parrot

2nd Project

Simpson Image Dataset Classification

Team 1 : 김민지 이영인 정상희 최보영 황선애 / 멘토 : 이수민

Contents

진행 과정

1주차
(11/11~11/13)

데이터 로드

데이터 타입 이해

- BatchDataSet
- Numpy Array

2주차
(11/13~11/20)

Numpy Array 사용

데이터 전처리

Simple CNN 모델

3주차
(11/20~11/27)

ImageNet 모듈 사용

ResNet / DenseNet / VGG
InceptionV3 / Xception /
MobileNet

4주차
(11/27~12/4)

모델 성능 향상

Ensemble

데이터

BatchDataSet : image_dataset_from_directory 사용

- 모델 학습 속도 느림
- 전이학습, 미세 조정 등의 기능 사용 어려움

Numpy Array : ImageDataGenerator & .next() 함수 사용

```
train_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
        "/content/drive/My Drive/parrot/2nd/train",target_size=(150, 150),
        batch_size=11248,class_mode='categorical')

train_x, train_y = train_generator.next()
```

데이터

데이터 전처리

- Data Augmentation / Gaussian Blur 등을 이용해 하이퍼 파라미터를 조절하는 방식으로 데이터 전처리를 해보았으나 별도의 데이터 전처리 과정이 없는 경우에 정확도가 가장 높아 최종 모델에서는 생략했음.

```
datagen = tf.keras.preprocessing.image.ImageDataGenerator(
    featurewise_center=False, samplewise_center=False,
    featurewise_std_normalization=False, samplewise_std_normalization=False,
    zca_whitening=False, zca_epsilon=1e-06, rotation_range=0, width_shift_range=0.1,
    height_shift_range=0.1, brightness_range=None, shear_range=0.0, zoom_range=0.1,
    channel_shift_range=0.0, fill_mode='nearest', cval=0.0, horizontal_flip=True,
    vertical_flip=False, rescale=None, preprocessing_function=None,
    data_format=None, validation_split=0.0, dtype=None
)
```

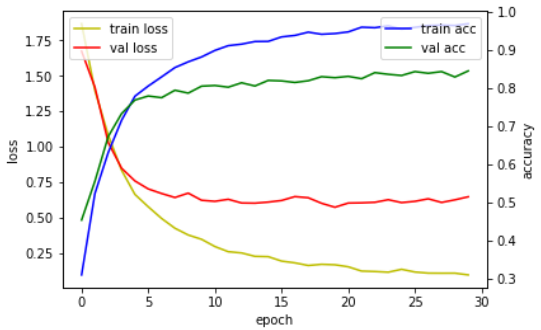
모델 학습 : 전이학습 사용

전이학습 (transfer learning) : 사전에 훈련된 모델을 사용하여 training set 학습

- ImageNet 모듈 사용 – ResNet / DenseNet / VGG /InceptionV3 / Xception / MobileNet
  - Image size / Batch size / Train Split 고정 후
  - Drop out rate / Learning Rate / Optimizer / Epoch 등에 변화를 주며 정확도 비교
  - https://www.tensorflow.org/tutorials/images/transfer_learning
- 직접 CNN 층을 쌓아 모델을 학습시킬 때는 정확도가 80% 초중반이었으나

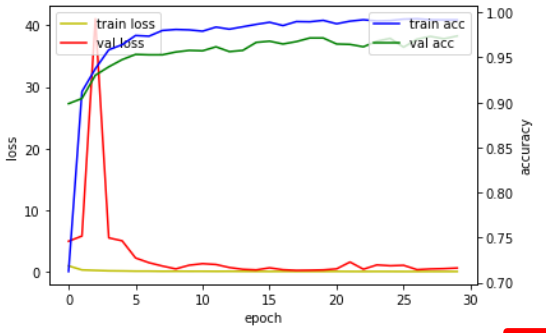
  이미지넷 모듈 사용 후 90% 후반까지 정확도가 올라감.

〈Simple CNN Model〉



val_loss: 0.6466 - val_accuracy: 0.8449

〈InceptionResNetV2 Module〉



val_loss: 0.5949 - val_accuracy: 0.9738

모델 학습 : ResNet (Residual Networks)

ResNet : 레이어가 깊어짐에 따라 최적의 학습 효과를 얻기 위한 모델

- shotcut 방식을 사용하여 비교적 속도가 빠르고 val_accuracy의 변동이 적음
- 이미지 사이즈에 크게 영향을 받지 않음.

