CNN 모델 보고서

제출일: 25.06.30

제출인: 서윤철

개요

- 서론
- 본론
- 결론

서론

내용

- CIFAR-10 데이터셋을 불러온 후 CNN 모델로 분류 문제 성능을 검토
- 하이퍼 파라미터의 여러 가능성을 점검해 보시고 최적의 설정을 찾아보기

목표

- CNN 모델 성능 확인하기
- 이전 실습했던 모델의 성능과 비교하기
- CNN 모델에서 하이퍼 파라미터의 여러 가능성을 점검해보고 최적의 설정 찾아보기

실습한 CNN 모델

- Layer

layer	목적
Conv2D	특징 추출 (로우레벨 → 하이레벨)
ReLU	비선형성 추가
MaxPool	차원 축소, 불변성
Flatten	CNN → MLP 연결용
Dense	분류 결정 로직 학습
Softmax	다중 클래스 확률 출력

- 구조

layer	특징	Output	역할
Input		(32, 32, 3)	
Conv2D	(32, (3, 3), relu')	(30, 30, 32)	경계, 모서리 등의 로우레벨 특징 추출
MaxPooling2D	pool_size=(2×2)	(15, 15, 32)	다운샘플링 (공간 차원 축소)
Conv2D	(64, (3×3), relu)	(13, 13, 64)	더 복잡한 특징 감지
MaxPooling2D	pool_size=(2×2)	(6, 6, 64)	정보 압축
Conv2D	(64, (3×3), relu)	(4, 4, 64)	하이레벨 추상적 특징 추출
Flatten		(4×4×64 = 1024)	Fully Connected 입력 변환
Dense	(64, relu)	(64,)	분류 의사 결정
Dense	(10, softmax)	(10,)	클래스별 확률 출력
Output		클래스별(10개) 확률	

