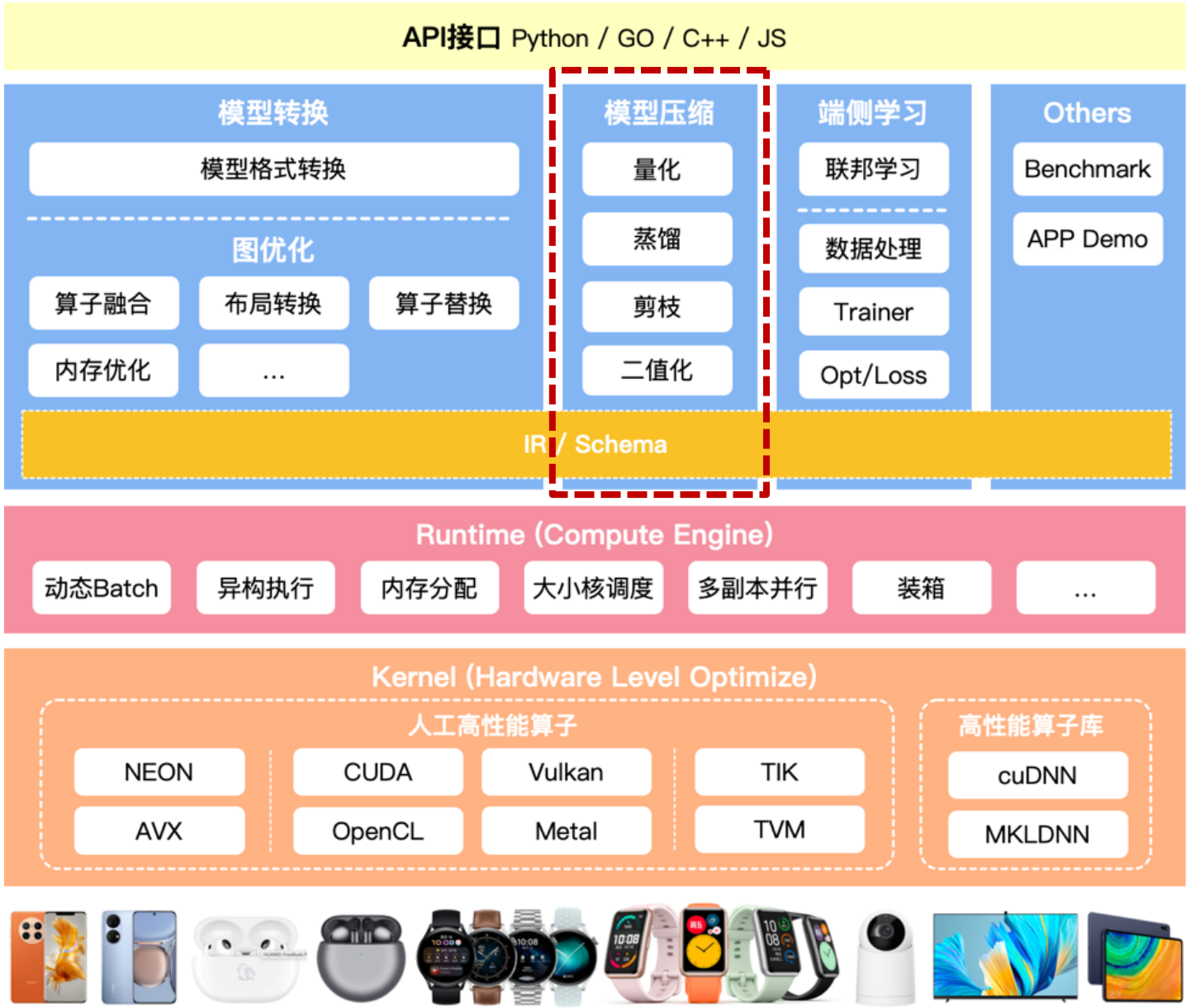
## 1. 模型剪枝

- Difference between pruning and quantification - 剪枝与量化的区别
- Classification of pruning methods - 剪枝算法分类
- Pruning process - 剪枝流程
- L1-norm based Channel Pruning - L1-norm剪枝算法