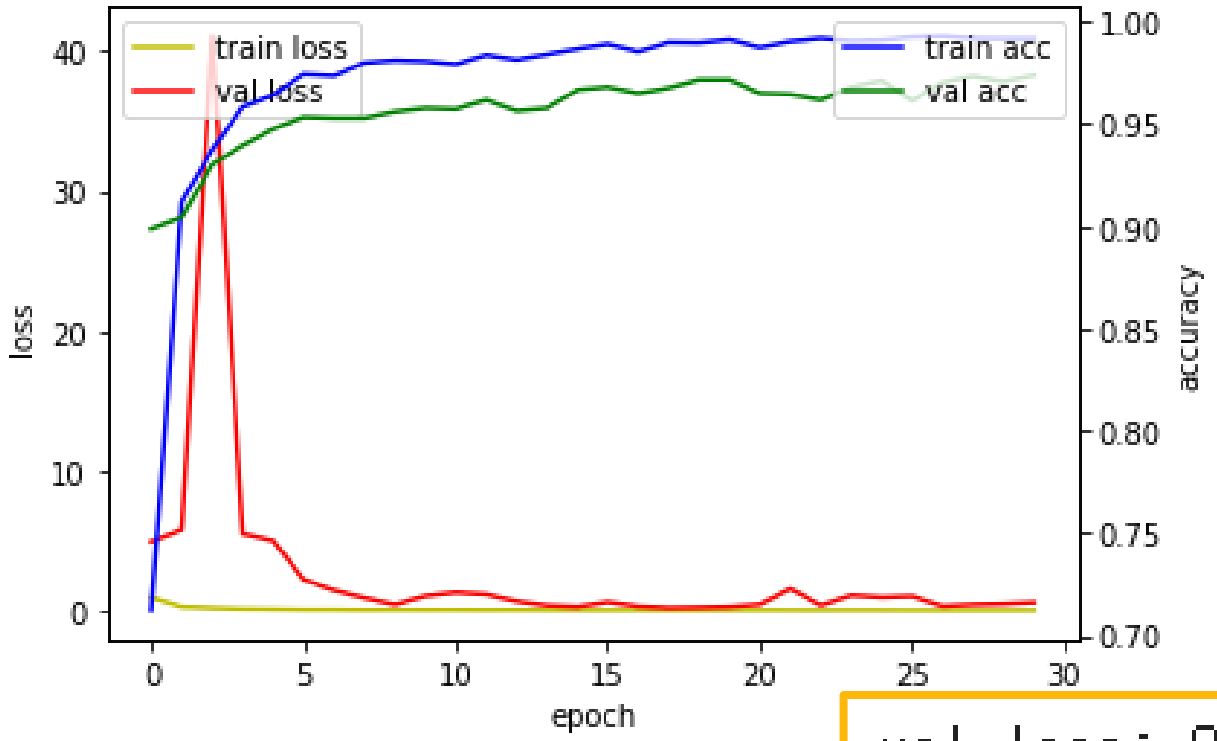
image-size	epoch	batch-size	dropout	learning-rate	optimizer	accuracy	max_acc
150	20	32	0.5	1.00E-04	adam	0.976	0.976
180	20	32	0.5	1.00E-04	adam	0.976	0.9822
150	30	32	0.4	1.00E-04	adam	0.96	0.9773
180	30	32	0.4	1.00E-04	adam	0.96	0.9747
150	50	32	0.4	1.00E-04	adam	0.9778	0.9778
180	50	32	0.4	1.00E-04	adam	0.9778	0.9778

- 종류 : 레이어 개수에 따라 ResNet50V2 / ResNet152V2 / InceptionResNetV2
- 정확도 : InceptionResNetV2 < ResNet50V2 < ResNet152V2

모델 학습 : InceptionResNetV2

InceptionResNetV2 :

Inception 구조와 Residual Network의 조합으로 만들어진 신경망. 164 개의 layer로 이루어짐.



