CIFAR10 실험 보고서

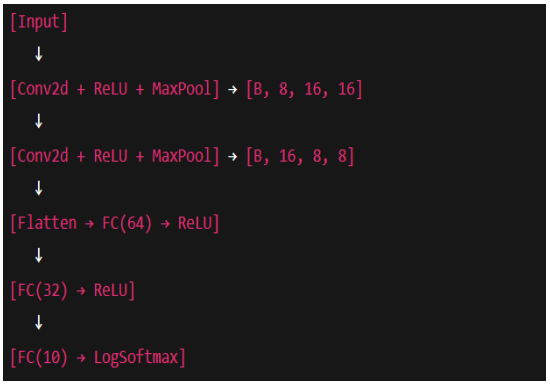
목차

Dataset-CIFAR10 Overview

- CIFAR10은 총 10개의 클래스로 구성된 컬러 이미지 분류 데이터셋이며, 각 클래스당 6000장의 32x32 크기 이미지를 포함
- 총 60000장의 이미지 중, 50000장은 학습용, 10000장은 테스트용으로 사용됨.
- 클래스는 비행기, 자동차, 새, 고양이, 사슴, 개, 개구리, 말, 배, 트럭으로 구성됨

Model Structure

초기 Baseline 모델의 구조와 성능이다.(B는 배치 사이즈)



Epoch	Train Loss	Train Accuracy (%)	Test Loss	Test Accuracy (%)
1	1.8166	33.35	1.5755	43.72
2	1.4928	45.85	1.4445	48.02
3	1.3373	51.70	1.2646	55.02
4	1.2375	55.46	1.2256	56.49
5	1.1745	58.35	1.2163	56.85
6	1.1297	59.97	1.1497	59.62
7	1.0880	61.57	1.1039	60.97
8	1.0564	62.85	1.1060	60.79
9	1.0263	63.84	1.0989	61.54
10	1.0025	64.67	1.0834	62.07

실험 내용

VGG-Inspired CNN Model 모델 설계 및 성능 향상 실험

실험 목적
기본 CNN 모델(위 baseline 모델)은 비교적 단순한 Conv-Pool 구조를 따르며 **CIFAR10**과 같은 복잡한 이미지 분류 작업에는 한계가 존재한다. 이에 따라, 대표적인 고성능 모델인 **VGGNet**의 설계 원리를 활용하여 더 깊은 구조와 작은 커널을 겹쳐 사용하는 방식으로 모델을 재구성 하였다.

VGGNet은 3x3 커널을 여러번 겹쳐 사용함으로써 적은 수의 파라미터로도 5x5, 7x7과 유사한 수용 영역(receptive field)를 확보할 수 있는 특징을 갖고 있다. 이러한 구조적 이점을 활용하면 **CIFAR-10**처럼 해상도가 낮지만 시각적 다양성이 높은 데이터셋에서도 효과적인 특징 추출이 가능할 것이라고 판단하였다.

실험 개요-모델 구조
VGGNet의 구조는 원래 224x224에 최적화되어 있으며, 다수의 Conv층과 5개의 MaxPooling이 포함되어 있다. CIFAR10은 32x32의 상대적으로 작은 이미지이므로, 이에 마자게 구조를 축소하였다. 전체 구조는 다음과 같이 두 개의 블록으로 구성된다.

- Block1 : Conv(32)-Conv(32)-Conv(32)-MaxPool
- Block2 : Conv(64)-Conv(64)-Conv(64)-MaxPool

이후 출력은 **[B, 64, 8, 8]**이 되며, 이를 Flatten하여 **[B, 4096]**의 벡터로 변환한 뒤 다음과 같은 FC Layer를 적용하였다.

- Linear(4096 > 128) > ReLU
- Linear(128 > 32) > ReLU
- Linear(32 > 10) > Softmax

실험 결과 및 성능 비교

모델	**Train Loss**	**Train Acc (%)**	**Test Loss**	**Test Acc (%)**
Baseline	1.0025	64.67	1.0834	62.07
VGG-Inspired	0.1171	96.07	1.8573	68.94

- 단순한 Conv-Pool 구조였던 BaseLine 모델에 비해, VGG 스타일의 깊은 네트워크는 Train Acc에서 매우 높은 성능(96%)을 기록하며 복잡한 특징 학습에 탁월한 효과를 보였다. 하지만 Test Acc가 68%에 그치며, Overfitting 문제가 발생하였다.

