

Kaggle flower classification based on ResNet

홍남표

Contents

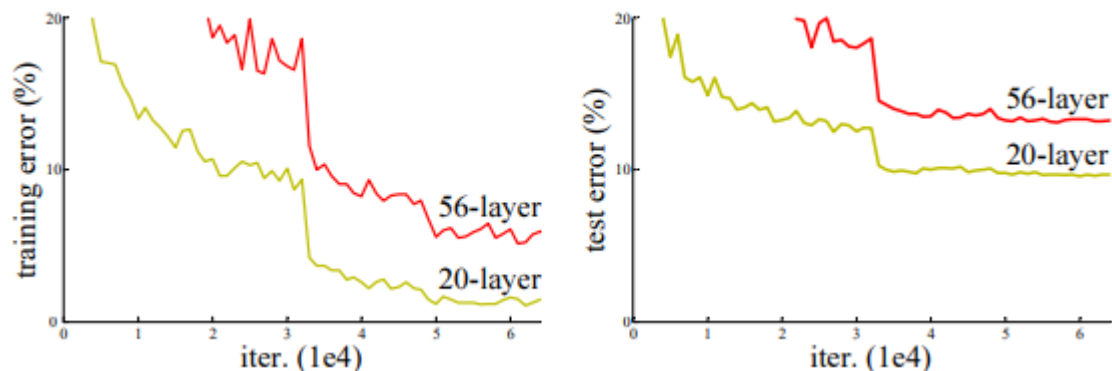
- 1 ResNet paper review
- 2 Kaggle flower classification dataset info
- 3 34-layer plain nets vs 34-layer ResNet
- 4 Basic block(34-layer ResNet) vs BottleNeck(50-layer ResNet)
- 5 Train from scratch vs Transfer learning from ImageNet
- 6 TODO

1

ResNet paper review (Deep Residual Learning for Image Recognition)

Introduction

AlexNet, VGGNet 등 ILSVRC competition(image classification) winner 모델들은 Convolution layer 를 깊게 쌓음으로써 분류 성능을 향상시킴

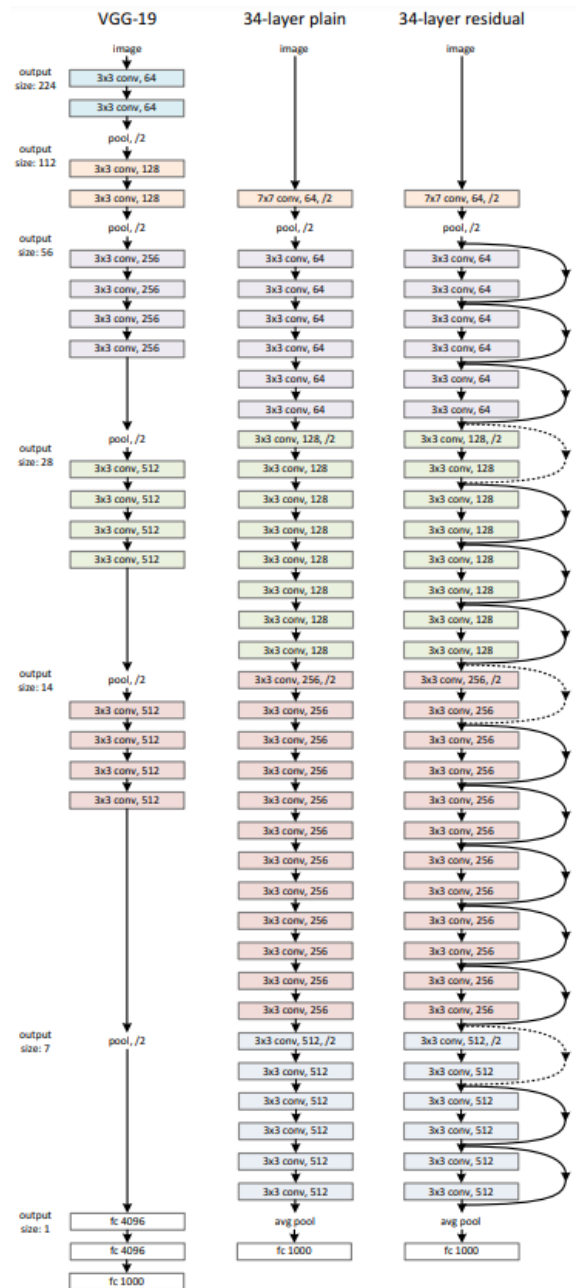


실제로는 conv layer를 깊게 쌓을수록 여러 문제가 발생함

1. vanishing/exploding gradients 로 인해 학습 초기부터 convergence 방해
 - normalized initialization, intermediate normalization layers(BatchNorm layers) 로 해결
2. deeper network 가 수렴을 시작할 때, degradation 문제가 발생함
 - network depth가 증가할수록, accuracy가 포화상태에 이르고, 빠르게 감소하기 시작
 - overfitting 의 문제는 아님(adding more layer -> higher training error)

