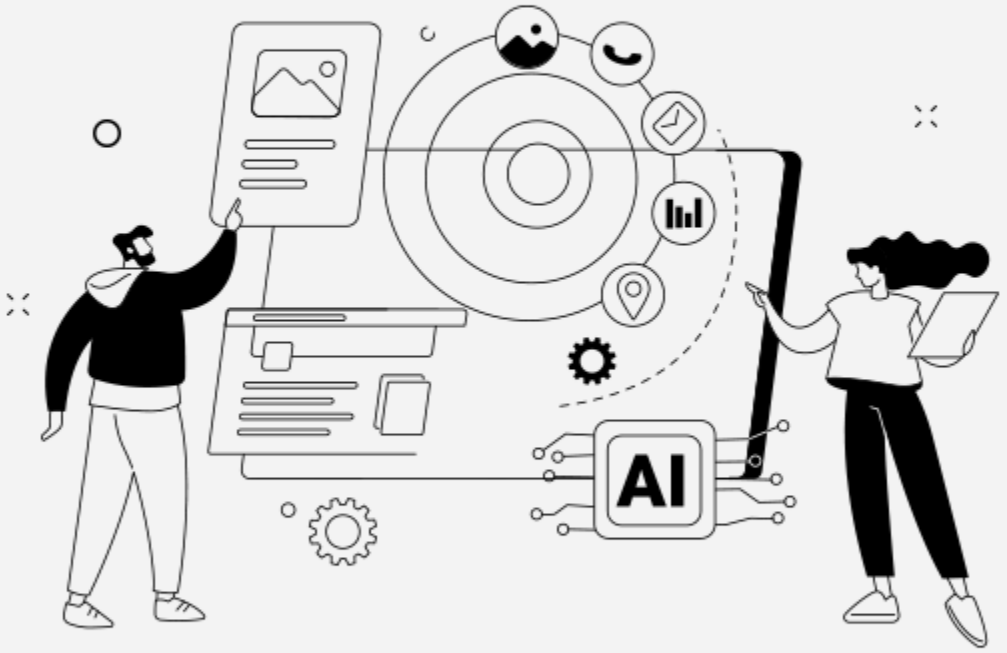
2022 데이터 크리에이터 캠프

Data Creator Camp



- ( )회차 사회관 507호

과학기술정보통신부 NIA 한국지능정보사회진흥원

전체 분석 흐름

0. 분석 과정 개요

EDA → 군집화 (사진 제거) → 랜덤 샘플링 → 모델 설정 → 모델 개선

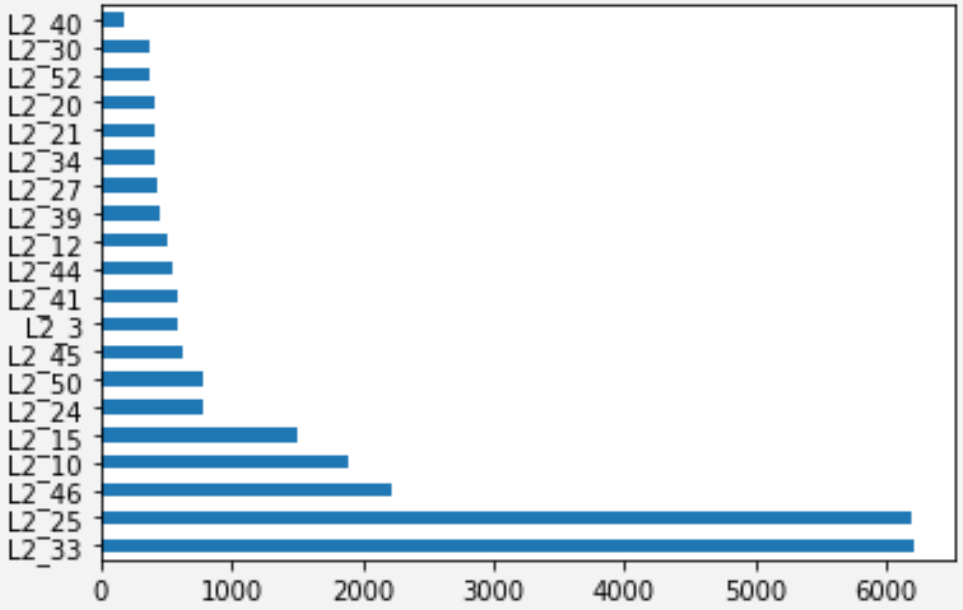
Mission #1	Mission #2	Mission #3
• 데이터 탐색 - 분포 확인 - 특성 확인 • 대응 전략 수립	• 병합(Agglomerative) 군집화 - HLS 기반 클러스터링 • 데이터 샘플링 (군집화 이후 시행) - Under-sampling - Over-sampling	• 모델 설정 - 베이스 모델 구현, 성능 비교 - Residual Block 기반 모델 설정 - 콜백 함수 선언 • 하이퍼파라미터 튜닝