VGG-Inspired CNN Model 모델 설계 및 성능 향상 실험 - Dropout, BatchNorm

실험 목적
Overfitting은 훈련 데이터에 과하게 적응해, 새로운 데이터에 대한 일반화 성능이 저하되는 대표적인 문제이다. 이를 해결하기 위해 본 실험에서는 대표적인 Regularization 기법인 **Dropout**과 **BatchNorm**을 비교하여 일반화 성능 향상에 어떤 방식이 더 효과적인지 실험적으로 검증하고자 하였다.

실험 개요**-**모델 구조
기존 VGG-Inspired 모델에서 **Dropout** 실험의 경우, fc1과 fc2 Layer 사이에 dropout을 적용하였으며,

BatchNorm의 경우, 모든 주요 층 뒤에 삽입하여 학습의 안정성과 일반화 성능을 향상시키고자 하였다.
- 모든 Conv Layer 뒤에 BatchNorm 삽입
- Fully Connected Layer 뒤에도 BatchNorm 추가

실험 결과 및 성능 비교

모델	**Train Loss**	**Train Acc (%)**	**Test Loss**	**Test Acc (%)**
VGG-Inspired	0.1171	96.07	1.8573	68.94
VGG + BatchNorm	0.0436	98.51	0.9117	81.53

- **Dropout**의 경우 위 표에는 작성이 되어있지 않지만, Train Loss(0.4225), Train Acc(85.07), Test Loss(0.8818), Test Acc(73.20%)의 성능이 나왔으며,
- **BatchNorm**을 적용했을 때, 더 높은 성능이 나옴을 확인할 수 있다.
- **BatchNorm**은 각 Conv Layer 이후에 적용되어 내부 공변량 변화(Internal Covariate Shift)를 줄이고, 안정적인 학습과 빠른 수렴을 유도하였다.

실험 결과, **Dropout**을 적용한 모델은 약간의 일반화 성능 향상을 보였지만**,** 여전히 과적합 문제는 완전히 해소되지 않았다.(Dropout을 여러 층에 적용해보는 것도 좋은 방향일 것 같다.) 반면, **BatchNorm**을 적용한 모델은 **Test Accuracy**가 약 **81.5%**로 크게 향상되었고, Loss 값 또한 크게 감소하며 학습의 안정성과 효율성을 함께 개선하는 모습을 보였다.

BatchNorm은 각 배치마다 feature의 분포를 정규화(Normalization)함으로써, 학습 도중 발생할 수 있는 입력 분포의 변화를 줄여준다. 이는 학습률에 덜 민감하고, 더 빠른 수렴을 유도하며, 특히 깊은 네트워크에서 매우 유리하게 작용한다.
또한, Dropout은 뉴런을 확률적으로 제거하기 때문에 정보 손실이 발생할 수 있다. 이는 네트워크가 복잡한 특징을 학습하는 데 있어 방해가 될 수 있으며, 특히 작은 크기의 이미지(CIFAR10)에서는 정보 손실이 성능저하로 이어질 수 있다. 반면 Batch Norm은 입력 정보를 보존하면서도 정규화만 수행하기 때문에 보다 효율적으로 판단된다.
Dropout은 주로 FCLayer에서 효과를 발휘하지만, Convolution Layer에서는 적용 범위가 제한적이며 오히려 성능을 저하시킬 수 있다. 반면 BatchNorm은 Convolution Layer에 직접 적용되어 각 채널 단위로 정규화를 수행하며 네트워크 전반에 걸쳐 안정적인 학습을 지원한다.

VGG-Inspired CNN Model 모델 설계 및 성능 향상 실험 - FC Layer 확장

실험 목적
FC Layer의 크기를 확장하여, 모델의 학습 용량을 증가시키고자 하였다. 이는 보다 풍부한 처리로 모델의 분류 결정 능력 강화 및 더 정밀한 결정 경계를 학습할 수 있어 성능향상이 될 것이라고 판단하였다.