# 推理引擎架构

对模型进行压缩

- 减少模型大小
- 加快训练速度
- 保持相同精度



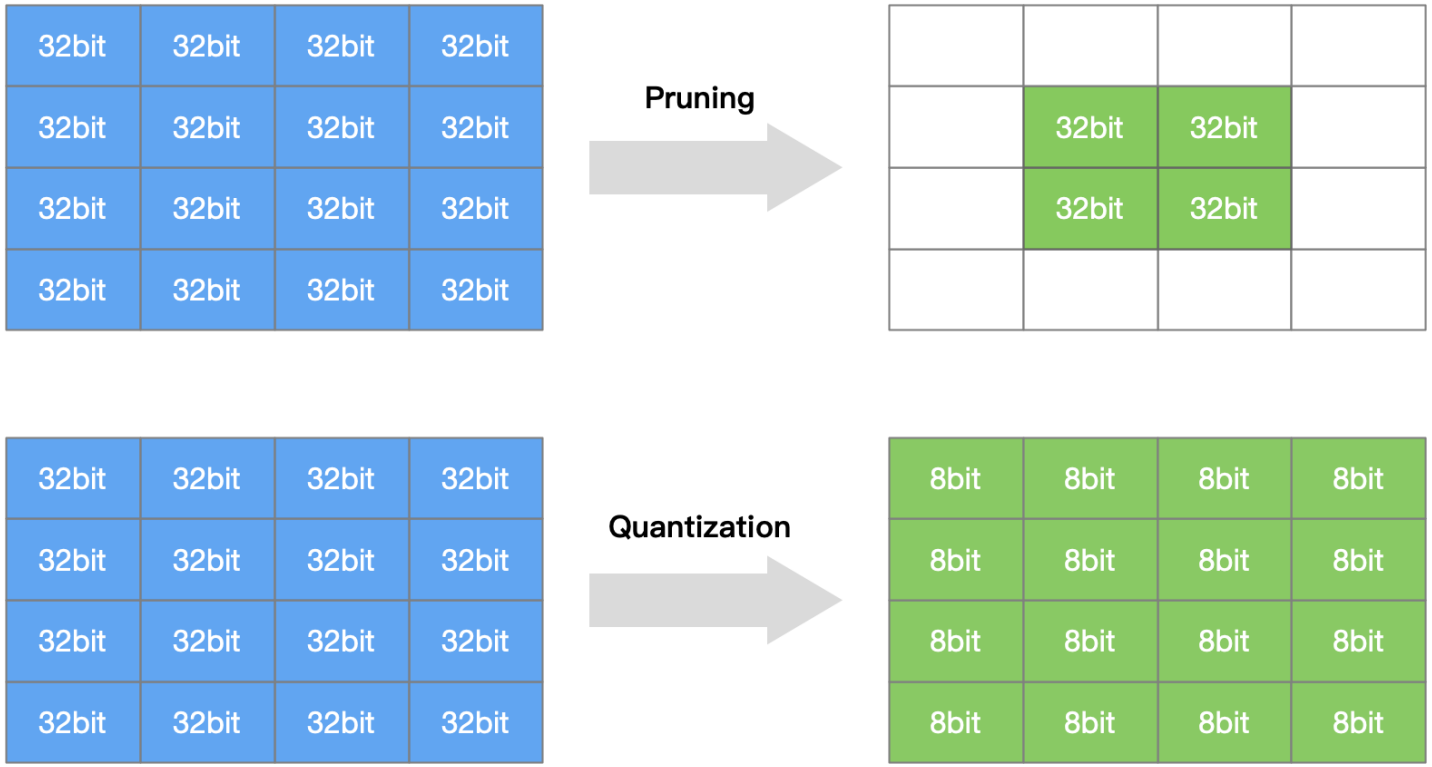
# 剪枝与量化区别

模型压缩提出了三部分优化：

1）是减少内存密集的访问量；2）提高获取模型参数的时间；3）加速模型推理时间。

# 量化压缩

- 模型量化是指通过减少权重表示或激活所需的比特数来压缩模型。
- 模型剪枝研究模型权重中的冗余， 并尝试删除/修剪冗余和非关键的权重。



# To prune, or not to prune: exploring the efficacy of pruning for model compression

1. 在内存占用相同情况下，大稀疏模型比小密集模型实现了更高的精度。
2. 经过剪枝之后稀疏模型要优于，同体积非稀疏模型。
3. 资源有限的情况下，剪枝是比较有效的模型压缩策略。
4. 优化点还可以往硬件稀疏矩阵储存方向发展。

