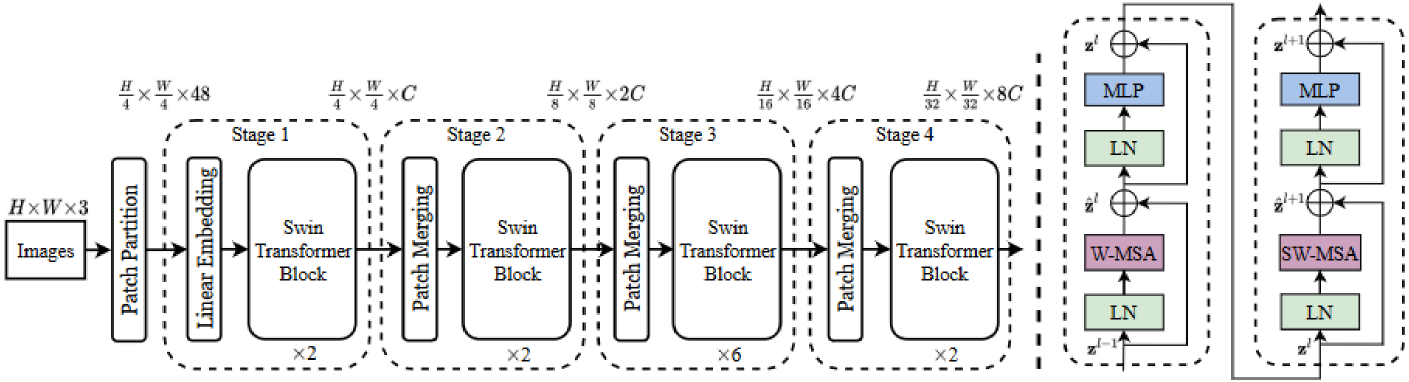
ConvNext and progress report

汇报人：王浩宇

https://blog.csdn.net/wqthaha/article/details/125492488
https://arxiv.org/abs/2201.03545

Architecture of Swin Transformer



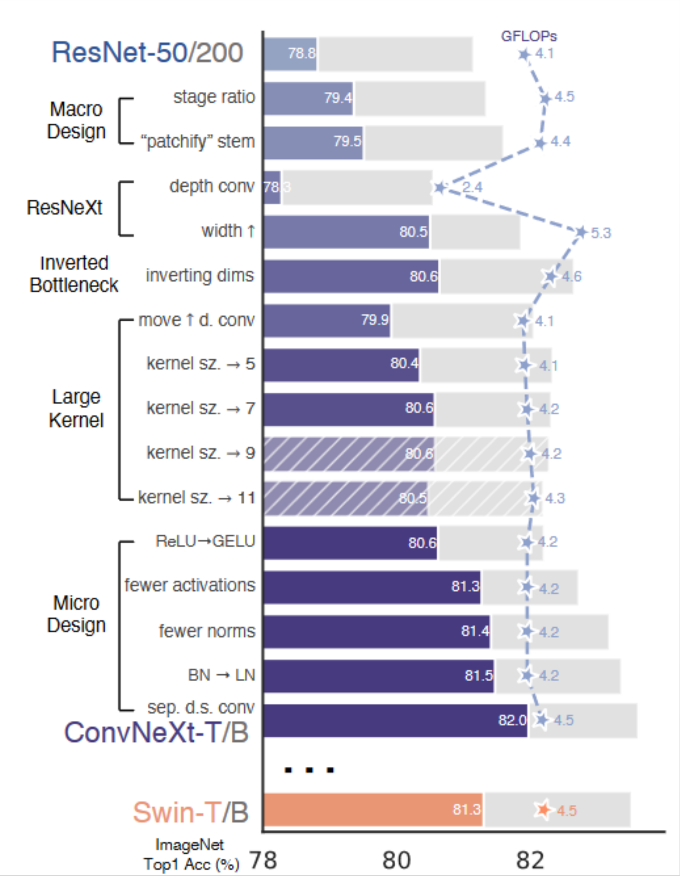
(a) Architecture (b) Two Successive Swin Transformer Blocks

Figure 3. (a) The architecture of a Swin Transformer (Swin-T); (b) two successive Swin Transformer Blocks (notation presented with Eq. (3)). W-MSA and SW-MSA are multi-head self attention modules with regular and shifted windowing configurations, respectively.

