데이터 가공 프로젝트 pt2

이미지 수집과 전처리, 증강

전진희

프로젝트 개요:

이번 프로젝트에서는 블루치즈, 체다치즈, 에멘탈치즈, 까망베르치즈의 4가지 치즈 이미지를 크롤링해서 수집했다. 전처리할 때는 필요없는 이미지를 제거하고, 데이터 증강으로 이미지를 추가했다. 회전, 좌우 반전, 상하 반전, 채도 조절 등 다양한 증강 기법을 활용해 이미지 데이터셋을 다양하게 만들었다. 마지막으로 이미지를 255x255 크기로 리사이즈했다.

데이터 수집 및 전처리

1. 크롤링

```
!pip install selenium==4.2.0
```

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.chrome.service import Service

import time
import os
import urllib, requests
```

```
##### 1. query, chromedriver 실행
query = "blue cheese"
service = Service("./chromedriver")
driver = webdriver.Chrome(service=service)

##### 2. query 검색창에 추가
driver.get("https://www.google.co.kr/imghp?hl=ko")

# 크롬 창 열리면 개발자 도구 열어서 검색창 선택하여 검색창 엘레먼트 셀렉트로 선택한 후
```

```
엘레먼트 창에 선택된 거 copy full Xpath 해주기

keyword =
driver.find_element_by_xpath("/html/body/div[1]/div[3]/form/div[1]/div[1]/div[1]/div/div[2]/textare
a")
keyword.send_keys(query) # 이제 검색창에 검색어기 들어가 있는 게 보인다

### 3. 검색창에 입력어 들어오면 검색 실행
driver.find_element_by_xpath("/html/body/div[1]/div[3]/form/div[1]/div[1]/div[1]/button").click()
# 검색 버튼의 full xpath

driver.implicitly_wait(3)

### 4. 스크롤 자동으로 내리고 더보기 버튼 나오면 클릭 하기
print(f"{query} 스크롤 내리는 중...")
elem = driver.find_element_by_tag_name('body')
for i in range(200):
    elem.send_keys(Keys.PAGE_DOWN)
    time.sleep(0.1)

try:
    driver.find_element_by_class_name('mye4qd').send_keys(Keys.ENTER)
    for i in range(200):
        elem.send_keys(Keys.PAGE_DOWN)
        time.sleep(0.1)
except Exception:
    pass


### 5. 이미지 개수 파악 하기
links = []
images = driver.find_elements_by_css_selector('img.rg_i.Q4LuWd') #  Selenium을 사용하여 웹
페이지에서 CSS 선택자를 사용하여 여러 개의 이미지 요소를 선택하는 코드
for image in images:

    if image.get_attribute('src') != None :
        links.append(image.get_attribute('src'))
    elif image.get_attribute('data-scr') != None :
        links.append(image.get_attribute('data-src'))
```

```
    elif image.get_attribute('data-iurl') != None:
        links.append(image.get_attribute('data-iurl'))

print("찾은 이미지 개수 : ", len(links))


### 6. 이미지 다운로드
count = 0
for i in links :
    start = time.time()
    url = i
    os.makedirs(f"./blue_cheese_img_download/", exist_ok=True)
    while True:
        try:
            urllib.request.urlretrieve(url,
f"./blue_cheese_img_download/{str(count)}_blue_cheese.png")
            print(f"{str(count + 1)} / {str(len(links))} / {query} / 다운로드 시간 : {str(time.time() -
start)[:5]} 초")
            break
        except urllib.error.HTTPError as e:
            print(f"HTTPError 발생 ({e}): 재시도 중...")
            time.sleep(5)
        except Exception as e:
            print(f"Error 발생 ({e}): 재시도 중...")
            time.sleep(5)
        if time.time() - start > 60:
            print(f"{query} 이미지 다운로드 실패")
            break
    count += 1

print(f"{query} 이미지 다운로드 완료")
driver.close()
```

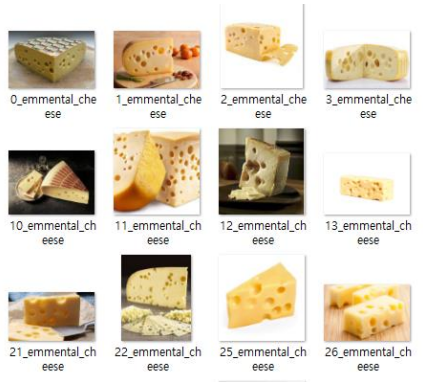
2. 전처리 – 불필요한 데이터 삭제, 결과 예제

전처리 후 데이터 양

Blue cheese image : 263개

Camembert cheese image : 147개

Cheddar cheese image : 123개



Emmental cheese image : 213개

데이터 증강

다양한 증강 기법을 활용하여 이미지를 변형시키기. 주로 사용한 증강 기법은 다음과 같다.

회전: 이미지를 일정 각도로 회전시켜 다양한 각도에서의 이미지를 얻음.
좌우 반전: 이미지를 좌우로 반전시켜 대칭 이미지를 생성함.
상하 반전: 이미지를 상하로 반전시켜 대칭 이미지를 생성함.
채도 조절: 이미지의 채도 값을 조절하여 다른 채도 수준에서의 이미지를 생성함.
이러한 데이터 증강 기법을 활용하여 이미지 데이터셋의 다양성을 높였다.