실험 개요**-**모델 구조
기존: 4096 → 128 → 32 → 10
확장: 4096 → 1024 → 128 → 10

실험 결과 및 성능 비교

모델	**Train Loss**	**Train Acc (%)**	**Test Loss**	**Test Acc (%)**
VGG + BatchNorm	0.0436	98.51	0.9117	81.53
VGG + BatchNorm + FC 확대	0.0994	96.53	0.7543	81.60

실험 결과, **Test Loss** 측면에서 의미 있는 개선**(Loss : 0.9117 → 0.7543)**을 보였다. FC 구조의 확장은 성능의 급격한 향상은 아니더라도, 신뢰도 높은 분류 성능 개선에 효과적임을 확인하였고, 이후 모든 실험에서 해당 확장 구조를 기본 FC 구성으로 채택하였다.

VGG-Inspired CNN Model 모델 설계 및 성능 향상 실험 - Data Augmentation

실험 목적
CIFAR10과 같은 시각적 다양성이 큰 데이터셋에서, 다양한 시각 변형이 모델에 강인하게 대응할 것이라고 판단하였다. 본 실험에서는 대표적인 증각 기법인 **RandomCrop(padding=4)**과 **RandomHorizontalFlip**을 적용해 학습 데이터의 시각적 다양성을 인위적으로 확장하고자 하였다. 이를 통해 모델이 위치나 방향에 민감하지 않고 보다 일반화된 특징을 학습할 수 있는지를 검증하고자 하였다.

추가적으로 **CollorJitter(brightness** 및 **saturation** 조합**)** 기법을 통해 색상 기반 변형에 대한 모델의 대응력도 실험적으로 분석하였다. 이는 CIFAR10의 일부 클래스에서 색상 정보가 주요 특징으로 작용하는 점을 고려한 것이다.

실험 개요**-**모델 구조
이전 실험과 동일한 모델을 사용하였다.

실험 결과 및 성능 비교

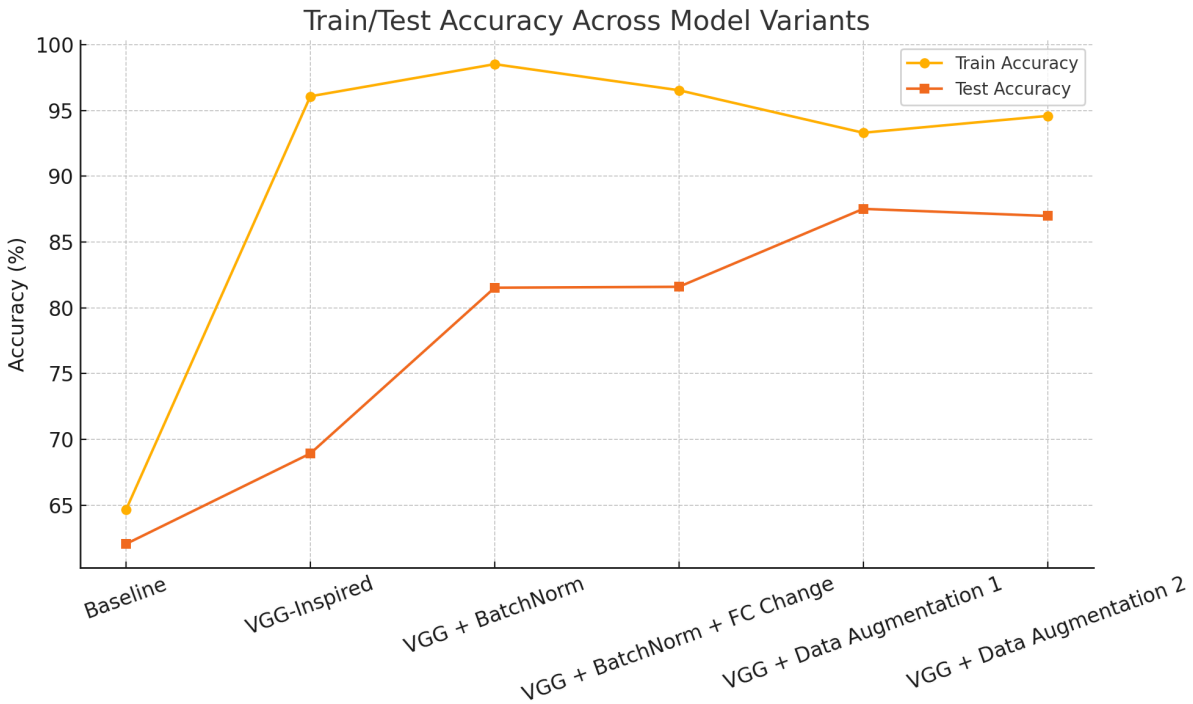
모델	**Train Loss**	**Train Acc (%)**	**Test Loss**	**Test Acc (%)**
VGG + BatchNorm + FC 확대	0.0994	96.53	0.7543	81.60
VGG + Data Augmentation (Crop+Flip)	0.1960	93.30	0.4279	87.52