	downsp. rate (output size)	Swin-T	Swin-S	Swin-B	Swin-L
stage 1	4× (56×56)	concat 4×4, 96-d, LN	concat 4×4, 96-d, LN	concat 4×4, 128-d, LN	concat 4×4, 192-d, LN
		[win. sz. 7×7, dim 96, head 3] × 2	[win. sz. 7×7, dim 96, head 3] × 2	[win. sz. 7×7, dim 128, head 4] × 2	[win. sz. 7×7, dim 192, head 6] × 2
stage 2	8× (28×28)	concat 2×2, 192-d , LN	concat 2×2, 192-d , LN	concat 2×2, 256-d , LN	concat 2×2, 384-d , LN
		[win. sz. 7×7, dim 192, head 6] × 2	[win. sz. 7×7, dim 192, head 6] × 2	[win. sz. 7×7, dim 256, head 8] × 2	[win. sz. 7×7, dim 384, head 12] × 2
stage 3	16× (14×14)	concat 2×2, 384-d , LN	concat 2×2, 384-d , LN	concat 2×2, 512-d , LN	concat 2×2, 768-d , LN
		[win. sz. 7×7, dim 384, head 12] × 6	[win. sz. 7×7, dim 384, head 12] × 18	[win. sz. 7×7, dim 512, head 16] × 18	[win. sz. 7×7, dim 768, head 24] × 18
stage 4	32× (7×7)	concat 2×2, 768-d , LN	concat 2×2, 768-d , LN	concat 2×2, 1024-d , LN	concat 2×2, 1536-d , LN
		[win. sz. 7×7, dim 768, head 24] × 2	[win. sz. 7×7, dim 768, head 24] × 2	[win. sz. 7×7, dim 1024, head 32] × 2	[win. sz. 7×7, dim 1536, head 48] × 2

Table 7. Detailed architecture specifications.

ConvNext


ResNet-50/200
GFLOPs
78.8
4.1
Macro Design
stage ratio
79.4
4.5
"patchify" stem
79.5
4.4
ResNeXt
depth conv
78.3
2.4
width ↑
80.5
5.3
Inverted Bottleneck
inverting dims
80.6
4.6
Large Kernel
move ↑ d. conv
79.9
4.1
kernel sz. → 5
80.4
4.1
kernel sz. → 7
80.6
4.2
kernel sz. → 9
80.6
4.2
kernel sz. → 11
80.5
4.3
Micro Design
ReLU→GELU
80.6
4.2
fewer activations
81.3
4.2
fewer norms
81.4
4.2
BN → LN
81.5
4.2
sep. d.s. conv
82.0
4.5
ConvNeXt-T/B
Swin-T/B
81.3
4.5
ImageNet Top1 Acc (%)
78
80
82

