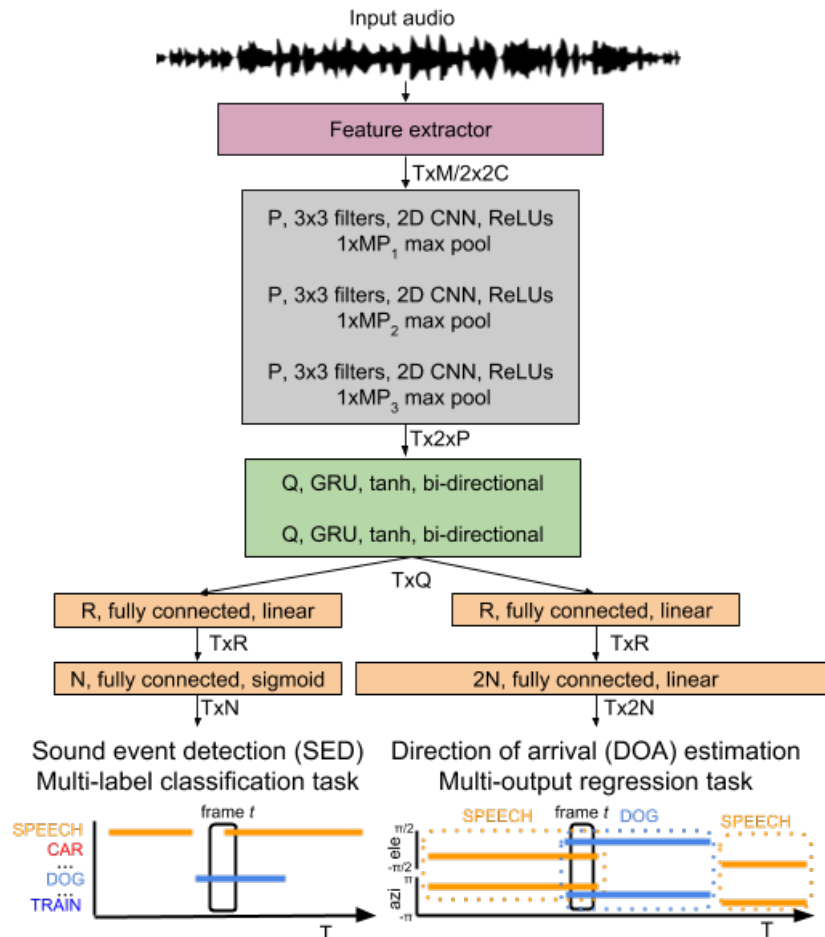
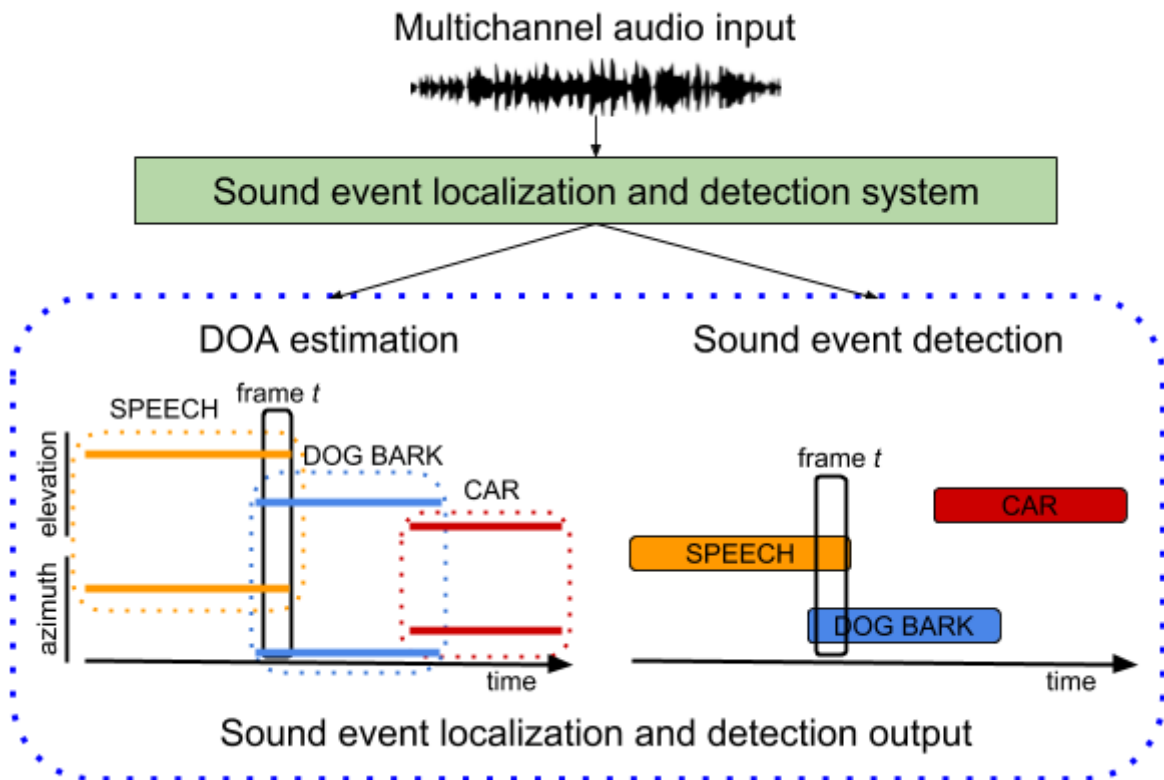