Table 4: NMT sparse vs dense results

| # units | Sparsity | NNZ params | EN-DE BLEU score | DE-EN BLEU score |
|---|---|---|---|---|
| 256 | 0% | 34M | 23.52 | 26.52 |
| 512 | 0% | 81M | 26.05 | 28.88 |
| 768 | 0% | 140M | 26.63 | 29.41 |
| 1024 | 0% | 211M | 26.77 | 29.47 |
| | 80% | 44M | 26.86 | 29.50 |
| | 85% | 33M | 26.52 | 29.24 |
| | 90% | 23M | 26.19 | 28.81 |

Table 1: Model size and accuracy tradeoff for sparse-InceptionV3

| Sparsity | NNZ params | Top-1 acc. | Top-5 acc. |
|---|---|---|---|
| 0% | 27.1M | 78.1% | 94.3% |
| 50% | 13.6M | 78.0% | 94.2% |
| 75% | 6.8M | 76.1% | 93.2% |
| 87.5% | 3.3M | 74.6% | 92.5% |

# 剪枝算法分类

# 剪枝算法

| Name | Brief Introduction of Algorithm |
| --- | --- |
| Level Pruner | Pruning the specified ratio on each weight element based on absolute value of weight element |
| L1 Norm Pruner | Pruning output channels with the smallest L1 norm of weights (Pruning Filters for Efficient Convnets) Reference Paper |
| L2 Norm Pruner | Pruning output channels with the smallest L2 norm of weights |
| FPGM Pruner | Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration Reference Paper |
| Slim Pruner | Pruning output channels by pruning scaling factors in BN layers(Learning Efficient Convolutional Networks through Network Slimming) Reference Paper |
| Activation APoZ Rank Pruner | Pruning output channels based on the metric APoZ (average percentage of zeros) which measures the percentage of zeros in activations of (convolutional) layers. Reference Paper |
| Activation Mean Rank Pruner | Pruning output channels based on the metric that calculates the smallest mean value of output activations |
| Taylor FO Weight Pruner | Pruning filters based on the first order taylor expansion on weights(Importance Estimation for Neural Network Pruning) Reference Paper |
| ADMM Pruner | Pruning based on ADMM optimization technique Reference Paper |
| Linear Pruner | Sparsity ratio increases linearly during each pruning rounds, in each round, using a basic pruner to prune the model. |
| AGP Pruner | Automated gradual pruning (To prune, or not to prune: exploring the efficacy of pruning for model compression) Reference Paper |
| Lottery Ticket Pruner | The pruning process used by "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks". It prunes a model iteratively. Reference Paper |
| Simulated Annealing Pruner | Automatic pruning with a guided heuristic search method, Simulated Annealing algorithm Reference Paper |
| Auto Compress Pruner | Automatic pruning by iteratively call SimulatedAnnealing Pruner and ADMM Pruner Reference Paper |
| AMC Pruner | AMC: AutoML for Model Compression and Acceleration on Mobile Devices Reference Paper |
| Movement Pruner | Movement Pruning: Adaptive Sparsity by Fine-Tuning Reference Paper |