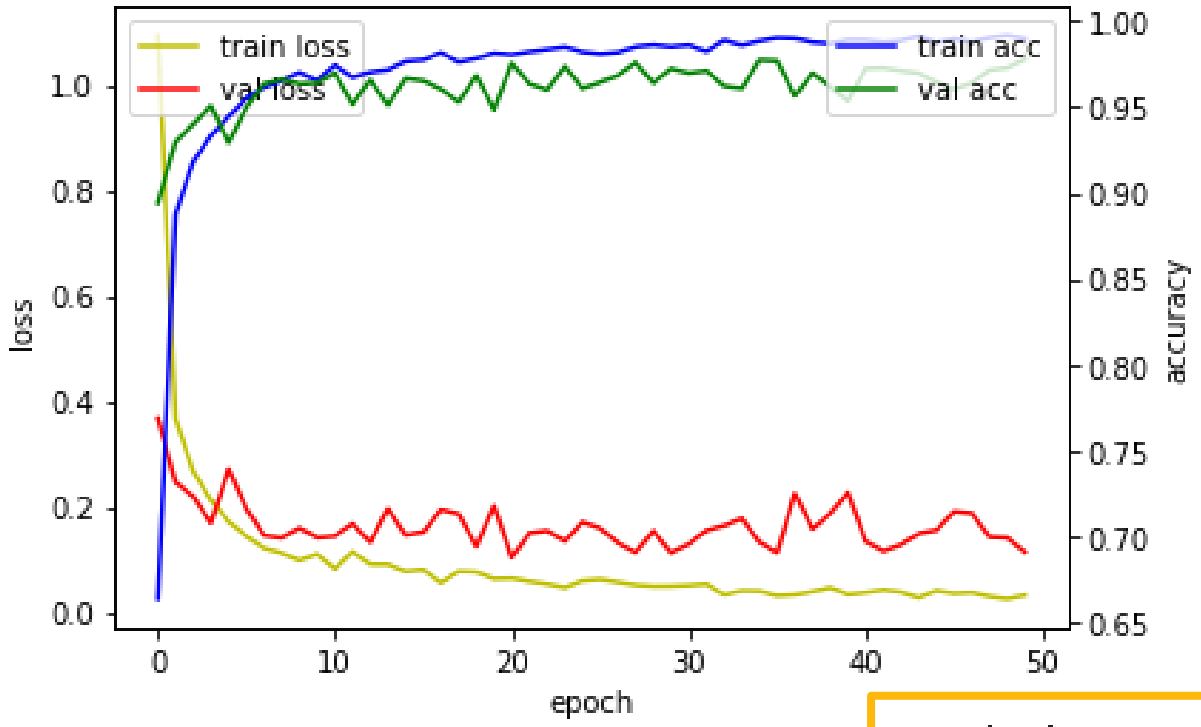

```
image_size = 150
drop_out_rate = 0.4
batch_size = 32
epoch = 30
optimizer = adam
learning rate : 1e-4
```

val_loss: 0.5949 - val_accuracy: 0.9738

모델 학습 : ResNet50V2

ResNet50V2 :

50개의 layer로 이루어진 Residual Network 구조의 신경망



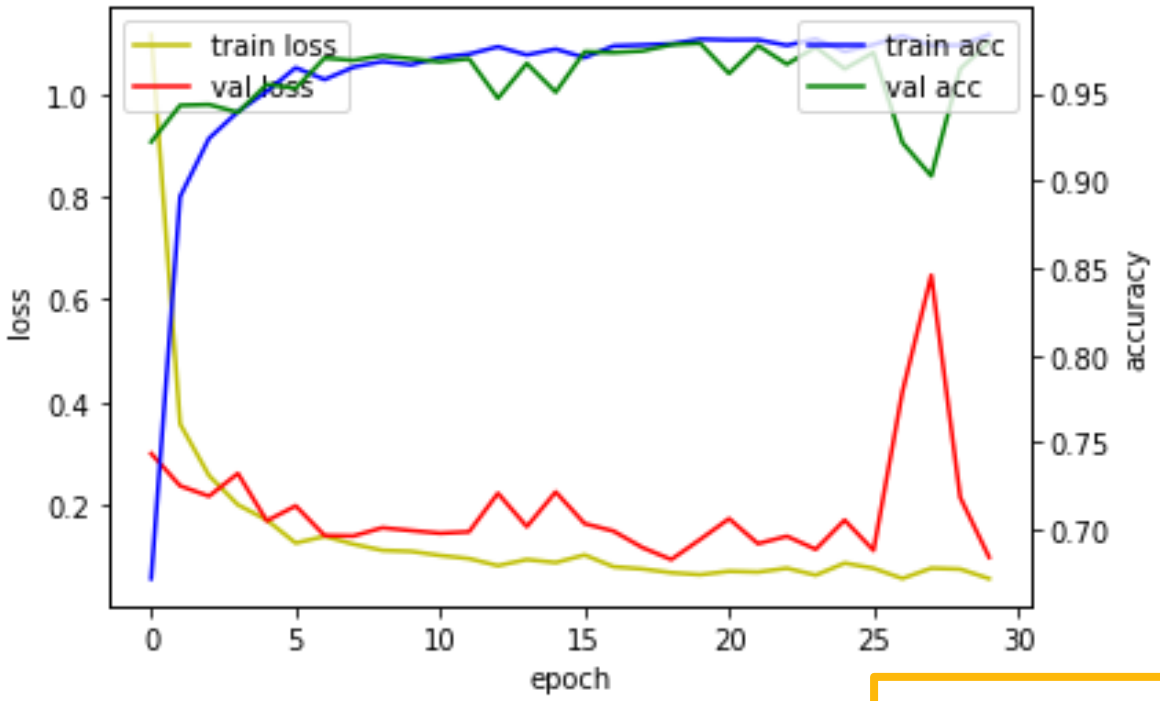

```
image_size = 150
drop_out_rate = 0.4
batch_size = 32
epoch = 50
optimizer = adam
learning rate : 1e-4
```

```
val_loss: 0.1146 - val_accuracy: 0.9778
```

모델 학습 : ResNet152V2

ResNet152V2 :