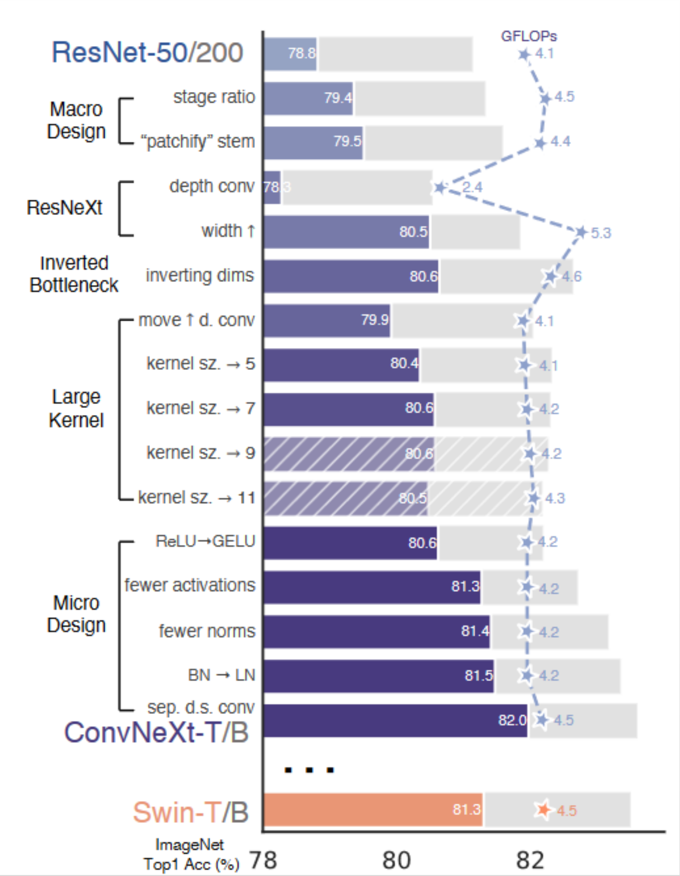
Macro design

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		[3×3, 64 3×3, 64] ×2	[3×3, 64 3×3, 64] ×3	[1×1, 64 3×3, 64 1×1, 256] ×3	[1×1, 64 3×3, 64 1×1, 256] ×3	[1×1, 64 3×3, 64 1×1, 256] ×3
conv3_x	28×28	[3×3, 128 3×3, 128] ×2	[3×3, 128 3×3, 128] ×4	[1×1, 128 3×3, 128 1×1, 512] ×4	[1×1, 128 3×3, 128 1×1, 512] ×4	[1×1, 128 3×3, 128 1×1, 512] ×8
conv4_x	14×14	[3×3, 256 3×3, 256] ×2	[3×3, 256 3×3, 256] ×6	[1×1, 256 3×3, 256 1×1, 1024] ×6	[1×1, 256 3×3, 256 1×1, 1024] ×23	[1×1, 256 3×3, 256 1×1, 1024] ×36
conv5_x	7×7	[3×3, 512 3×3, 512] ×2	[3×3, 512 3×3, 512] ×3	[1×1, 512 3×3, 512 1×1, 2048] ×3	[1×1, 512 3×3, 512 1×1, 2048] ×3	[1×1, 512 3×3, 512 1×1, 2048] ×3
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

	downsp. rate (output size)	Swin-T	Swin-S	Swin-B	Swin-L
stage 1	4× (56×56)	concat 4×4, 96-d, LN	concat 4×4, 96-d, LN	concat 4×4, 128-d, LN	concat 4×4, 192-d, LN
		[win. sz. 7×7, dim 96, head 3] × 2	[win. sz. 7×7, dim 96, head 3] × 2	[win. sz. 7×7, dim 128, head 4] × 2	[win. sz. 7×7, dim 192, head 6] × 2
stage 2	8× (28×28)	concat 2×2, 192-d , LN	concat 2×2, 192-d , LN	concat 2×2, 256-d , LN	concat 2×2, 384-d , LN
		[win. sz. 7×7, dim 192, head 6] × 2	[win. sz. 7×7, dim 192, head 6] × 2	[win. sz. 7×7, dim 256, head 8] × 2	[win. sz. 7×7, dim 384, head 12] × 2
stage 3	16× (14×14)	concat 2×2, 384-d , LN	concat 2×2, 384-d , LN	concat 2×2, 512-d , LN	concat 2×2, 768-d , LN
		[win. sz. 7×7, dim 384, head 12] × 6	[win. sz. 7×7, dim 384, head 12] × 18	[win. sz. 7×7, dim 512, head 16] × 18	[win. sz. 7×7, dim 768, head 24] × 18
stage 4	32× (7×7)	concat 2×2, 768-d , LN	concat 2×2, 768-d , LN	concat 2×2, 1024-d , LN	concat 2×2, 1536-d , LN
		[win. sz. 7×7, dim 768, head 24] × 2	[win. sz. 7×7, dim 768, head 24] × 2	[win. sz. 7×7, dim 1024, head 32] × 2	[win. sz. 7×7, dim 1536, head 48] × 2

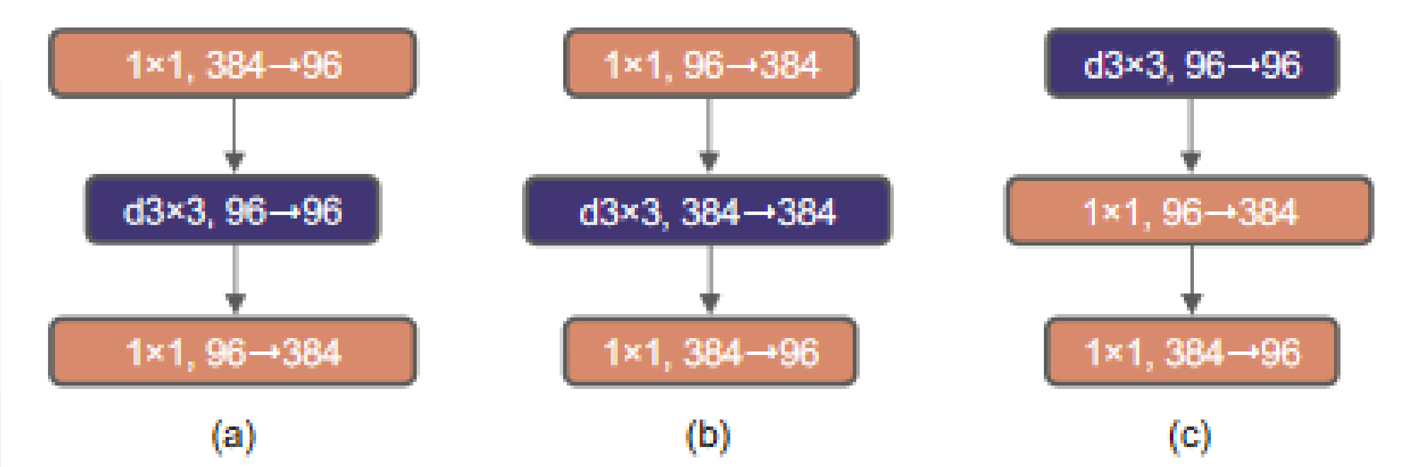
Table 7. Detailed architecture specifications.