Mission #1.

1. 탐색적 데이터 분석(Exploratory Data Analysis, EDA)

- 데이터 분포 및 특성 확인

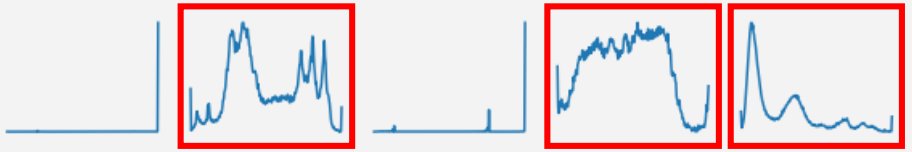
- 범주별 이미지 빈도(개수) 그래프



- 범주별 이미지 밝기 그래프

- 일러스트만 있는 15개 범주

- 사진이 포함된 총 5개의 범주 (L2_3, L2_12, L2_24, L2_41, L2_50)



- Max: 6206(L_33) / Min: 180(L2_40) ➔ 데이터 불균형
- 일러스트가 아닌 사진이 포함된 5개 범주 ➔ 이질적인 데이터 포함

Mission #1.

1. 탐색적 데이터 분석(Exploratory Data Analysis, EDA)

- 데이터 불균형 문제 대응 전략 : 랜덤 언더/오버 샘플링(Random Under/Over sampling)

✓ "각 클래스 별로 250-1,000개의 데이터일 때 성능이 가장 좋았음" – 배은지, & 이성진. (2021).

➔ 각 클래스의 데이터가 250-1,000개가 되도록 두 샘플링 방식을 모두 사용함.

250 ~ 1,000개의 데이터 범위를 벗어나는 범주 식별

L2_33, L2_25, L2_46, L2_10, L2_15 (5개) > 1000개 초과 ➔ 언더 샘플링

L2_41, L2_3, L2_40, L2_12 (4개) < 250개 미만 ➔ 오버 샘플링

* 사진이 샘플링되는 것을 막기 위해 이 과정은 군집화 이후 시행됨.

```
L2_10    1000
L2_15    1000
L2_33    1000
L2_25    1000
L2_46    1000
L2_45     631
L2_44     547
L2_39     454
L2_27     426
L2_34     419
L2_24     416
L2_21     410
L2_20     410
L2_50     404
L2_52     382
L2_30     364
L2_12     250
L2_40     250
L2_3      250
L2_41     250
Name: label, dtype: int64
```

Mission #2.

2. 군집화(Clustering)

- 분석 개요

  그림과 그림이 아닌 사진의 구분을 위한 비지도 학습의 필요성

  : 라벨이 붙어 있진 않지만, EDA의 밝기 그래프 상 명확한 차이를 보이고 있음.

  ➔ 군집화(Clustering) 기법으로 구분이 가능할 것으로 판단됨.

