mediapipe를 이용한 ASL 숫자 제스쳐 인식
2024 인공지능시스템설계프로젝트

2024-19448
김민재

2024.12.16

1 팀프로젝트 진행내용

데이터 생성

데이터의 생성은, 웹상에서 검색을 통해 얻은 이미지와, 웹캠을 통해 얻어진 랜드마크 각각을 ndarray타입의 .npy파일로 class별 디렉토리에 저장하여 준비하였다.

Figure 1: 데이터 생성 실행화면

데이터 전처리

데이터에서 의미 없는 정보를 제외하고자 노력했다. 전체 스켈레톤의 규모, 각 랜드마크의 화면 안에서의 절대좌표는 제스쳐 인식에 있어 유효한 정보가 되지 않으므로, 0번 랜드마크(손목)을 원점으로 두고 상대좌표로 변환을 수행하고, 0번과

1번 랜드마크 사이의 길이를 이용해 정규화를 수행해 규모의 차이를 줄였다. 해당 처리는 제스쳐 인식에 있어서도 동일하게 적용하였다.

```
for i in range(len(landmarks)):
    landmarks[i]=landmarks[i]-landmarks[0]
landmarks=landmarks/np.linalg.norm(landmarks[1],axis=0)
```

Figure 2: 데이터 전처리 코드

데이터 증강

학습을 위해 준비한 데이터들은 대부분 정방향으로 향한 오른손이었고, 이는 편향으로 이어질 수 있다고 판단했다. 이를 해결하기 위해 전체 학습 데이터에 대해 회전 및 좌우반전을 여러차례 수행해 데이터의 편향을 줄이고, 모델의 안정성을 높였다.

```
def rotate(landmarks_,angle):#angle: radian
    rotation_matrix_z = np.array([
    [np.cos(angle), -np.sin(angle), 0],
    [np.sin(angle), np.cos(angle), 0],
    [0, 0, 1]
    ])
    for i in range(len(landmarks_)):
        landmarks_[i] = np.dot(rotation_matrix_z, landmarks_[i])

def mirror(landmarks_):
    for i in range(len(landmarks_)):
        landmarks_[i] = landmarks_[i] * np.array([-1, 1, 1])
```

Figure 3: 데이터 증강코드

모델학습

제스쳐 인식 모델은, (21,3)크기의 랜드마크 텐서를 입력받아 출력단의 softmax를 통해 10개 클래스 중 하나로 분류를 수행하는 형태이다. 입력단에는 (63,)로 flatten된 데이터가 입력되고, 너비 128의 완전연결계층, relu를 지나, 다시 너비 64의 완전연결계층 및 relu, 마지막의 softmax로 연결되는 뉴럴넷을 사용하였다. 이때 과적합을 막기위해 중간에 dropout을 사용하였다. 학습 epoch는 120으로 설정하였고, 전체데이터 중 20%를 분리해 생성한 테스트셋에 대해 약 99%의 accuracy를 보였다.

이미지 변화 인식

주어진 영상에서 이미지의 변화를 인식하기 위해 초반에는 각 frame을 흑백으로 변환하고 연속된 프레임의 전체 픽셀간 차이의 절댓값을 합해 임계치와 비교하는 방식을 사용했다. 하지만, 그 한계가 명확했고, 전체 이미지의 색의 평균을 비교하는 방식을 채택했다. 하지만 전체 색의 평균이 유사한 이미지가 많아, 전체 이미지를

4개의 섹션으로 나누고, 각각의 섹션에 대해 평균색의 차이를 구하고 이를 합한 값의 변동의 크기를 임계치와 비교해 이미지의 변화여부를 인식하도록 하였다.

```
h,w,_=frame.shape
c1 = abs(np.sum(cv2.mean(frame[0:h//2,0:w//2]))-np.sum(cv2.mean(previous_frame[0:h//2,0:w//2])))
c2 = abs(np.sum(cv2.mean(frame[0:h//2,w//2:w]))-np.sum(cv2.mean(previous_frame[0:h//2,w//2:w])))
c3 = abs(np.sum(cv2.mean(frame[h//2:h,0:w//2]))-np.sum(cv2.mean(previous_frame[h//2:h,0:w//2])))
c4 = abs(np.sum(cv2.mean(frame[h//2:h,w//2:w]))-np.sum(cv2.mean(previous_frame[h//2:h,w//2:w])))
prev_difference=img_difference
img_difference = c1+c2+c3+c4
```

Figure 4: 섹션별 평균색 차이 합연산 코드

```
if recogTerm==1:
    change_count += 1
    print(f"{change_count}:  {gesture_name}")
if recogTerm>=1:
    recogTerm-=1
elif abs(prev_difference-img_difference)>3.23:
    recogTerm=5
```

Figure 5: 평균색 차이 합의 변동 임계치 비교 코드

제스쳐인식

제스쳐 인식은, mediapipe 모델이 이미지에서 손의 랜드마크를 인식하면 해당 스켈레톤에 대해 상대좌표 변환, 크기 정규화를 수행하고, 학습한 제스쳐 인식 모델을 사용해 제스쳐 분류를 수행하는 방식으로 수행되었다.