-> degradation problem 을 해결하기 위한 deep residual learning framework 제안

Deep Residual Learning – Baseline Network Architectures

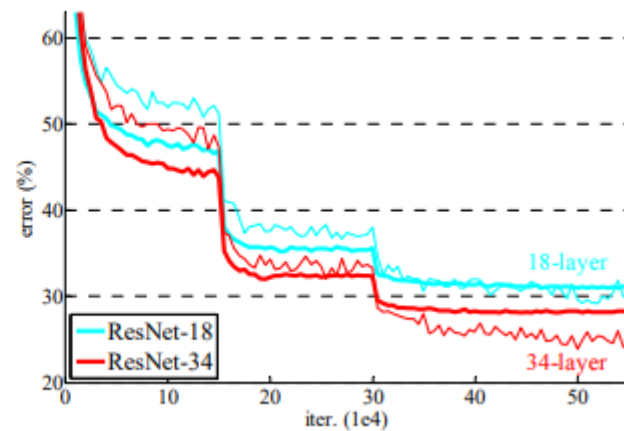
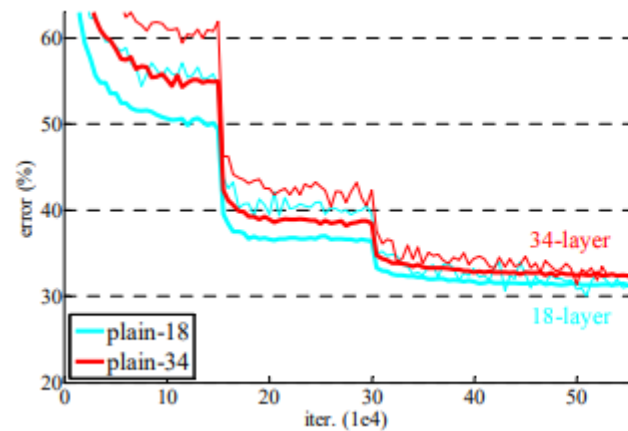


Plain Network	<ul style="list-style-type: none"> - VGG net 에 많은 영감을 받음 (3x3 크기의 convolution filter 를 거의 모든 레이어에서 사용) - 두가지 Design Rules <ol style="list-style-type: none"> 1. 같은 output feature map 크기에 대해서는, 같은 수의 filter를 사용한다 2. feature map 크기가 절반으로 줄어든 경우, filter 수는 2배로 증가 - Downsampling stride : 2 - 마지막 fc layer 전에 global average pooling - 1000-way(ImageNet) fully-connected layer + softmax - Weight layer 의 총 개수 : 34
Residual Network	<ul style="list-style-type: none"> - Plain network 를 기반으로, shortcut connection을 추가함 - input feature dim == output feature dim 일 경우, identity shortcuts 사용 - output feature 의 dimension 이 증가할 경우 2가지 옵션이 고려됨 <ol style="list-style-type: none"> 1. 증가한 dimension 에 대한 extra padding 을 수행한 다음 identity mapping 2. 1x1 convolution 을 이용한 projection shortcut - 두 옵션 모두 feature map size 는 2배로 증가하고, stride 2로 진행함

Deep Residual Learning – Train Details

Categories	Details
Preprocessing	<ul style="list-style-type: none">- 이미지는 단축 기준 [256, 480] 크기로 randomly sampled 되는 scale augmentation 수행- 224x224 size 로 random crop- random horizontal flip- per-pixel mean subtracted- standard color augmentation
Regularization	<ul style="list-style-type: none">- 각 convolution layer 뒤, activation layer 앞에 batch normalization 적용- Dropout 은 사용하지 않음
Initialization	<ul style="list-style-type: none">- He initialization 적용- 모든 plain, residual nets 는 scratch 에서 학습 진행
Optimizer	<ul style="list-style-type: none">- Stochastic gradient descent, mini-batch size 256- learning rate : 0.1 부터 시작해서, validation error 가 감소하지 않으면 10으로 나눠 줌- weight decay 0.0001, momentum 0.9
Dataset	<ul style="list-style-type: none">- ImageNet 2012 classification dataset- 1000 classes, 1.28M train images, 50k validation images, 100k test images
Evaluation Metrics	<ul style="list-style-type: none">- Top-1 error rates- Top-5 error rates