  - 사진이 포함된 총 5개의 범주 (L2_3, L2_12, L2_24, L2_41, L2_50)

- 대응 전략 : HLS (색상, 명도, 채도)에 기반한 병합 군집화(Agglomerative Clustering)

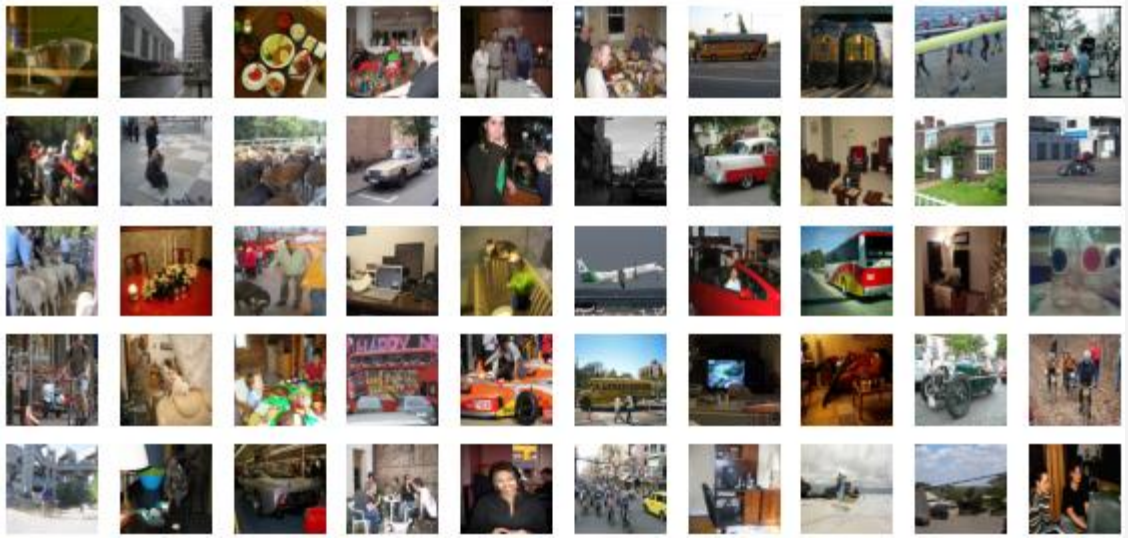
  ✓ "K-평균 군집화는 데이터의 정규화 여부에 따라 결과가 달라질 수 있음" - Visalakshi, N. K., & Thangavel, K. (2009)

  ✓ "색상에 의존하는 RGB 기반 수행보다 HVS, HLS 등의 요소를 추가한 군집화 성능이 더 좋음." - Jurio, A. et al., (2010)

Mission #2.

2. 군집화(Clustering)

- 클러스터링 결과표



클러스터 1 - 사진

클러스터 2 - 일러스트

클러스터링이 전반적으로 잘 수행되었으며, 기존의 방식보다 오분류 빈도가 낮았음.

Mission #3.

4. 모델 설정(Model Specification)

- 분석 개요

• ImageDataGenerator

➔ Batch_size만큼 이미지를 로드함.

• Bottleneck 구조 도입

➔ 추정 파라미터 수를 감소시킴.