ConvNext

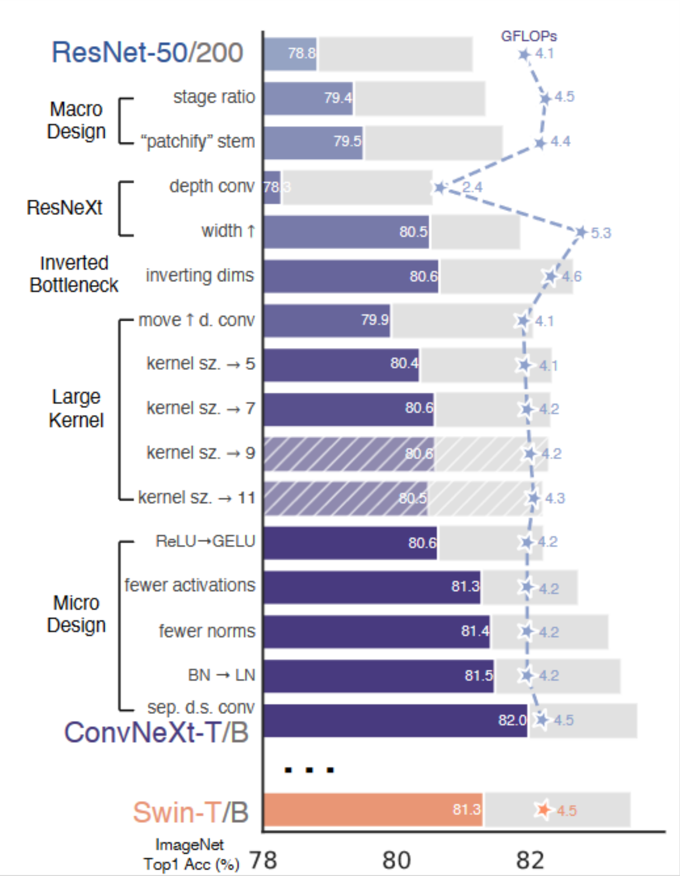


ResNet-50/200
GFLOPs
78.8
4.1
Macro Design
stage ratio
79.4
4.5
"patchify" stem
79.5
4.4
ResNeXt
depth conv
78.3
2.4
width ↑
80.5
5.3
Inverted Bottleneck
inverting dims
80.6
4.6
Large Kernel
move ↑ d. conv
79.9
4.1
kernel sz. → 5
80.4
4.1
kernel sz. → 7
80.6
4.2
kernel sz. → 9
80.6
4.2
kernel sz. → 11
80.5
4.3
Micro Design
ReLU→GELU
80.6
4.2
fewer activations
81.3
4.2
fewer norms
81.4
4.2
BN → LN
81.5
4.2
sep. d.s. conv
82.0
4.5
ConvNeXt-T/B
Swin-T/B
81.3
4.5
ImageNet Top1 Acc (%)
78
80
82