모델	**Train Loss**	**Train Acc (%)**	**Test Loss**	**Test Acc (%)**
VGG + Data Augmentation (Crop+Flip)	0.1960	93.30	0.4279	87.52
VGG + Crop+Flip + ColorJitter	0.1613	94.58	0.4943	86.98

- **RandomCrop + Flip** : Test Acc는 **87.52%**로 가장 높은 성능을 기록하였다. 이는 모델이 다양한 위치와 방향의 이미지에 대해 보다 강인하게 일반화되었음을 보여주는 결과로, 데이터 증강이 효과적인 정규화 수단임을 입증한다.
- **ColorJitter** 추가 실험 : Test Acc가 소폭 감소하였다. 이는 색상 변형(무분별한 색상 왜곡)이 오히려 원래의 시각적 패턴을 흐리게 만들어, 일부 클래스에서 특징 분리를 어렵게 만들었을 가능성을 시사한다.

전체 성능 비교 표

모델	**Train Loss**	**Train Acc (%)**	**Test Loss**	**Test Acc (%)**
Baseline	1.0025	64.67	1.0834	62.07
VGG-Inspired	0.1171	96.07	1.8573	68.94
VGG + BatchNorm	0.0436	98.51	0.9117	81.53
VGG + BatchNorm + FC 확대	0.0994	96.53	0.7543	81.60
VGG + Data Augmentation (Crop+Flip)	0.1960	93.30	0.4279	87.52
VGG + Crop+Flip + ColorJitter	0.1613	94.58	0.4943	86.98



Curriculum Learning 실험

실험 목적

Curriculum Learning은 인간이나 동물의 학습 방식에서 영감을 받은 전략으로, 모델이 쉬운 예제부터 점진적으로 어려운 예제를 학습함으로써 효율적인 학습과 더 나은 일반화 성능을 달성하는 것을 목표로한다.

이 전략은 특히 비선형 딥러닝 모델에서 더 나은 지역 최적점**(Local minima)**을 찾는 데 유리하며, 경사하강법 최적화 경로를 안정화시키는 데도 도움이 되는 것으로 알려져 있다.(Bengio et al., 2009).

본 실험에서는 Curriculum Learning의 원리를 CIFAR10 분류 문제에 적용하여, 쉬운 데이터 **->** 어려운 데이터 순으로 학습시키는 전략이 실제 성능 향상에 도움이 되는지를 두 가지 다른 접근 방식으로 검증하였다.

기본 모델 구조
이전 실험에서 진행한 Conv-Conv-Pool Conv-Conv-Conv-pool 구조 그대로 사용(VGG-Inspired + BatchNorm + FC 확대, 위 내용 참고)

Curriculum Learning 실험 1 : 시각 난이도 기반

학습 설계
인간의 시각적 인식 흐름을 참고하여 학습 난이도를 다음과 같이 단계적으로 구성:

- **Stage1** : 객체의 윤곽선만 추출한 데이터셋(검은 배경 + 흰 선)
- **Stage2** : 객체의 단순 색을 입힌 데이터 셋
- **Stage3** : 배경을 추가하되 약간 블러처리된 이미지
- **Stage4** : 원본 이미지 데이터셋