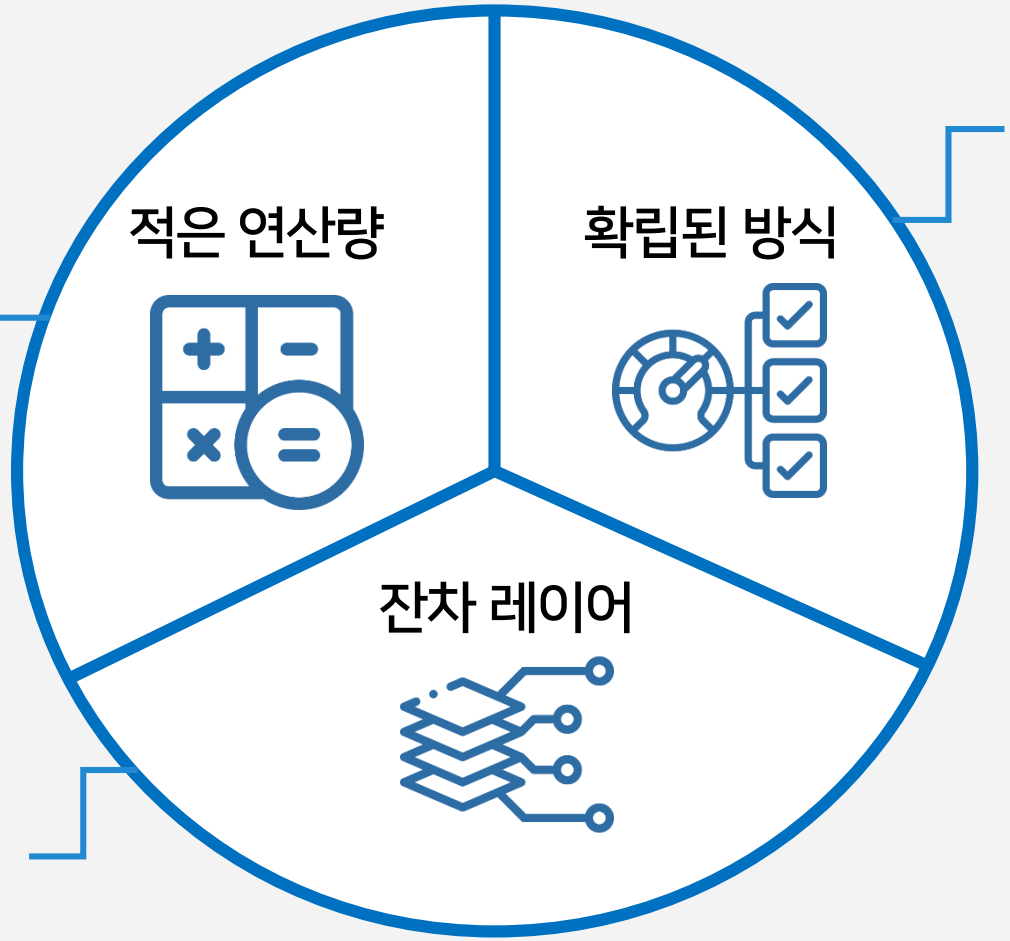
• Batch_Size 조정

➔ 64보다 큰 사이즈는 메모리 초과

VGG16, resnet 등의 기본 구조 이해

➔ Residual Block 활용 모델 구현

분석 모델 설정의 3가지 기준점



• 성과 최대화 / 간명한 모델

➔ 한 층씩 쌓아가면서 모델 탐색

: 가장 효율적인 모델 구조 확인

• 분석 시간의 한계

➔ 선행연구 기반 파라미터 설계

➔ 이론에 근거한 연역적 접근

Mission #3.

4. 모델 설정(Model Specification)

- ImageDataGenerator 기반 분석 – GPU 기반 처리

  : Image Data + Generator ➔ Batch_size만큼 '전처리가 된' 이미지를 불러오는 생성 기법

이미지 증식 기법(Image Augmentation)

: 기존의 이미지와 약간의 차이가 있는 이미지를 생성하여 데이터 수와 학습 난이도를 증가시키는 기법

✓ "데이터의 특성과 분석 목적, 분야 지식에 따라 증식 기법을 결정해야 한다." - Buslaev, A. et al., (2020)

```
ImageDataGenerator(horizontal_flip = True,
                   width_shift_range = .3,
                   height_shift_range = .3,
                   rotation_range = 30.0,
                   zoom_range = 0.2,
                   fill_mode = 'nearest',
                   rescale = 1./255)
```

(1) 분류(Classification) 문제 (Zoom in/out 사용 가능)

(2) 거꾸로 뒤집힌 일러스트는 거의 없음. (Horizontal Flip 사용, Vertical Flip 배제)

(3) 일러스트의 여백은 대부분 흰색으로 고정됨. (fill_mode = "nearest" 사용)

Mission #3.

