**Datasets**

PyTorch의 **Dataset** 클래스를 상속받아 세 가지 다른 데이터셋(**Gaze360**, **Mpiigaze**, **NIA2022**)을 처리하는 클래스들을 정의. 각 클래스는 이미지와 레이블을 로드하고 전처리하며, 모델 학습 및 평가에 사용할 수 있는 형태로 데이터를 제공

1. 이 메서드는 데이터 파일을 읽고 각도 제한에 따라 데이터를 필터링합니다.
2. 데이터셋의 길이를 반환 (**\_\_len\_\_** 메서드)
3. 데이터를 읽고, 변환하고, 각도 범위로 양자화된 레이블을 반환 (**\_\_getitem\_\_** 메서드)

**Loader**

비디오 파일에서 프레임을 비동기적으로 읽어오는 **FileVideoStream** 클래스를 정의합니다. 이 클래스는 스레드를 사용하여 프레임을 읽고, 읽은 프레임을 큐에 저장합니다. 이렇게 하면 프레임을 읽는 동안 메인 프로그램의 성능이 향상되고, 비디오 스트림을 실시간으로 처리할 수 있습니다.

 \_\_init\_\_: 비디오 파일 스트림을 초기화하고 큐를 초기화합니다.

 start: 비디오 파일 스트림을 읽는 스레드를 시작합니다.

 update: 파일 스트림에서 프레임을 읽고 큐에 추가합니다.

 read: 큐에서 다음 프레임을 읽습니다.

 running: 여전히 프레임이 큐에 있는지 여부를 확인합니다.

 more: 큐에 여전히 프레임이 있는지 여부를 확인합니다.

 stop: 스레드를 중지하고 스트림 리소스를 해제합니다.

**Model**

이 코드는 L2CS 모델을 정의하는 L2CS 클래스를 보여줍니다. 이 모델은 주어진 이미지에서 yaw와 pitch의 gaze(시선)를 예측하는 데 사용될 수 있습니다.

이 모델의 주요 특징은 다음과 같습니다:

1. **Convolutional Layers**: 모델의 첫 번째 레이어는 7x7 커널 크기를 가진 64개의 필터로 구성된 Convolutional 레이어입니다. 이후에는 각 레이어마다 적절한 수의 레이어와 블록을 생성하는 \_make\_layer 메서드를 사용하여 추가적인 Convolutional 레이어를 만듭니다.
2. **Fully Connected Layers**: 마지막으로, Adaptive Average Pooling을 통해 입력된 이미지의 특징을 압축하고, 그 다음에는 이를 완전히 연결된 레이어(fc\_yaw\_gaze, fc\_pitch\_gaze)에 전달하여 yaw와 pitch의 gaze를 예측합니다.
3. **Activation Functions**: ReLU(Rectified Linear Unit) 함수를 사용하여 각 레이어의 출력을 활성화합니다.
4. **Batch Normalization**: 각 Convolutional 레이어 뒤에는 Batch Normalization 레이어가 있어 네트워크를 안정화시킵니다.
5. **Weight Initialization**: Convolutional 레이어의 가중치는 Xavier 초기화를 사용하여 초기화됩니다.
6. **Forward Pass**: forward 메서드에서는 입력 이미지를 받아서 각 레이어를 통과시키고, 마지막에는 yaw와 pitch의 gaze를 예측합니다.

이 모델은 학습 데이터에서 시선을 얻기 위해 사용될 수 있으며, 이미지나 비디오에서 얼굴의 시선을 감지하거나 추적하는 등의 작업에 활용될 수 있습니다.

**Utils**

이 코드는 L2CS 모델을 훈련하고 평가하는 데 사용되는 여러 유틸리티 함수들을 포함하고 있습니다. 각 함수의 역할은 다음과 같습니다:

1. **atoi(text), natural\_keys(text):** 숫자가 포함된 문자열을 정렬하기 위한 유틸리티 함수입니다. natural\_keys 함수는 숫자가 포함된 문자열을 인식하여 숫자 순서대로 정렬합니다.
2. **gazeto3d(gaze):** 주어진 gaze(시선)를 3D 공간 좌표로 변환합니다.
3. angular(gaze, label): 두 벡터 사이의 각도를 계산합니다.
4. **draw\_gaze(a, b, c, d, image\_in, pitchyaw, thickness=2, color=(255, 255, 0), scale=2.0):** 주어진 이미지에 시선 각도를 그리는 함수입니다.
5. **select\_device(device='', batch\_size=None):** 디바이스를 선택하고 CUDA 디바이스의 정보를 제공하는 함수입니다.
6. **spherical2cartesial(x):** 구면 좌표를 직교 좌표로 변환하는 함수입니다.
7. **compute\_angular\_error(input, target):** 입력과 목표 간의 각도 오차를 계산하는 함수입니다.
8. **softmax\_temperature(tensor, temperature):** 온도가 있는 소프트맥스 함수입니다.
9. **git\_describe(path=Path(\_\_file\_\_).parent):** git 저장소의 설명을 반환하는 함수입니다.

이러한 유틸리티 함수들은 모델의 훈련 및 평가 프로세스에서 다양한 작업을 수행하는 데 사용됩니다. 예를 들어, 시선 각도를 그리거나, 디바이스를 선택하거나, 구면 좌표를 직교 좌표로 변환하는 등의 작업을 수행합니다.

**Run\_valid\_I2cs**

이 **run\_valid\_I2cs.py** 스크립트는 L2CSNet 네트워크를 사용한 시선 추정(gaze estimation)을 수행하는 코드입니다. 이 코드는 주어진 이미지와 레이블 데이터셋에 대해 모델을 평가하고, 각 에폭(epoch)에 대한 평균 절대 오차(MAE)를 계산합니다.:

1. **인수 파싱**: 스크립트가 실행될 때 사용자가 지정한 인수들을 파싱합니다. 이를 통해 데이터셋의 경로, 모델의 스냅샷 경로, GPU 설정 등을 제어할 수 있습니다.
2. **모델 및 데이터셋 설정**: 지정된 아키텍처로부터 모델을 만들고, 데이터셋을 로드합니다. 여기서는 NIA2022 데이터셋을 사용하고 있습니다.
3. **테스트 루프 실행**: 테스트 루프는 모델을 로드하고 각 이미지에 대한 예측을 수행합니다. 예측된 시선은 실제 라벨과 비교되어 오차를 계산합니다. 이 오차는 평균 각도 오차(MAE)로 계산되고 기록됩니다.
4. **결과 기록**: 각 에포크에서 평균 각도 오차를 기록하고, 최종 결과는 로그 파일에 저장됩니다.

이 스크립트를 사용하면 훈련된 모델의 성능을 검증하고, 모델의 개선을 위해 필요한 조치를 취할 수 있습니다.

양식의 맨 위

양식의 맨 아래

**Setup**

‘l2cs’ 패키지를 깔기위한 셋업