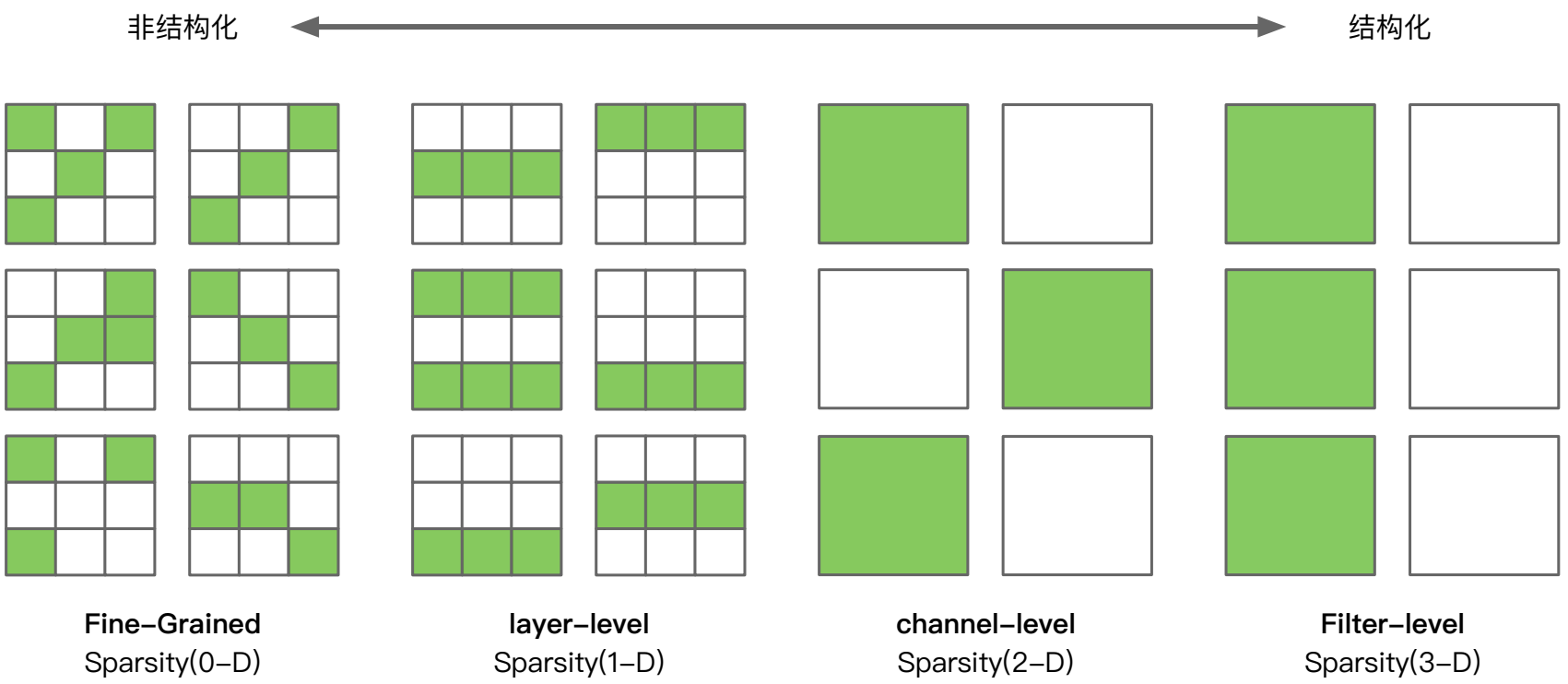
# 模型剪枝分类

- **Unstructured Pruning（非结构化剪枝）**

  随机对独立的权重或者神经元链接进行剪枝

- **Structured Pruning（结构化剪枝）**

  对 filter / channel / layer 进行剪枝