4. 모델 설정(Model Specification)

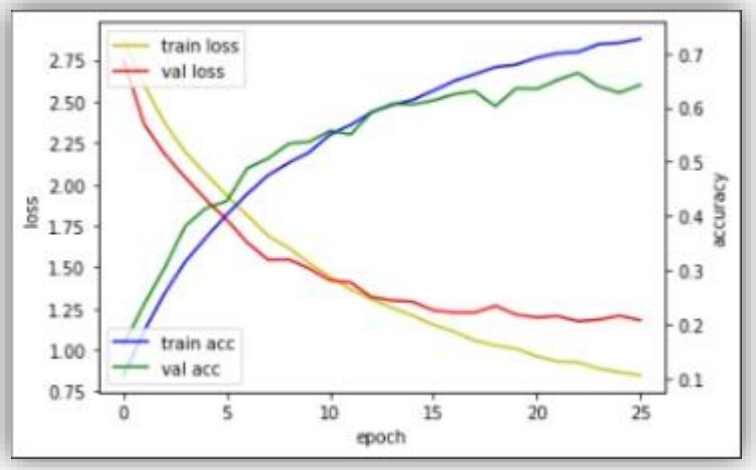
- Baseline 모델 설정

```
# 베이스 모델 구축
model = Sequential()
model.add(Convolution2D(32, 3, 3, padding = 'same',
    input_shape = (224,224,3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Convolution2D(64, 3, 3, padding='same'))
model.add(Activation('relu'))
model.add(Convolution2D(64, 3, 3))
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2,2), padding='same'))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(20))
model.add(Activation('softmax'))
```

기본적인 형태의 CNN 모형

Convolution, Flatten, Dense 등의 레이어를 보유한 형태

-> Keras 기반 모형 설정 연습 / Baseline 성과 비교 목적




```
*** CLASSWISE RESULT ***
                0         1         2         3         4         5  #
f1       0.632768  0.512821  0.723005  0.650602  0.507042  0.542373
acc      0.708861  0.400000  0.793814  0.675000  0.409091  0.432432
support 79.000000 25.000000 97.000000 40.000000 44.000000 37.000000

                6         7         8         9        10        11  #
f1       0.642857  0.551724  0.578947  0.540541  0.680000  0.786517
acc      0.642857  0.521739  0.500000  0.571429  0.731183  0.760870
support 98.000000 46.000000 22.000000 35.000000 93.000000 46.000000

               12        13        14        15        16        17  #
f1       0.660377  0.799474  0.619048  0.661017  0.607843  0.635983
acc      0.729167  0.652174  0.565217  0.709091  0.516667  0.723810
support 48.000000 23.000000 23.000000 55.000000 60.000000 105.000000

               18        19
f1       0.682353  0.666667
acc      0.557692  0.666667
support 52.000000 36.000000

*** AVG RESULT ***
f1      0.633598
acc     0.645677
dtype: float64
```

학습은 안정적이었으나 Valid Accuracy가 0.7을 넘지 못했음.

Mission #3.

4. 모델 설정(Model Specification)

- 모델 개선 과정 1. 모델 구조 변경

"Vgg19의 성능이 Resnet보다 높을 수 있음." - Ikechukwu, A. V., Murali, S., Deepu, R., & Shivamurthy, R. C. (2021)

```
self.fc1 = keras.layers.Dense(4096, activation='relu')
self.fc2 = keras.layers.Dense(4096, activation='relu')
self.fc3 = keras.layers.Dense(2048, activation='relu')
self.fc4 = keras.layers.Dense(1024, activation='relu')
self.fc5 = keras.layers.Dense(20, activation='softmax')

    # Classification block
  x = self.flat(x)
  x = self.fc1(x)
  x = self.fc2(x)
  x = self.fc3(x)
  x = self.fc4(x)
  x = self.dropout2(x)
  x = self.fc5(x)

  return x
```