2 적용아이디어

적절한 속도로 학습 및 인식을 실행하기 위해서는 주어진 이미지데이터에서 유의미한 정보만을 추출 할 수 있어야 했다. 이미지 전체를 인식하기보다, 스켈레톤 모델을 사용해 손의 각 지점들만 추출해 인식과정을 효율화 하고자 했다. 스켈레톤 모델은 mediapipe의 hands모델을 사용하였다. 이때, 학습 데이터로는 웹상의 이미지 파일과, 웹캠에서 촬영한 데이터를 사용하였다. 데이터의 다양성을 높이기 위해 각 랜드마크 배열별로 회전 및 좌우반전을 여러번 실행하였다. 테스트셋과 학습데이터의 랜드마크의 절대좌표의 차이는 유의미한 정보가 아니므로, 0번째 랜드마크(손목) 을 [0,0,0]으로 설정하고, 그에대한 상대좌표로 21개의 랜드마크의 좌표를 변환하였다. 또한 전체 스켈레톤의 크기도 유의미한 정보가 아니므로, 0번과 1번 랜드마크 사이의 거리를 이용해 전체 랜드마크의 규모를 정규화 하였다. 스켈레톤 모델을 사용해 접근하는 방법론은 새로운 것이 아니나, 데이터에서 유의미한 feature만을 추출하고, 데이터 자체의 편향을 줄이기위한 증강의 수행과정이 본 프로젝트의 유의미한 점이라 생각한다.

3 기존 모델과의 차이점

기존의 이미지 기반 제스쳐 인식과의 차이는, 공개된 스켈레톤 모델을 이용해 전체 모델의 학습과 인식과정을 단순화하고, 효율화를 달성했다는 점을 들 수 있다. 스켈레톤 모델을 사용해 제스쳐 인식을 수행한 것이 새로운 접근방식은 아니나, 상술했듯, 데이터의 전처리와 증강을 통해 편향을 줄이고 중요한 feature만을 남겨, 모델이 일반성을 잃지 않도록 했다는 점이 차별점이라고 할 수 있다.

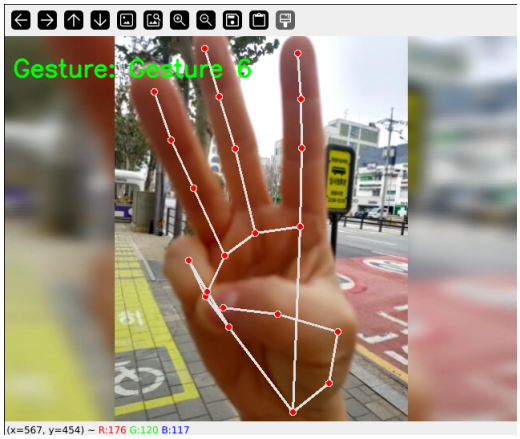

Figure 6: 제스쳐 인식 실행화면

4 팀프로젝트 실행 결과

두개의 모델을 함께 사용한 특성상, 잘못된 인식의 원인은 크게 두가지로 나눌 수 있다. 첫번째는 학습한 제스쳐 분류모델의 오류이다. 이는 학습데이터의 다양성, 특히 3차원 회전각에 대한 다양성을 높이고 다시 학습시키는 방식으로 개선할 수 있었다. 두번째 원인은 mediapipe 모델의 한계이다. mediapipe모델이 주어진 이미지의 랜드마크를 적절히 추출하지 못한다면, 분류모델 또한 적절한 결과를 내놓을 수 없다. 100개의 이미지 중 2개 정도의 이미지의 제스쳐 분류에서 오류를 발견했고, 학습데이터 추가 및 재학습을 수행했으나, 유의미한 개선은 없었다. 따라서 mediapipe의 한계가 잘못된 인식의 오류라고 판단할 수 있었다. 하지만, 현재 주어진 자원에서, mediapipe를 뛰어넘는 스켈레톤 모델을 생성하는것은 한계가 있다고 판단된다. 향후 다른 접근 방식을 사용하거나, 랜드마크의 분류에 대한 다른 전처리 및 인공신경망을 사용한다면 개선의 여지가 있을 것으로 생각한다.

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 5, 10, 1, 9, 2, 8, 3, 7, 6, 4, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 3, 1, 4, 2, 6, 10, 5, 7, 9, 8, 2, 1, 3, 10, 4, 7, 5, 6, 9, 8, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 10, 9, 2, 8, 3, 7, 4, 6, 5, 10, 8, 6, 6, 2, 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, 10, 8, 6, 4, 2, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1,

5 배운 점

인공신경망의 적용에 있어, 모델의 학습과 실행시간은 프로젝트의 수행과 실행에 있어 작지 않은 요소임을 깨달았다. 데이터와 모델의 규모를 줄여서 접근하는 방식을 사용했는데, 처음부터 의도한 것은 아니나, 효율화된 학습시간으로 데이터의 전처리에 대한 여러 시도들을 하는 것이 가능했다. 그리고 이는 상대적으로 나은 결과로 이어졌다고 생각한다. 컴퓨팅 파워가 증가하면서 YOLO등, 전체 프로세스를 하나의 모델로 처리하는 경우가 많은 것 같다고 느끼는데, 때론 다른 기능을 수행하는 여러 모델을 함께 사용하는 것 또한, 디버깅과 효율성 면에서 유의미함을 확인할 수 있었다. 총평은, 기대한 것 이상의 많은 것들을 배울 수 있는 수업이었고, 재미있는 프로젝트였다. 한학기 동안 많은 가르침을 주신 교수님과 조교님께 감사한다.