Inverted Bottleneck



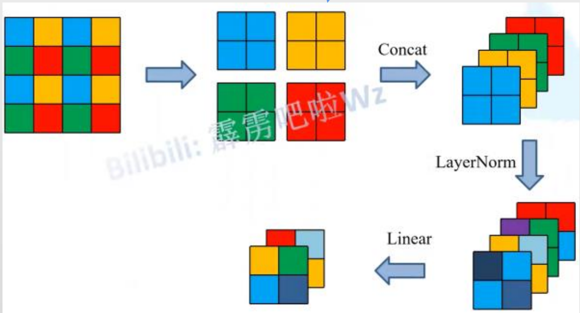
ConvNext

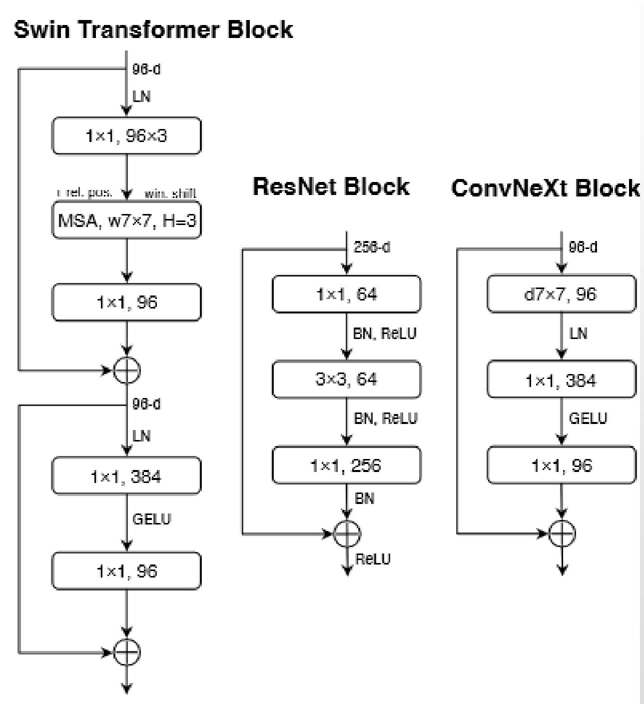


ResNet-50/200
Macro Design
stage ratio
"patchify" stem
ResNeXt
depth conv
width ↑
Inverted Bottleneck
inverting dims
Large Kernel
move ↑ d. conv
kernel sz. → 5
kernel sz. → 7
kernel sz. → 9
kernel sz. → 11
Micro Design
ReLU→GELU
fewer activations
fewer norms
BN → LN
sep. d.s. conv
ConvNeXt-T/B
Swin-T/B
GFLOPs
78.8
79.4
79.5
78.3
80.5
80.6
79.9
80.4
80.6
80.6
80.5
80.6
81.3
81.4
81.5
82.0
81.3
4.1
4.5
4.4
2.4
5.3
4.6
4.1
4.1
4.2
4.2
4.3
4.2
4.2
4.2
4.2
4.5
4.5
ImageNet Top1 Acc (%)
78
80
82

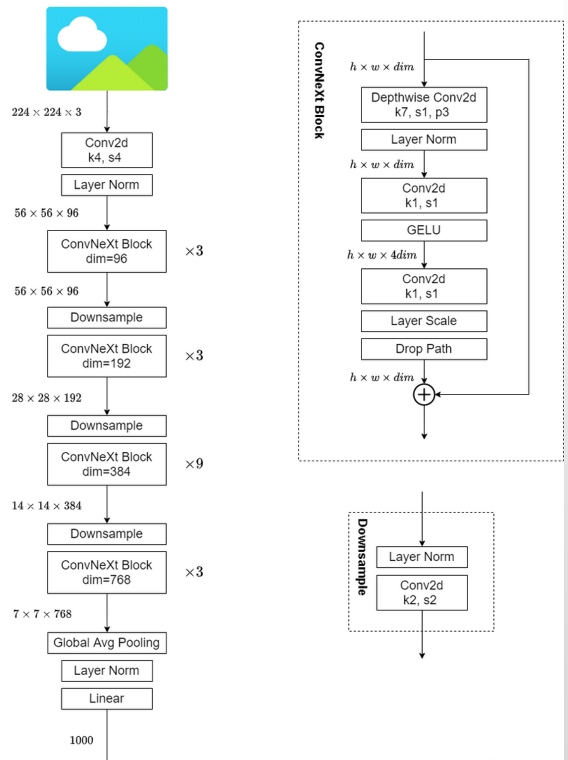
Micro Design

- Replacing ReLU with GELU
- Fewer activation functions
- Fewer normalization layers
- Substituting BN with LN
- Separate downsampling layers