```
import cv2
import glob
import os
from tqdm import tqdm
import numpy as np
```

```
# 이미지 증강 함수
def image_aug(img):
    # BGR -> RGB
    image = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    # 회전할 각도 설정
    angle = 30

    # 이미지 중심점 기준 회전 행렬 생성
    (h, w) = image.shape[:2]
    center = (w // 2, h // 2)
    M = cv2.getRotationMatrix2D(center, angle, 1.0)

    # 회전 적용
    rotated = cv2.warpAffine(image, M, (w, h))

    # 이미지 좌우반전
    flipped_right_and_left = cv2.flip(image, 1)
    flipped_up_and_down = cv2.flip(image, 0)

    # 채도값 변경하려면 BGR -> HSV
    img_hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

    # 채도 값을 0.8배로 증가시키기
```

```
    saturation_factor = 0.8
    img_hsv[:, :, 1] = img_hsv[:, :, 1] * saturation_factor

    # HSV -> BGR
    img_saturated = cv2.cvtColor(img_hsv, cv2.COLOR_HSV2BGR)

    # BGR -> RGB
    img_saturated = cv2.cvtColor(img_saturated, cv2.COLOR_BGR2RGB)

    # 증강된 이미지 리턴
    return [image, rotated, flipped_right_and_left, flipped_up_and_down, img_saturated]


# 이미지가 저장된 폴더 경로
image_dir = "./cheese/emmental_cheese_img_download/"

# 이미지 파일 경로들 가져오기
image_path = glob.glob(os.path.join(image_dir, "*.png"))

# 증강 처리된 데이터와 원본 데이터 저장할 폴더
os.makedirs("./cheese_aug/emmental_cheese_org_aug_dataset", exist_ok=True)
for path in tqdm(image_path):
    image_name = os.path.splitext(os.path.basename(path))[0]  # 확장자 제외한 파일 이름 추출
    img = cv2.imread(path)

    # 증강된 이미지와 원본 이미지를 저장
    for idx, aug_image in enumerate(image_aug(img)):
        file_name = f"{str(idx).zfill(4)}_{image_name}.png"  # 파일 이름에 확장자 추가
        file_path = os.path.join("./cheese_aug/emmental_cheese_org_aug_dataset", file_name)
        image = cv2.cvtColor(aug_image, cv2.COLOR_RGB2BGR)  # BGR 형식으로 변환하여 저장

        cv2.imwrite(file_path, image)
```

결과 예제

이미지 순서는 원본, 회전, 좌우반전, 상하반전, 채도조절 순으로 되어있다.

blue cheese augs

camembert cheese augs

cheddar cheese augs

Emmental cheese augs

이미지 리사이즈

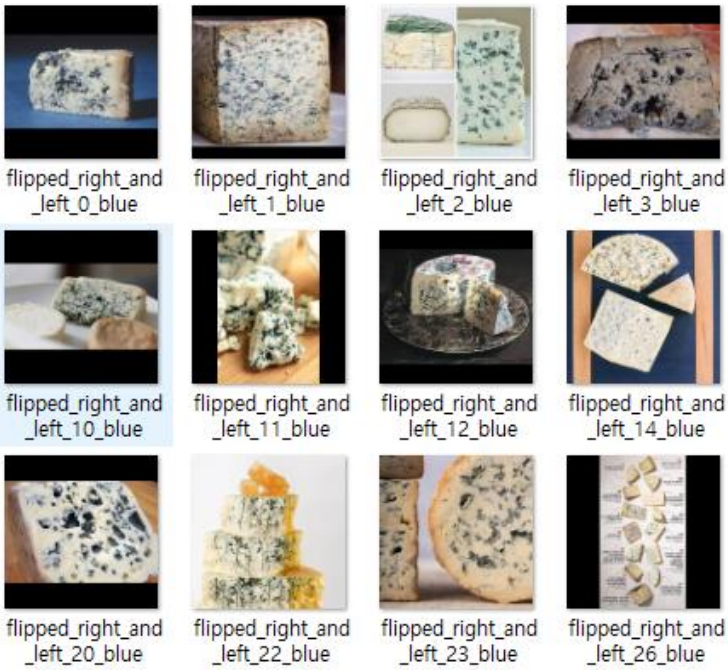
이미지의 비율을 유지하면서 255x255 크기로 리사이즈하는 작업을 수행했다. 이미지 리사이즈 과정에서는 이미지의 비율을 유지하여 왜곡이 없도록 조절하였다. 이를 통해 동일한 크기의 이미지를 얻을 수 있었다.

