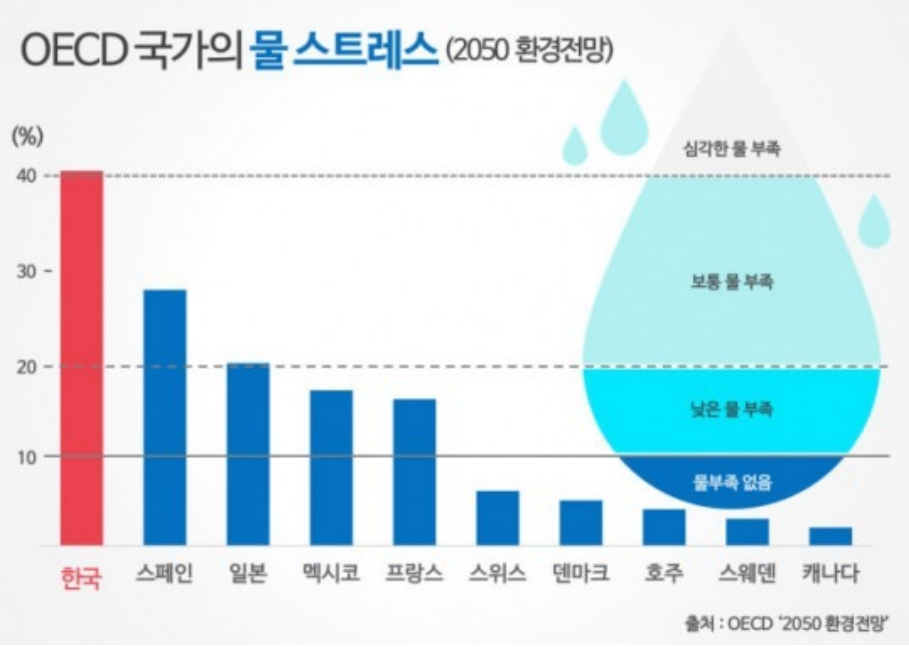
상수도 배관 누수탐지를 위한 AI 모델 개발

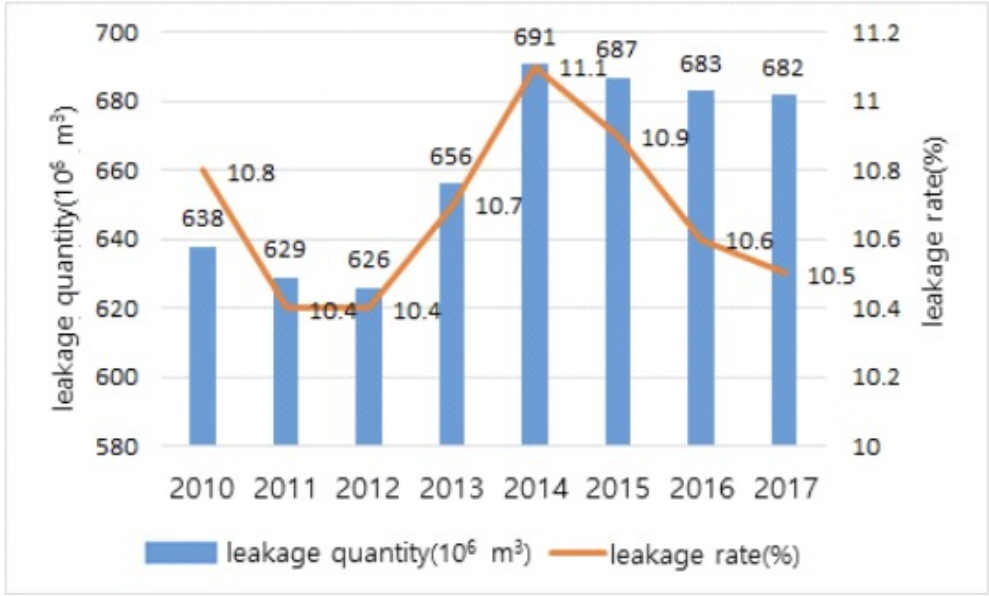
성 균 관 대 학 교
기계공학연구현장실습
이 성 윤
2023.05.29

Table of contents

연구배경 (1)

- 경제협력개발기구(OECD) 환경전망 2050 보고서는 대한민국은 2025년에 '물 기근 국가'를 거쳐, 2050년에는 물 부족지수가 가장 높은 국가가 될 것으로 예상.
- 한국 상수도관을 통해 물이 전달되는 중 상당수가 누수되며, 2012년 한 해 동안 6억2600만 톤(5100억 원)의 수돗물이 중간에 샌 것으로 집계됨.
- 환경부의 통계에 따르면 2017년 기준 국내 연간 수돗물 생산량 65억m^3 으로 전체공급량의 10.5%를 차지하고 있어 환경적, 경제적 손실이 발생하고 있음.