Architecture



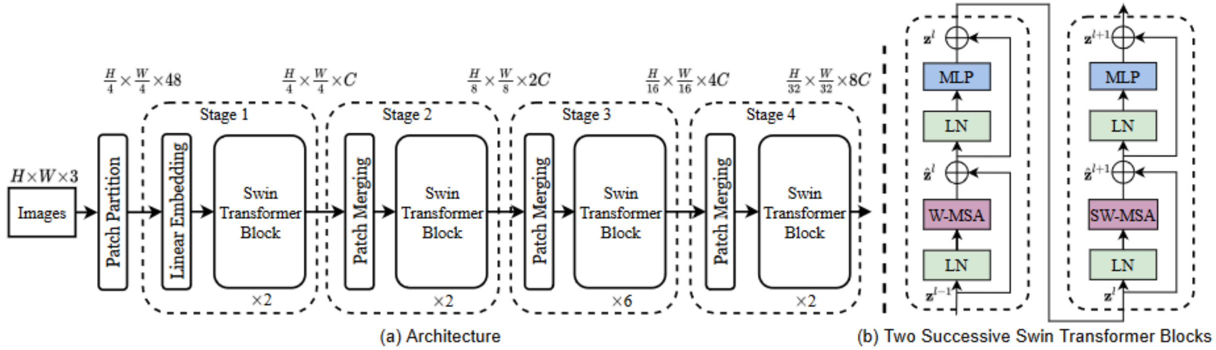
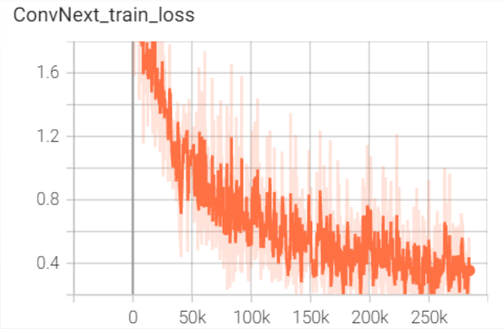
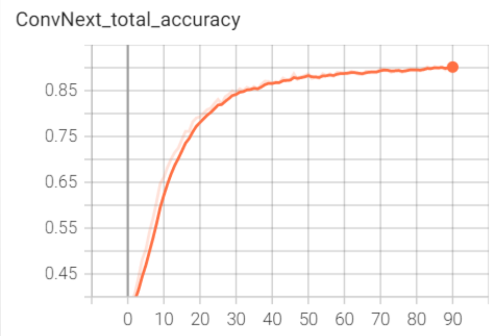


Figure 3. (a) The architecture of a Swin Transformer (Swin-T); (b) two successive Swin Transformer Blocks (notation presented with Eq. (3)). W-MSA and SW-MSA are multi-head self attention modules with regular and shifted windowing configurations, respectively.

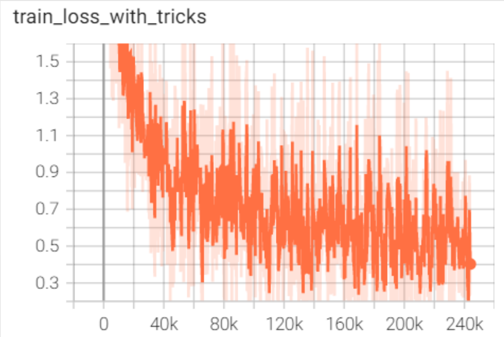
Experiments

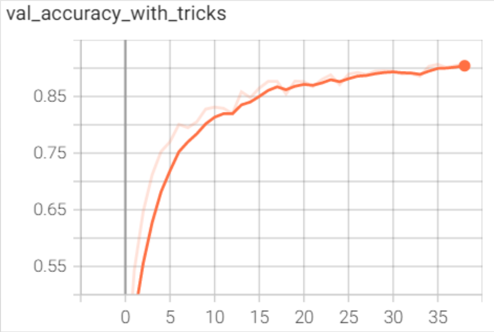
Training with Adam lr=0.0001 on CIFAR10 without augmentation