# 模型剪枝分类

- **Unstructured Pruning（非结构化剪枝）**

  Pros：剪枝算法简单，模型压缩比高

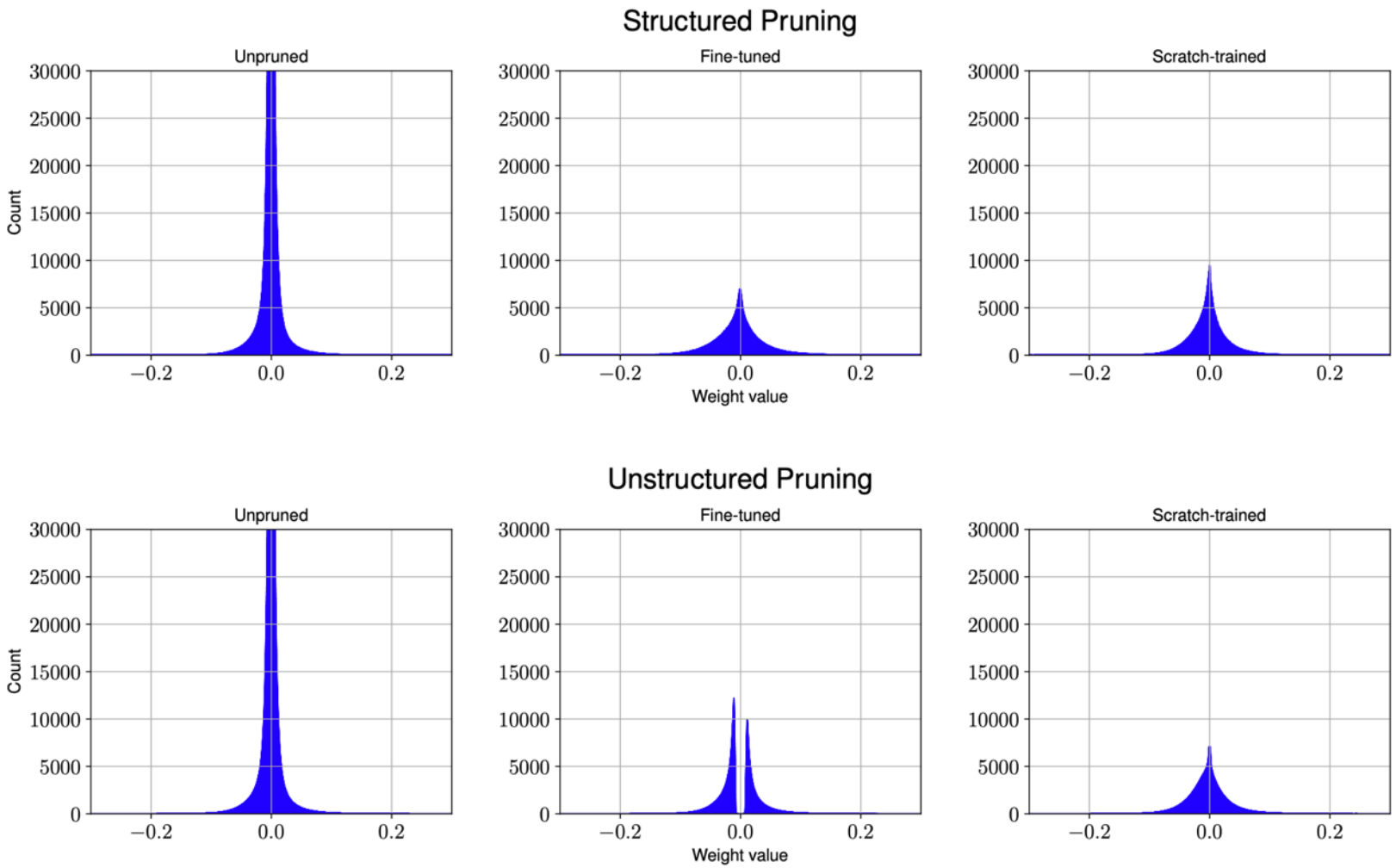
  Cons：精度不可控，剪枝后权重矩阵稀疏，没有专用硬件难以实现压缩和加速的效果