시스템 구조



- 입력 특징으로 FFT의 크기 및 위상 구성 요소, 원-핫 인코딩으로 표시되는 SED 레이블 및 라디안 단위의 방위각 및 앙각으로 표시되는 DOA 사용.
- 일련의 연속 스펙트로그램 프레임을 입력으로 사용하고 각각의 공간 위치와 함께 각 입력 프레임에 대해 활성화된 모든 사운드 이벤트 클래스를 예측하여 각 사운드 이벤트 클래스에 대한 시간 활동 및 DOA 궤적을 생성.
- CRNN(Convolutional Recurrent Neural Network)은 프레임 시퀀스를 두 개의 출력에 병렬로 매핑하는 데 사용.
- 첫 번째 출력에서 사운드 이벤트 감지(SED)는 다중 레이블 다중 클래스 분류 작업으로 수행되어 네트워크가 각 프레임에 대해 여러 사운드 이벤트의 존재를 동시에 추정할 수 있음.
- 두 번째 출력에서 연속 3D 공간의 도착 방향(DOA) 추정치는 다중 출력 회귀 작업으로 얻어지며, 각 사운드 이벤트 클래스는 마이크 주변의 단위 구에서 DOA의 구형 좌표 방위각(azi) 및 고도(ele)를 추정하는 두 개의 회귀자와 연결됨.

Sound event class	SED output	Sound event activity	DOA estimates	
			azi	ele
SPEECH	0.8	Threshold > 0.5	1.4	-0.4
CAR	0.1		2.3	-1.1
...	0.2		0.1	0.2
DOG	0.7		-0.8	1.1
...
TRAIN	0.1		-3.1	0.0


 Sound event active


 Sound event inactive

- 네트워크의 SED 출력은 데이터 세트의 각 사운드 이벤트에 대해 [0 1]의 연속 범위에 있으며 이 값은 그림과 같이 각 사운드 이벤트 활동에 대한 이진 결정을 얻기 위해 임계값이 지정됨.
- 활성화 사운드 이벤트 클래스에 대한 각 DOA 추정치는 공간 위치를 제공.

데이터세트

- 오디오 형식만 다른 동일한 사운드 장면의 두 가지 데이터세트 제공.
- 두 가지 데이터 세트 (Ambisonic, Microphone Array) 모두 개발 및 평가 세트로 구성.
- Development Datasets
 - 48000Hz로 샘플링된 1분 길이의 녹음 400개로 구성되며, 각각 100개의 녹음으로 구성된 4개의 교차 검증 분할로 나뉨.
- Evaluation Datasets
 - 1분 길이의 녹음 100개로 구성.
- 504개의 고유한 방위각-고도-거리 조합에서 5개의 실내 위치에서 수집.
- 사실적인 녹음을 위해 녹음 위치에서 수집된 주변 소음을 합성된 녹음에 추가하여 평균 SNR이 30dB가 되도록 함.
- 데이터 세트에 사용된 11개의 사운드 이벤트 클래스와 해당 인덱스 값.

Sound class	Index
knock	0
drawer	1
clearthroat	2
phone	3
keysDrop	4
speech	5
keyboard	6
pageturn	7
cough	8
doorslam	9
laughter	10

데이터세트 오디오 파일 녹음환경

- Impuse Responses(IR) 수집
 - Eigenmike에서 1m 떨어진 거리에서 10° 증분으로 -40°에서 40°까지 9개의 고도에 대해 10° 방위각마다 36개의 IR로 324개의 개별 Directions-Of-Arrival (DOA) 생성.
 - Eigenmike에서 2m 떨어진 거리에서 10° 증분으로 -20°에서 20°까지 5개의 고도에 대해 10° 방위각마다 36개의 IR이 있어 180개의 개별 DOA 생성.
 - IR은 핀란드 Hervanta에 있는 Tampere 대학 캠퍼스 내부의 5개 실내 위치에서 기록.
 - IR 기록 설정을 변경하지 않고 5개 위치에서 30분간의 주변 소음 기록을 수집.
- 실내 위치
 - Language Center: 여러 개의 좌석 테이블과 카펫 바닥이 있는 넓은 공용 공간. 사람들이 대화하며 일하고 있음.
 - Reaktori Building: 여러 개의 좌석 테이블과 카펫 바닥이 있는 대형 카페테리아. 사람들이 대화하며 음식을 먹고 있음.
 - Festia Building: 단단한 바닥이 있는 높은 천장 복도. 사람들이 돌아다니며 대화하고 있음.
 - Tietotalo Building: 주변에 교실이 있고 단단한 바닥이 있는 복도. 사람들이 돌아다니며 대화하고 있음.
 - Sähköotalo Building: 여러 개의 소파와 테이블이 있는 넓은 복도, 서로 다른 부분의 단단한 카펫 바닥. 사람들이 돌아다니며 대화하고 있음.