실험 목적
사람처럼 시각적으로 단순한 특징(윤곽선)부터 학습함으로써, 점진적으로 복잡한 시각적 요소(색, 배경 등)을 일반화할 수 있는지 확인

실험 결과

Stage	Epoch	Train Loss	Train Accuracy	Test Loss	Test Accuracy
Edge Only	1	1.3370	53.01%	2.5058	28.25%
	2	1.0561	63.29%	2.3582	29.54%
	3	0.8968	68.69%	2.9093	21.63%
	4	0.7427	74.15%	3.2549	21.50%
	5	0.5684	80.38%	3.3003	23.79%
Filled Object Only	1	0.5013	82.39%	6.0361	19.87%
	2	0.2759	90.54%	7.2122	19.65%

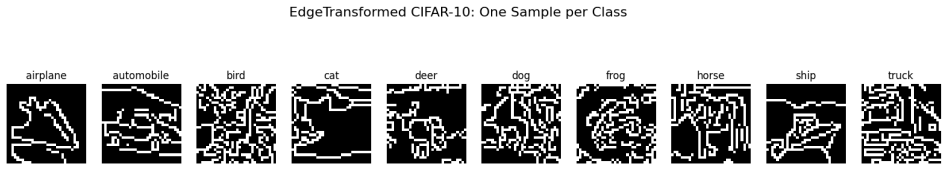
Stage	Epoch	Train Loss	Train Accuracy	Test Loss	Test Accuracy
	3	0.1738	94.06%	9.9876	17.05%
	4	0.1352	95.32%	14.3340	17.70%
Color + Background Blurred	1	0.6774	77.60%	3.1420	51.50%
	2	0.2814	90.33%	4.7278	43.17%
	3	0.1485	95.02%	3.1110	56.92%
	4	0.1073	96.30%	3.8780	53.84%
	5	0.0879	96.96%	4.4326	51.91%
	6	0.0810	97.21%	4.0953	56.13%
Original	1	0.2245	92.29%	0.8621	77.26%
	2	0.0633	97.98%	0.9569	77.15%
	3	0.0445	98.50%	1.0603	77.28%
	4	0.0542	98.15%	1.0988	76.91%

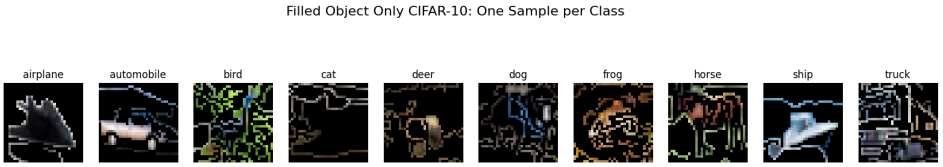
- 최종 Test Accuracy : 약 77%
- 기대했던 성능보다 낮았으며, 커리큘럼이 오히려 학습을 방해한 것으로 보임

실패 요인 분석

- **Stage1~3**의 데이터셋 품질 저하 : 직접 생성한 윤곽선 데이터에서 객체 인식이 불완전함. 객체 외곽이 명확히 잡히지 않거나, 배경까지 함께 추출되는 오류 발생
- 해상도 문제 : CIFAR10은 32x32의 매우 작은 이미지로 윤곽선이나 색상 분리 작업에서 정보손실이 발생
- “쉬운 데이터”라기보다 일관성 없는 데이터로 학습이 시작되어 모델이 오히려 일반화에 실패했을 가능성이 존재한다고 판단

실제 **Stage**별 데이터셋 예시





Curriculum Learning 실험 2 : Confidence 기반 데이터 난이도 정렬

학습 설계

- pretrained 모델로 예측한 Confidence값을 기준으로 학습 데이터를 정렬
  - 각 이미지마다 Softmax 출력에서 가장 큰 확률 값을 confidence score로 사용
  - 50000개의 confidence 값이 생성됨
  - 확률 높은 순서대로 정렬 > 상위일수록 예측에 자신있는 데이터
- 정규화 및 데이터 증강 구성:

Stage	데이터 난이도	**Confidence Score** 기준	이미지 증강 전략	목적
1	쉬움	상위 30%	원본 (No Augmentation)	기본 개념 형성
2	중간	중간 40%	Crop + Flip	일반화 성능 확보
3	어려움	하위 30%	Crop + Flip	난이도 높은 데이터 대응