*** CLASSWISE RESULT ***

	0	1	2	3	4	5
f1	0.912500	0.916667	0.921466	0.84507	0.790698	0.864865
acc	0.924051	0.880000	0.907216	0.75000	0.772727	0.864865
support	79.000000	25.000000	97.000000	40.00000	44.000000	37.000000

	6	7	8	9	10	11
f1	0.927835	0.831461	0.883721	0.864865	0.950820	0.947368
acc	0.918367	0.804348	0.863636	0.914286	0.935484	0.978261
support	98.000000	46.000000	22.000000	35.000000	93.000000	46.000000

	12	13	14	15	16	17
f1	0.929293	0.977778	0.933333	0.915254	0.905983	0.894231
acc	0.958333	0.956522	0.913043	0.981818	0.883333	0.885714
support	48.000000	23.000000	23.000000	55.000000	60.000000	105.000000

	18	19
f1	0.859649	0.972973
acc	0.942308	1.000000
support	52.000000	36.000000

*** AVG RESULT ***
f1 0.902291
acc 0.905075
dtype: float64

Valid Set에 대해 적용한 결과 F1-Score

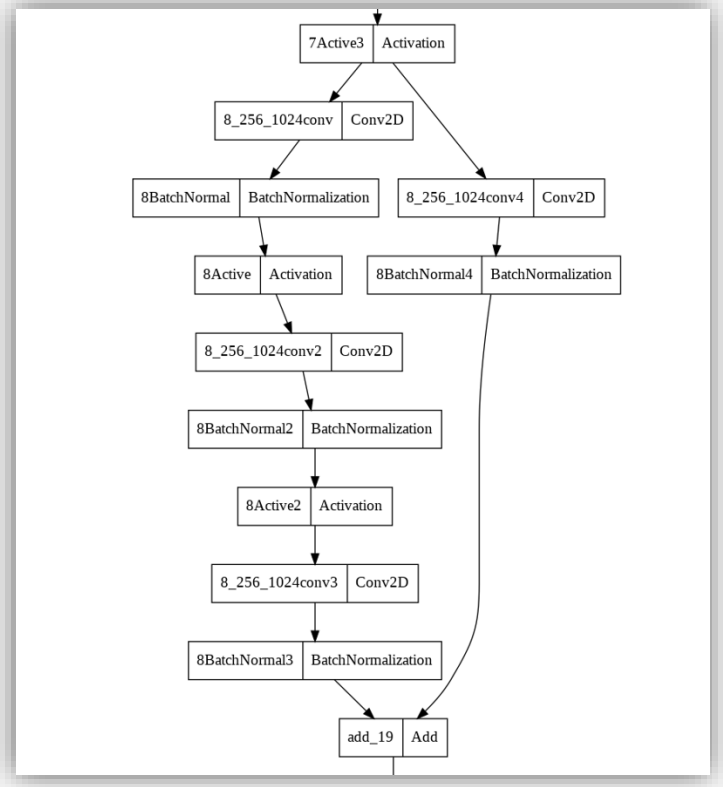
Vgg16 형태의 모델은 Baseline 모델보다 성과가 뛰어남.

Residual Block 기반 모형보다 성과가 낮아 최종 모형으로는 채택되지 않았음.

Mission #3.

4. 모델 설정(Model Specification)

- 모델 개선 과정 1. 모델 구조 변경



Residual Block 구조 도식

1) Residual Block 형태 구현

: Convolution Layer 2개를 하나의 Block으로 보고, 우회로 Shortcut 포함

: 위의 형태에서 연산량을 낮추기 위한 Bottleneck 구조로 변경

- 64 * 3, 128 * 3, 256 * 3, 512 * 3 (Bottleneck) 구조가 가장 성능이 좋았음.

2) 최선의 모형 구조 탐색

✓ "Dropout과 BatchNormalization은 하나의 Block에 같이 사용되지 않음."

- Li, X., Chen, S., Hu, X., & Yang, J. (2019).

✓ "Global Average Pooling은 Flatten보다 파라미터 수가 작아 복잡성이 낮음."