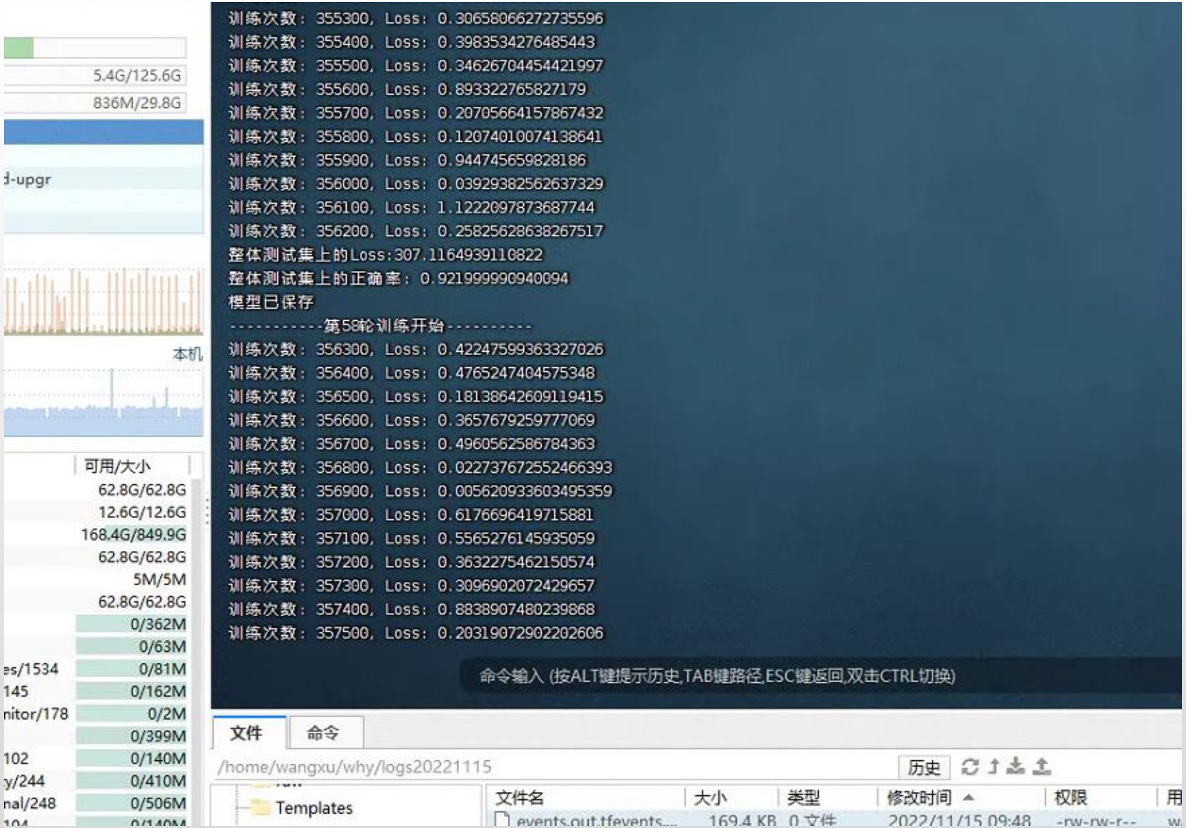


Training with some tricks(AdamW ;cos lr) on CIFAR10 without augmentation





Experiments



Init of distributed training

```python
# 初始化各进程环境
init_distributed_mode(args=args)

rank = args.rank
device = torch.device(args.device)
batch_size = args.batch_size
weights_path = args.weights
args.lr *= args.world_size  # 学习率要根据并行GPU的数量进行倍增
```

```python
def init_distributed_mode(args):
    if 'RANK' in os.environ and 'WORLD_SIZE' in os.environ:
        args.rank = int(os.environ["RANK"])
        args.world_size = int(os.environ['WORLD_SIZE'])
        args.gpu = int(os.environ['LOCAL_RANK'])
    elif 'SLURM_PROCID' in os.environ:
        args.rank = int(os.environ['SLURM_PROCID'])
        args.gpu = args.rank % torch.cuda.device_count()
    else:
        print('Not using distributed mode')
        args.distributed = False
        return

    args.distributed = True

    torch.cuda.set_device(args.gpu)
    args.dist_backend = 'nccl'  # 通信后端，nvidia GPU推荐使用NCCL
    print('| distributed init (rank {}): {}'.format(
        args.rank, args.dist_url), flush=True)
    dist.init_process_group(backend=args.dist_backend, init_method=args.dist_url,
                            world_size=args.world_size, rank=args.rank)
    dist.barrier()
```