실험 목적

사람처럼 쉬운 예제부터 익숙해지는 흐름을 그대로 구성하되, 난이도를 “모델 입장”에서 정의를 하여 테스트를 하면 성능이 향상될 것이라고 생각. 쉬운 예제를 먼저 학습시켜 모델 파라미터의 초기화를 더 나은 방향으로 유도

실험 결과

Stage	Epoch	Train Loss	Train Acc	Test Loss	Test Acc
1	6	0.3347	88.51	0.9864	69.66
2	10	0.4949	82.81	0.5557	81.6
3	9	0.3227	89.03	0.5474	81.58

- Stage3까지 학습 후 Test Accuracy : 81.58%

Full Model Fine-Tuning 결과

Stage	Epoch	Train Loss	Train Acc	Test Loss	Test Acc
1	6	0.3347	88.51	0.9864	69.66
2	10	0.4949	82.81	0.5557	81.6
3	9	0.3227	89.03	0.5474	81.58
4 (Fine-tune)	9	0.3266	88.58	0.384	87.51

Fine-tuning으로 성능이 소폭 향상됨을 확인 가능
- 초기에 쉬운 예제로 파라미터가 잘 초기화된 상태에서 전체 데이터셋을 다시 학습 > 모델이 더 넓은 분포에 일반화됨
- 각 스테이지에서 학습한 편향된 분포를 원래의 전체 분포로 보정하는 효과가 있음
- 기존 커리큘럼에서 보지 못한 샘플을 다시 학습하면서 성능이 미세 조정되고 보완됨으로 판단

실험 방식	최종 **Test Accuracy**
커리큘럼 1 (윤곽선 → 색 → 배경)	77% (약)
커리큘럼 2 (confidence 기반)	87.58%

- 모델 입장에서 진짜 **'**쉬운**'** 데이터가 무엇인지 판단하게 하고, 이를 기반으로 커리큘럼을 설계하는 confidence 전략이 더 높은 성능과 안정성을 제공한다.
- 커리큘럼 학습의 핵심인 난이도 정렬의 객관성과 점진적 학습 구조가 잘 지켜졌을 때, 모델의 일반화 능력이 크게 향상됨을 보여준다.

이전 실험 결과와 비교**(Shuffle vs Curriculum)**

모델 구성	**Train Loss**	**Train Accuracy (%)**	**Test Loss**	**Test Accuracy (%)**
Shuffle (VGG + Data Augmentation (Crop+Flip))	0.1960	93.30	**0.4279**	87.52
Curriculum Learning	0.3266	88.58	**0.3840**	87.51

- Train 데이터 성능에서는 확실히 Shuffle 모델이 우세한 것으로 판단되나, Test 데이터에 대해서는 정확도는 비슷하나, Loss값이 Curriculum 학습이 우세한 것으로 판단됨.
- 즉, 정답/오답 비율은 같지만, 정답에 도달하는 "확신의 정도"가 다르다는 의미. Curriculum Learning을 한 모델이 정답에 대한 확신이 더 크게 나타남.

Residual Module

실험 목적
이번 실험에서는 ResNet 기반의 핵심 구조인 **Residual Module**을 적용하여 CIFAR-10 분류 성능을 평가하였다.
특히 학습 방식으로는 두 가지를 비교하였다.

- Shuffle Training : 일반적인 데이터 셔플 기반 학습
- Curriculum Learning(Confidence 기반) : 모델이 예측하기 쉬운 샘플 부터 점진적 학습

두 방식 모두 동일한 ResidualCNN 구조를 사용하였고, 성능의 차이를 통해 데이터 구성 방식과 학습 흐름이 모델 일반화에 미치는 영향을 분석하고자 하였다.

모델 개요
Input: 3 x 32 x 32
▼ Convolution Block 1

ResidualBlock(3 → 32)
ResidualBlock(32 → 32)
MaxPool(2x2) → Output: 32 x 16 x 16

▼ Convolution Block 2

ResidualBlock(32 → 64)
ResidualBlock(64 → 64)
ResidualBlock(64 → 64)
MaxPool(2x2) → Output: 64 x 8 x 8

▼ Fully Connected Block

Flatten → 8 x 8 x 64 = 4096
Linear(4096 → 1024) + BatchNorm + ReLU
Linear(1024 → 128) + BatchNorm + ReLU
Linear(128 → 10)

▼ Output
LogSoftmax (for 10-class classification)

실험 결과
실험 결과는 Shuffle Training과 Curriculum Learning 두 가지 학습 방식에서 각각 잔차 모듈의 적용 유무에 따라 성능 차이가 뚜렷하게 나타남을 보여준다.