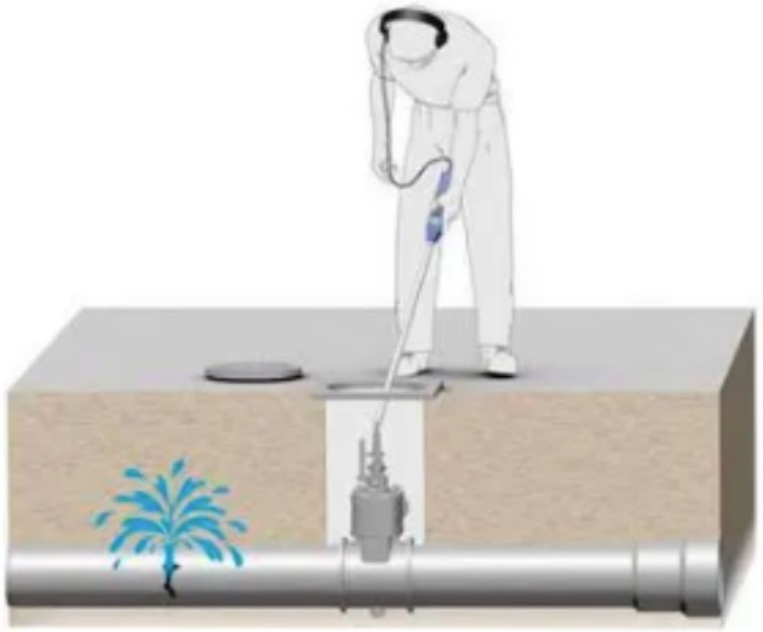




연구배경 (2)