- Optimizer

  RMSprop

- batch_size, learning_rate, epoch

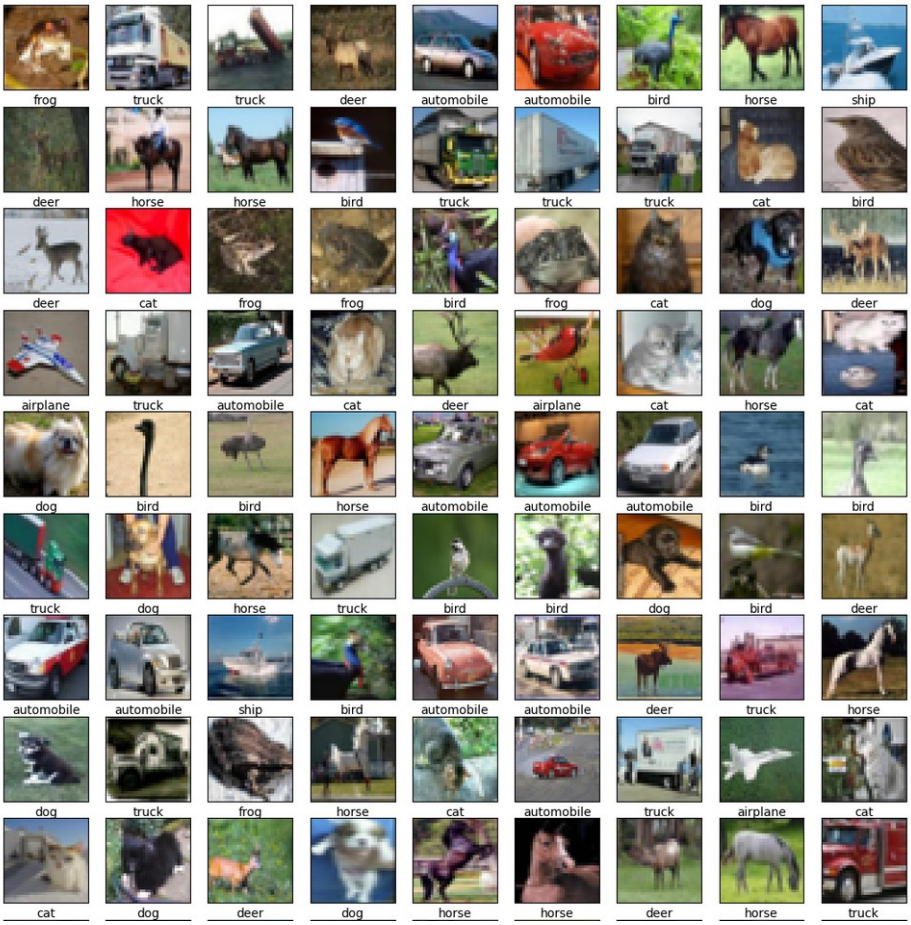
  20, 256, 0.2

본론

CNN 모델 성능 확인하기

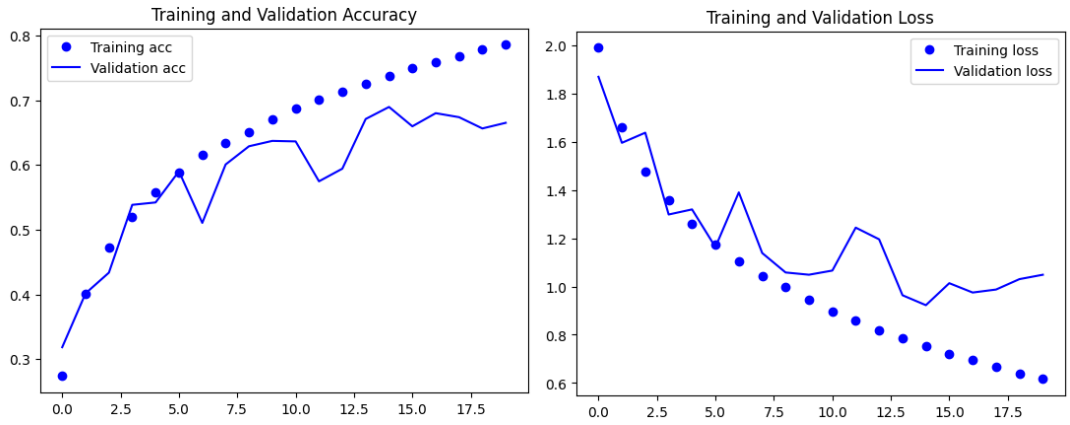
- CIFAR-10

  32×32 크기의 RGB 컬러 이미지 60,000장 (10개 클래스)



...