```
import os
import glob
import cv2
from tqdm import tqdm
from PIL import Image
```

```
# 이미지 전처리 함수
def resize_with_padding(pil_img, new_size, background_color):
    width, height = pil_img.size

    # 정사각형이면 수정할 필요 없음
    if width == height:
        return pil_img

    # 너비가 높이보다 큰 경우 처리하는 코드
    elif width > height:
        # 너비와 같은 크기의 빈 이미지(result)를 생성
        result = Image.new(pil_img.mode, (width, width), background_color)
        # 원본 이미지를 생성한 빈 이미지의 왼쪽 상단에 붙이기
        result.paste(pil_img, (0, (width - height) // 2))
        return result

    # 높이가 너비보다 큰 경우 처리하는 코드
    else:
        result = Image.new(pil_img.mode, (height, height), background_color)
        # 원본 이미지를 생성한 빈 이미지의 상단 중앙에 붙이기
        result.paste(pil_img, ((height - width) // 2, 0))
        return result

# 기존 데이터 증강 처리된 데이터 폴더 경로
img_dir = "./cheese_aug/blue_cheese_org_aug_dataset"
img_path = glob.glob(os.path.join(img_dir, "*.png"))

# 이미지 저장할 폴더 생성
os.makedirs("./cheese_final_dataset/blue_cheese", exist_ok=True)

# 라벨 딕셔너리
label_dict = {"0000":"org",
              "0001": "rotated",
              "0002": "flipped_right_and_left",
              "0003": "flipped_up_and_down",
              "0004": "saturat"}
```

```
# 이미지 처리 및 저장 반복문
for path in tqdm(img_path):
    # 이미지 경로로부터 이미지 이름 추출
    img_name = os.path.basename(path)
    # 이미지 이름을 _로 분할
    img_name_split = img_name.split('_')
    # 라벨 값을 추출하여 적용
    label = label_dict[img_name_split[0]]
    # 새로운 이미지 이름 생성
    new_img_name = f"{label}_{img_name_split[1]}_{img_name_split[2]}"

    # PIL 라이브러리를 사용하여 이미지 열기
    img_pil = Image.open(path)
    # 이미지 전처리 수행
    img_pil_processed = resize_with_padding(img_pil, (255,255), (0,0,0))
    # 이미지 저장
    img_pil_processed.save(f"./cheese_final_dataset/blue_cheese/{new_img_name}.png", "PNG")
```

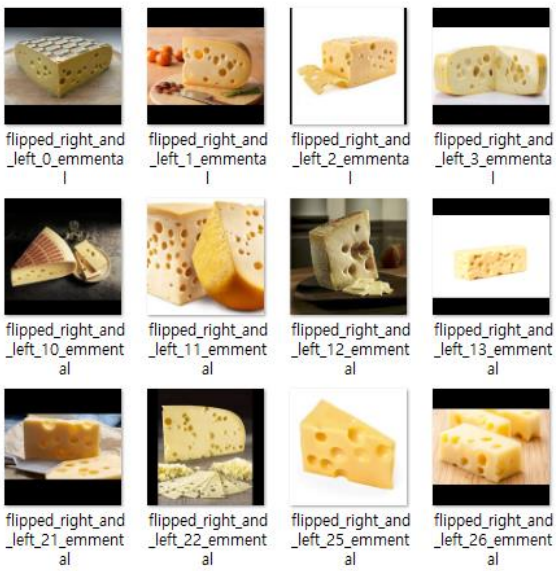
결과 예제



Blue cheese 리사이징

Cheddar cheese 리사이징

Carmembert cheese 리사이징



Emmental cheese 리사이징

최종적으로 Blue cheese는 1315개, Carmembert cheese는 735개, Cheddar cheese는 615개, Emmental cheese는 1065개로 이미지를 증강하였다.

해결해야 할 점 : 데이터 불균형

추후 데이터 개수를 일정 수준에 맞추는 작업이 필요할 수도 있음.