Experiments – ImageNet Classification – Residual Networks

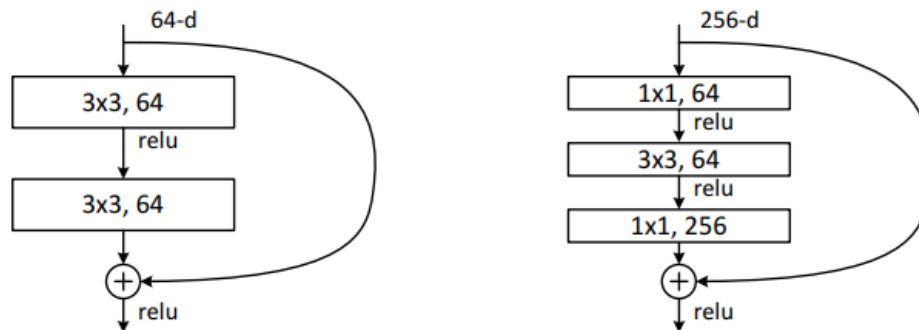


PlainNet	ResNet
<p>34-layer plain 모델의 학습 에러율이 18-layer plain 모델의 학습 에러율보다 더 높음</p>	<p>34-layer ResNet 모델의 학습 에러율이 18-layer ResNet 모델의 학습 에러율보다 더 낮음</p> <ol style="list-style-type: none"> 1. Degradation 문제 해결 : 상당히 낮은 training error 율을 보이고, validation data 에 대해서도 generalizable 2. 18-layer ResNet 도 18-layer plain net 보다 빨리 수렴(converge)한다. ResNet은 early stage 에서 optimizer 의 빠른 수렴을 도와줌

Experiments – ImageNet Classification – Identity vs. Projection Shortcuts

Shortcut Connection Options	increasing dimensions	other shortcuts
option 1	zero-padding	identity shortcut connections
option 2	projection shortcuts	identity shortcut connections
option 3	모든 shortcut connections 에 대해서 projection 사용	

Experiments – ImageNet Classification – Deeper Bottleneck Architectures



- Bottleneck block : 34-layer ResNet의 block 에서, 2번째 3x3 convolution을 빼고, 처음과 끝에 1x1 convolution layer 추가함
 - 1x1 layer 는 dimension 의 감소, 증가에 모두 대응할 수 있음
- time complexity 는 비슷함
 - 34-layer ResNet : 3.6×10^9 FLOPs
 - 50-layer ResNet : 3.8×10^9 FLOPs
- parameter-free identity shortcut 사용 (shortcut connection 의 option 2)
- 152-layer ResNet의 경우에도 VGG-16/19 nets 보다 낮은 complexity 를 가짐
- 6개의 서로 다른 depth 를 가진 ResNet 모델을 ensemble, ImageNet testset 에 대해서 3.57% top-5 error 달성