- 결과

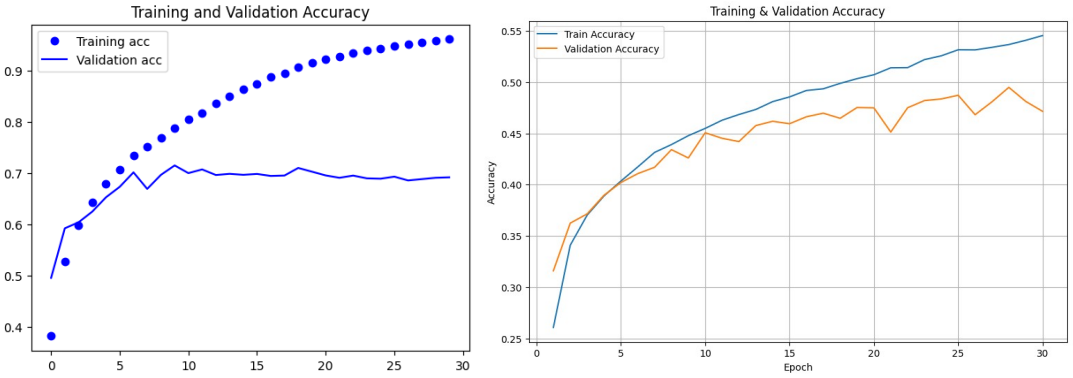


=> val_accuracy: 66.5 %, val_loss: 1.0490

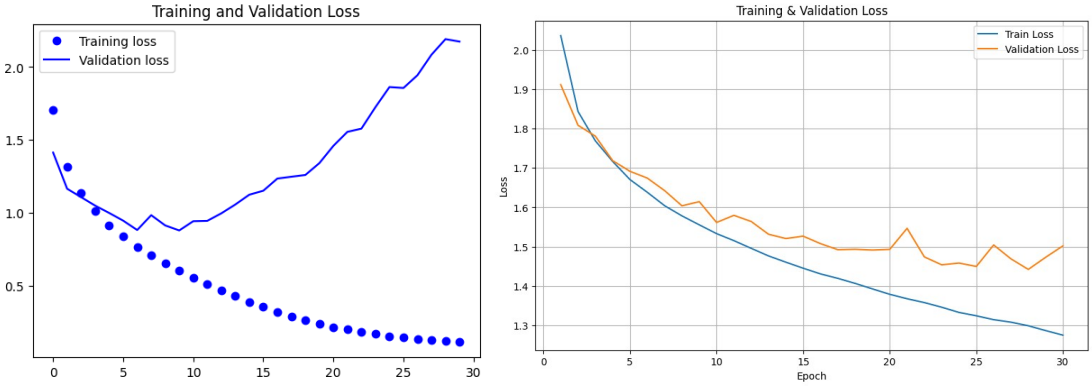
MLP와 성능 비교

파란색 라인: CNN, other: MLP

- 동일한 epoch(30), batch_size(64), validation(0.2)



val_accuracy: CNN(69.22%) > MLP(47.16%)



val_loss: CNN(2.1742) > MLP(1.5019)

: val_accuracy 측면에서 20%이상의 차이를 보인다. CIFAR-10을 처리하기 위해서 CNN이 MLP보다 더 적합함.

하이퍼 파라미터의 여러 가능성 점검

- 성능 향상 방법

범주	하이퍼파라미터	설명
모델 구조	Conv2D filter 수	필터 수를 늘리면 더 복잡한 패턴을 학습
	kernel_size	일반적으로 (3, 3), (5, 5) 실험
	layer 수	Conv 레이어 3~5개까지 실험
	Dense layer 수 및 units 수	64 대신 128, 256 등 실험
Regularization	Dropout	과적합 방지를 위해 Dropout 추가 (예: 0.3~0.5)
Optimizer	Adam, RMSprop, SGD 등	학습 속도, 수렴 여부에 영향
Learning rate	optimizer=Adam(learning_rate=0.001	학습률 변화가 성능에 큰 영향
Training	Epoch 수	20보다 더 늘려서 학습 가능
	Batch size	64, 128, 256 등 실험

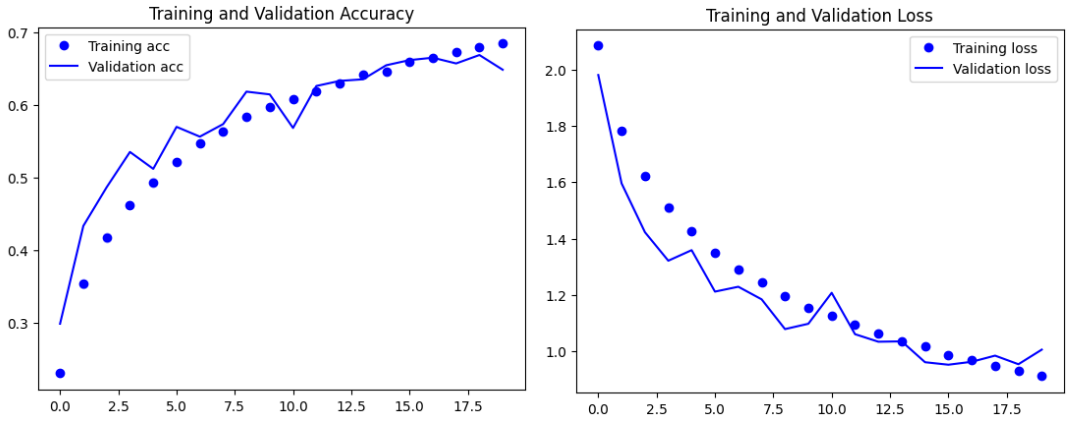
- 전략

1. Dropout 추가
2. 필터 수, 레이어 수 증가
3. Learning rate 튜닝
4. Epoch 늘려 시각화 성능 확인 with EarlyStopping(검증 손실이 더 이상 줄어들지 않으면 자동으로 학습 중단)