- Maisano, R. et al., (2018).

Mission #3.

4. 모델 설정(Model Specification)

- 모델 개선 과정 2. 콜백 함수 선언

  : 특정한 시점이나 어떤 이벤트가 발생했을 때 시스템에서 호출되는 함수

  - 학습의 조기 종료, 가장 좋은 모델의 가중치 저장, 학습률 조정의 역할 수행

1) Model Checkpoint : Valid Set의 Loss가 가장 낮은 모형의 가중치만 저장

2) Early Stopping : Valid Set의 Loss가 10회 동안 감소하지 않는 경우 학습을 중단

3) Reduce Learning Rate : Valid Set의 Loss가 5회 동안 감소하지 않는 경우 0.5 * rate로 조정

Mission #3.

4. 모델 설정(Model Specification)

- 모델 개선 과정 3. 하이퍼파라미터(Hyperparameter) 튜닝

: Gradient Student Descent 방식으로 조건 탐색

✓ "SGD 방식에선 Batch_size = 64와 Learning_rate = 0.001의 ACC가 가장 높음." - Kandel, I., & Castelli, M. (2020).

1) Optimizer : SGD가 Adam보다 안정적이었음. - Charles, Z., & Papailiopoulos, D. (2018, July).

2) Batch_size : 64개(.9467)에서 성능이 좋았음. (32개 : 성능 저하, 128개 : 런타임 강제 종료)

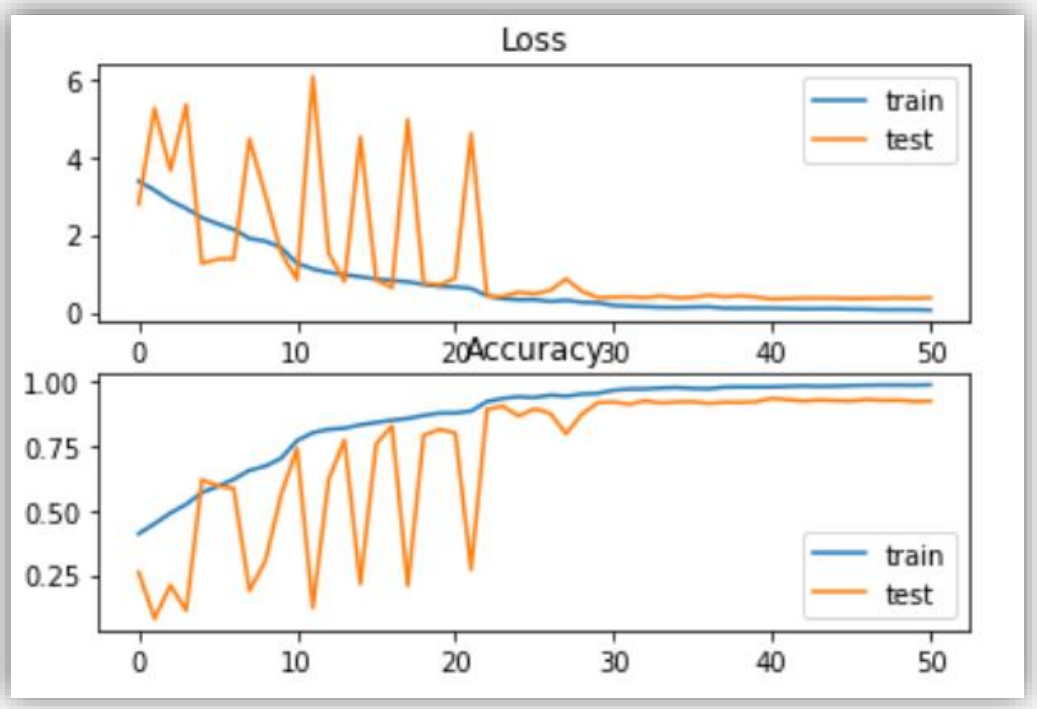
3) Learning Rate : 0.01에서 시작해 0.001 부근에서 최종 Valid Accuracy가 높아지며 안정화됨.