2

Kaggle flower classification dataset info

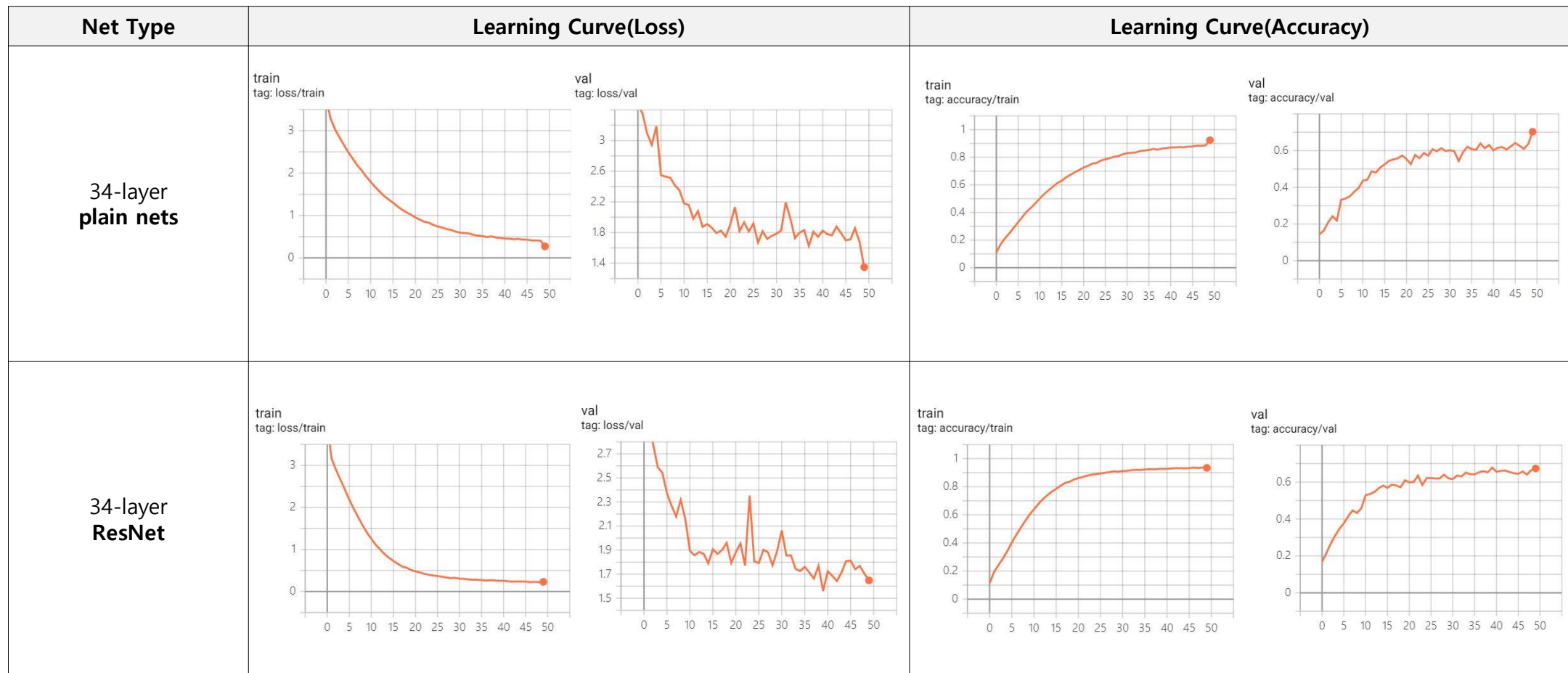
Dataset information

Categories	Details
# classes	104 classes
# images	51012 train images, 14848 validation images
size of images	192x192, 224x224, 331x331, 512x512
pixel mean (train dataset)	[0.485, 0.456, 0.406]
pixel std (train dataset)	[0.229, 0.224, 0.225]
image augmentation (train)	<ol style="list-style-type: none">1. The image is resized with its shorter side randomly sampled in [256, 480] for scale augmentation2. 224x224 crop is randomly sampled from an image3. Random horizontal flip4. Per-pixel mean subtracted(normalize) : For pretrained resnet50, ImageNet mean, std are required For scratch training for plain34, resnet34, resnet50, flower dataset mean, std are required5. Standard color augmentation is used : Not apply color augmentation, I thought color is an important feature for classification of flowers
image augmentation (val)	<ol style="list-style-type: none">1. Centor crop image, size 224x2242. Per-pixel mean subtracted(normalize)