152개의 layer로 이루어진 Residual Network 구조의 신경망




```
image_size = 150
drop_out_rate = 0.5
batch_size = 32
epoch = 30
optimizer = adam
learning rate : 1e-4
```

val_loss: 0.0971 - val_accuracy: 0.9791

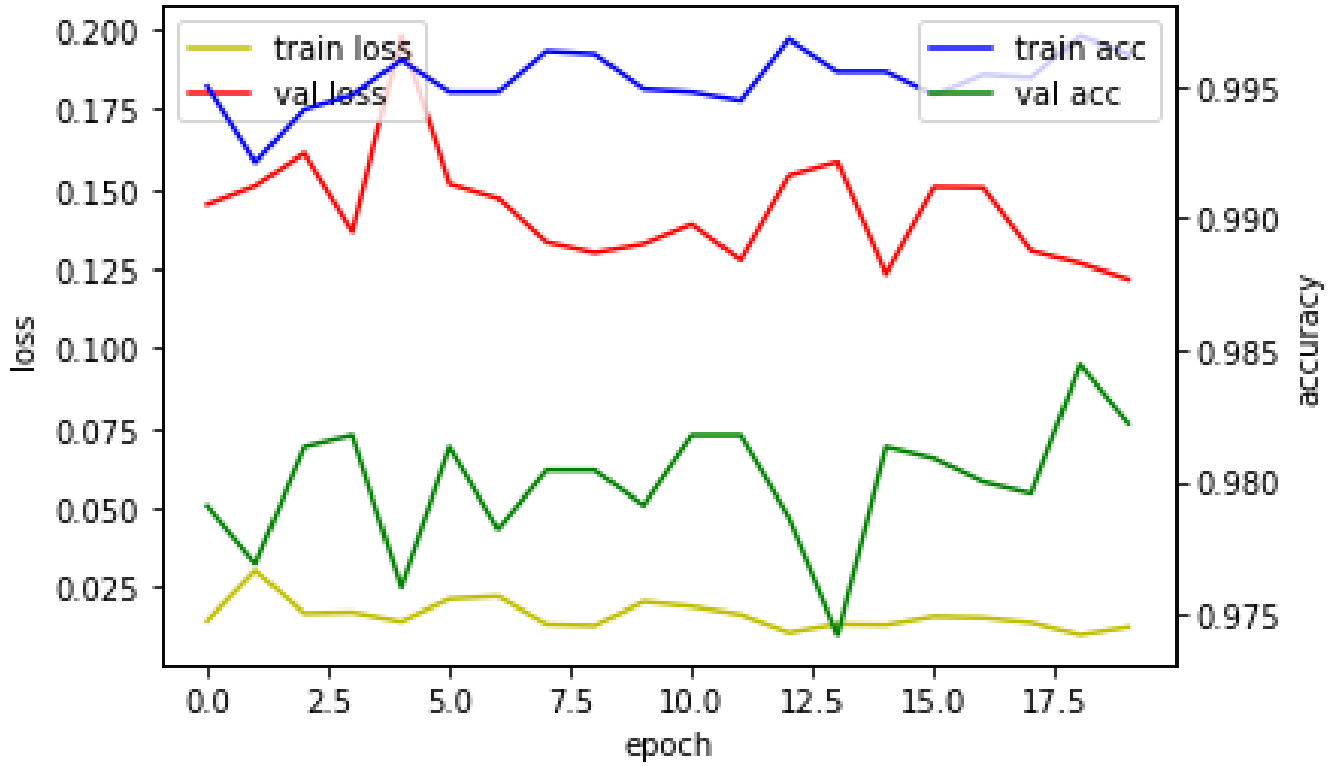
모델 학습 : Xception

Xception :

- train 정확도는 높지만 val_acc의 변동이 심함
- 이미지 사이즈가 150일 때 180일 때보다 정확도가 높음.

image-size	epoch	batch-size	dropout	learning-r	optimizer	accuracy	max_acc
150	30	32	0.4	1.00E-04	adam	0.9582	0.9809
180	30	32	0.4	1.00E-04	adam	0.9316	0.9462
150	30	32	0.5	1.00E-04	adam	0.9773	0.9787
180	30	32	0.5	1.00E-04	adam	0.9218	0.9524

모델 학습 : Xception




```
image_size = 150
drop_out_rate = 0.4
batch_size = 32
epoch = 30
optimizer = adam
learning rate : 1e-4
```