1. Shuffle Training 비교

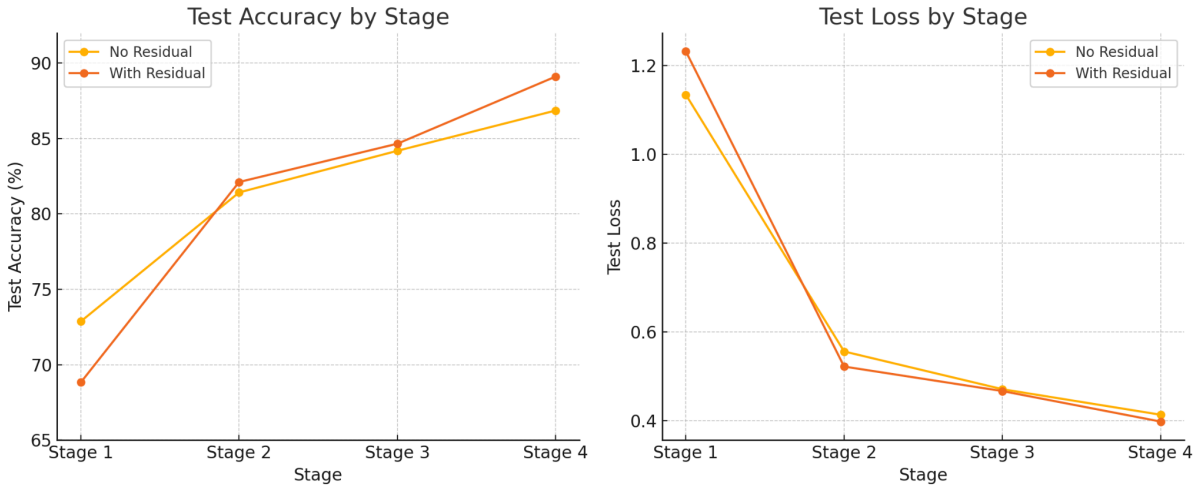
Method	Epoch	Train Acc (%)	Test Acc (%)
Shuffle + Residual Module	15	95.55	88.80
Shuffle Only	11	93.30	87.52

- 같은 조건에서 학습한 경우, 잔차 모듈을 적용한 모델이 더 빠르게 수렴하고, 최종 정확도 역시 약 **1.3%** 향상되었다.
- 이는 Residual Block이 입력 정보 손실 없이 deeper한 모델 구성이 가능하게 하며, 더 풍부한 표현 학습이 가능했음을 의미한다.

2. Curriculum Learning 비교

Method	**Epoch**	**Train Acc (%)**	**Test Acc (%)**
No Residual	10	89.00	**86.85**
With Residual	10	95.45	**89.10**

- Curriculum(Confidence 기반 샘플 학습) 조건에서도, 잔차 모듈을 적용한 모델은 약 **2.25%**의 정확도 향상을 보였다.
- 특히 훈련 정확도가 더 높게 나왔음에도 불구하고 과적합 없이 일반화 정확도도 같이 상승했다는 점에서, 모델의 표현력과 안정성을 동시에 확보한 것으로 해석된다.



Schedular

실험 목적
이번 실험에서는 ReduceLROnPlateau 스케줄러를 도입하였다.
스케줄러 도입의 주요 목적은 다음과 같다.

“학습 중 성능이 정체되는 구간에서 학습률을 자동으로 조절함으로써, 모델이 더 정밀하게 학습되도록 유도하는 것”

딥러닝 학습에서는 일반적으로 학습 초반에 손실이 빠르게 줄어들지만, 중반 이후로는 성능이 정체되는 plateau 현상이 나타나기 쉽다. 이러한 구간에서 학습률이 너무 크면 모델이 최적점을 지나치거나 진동할 수 있고, 반대로 너무 작게 고정해두면 학습 시간이 불필요하게 늘어나거나 수렴하지 못할 수 있다.