3

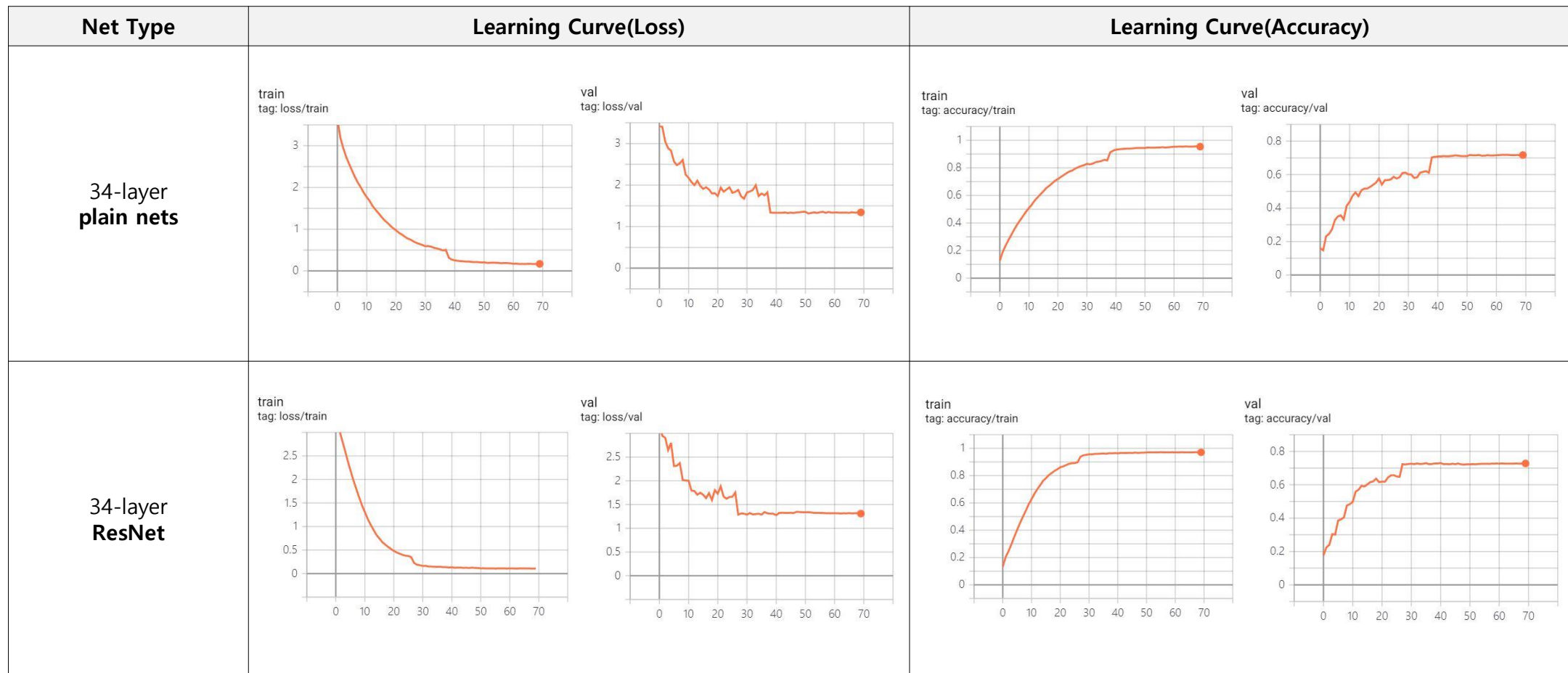
34-layer plain nets vs 34-layer ResNet

Learning curve – 34-layer plain nets vs 34-layer ResNet



- **LR scheduler** : 10 epochs 동안 validation loss 가 min 값을 찍지 못하면 lr /= 10, 전체 50 epochs 학습
- **Convergence rate analysis** : loss, accuracy 모두 ResNet 에서 더 빠르게 수렴
- **Loss/accuracy analysis** : plain-net 의 경우 (27, 37, 47) epoch 에서 3번의 learning rate 감소 (0.1-> 0.01 -> 0.001 -> 0.0001), ResNet의 경우 (33, 49) epoch 에서 2번의 learning rate 감소, 최종적으로 50 epoch 에서 더 높은 accuracy 를 달성(plain 0.7, resnet 0.68)
- Validation loss를 기준으로 한 patience = 10 인 ReduceLROnPlateau scheduler를 사용했는데, 전체적으로 학습 epoch을 늘리고, patience 값을 더 낮게 조정해야 할 필요성을 느꼈습니다.

Learning curve – 34-layer plain nets vs 34-layer ResNet (more training epochs, less patience for lr scheduler)