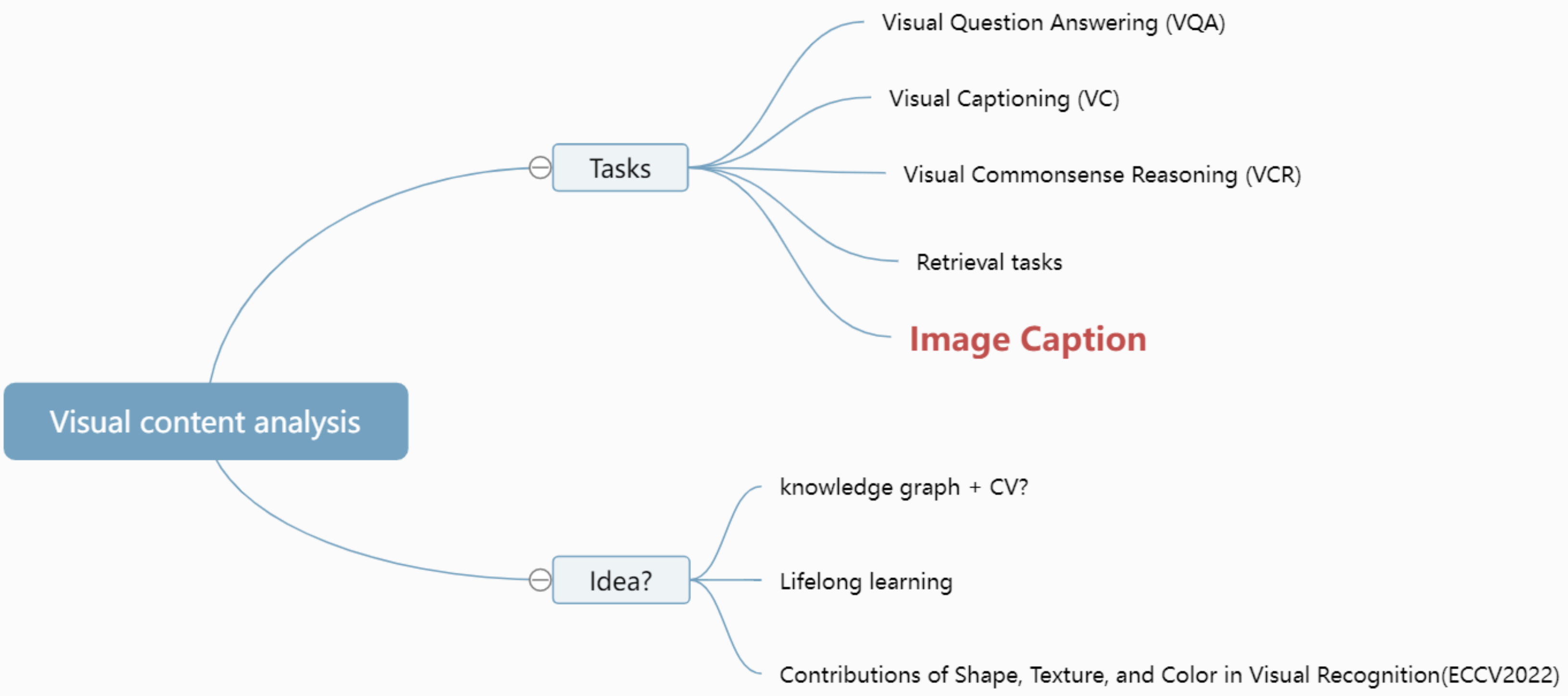
Update the model

```
# 给每个rank对应的进程分配训练的样本索引
train_sampler = torch.utils.data.distributed.DistributedSampler(train_data_set)
val_sampler = torch.utils.data.distributed.DistributedSampler(val_data_set)
```

```
# 转为DDP模型
model = torch.nn.parallel.DistributedDataParallel(model, device_ids=[args.gpu])
```

```
# 在进程0中打印训练进度
if is_main_process():
    data_loader = tqdm(data_loader, file=sys.stdout)

for step, data in enumerate(data_loader):
    images, labels = data

    pred = model(images.to(device))

    loss = loss_function(pred, labels.to(device))
    loss.backward()
    loss = reduce_value(loss, average=True)
    mean_loss = (mean_loss * step + loss.detach()) / (step + 1)  # update mean losses

    # 在进程0中打印平均loss
    if is_main_process():
        data_loader.desc = "[epoch {}] mean loss {}".format(epoch, round(mean_loss.item(), 3))
```

Future work



Visual content analysis
Tasks
Visual Question Answering (VQA)
Visual Captioning (VC)
Visual Commonsense Reasoning (VCR)
Retrieval tasks
Image Caption
Idea?
knowledge graph + CV?
Lifelong learning
Contributions of Shape, Texture, and Color in Visual Recognition(ECCV2022)

Thank you！