따라서 일정 기준(예: validation/test loss)으로 성능 향상이 멈췄을 때, 학습률을 자동으로 줄이는 ReduceLROnPlateau 스케줄러는 효과적인 해결책이 될 수 있다

Shedular 적용
scheduler = torch.optim.lr_scheduler.ReduceLROnPlateau(optimizer, mode='min', factor=0.5, patience=3,)

파라미터	설정값	설명
optimizer	-	학습률을 조정할 옵티마이저 (Adam 등)
mode	'min'	loss가 감소해야 하는 지표일 때 사용. (validation loss 등)
factor	0.5	학습률을 줄이는 비율. 현재 학습률의 0.5배로 감소함

파라미터	설정값	설명
patience	3	3번 연속 개선이 없을 경우 학습률을 감소시킴

- 매 epoch마다 Scheduler.step(val_loss) 호출
- val_loss가 patience만큼 연속으로 감소하지 않으면, 현재 학습률 x factor 만큼 감소
- 예를 들어, 초기 학습률이 0.001이고, 3 epoch 동안 val_loss가 줄지 않았다면: 학습률이 0.001 x 0.5 = 0.0005 감소

실험 결과

학습 방식	최고 **Test Accuracy**	최고 **Epoch**	대응 **Loss**
Shuffle + Augmentation	**90.21%**	16	0.3821
Curriculum + Fine-Tuning	89.34%	10 (Stage 4)	0.3842

1. **Shuffle + Augmentation** 방식

학습 데이터가 무작위로 고르게 분포되어 있기 때문에 모든 난이도의 데이터가 동시에 반복적으로 학습됨. Residual 구조와 Data Augmentation(Crop+Flip) 덕분에 모델이 다양한 변형에 강해지고 빠르게 일반화 성능 확보 가능

사용한 스케줄러 **ReduceLROnPlateau**는 Test loss가 3번 이상 감소하지 않으면 학습률을 절반으로 줄임 -> 후반부 학습률이 낮아지면서 모델이 미세한 조정 단계로 넘아가면서 정확도 최대화

Epoch 11~16 사이에서 성능이 천천히 수렴하며 90%를 초과함

2. Curriculum Learning(Confidence 기반)

Stage 1: 쉬운 데이터부터 학습하여 빠르게 높은 정확도 도달(99% Train Acc)
Stage 2~3 : 중간 및 어려운 데이터를 점진적으로 학습하면서 모델 분포를 확장
Fine-Tune 단계 : 전체 데이터를 다시 학습하며 초기화된 파라미터 기반으로 미세조정. 이 시점에서 ReduceLROnPlateau가 적용되어 학습률이 감소하면서 정확도가 89.34%까지 향상

다만, Shuffle 대비 다양한 난이도가 반복적으로 제공되지 않아 최종 성능은 살짝 낮음

ReduceLROnPlateau 스케줄러의 역할

역할	적용 방식
성능 정체 구간 감지	`test_loss`가 `patience=3` 동안 줄지 않으면 감지
학습률 감소 **(factor=0.5)**	현재 학습률을 절반으로 줄여, **fine-tuning** 정밀도 향상
정확도 안정화에 기여	학습 후반부에 **loss** 진동 없이 수렴 유도

특히 Shuffle에서는 스케줄러가 후반부 학습률을 낮춰 폭발적 정확도 증가 없이 안정적 수렴을 만들고, Curriculum에서는 Fine-Tuning 단계에서 초기 좋은 파라미터를 미세하게 다듬어 높은 정확도를 달성하는 데 기여했다.

항목	분석 요약
Shuffle 방식 장점	다양한 데이터에 대해 빠르고 고르게 학습 → 최종 정확도 높음
Curriculum 방식 장점	파라미터 초기화를 유리하게 유도하여 학습 안정성 확보
공통점 (스케줄러 역할)	후반부 학습률 조절로 **loss** 안정화 및 과적합 방지
핵심 차이	Curriculum은 훈련 과정의 흐름에 집중, Shuffle은 데이터 다양성에 집중

(세부사항 및 코드)
GitHub Link : https://github.com/yeontachi/deep-learning-lab/tree/main