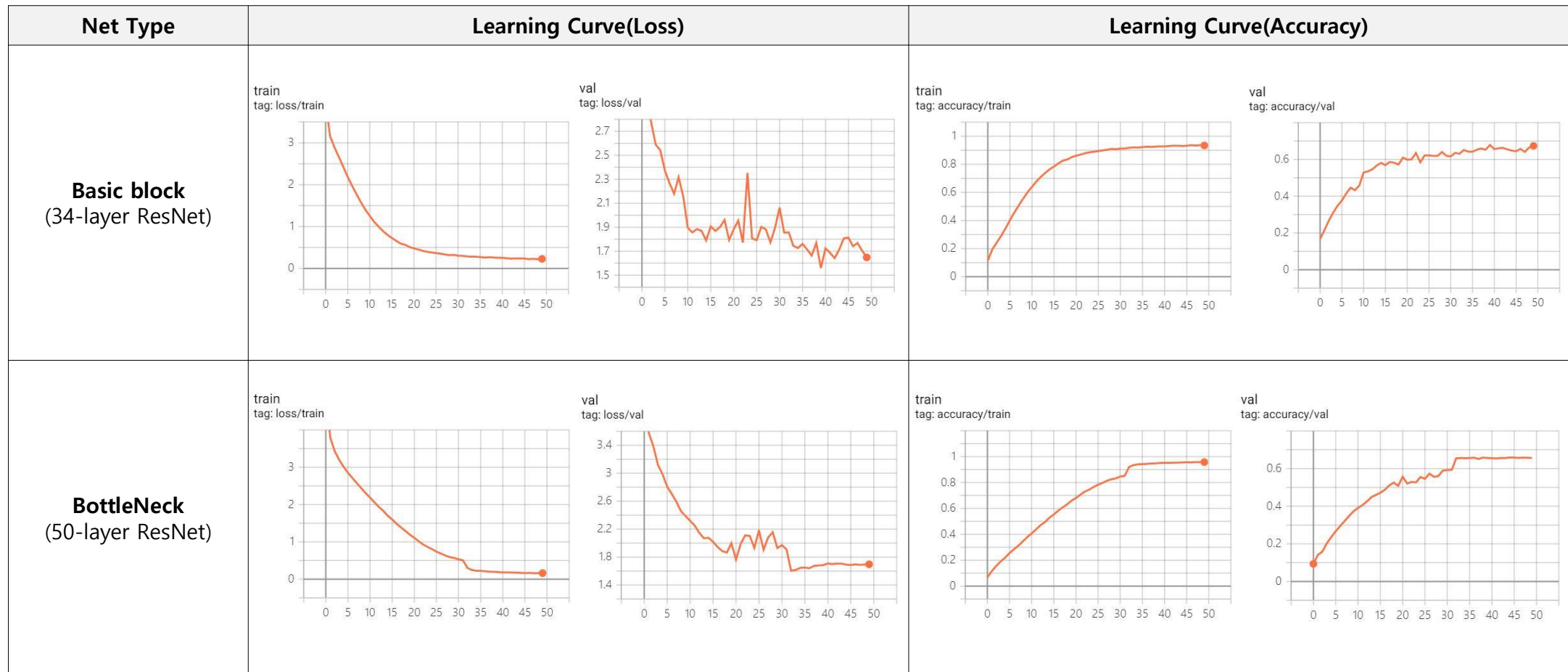


- **LR scheduler** : 7 epochs 동안 validation loss 가 min 값을 찍지 못하면 lr /= 10, 전체 70 epochs 학습
- **Convergence rate analysis** : ResNet이 plain net보다 10epoch 정도 더 빠르게 수렴
- **Loss/accuracy analysis** : ResNet의 최종 accuracy 가 plain net의 최종 accuracy 보다 근소하게 높음(ResNet 72%, PlainNet 71%)
- 성능도 좋아지고 무엇보다도 모델 자체가 훨씬 빠르게 수렴되기 때문에, Flower Dataset의 학습에 있어서 residual connection 을 사용하는 것이 좋다는 결론을 내렸습니다.

4

Basic block(34-layer ResNet) vs
BottleNeck(50-layer ResNet)

Learning curve – Basic block(34-layer ResNet) vs BottleNeck(50-layer ResNet)



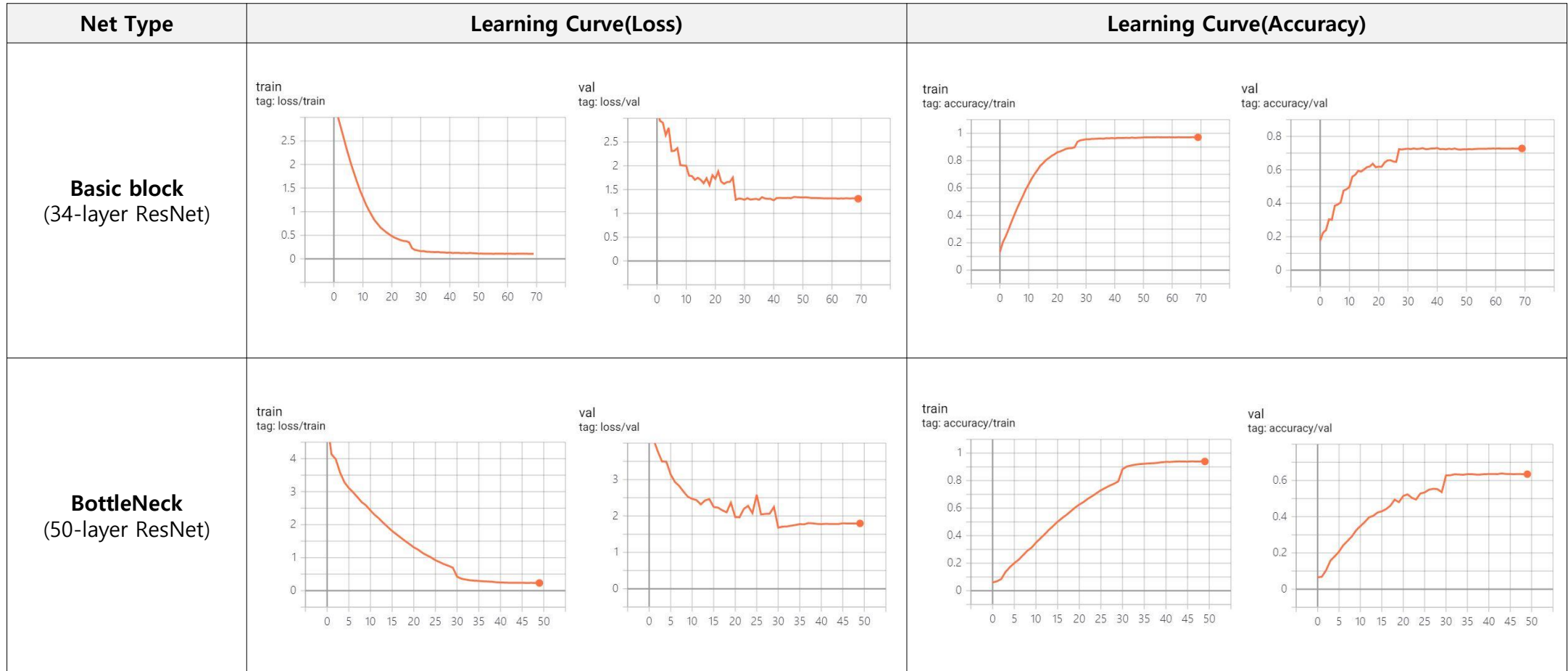
- **LR scheduler** : 10 epochs 동안 validation loss 가 min 값을 찍지 못하면 lr /= 10, 전체 50 epochs 학습
- **Convergence rate analysis** : ResNet block architecture 는 Basic Block을 사용했을 때, 더 빠른 수렴을 보임
- **Loss/accuracy analysis** : BottleNeck에서 Validation loss 가 튀지 않고 안정적으로 수렴했고, 두 모델 전부 learning rate 가 2번 감소해서 epoch 50 에서는 최종 0.001의 learning rate 로 학습됨, 50epoch 에서 최종 accuracy 는 basic block이 더 높음
- 두 모델 전부 최종적으로 수렴이 덜 된 상태에서 학습을 종료했기 때문에, **34-layer plain nets vs 34-layer ResNet** task 와 마찬가지로 전체 적으로 학습 epoch을 늘리고, patience 값을 더 낮춰서 학습을 진행했습니다.