데이터세트 오디오 파일명 규칙

- Development Datasets
split[number]_ir[location number]_ov[number of overlapping sound events]_[recording number per split].wav
- Evaluation Datasets
split[number]_[recording number per split].wav

파이썬 스크립트

- python 2.7.10/3.5.3.
- Keras 2.2.2./2.2.4
- batch_feature_extraction.py: 특징, 레이블을 추출하고 지정된 데이터세트에 대한 학습 및 테스트 분할 특징을 정규화하는 독립형 래퍼 스크립트.
- parameter.py: 모든 학습, 모델 및 특징 매개변수로 구성. 사용자가 일부 매개변수를 변경해야 하는 경우 여기에서 고유 ID로 하위 작업을 생성해야 함.
- cls_feature_class.py: 레이블 생성, 특징 추출 및 정규화를 위한 루틴이 있음.
- cls_data_generator.py: 학습을 위해 generator 모드에서 특징 + 레이블 데이터를 제공.
- keras_model.py: SELDnet 아키텍처를 구현.
- evaluation_metrics.py: 사운드 이벤트 감지 평가 모듈의 핵심 메트릭과 DOA 메트릭을 구현.
- seld.py: 네트워크 학습 래퍼 스크립트. SELD 오류(check paper)가 개선되지 않으면 학습이 중지. 데이터 세트와 결과를 분석하는 데 도움이 되는 지원 스크립트도 제공.
- check_dataset_distribution.py: 다양한 구성에서 데이터세트 분포를 시각화.
- Visualize_SELD_output.py: 네트워크 출력 시각화.
- test_SELD_metrics.py: 사용된 다양한 메트릭을 평가하는 테스트 스크립트.
- 네트워크 학습 절차
 - 데이터세트 디렉토리: 예를 들어, Ambisonic 데이터 세트인 경우 'base_folder/'에는 'foa_dev/' 및 'metadata_dev/'라는 두 개의 폴더가 있어야 함.
 - parameter.py 스크립트에서 해당 데이터세트 경로를 업데이트.
 - 위의 예시의 경우는 dataset_dir='base_folder/'로 변경.
 - 모든 특징과 레이블이 덤프되는 동일한 parameter.py 스크립트에서 디렉토리 경로 feat_label_dir을 제공해야 함.
 - batch_feature_extraction.py 스크립트를 실행하여 다운로드한 데이터세트에서 특징을 추출. 먼저 스크립트의 매개변수를 업데이트하고 더 많은 주석이 있는지 Python 파일을 확인. 이제 아래와 같이 스크립트를 실행할 수 있음. 이렇게 하면 정규화된 특징과 레이블이 여기에 덤프됨. 특징 추출은 일회성이므로 이 스크립트는 독립 실행형이며 parameter.py 파일을 사용하지 않음.
python batch_feature_extraction.py
 - python seld.py를 사용하여 기본 매개변수를 사용하여 네트워크 학습 가능.
 - parameter.py 스크립트에서 볼 수 있듯이 if-else 루프에서 고유 식별자 <task-id>를 사용하여 매개 변수를 추가/변경하고 다음과 같이 호출할 수 있음.
 - python seld.py <task-id> <task-id>
<job-id>는 출력 파일 이름(모델, 학습 플롯)에 사용되는 고유 식별자. 이를 위해 임의의 숫자 또는 문자열을 사용할 수 있음.
 - 마이크 어레이 녹음을 위한 Development 세트에 대한 결과를 얻으려면 다음 명령을 실행.
python seld.py 2

- Ambisonic 녹음을 위한 결과를 얻으려면 다음 명령을 실행.
`python seld.py 4`
- 기본적으로 코드는 `quick_test = True` 모드에서 실행됨. 단, 2개의 미니 배치에서 2개의 epoch 동안 네트워크를 훈련시킴.
- 코드를 성공적으로 실행하게 되면 `parameter.py` 스크립트에서 `quick_test = False`를 설정하고 전체 데이터에 대해 학습해야 함.
- 본 코드는 학습 곡선, 중간 결과를 플로팅하고 `parameter.py` 파일에서 사용자가 제공한 `model_dir` 경로에 모델을 저장함.
- 네트워크의 출력을 시각화하기 위해 `dcase_output=True`로 설정하고 `dcase_dir` 디렉토리를 제공. 그러면 `misc_files/visualize_SELD_output.py` 스크립트를 사용하여 개별적으로 시각화할 수 있는 디렉토리에 파일별 결과가 덤프됨.
- 마지막으로, `compute_SELD_metrics.py` 스크립트를 사용하여 4개의 폴드에 걸친 평균 Development 데이터셋 점수를 얻을 수 있음. 위의 파일별 결과를 덤프한 디렉토리와 참조 메타데이터 폴더를 제공해야 함.