- **Structured Pruning（结构化剪枝）**

  Pros：大部分算法在 channel 或者 layer 上进行剪枝，保留原始卷积结构，不需要专用硬件来实现

  Cons：剪枝算法相对复杂

# Weight distribution of CNN layers for different pruning methods



# 剪枝流程

# 模型剪枝流程

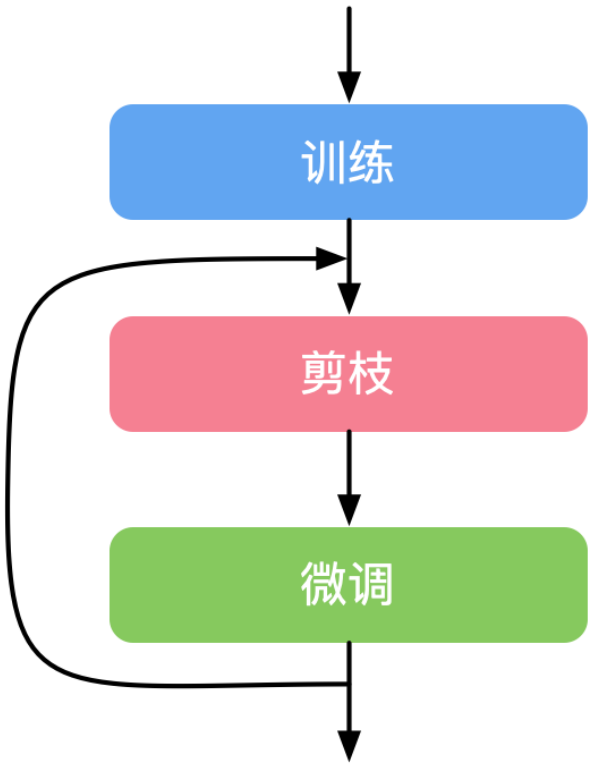
**对模型进行剪枝三种常见做法：**

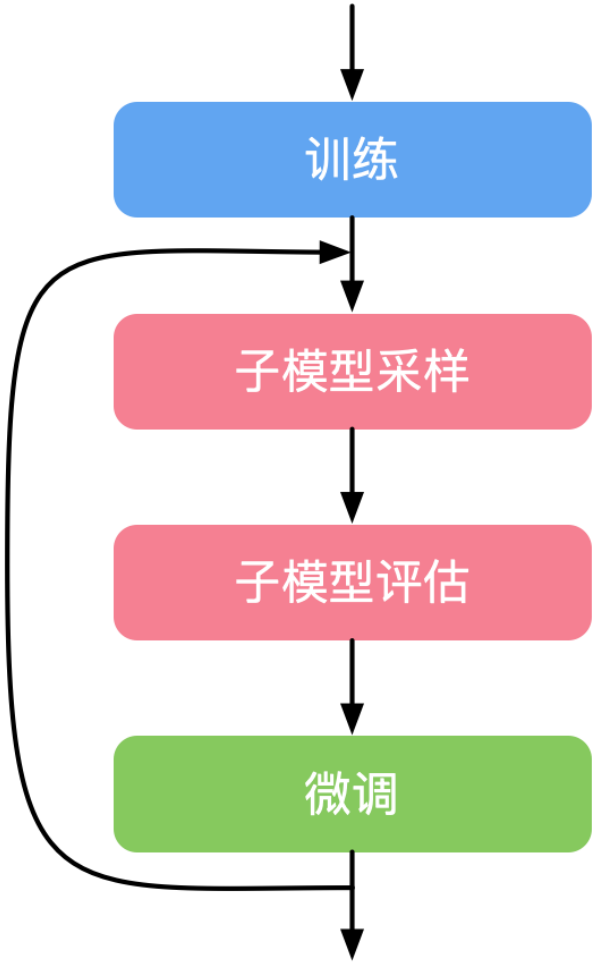
1. 训练一个模型 -> 对模型进行剪枝 -> 对剪枝后模型进行微调
2. 在模型训练过程中进行剪枝 -> 对剪枝后模型进行微调
3. 进行剪枝 -> 从头训练剪枝后模型

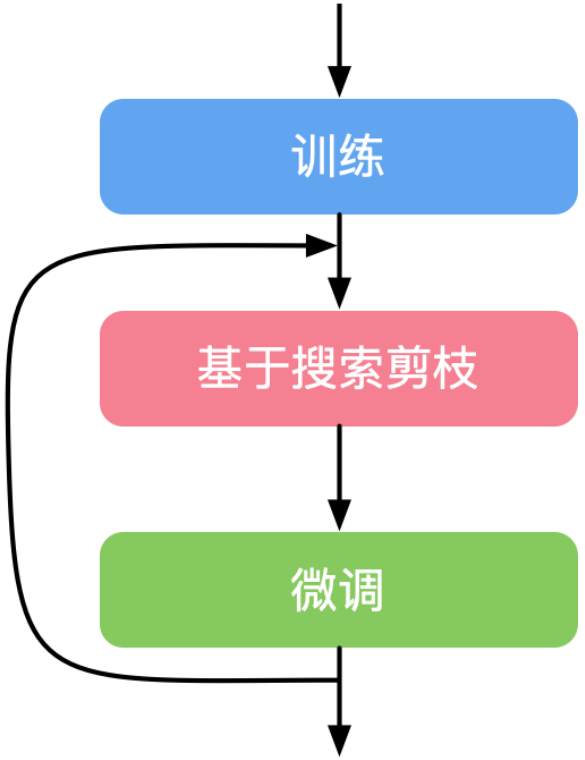
# 模型剪枝主要单元

- **训练 Training**：训练过参数化模型，得到最佳网络性能，以此为基准；
- **剪枝 Pruning**：根据算法对模型剪枝，调整网络结构中通道或层数，得到剪枝后的网络结构；
- **微调 Finetune**：在原数据集上进行微调，用于重新弥补因为剪枝后的稀疏模型丢失的精度性能。