Learning curve – Basic block vs BottleNeck (more training epochs, less patience for lr scheduler, trial 1)

Net Type	Learning Curve(Loss)	Learning Curve(Accuracy)
Basic block (34-layer ResNet)		
BottleNeck (50-layer ResNet)		

- **LR scheduler** : 7 epochs 동안 validation loss 가 min 값을 찍지 못하면 lr /= 10, 전체 70 epochs 학습
- **Convergence rate analysis** : ResNet50모델의 validation loss가 17epoch 때 최소값이었는데 24epoch까지 감소하지 않고, 높은 상태에서 learning rate 가 감소해서 model weight가 local minima에 빠짐
- 1) 현 LR Scheduler(ReduceLROnPlateau)의 patience 를 더 늘리고 학습 epoch을 늘림
- 2) 다른 LR Scheduler 를 사용하거나 학습이 정상적으로 된 checkpoint 에서 시작해서 다시 시작
- 1) 을 하기에는 시간이 부족하고, 2)를 적용하기에는 epoch 10 부터 validation loss 가 크게 튀어서 처음부터 그대로 진행했습니다.

Learning curve – Basic block vs BottleNeck (more training epochs, less patience for lr scheduler, trial 2)

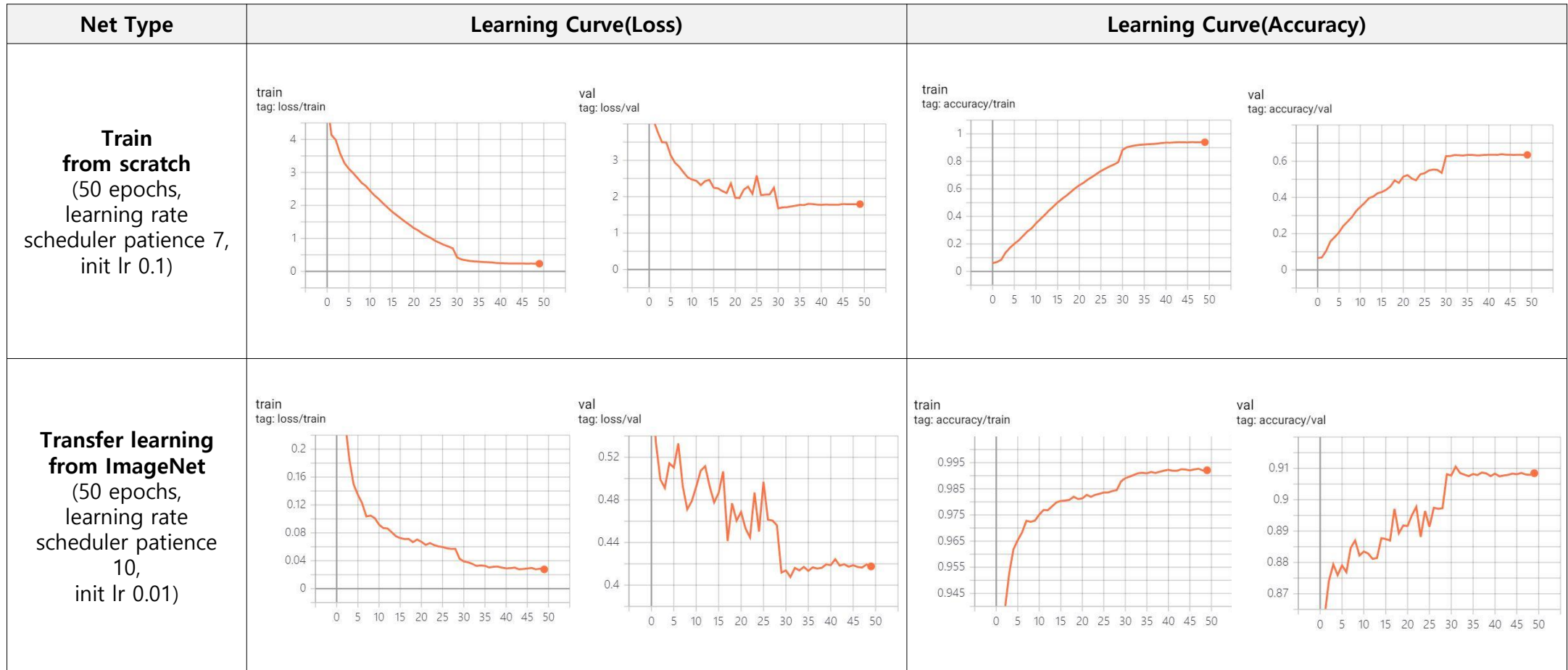


- **Convergence rate analysis** : Basic Block에서 3 epochs 정도 더 빠르게 validation loss가 수렴, 두 모델 전부 30Epoch 이전에 수렴이 되어서 50-layer ResNet 모델은 50Epoch 까지만 학습 진행
- **Loss/accuracy analysis** : 34-layer ResNet의 validation accuracy 가 50-layer ResNet의 validation accuracy 보다 약 7~8% 정도 높음
- Kaggle Flower Dataset의 scratch training 에서는 BottleNeck보다 BasicBlock을 적용하는 것이 좋다는 결론을 내렸습니다.
(Tensor initialization, dataloader sampler 등 여러 요인에 의해 accuracy 가 달라질 수 있어서, 추가적인 검증 필요)

5

Train from scratch vs Transfer learning from ImageNet

Learning curve – Train from scratch vs Transfer learning from ImageNet



- **Implementation** : Transfer learning model에서는 마지막 fc-layer를 제외한 나머지 backbone에 ImageNet Pre-trained weight를 사용
- **Convergence rate analysis** : transfer learning의 경우에는 28epoch에서 learning rate의 감소가 일어났고 31epoch에서 완전히 수렴함
- **Loss/accuracy analysis** : transfer learning model이 학습 시작부터 학습 종료까지 50Epochs 전부 loss, accuracy가 좋았음