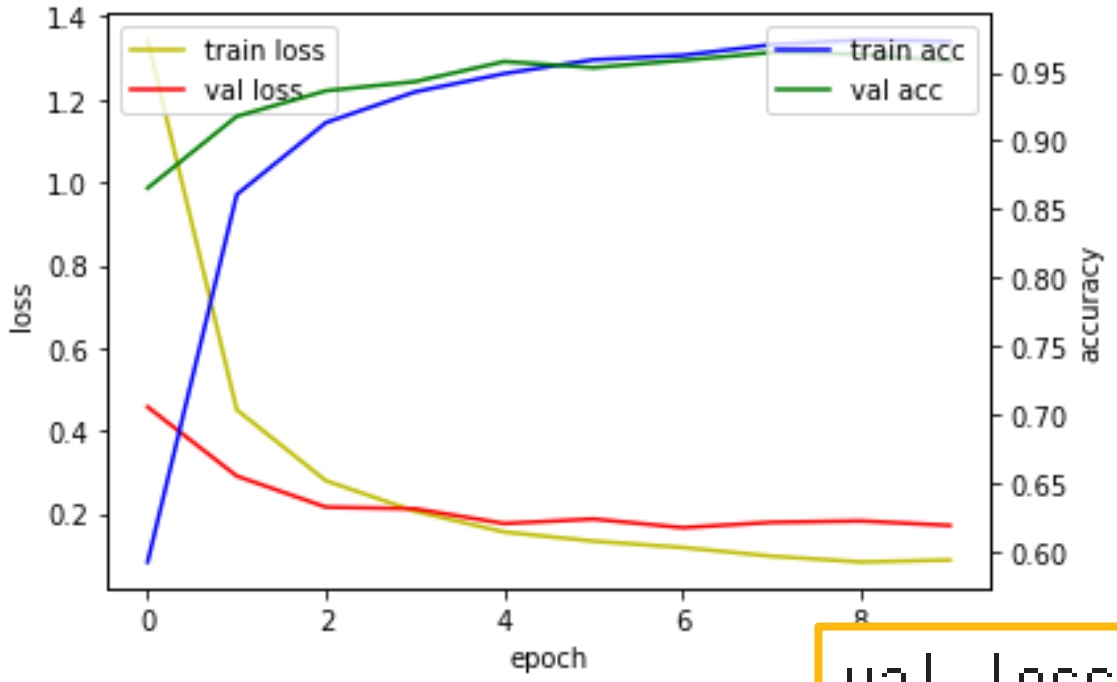
```
val_loss: 0.1217 - val_accuracy: 0.9822
```

모델 학습 : InceptionV3

InceptionV3 :

다른 모듈에 비해 레이어 수가 많지만 파라미터를 적게 사용하여

학습 속도가 높고 메모리 사용량이 적은 모듈



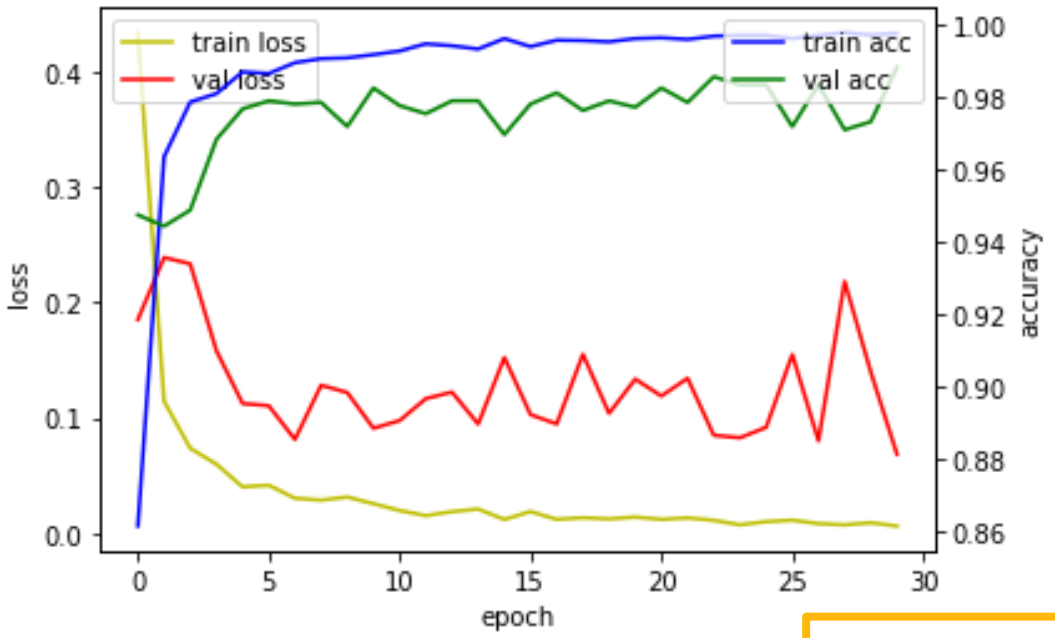

```
image_size = 150
drop_out_rate = 0.4
batch_size = 32
epoch = 10
optimizer = adam
learning rate : 1e-4
```

```
val_loss: 0.1715 - val_accuracy: 0.9582
```

모델 학습 : DenseNet201 (Densely Connected Convolutional Networks)

DenseNet201 :