# 训练一个模型 -> 对模型进行剪枝 -> 对剪枝后模型进行微调



# L1-norm
# 剪枝算法

# L1-norm based Channel Pruning

- 使用 L1-norm 标准来衡量卷积核的重要性，L1-norm 是一个很好的选择卷积核的方法，认为如果一个filter的绝对值和比较小，说明该filter并不重要。论文指出对剪枝后的网络结构从头训练要比对重新训练剪枝后的网络。

# L1-norm based Channel Pruning

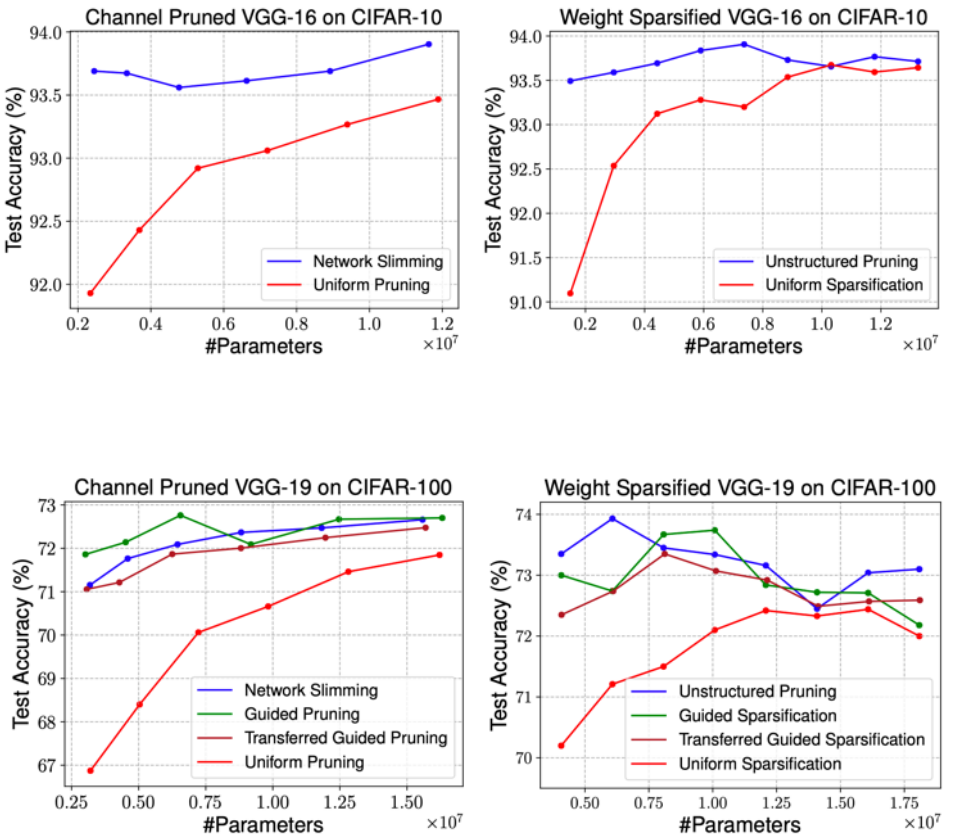
**算法步骤**

1. 对每个卷积核 $F_{ij}$，计算它的权重绝对值（L1-norm）之和 $S_j = \sum_{l=1}^{ji} \sum |Kl|$；
2. 根据卷积核的L1-norm 值 $S_j$ 进行排序；
3. 将 m 个权重绝对值之和最小的卷积核以及对应 feature maps进行剪枝；
4. 下一个卷积层中与剪掉 feature maps 相关的卷积核 $F_{i+1,j}$ 也要剪枝；
5. 对于第 I 层和第 i+1 层的新权重矩阵被创建，剩下权重参数被复制到新模型中。