+ 범주마다 비중이 다른 점을 고려하여 Class_weight 부여

Mission #3.

5. 모델 성과

- 최종 모델의 성과



0.01 0.005 0.0025 0.00125

Learning Rate

*** CLASSWISE RESULT ***

	0	1	2	3	4	5	₩
f1	0.975309	0.958333	0.989796	0.987654	0.977273	0.945946	
acc	1.000000	0.920000	1.000000	1.000000	0.977273	0.945946	
support	79.000000	25.000000	97.000000	40.000000	44.000000	37.000000	

	6	7	8	9	10	11	₩
f1	0.968421	0.967742	0.954545	0.916667	0.972678	1.0	
acc	0.938776	0.978261	0.954545	0.942857	0.956989	1.0	
support	98.000000	46.000000	22.000000	35.000000	93.000000	46.0	

	12	13	14	15	16	17	₩
f1	0.968421	0.909091	0.938776	0.990991	0.983333	0.995261	
acc	0.958333	0.869565	1.000000	1.000000	0.983333	1.000000	
support	48.000000	23.000000	23.000000	55.000000	60.000000	105.000000	

	18	19
f1	0.990291	0.972222
acc	0.980769	0.972222
support	52.000000	36.000000

*** AVG RESULT ***
f1 0.968137
acc 0.974624
dtype: float64

Valid Set에 대해 적용한 결과 F1-Score

Baseline 모델과 vgg16 기반 모델보다 성과가 뛰어나 최종 모델로 선정됨.

시간상 문제로 교차타당화(10-Fold Cross Validation)는 진행하지 못함.

팀원 역할분담

부산대학교

계량심리연구실

(사회관 507호)

정지헌 Oh Captain My Captain	병합 군집화 코드 작성 및 시각화 / vgg16 모델 작성 및 개선 Model.py 파일 작성
노치현	Residual Block 기반 모델 설계 및 개선, (교차 타당화) 이론적 배경, 분석 기법 적용, 성능 비교 근거 등 논문 서칭, ppt 전체 제작
이현우	Residual Block 기반 모델 작성 및 개선 Google Colab 전체 코드 호환성 확인, 주석 작성, ppt 수정, 발표
최범식	EDA 코드 작성 및 시각화, HLS 기반 병합 군집화 코드 개선 / vgg16 모델 개선 Eval.py 파일 작성 및 호환성 확인

참고문헌

배은지, & 이성진. (2021). 이미지 분류 네트워크에서의 효율적 훈련 기법. *한국통신학회논문지*, *46*(6), 1087-1096.

Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., & Kalinin, A. A. (2020). Albumentations: fast and flexible image augmentations. *Information*, *11*(2), 125.

Charles, Z., & Papailiopoulos, D. (2018, July). Stability and generalization of learning algorithms that converge to global optima. In *International Conference on Machine Learning* (pp. 745-754). PMLR

Ikechukwu, A. V., Murali, S., Deepu, R., & Shivamurthy, R. C. (2021). ResNet-50 vs VGG-19 vs training from scratch: a comparative analysis of the segmentation and classification of Pneumonia from chest X-ray images. Global Transitions Proceedings, 2(2), 375-381.

Jurio, A., Pagola, M., Galar, M., Lopez-Molina, C., & Paternain, D. (2010, June). A comparison study of different color spaces in clustering based image segmentation. In *International conference on information processing and management of uncertainty in knowledge-based systems* (pp. 532-541). Springer, Berlin, Heidelberg.

Kandel, I., & Castelli, M. (2020). The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. *ICT express*, *6*(4), 312-315.

Visalakshi, N. K., & Thangavel, K. (2009). Distributed data clustering: A comparative analysis. In *Foundations of Computational, IntelligenceVolume 6* (pp. 371-397). Springer, Berlin, Heidelberg.

감사합니다

2022 DATA CREATOR CAMP

과학기술정보통신부 NIA 한국지능정보사회진흥원