성능 개선 과정

- Dropout 2개 추가

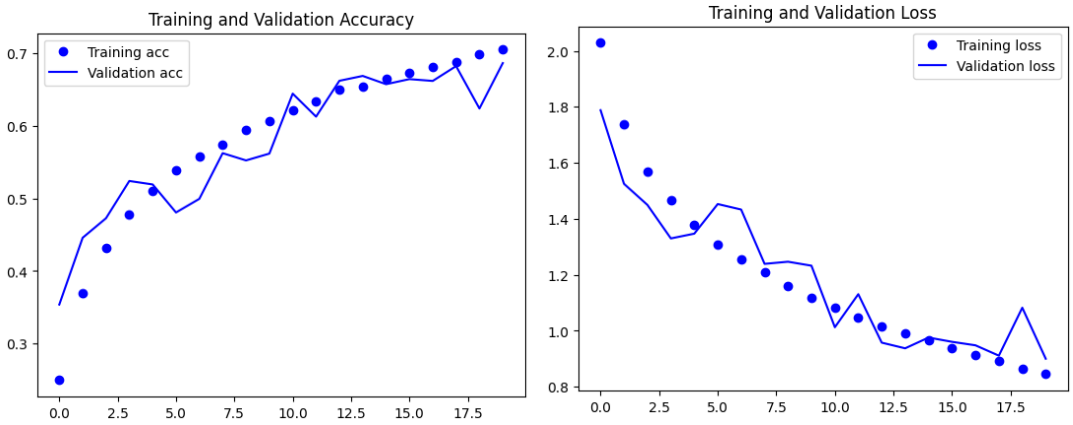
  위치: Conv2D 뒤(비율:0.3), Dense 뒤(비율: 0.5)



=> val_accuracy: 64.86 %, val_loss: 1.0060 오히려 val_accuracy 하락 미채택

- Dropout 2개 추가

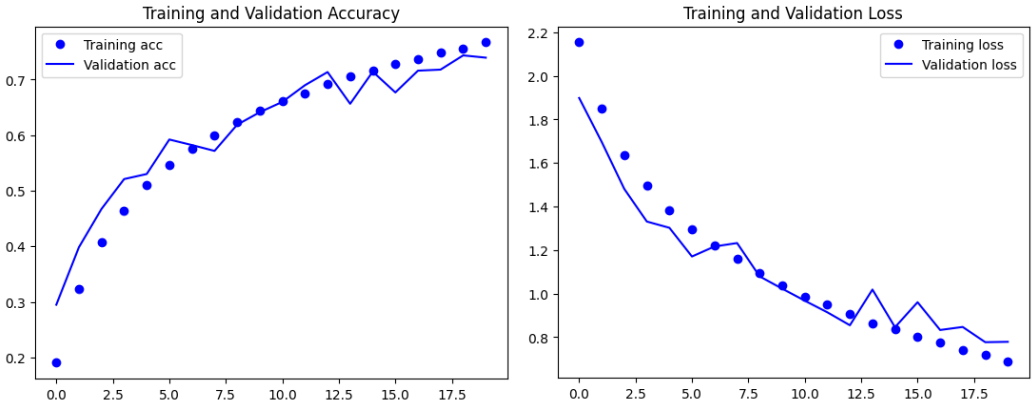
  위치: Conv2D 뒤(비율:0.2), Dense 뒤(비율: 0.4)



=> val_accuracy: 68.65 %, val_loss: 0.9003 채택

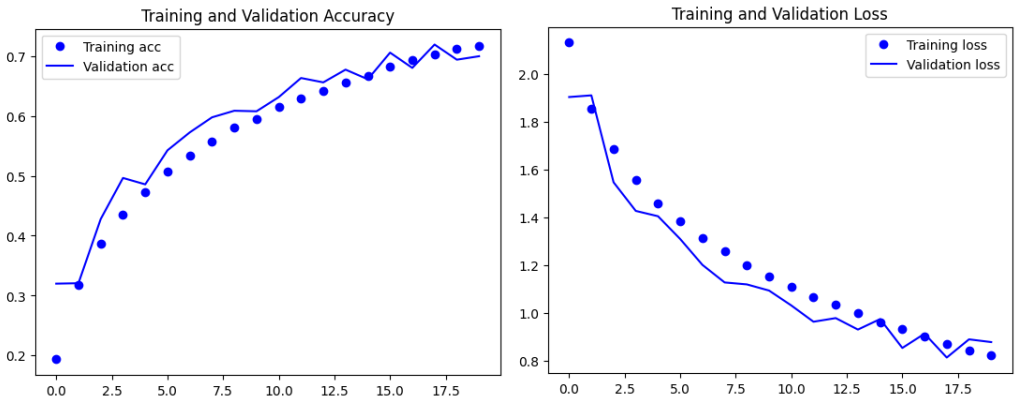
- 필터 수, 레이어 수 증가

layer	역할
Conv2D 64	가장 낮은 수준의 특징 추출
Conv2D 128 2개	더 복잡한 패턴 인식
Conv2D 256	추상화된 고차원 특징 추출
Dropout	과적합 방지
Dense 128	분류기 전 중간 표현 학습
Dense 10	CIFAR-10 클래스 분류