이전 feature map을 다음 레이어의 입력과 연결하여 그래디언트 소실 문제를 개선하고 파라미터 수를 줄여주는 모듈




```
image_size = 150
drop_out_rate = 0.2
batch_size = 32
epoch = 30
optimizer = RMSprop
learning rate : 1e-4
```

```
val_loss: 0.0684 - val_accuracy: 0.9884
```

모델 학습 : VGG

VGG :

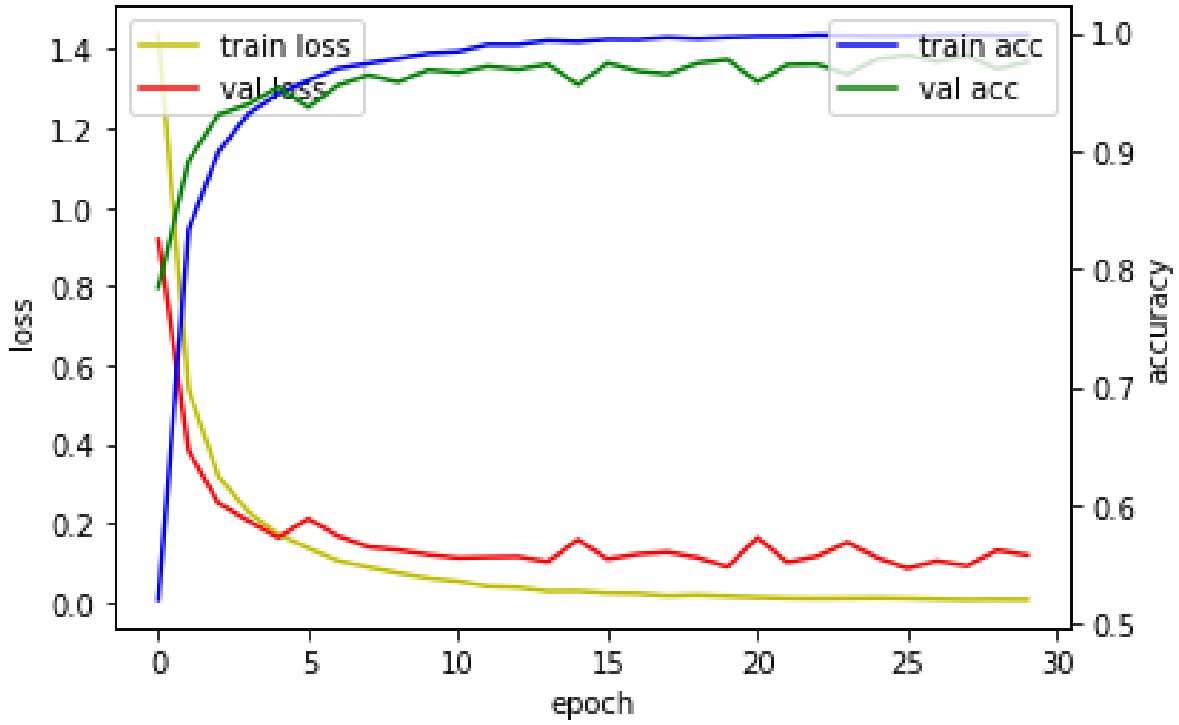
- 3x3 conv 필터를 중첩하여 더 많은 비선형성을 얻으며 파라미터 수를 줄여주는 모듈
- drop-out은 정확도에 큰 영향을 주지 못함.

image-size	epoch	batch-size	dropout	learning-r	optimizer	accuracy	max_acc
150	10	32	0.2	1.00E-04	adam	0.9538	0.9604
150	10	32	0.3	1.00E-04	adam	0.96	0.96
150	10	32	0.4	1.00E-04	adam	0.9542	0.9596
150	10	32	0.2	1.00E-04	rmsprop	0.9613	0.9613
150	10	32	0.3	1.00E-04	rmsprop	0.9516	0.9582
150	10	32	0.4	1.00E-04	rmsprop	0.9596	0.9596