# L1-norm based Channel Pruning

| Dataset | Model | Unpruned | Prune Ratio | Fine-tuned | Scratch-E | Scratch-B |
|---|---|---|---|---|---|---|
| CIFAR-10 | VGG-19 | 93.50 (±0.11) | 30% | 93.51 (±0.05) | **93.71** (±0.09) | 93.31 (±0.26) |
| | | | 80% | 93.52 (±0.10) | **93.71** (±0.08) | 93.64 (±0.09) |
| | | | 95% | 93.34 (±0.13) | 93.21 (±0.17) | **93.63** (±0.18) |
| | PreResNet-110 | 95.04 (±0.15) | 30% | 95.06 (±0.05) | 94.84 (±0.07) | **95.11** (±0.09) |
| | | | 80% | **94.55** (±0.11) | 93.76 (±0.10) | 94.52 (±0.13) |
| | | | 95% | **92.35** (±0.20) | 91.23 (±0.11) | 91.55 (±0.34) |
| | DenseNet-BC-100 | 95.24 (±0.17) | 30% | 95.21 (±0.17) | 95.22 (±0.18) | **95.23** (±0.14) |
| | | | 80% | 95.04 (±0.15) | 94.42 (±0.12) | **95.12** (±0.04) |
| | | | 95% | **94.19** (±0.15) | 92.91 (±0.22) | 93.44 (±0.19) |
| CIFAR-100 | VGG-19 | 71.70 (±0.31) | 30% | 71.96 (±0.36) | 72.81 (±0.31) | **73.30** (±0.25) |
| | | | 50% | 71.85 (±0.30) | 73.12 (±0.36) | **73.77** (±0.23) |
| | | | 95% | 70.22 (±0.38) | 70.88 (±0.35) | **72.08** (±0.15) |
| | PreResNet-110 | 76.96 (±0.34) | 30% | 76.88 (±0.31) | 76.36 (±0.26) | **76.96** (±0.31) |
| | | | 50% | **76.60** (±0.36) | 75.45 (±0.23) | 76.42 (±0.39) |
| | | | 95% | 68.55 (±0.51) | 68.13 (±0.64) | **68.99** (±0.32) |
| | DenseNet-BC-100 | 77.59 (±0.19) | 30% | 77.23 (±0.05) | 77.58 (±0.25) | **77.97** (±0.31) |
| | | | 50% | 77.41 (±0.14) | 77.65 (±0.09) | **77.80** (±0.23) |
| | | | 95% | **73.67** (±0.03) | 71.47 (±0.46) | 72.57 (±0.37) |
| ImageNet | VGG-16 | 73.37 | 30% | 73.68 | 72.75 | **74.02** |
| | | | 60% | **73.63** | 71.50 | 73.42 |
| | ResNet-50 | 76.15 | 30% | **76.06** | 74.77 | 75.70 |
| | | | 60% | **76.09** | 73.69 | 74.91 |

**Table 6:** Results (accuracy) for unstructured pruning (Han et al., 2015). "Prune Ratio" denotes the percentage of parameters pruned in the set of all convolutional weights.



# 参考文献


- 模型压缩：剪枝算法 https://zhuanlan.zhihu.com/p/462026539
- To prune, or not to prune: exploring the efficacy of pruning for model compression
- The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks
- Zhu M, Gupta S. To prune, or not to prune: exploring the efficacy of pruning for model compression[J]. arXiv: Machine Learning, 2017.
- Han S, Pool J, Tran J, et al. Learning both weights and connections for efficient neural network[C]. Advances in neural information processing systems..
- LeCun Y, Denker J S, Solla S A. Optimal brain damage[C]//Advances in neural information processing systems. 1990: 598-605.
- Guo Y, Yao A, Chen Y. Dynamic network surgery for efficient dnns[C]//Advances In Neural Information Processing Systems. 2016: 1379-1387.
- AutoCompress: An Automatic DNN Structured Pruning Framework for Ultra-High Compression Rates
- A Systematic DNN Weight Pruning Framework using Alternating Direction Method of Multipliers
- Network Trimming: A Data-Driven Neuron Pruning Approach towards Efficient Deep Architectures
- Learning Efficient Convolutional Networks through Network Slimming
- Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration
- Pruning Filters for Efficient ConvNets.