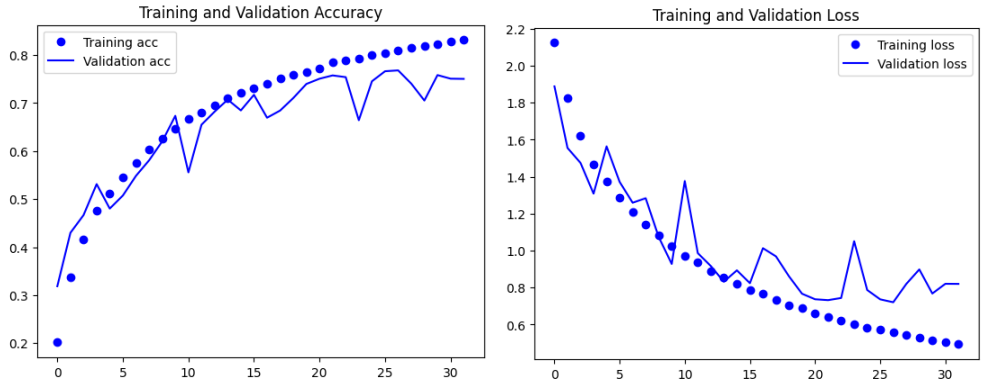
=> val_accuracy: 73.95 %, val_loss: 0.7788 채택

- Learning rate 튜닝 (0.001 → 0.0005)



=> val_accuracy: 73.95 %, val_loss: 0.7788 미채택

- Epoch 조정 with EarlyStopping (epoch: 20 → 50)



=> val_accuracy: 75.05 %, val_loss: 0.8200 채택

-EarlyStopping에 의해 32에서 stop!!

결론

기존 CNN 모델의 성능을 향상시키기 위해 세운 전략은 Dropout 추가, 필터 수, 레이어 수 증가, Learning rate 튜닝, Epoch 늘려 시각화 성능 확인 with EarlyStopping이 있었다. 이 중 val_accuracy, val_loss에 효과적이었던 전략은 필터 수 증가, epoch 증가 가 있었다. 이 두 전략을 통해 기존 성능의 5~8%의 향상을 확인할 수 있었다. 최종적으로 하이퍼파라미터를 조정하여 성능을 향상시킨 모델은 다음과 같다.

Layer	Output	설명
Input	(32, 32, 3)	CIFAR-10 이미지 (RGB)
Conv2D (64)	(30, 30, 64)	커널 (3x3), 필터 64개, ReLU
MaxPool2D	(15, 15, 64)	풀링 크기 (2x2), 다운샘플링
Dropout(0.2)	same	20% 확률로 뉴런 비활성화
Conv2D (128)	(13, 13, 128)	커널 (3x3), 필터 128개
MaxPool2D	(6, 6, 128)	다운샘플링
Dropout(0.3)	same	30% Dropout
Conv2D (128)	(4, 4, 128)	추가 Conv 레이어
Conv2D (256)	(2, 2, 256)	필터 수 증가
MaxPool2D	(1, 1, 256)	거의 정보 요약 완료
Dropout(0.4)	same	강한 정규화
Flatten	(256,)	3D → 1D로 변환
Dense(128)	(128,)	완전연결층, ReLU
Dropout(0.5)	same	강한 Dropout
Dense(10)	(10,)	softmax 출력 (클래스 확률)

위 모델을 통해 CIFAR-10을 처리한 결과, 선형모델이나 MLP로 이미지를 처리한 성능보다 확연히 향상된 것을 확인할 수 있었다. 하지만 위 모델의 val_accuracy는 75.05 %로 인간의 이미지 처리 능력보다 좋다고 볼 수는 없다. 그러므로 성능을 더 개선하기 위해 Optimizer를 변경하거나, 데이터를 증강(data augmentation)하거나 batchnorm을 추가하거나 학습률 스케줄러 등을 통해 더 좋은 성능을 구현할 수 있을 것이다.