- 종류 : VGG16 / VGG19

모델 학습 : VGG16

VGG16 : 16개의 레이어로 이루어진 신경망

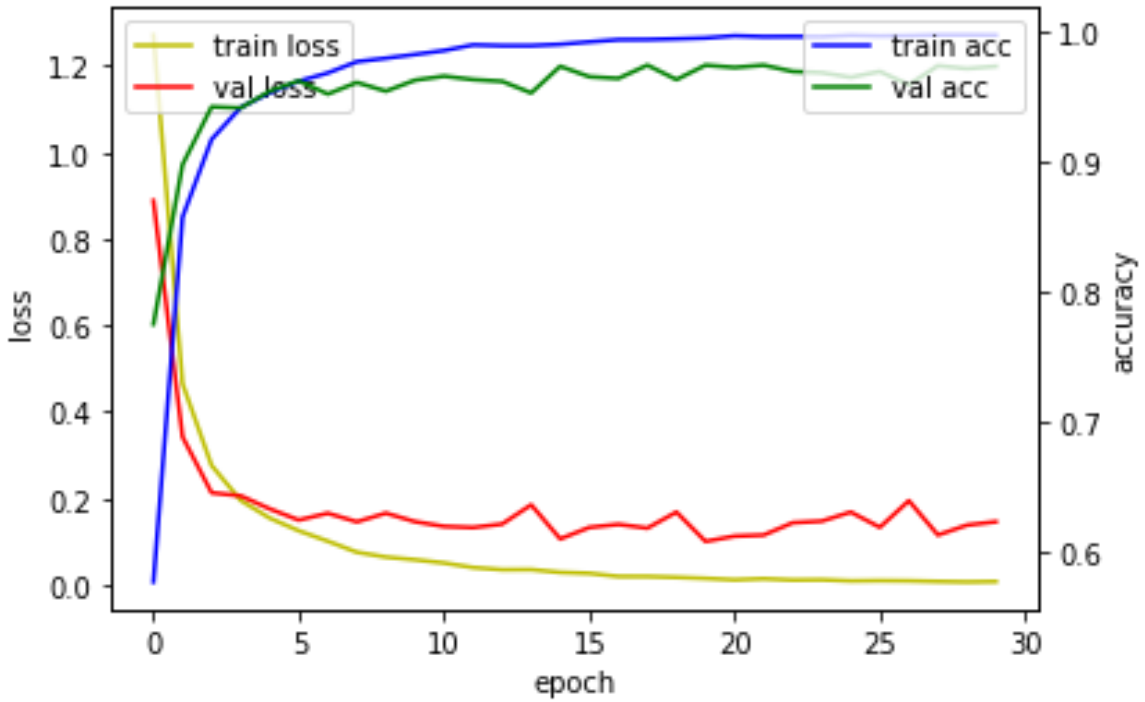



```
image_size = 150
drop_out_rate = 0.2
batch_size = 32
epoch = 30
optimizer = RMSprop
learning rate : 1e-4
```

```
val_loss: 0.1315 - val_accuracy: 0.9693
```

모델 학습 : VGG19

VGG19 : 19개의 레이어로 이루어진 신경망



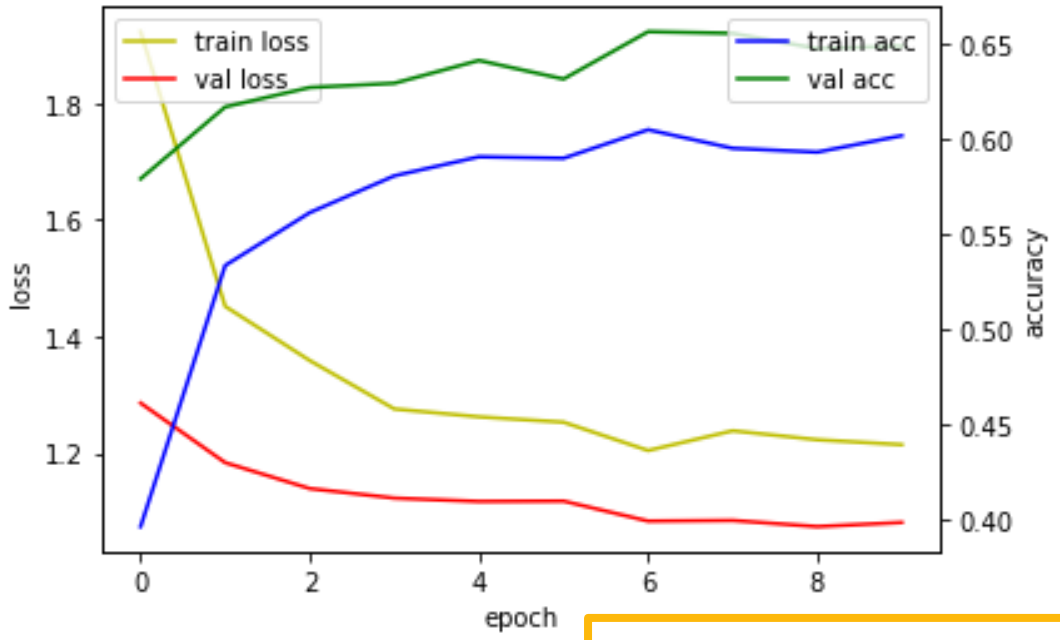

```
image_size = 150
drop_out_rate = 0.2
batch_size = 32
epoch = 30
optimizer = RMSprop
learning rate : 1e-4
```

```
val_loss: 0.1456 - val_accuracy: 0.9738
```

모델 학습 : MobileNetV2

MobileNetV2 :

- Mobile device에서도 돌아갈 수 있도록 만든 가벼울 구조의 모듈
- 가벼운 구조이다 보니 다른 모듈에 비해 정확도가 떨어짐




```
image_size = 150
drop_out_rate = 0.4
batch_size = 32
epoch = 10
optimizer = adam
learning rate : 1e-4
```

```
val_loss: 1.0634 - val_accuracy: 0.6533
```

모델 학습 : ImageNet 최종

MobileNet 모듈을 제외하고는 모두 95-98%의 정확도를 냈음.