- Google의 BiT(Big Transfer), ViT(Visual Transformer)와 같이 최대한 많은 이미지로 pre-train하고(JFT-300M dataset), pre-trained weight에서 특정 dataset, task로의 transfer learning이 강력한 성능을 발휘한다는 것을 보여준 바 있습니다. 강력한 성능의 모델을 위해서는 large-scale supervised pre-training이 중요하다는 것을 다시 한번 경험하는 계기가 되었습니다.

6

TODO

TODO

Categories	Details
Hyper parameters	<ul style="list-style-type: none"> - 처음에는 train epochs 를 50으로 설정했지만, 실제 ResNet paper 에서는 600,000 iterations 까지 학습을 진행함 - We use SGD with a mini-batch size of 256. The learning rate starts from 0.1 and is divided by 10 when the error plateaus, and the models are trained for up to 60×10^4 iterations - 600,000 batch x 256 images/batch x (1/128,000) epoch/images = 120 epochs - Train epochs 를 50에서 원래 논문처럼 120 Epochs 까지 학습하면 더 뚜렷한 convergence rate 를 확인할 수 있을 것으로 예상됨 -> ImageNet보다 데이터셋 사이즈가 작아서 실제로는 70Epoch 의 학습도 충분했음 - LR scheduler 의 patience 값 감소(validation loss 가 진전이 없는 상태에서 더 빠른 convergence를 유도) - 실제로 LR scheduler의 patience 값을 10에서 7로 바꾸고, 학습을 70 Epoch 까지 해서 모델 수렴 결과를 볼 수 있었음
Preprocess	<ul style="list-style-type: none"> - Color transform 사용 여부에 따른 ablation study - Preprocess tensor normalization 에서 train, validation dataset 의 pixel mean, std를 한번에 구하고, cross validation 적용
Test	<ul style="list-style-type: none"> - Test dataset 에 대한 testing process 구현 - 10-crop testing (horizontal flips x FiveCrop) 적용 - 마지막 fc layer 대신 convolution layer를 사용한 Fully-convolution form 을 통한 multiple scales size {224, 256, 384, 480, 640} 에서 테스트
Ensemble	<ul style="list-style-type: none"> - plain34, resnet34, resnet50, pre-trained resnet50 각 모델의 classification 결과의 vote 기반 ensemble 을 통한 성능 향상
Code Refactoring	<ul style="list-style-type: none"> - Tensorboard 에서 epoch 에 따른 loss 를 바로 확인할 수 있도록 SummaryWriter hook 에 추가 - Validation loss 가 min 값일 때만 Checkpoint 저장하도록 변경 - Hyper parameter, Image preprocessing pipeline 에 대한 configuration file 작성 및 연동 - Kaggle submission format에 따른 test result csv builder script 작성 - 특정 checkpoint 부터 이어서 학습하는 script 작성 필요