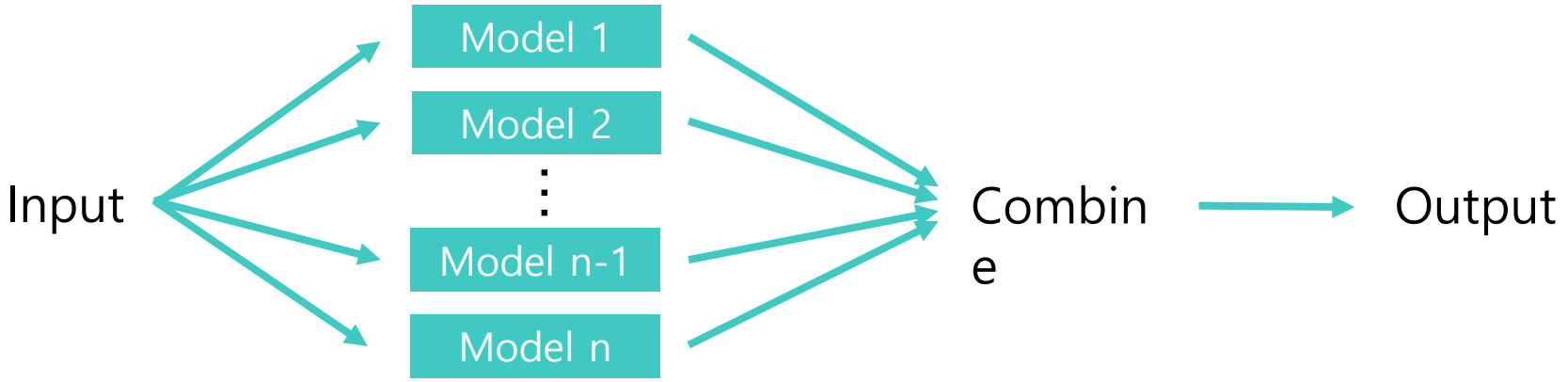
learning rate는 1e-4 일 때 가장 정확도가 높았음.

module	image_size	drop_out	batch_size	epoch	optimizer	learning_rate	val_acc	val_loss
DenseNet201	150	0.2	32	30	RMSprop	1.00E-04	0.9884	0.0684
Xception	150	0.4	32	30	Adam	1.00E-04	0.9822	0.1217
ResNet152V2	150	0.5	32	30	Adam	1.00E-04	0.9791	0.0971
ResNet50V2	150	0.4	32	50	Adam	1.00E-04	0.9778	0.1146
InceptionResNetV2	150	0.4	32	30	Adam	1.00E-04	0.9738	0.5949
VGG19	150	0.2	32	30	RMSprop	1.00E-04	0.9738	0.1456
VGG16	150	0.2	32	30	RMSprop	1.00E-04	0.9693	0.1315
InceptionV3	150	0.4	32	10	Adam	1.00E-04	0.9582	0.1715
MobileNetV2	150	0.4	32	10	Adam	1.00E-04	0.6533	1.0634

모델 학습 : 모델 성능 향상 - Ensemble

- 이미지넷 모듈 사용 후 모델 성능 향상을 위해
  미세조정, grid search, coarse&finer search, early stopping 등의 방법을 찾아보았으나
  RAM 용량 문제와 속도의 문제로 시도하지 못했음.

- Ensemble 사용 :
  학습시킨 두 개 이상 모델의 예측치 평균을 구해 예측 결과를 출력하는 방법



모델 학습 : 모델 성능 향상 - Ensemble

아래 5개 모델을 사용하여 앙상블 - 따로 정확도를 측정했을 때에 비해 정확도가 올라감

Module	Drop-out	Batch	Epoch	Optimizer	Learning Rate	Val_acc
ResNet152V2	0.4	32	30	Adam	1.00E-04	0.9738
Xception	0.4	32	30	Adam	1.00E-04	0.9818
InceptionV3	0.4	32	30	Adam	1.00E-04	0.976
DenseNet201	0.2	32	30	RMSprop	1.00E-04	0.9849
VGG16	0.4	32	50	RMSprop	1.00E-04	0.9387

- 이미지 사이즈가 커짐에 따라 test accuracy 높아짐
  - 150 : 98.5267%
  - 180 : 98.9871%
  - 200 : 99.1712%

최종 모델

https://colab.research.google.com/drive/1aGAajWBe08VeOXHJWia6tW0iQdYmc5ak?usp=sharing

정확도 : 99.1712%

Q&A