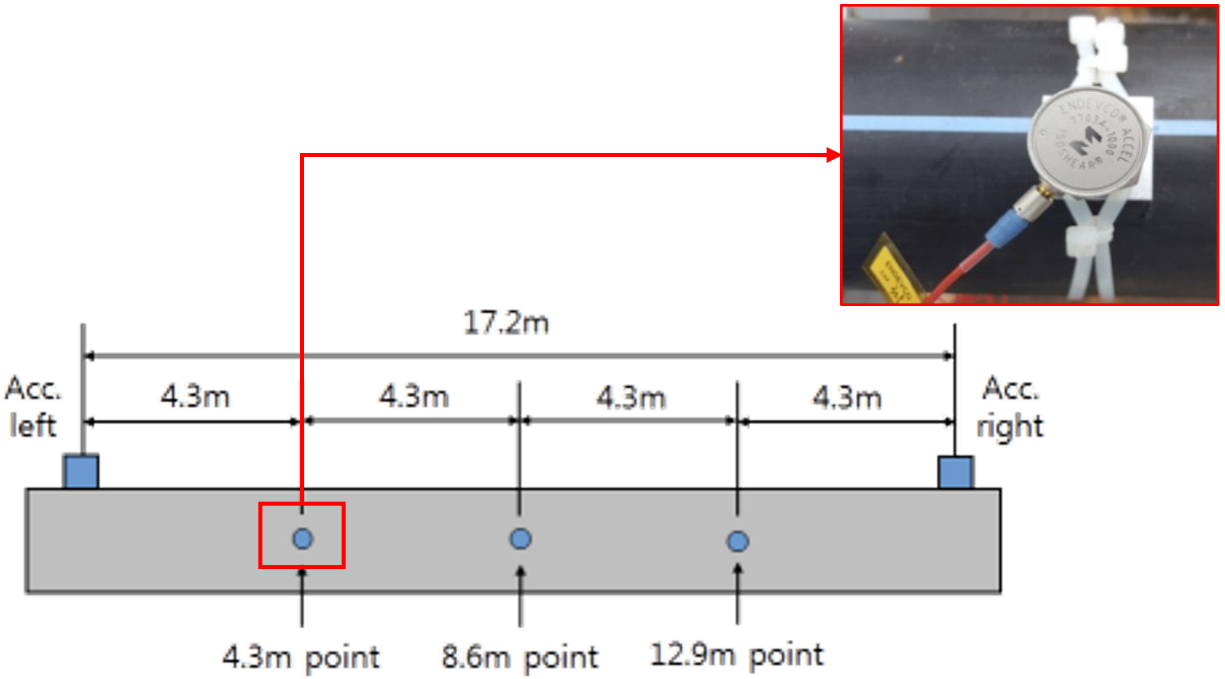
기존 누수탐지 방법

- 상수도 배관은 땅속에 매설되어 있으며, 누수탐지사가 귀로 듣고 배관의 누수 여부를 탐지하는 기술이 주로 사용됨.
- 배관의 누수가 발생할 경우, 기존의 방법으로는 신속하게 대응하지 못함.
- 정확도가 낮고 전문인력 양성하는 비용이 크다.



선행연구

- 배관 외벽에 가속도계를 부착하여, 유체-배관 연성 진동수를 측정하여 누수여부를 평가하는 기술.
- 가속도계를 따로 마련해야하며, 관리해야하는 상수도 배관 수를모두 포괄하기 어려움.



수도배관 음향 데이터를 이용한 AI 기반 누수 탐지 시스템 개발

- 음향 데이터 전처리
  - ✓ 이미지 분류 모델 사용을 위한 음향데이터 이미지화

- 데이터 학습 및 성능평가
  - ✓ Kfold 방법을 이용한 이미지 분류 모델 성능 평가

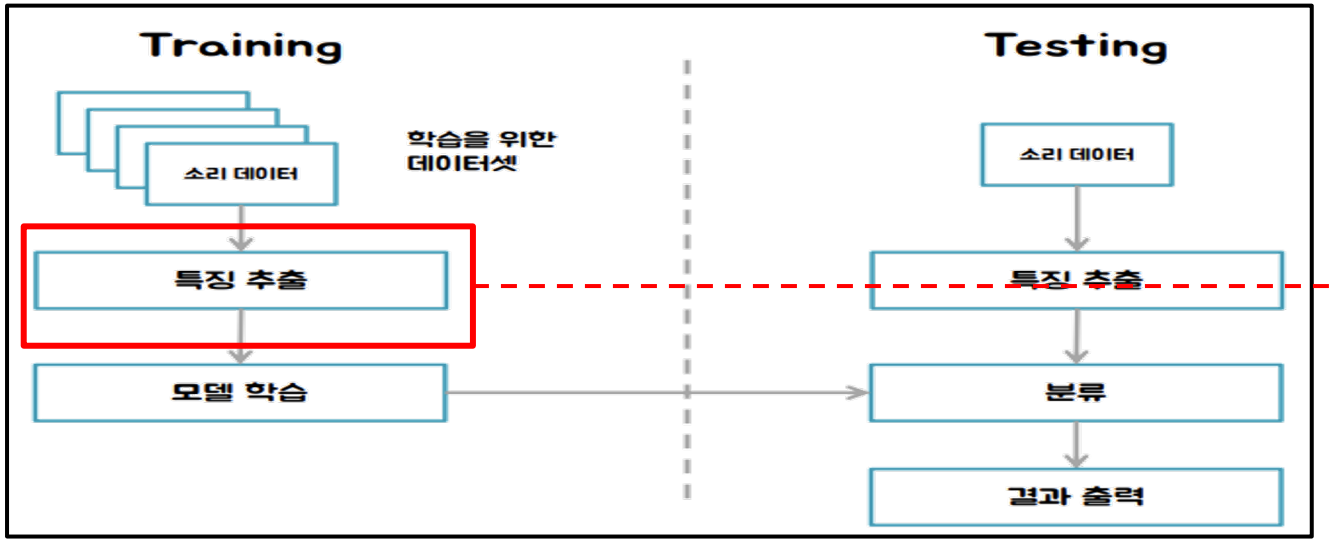
연구내용 (데이터 분석 프로세스)

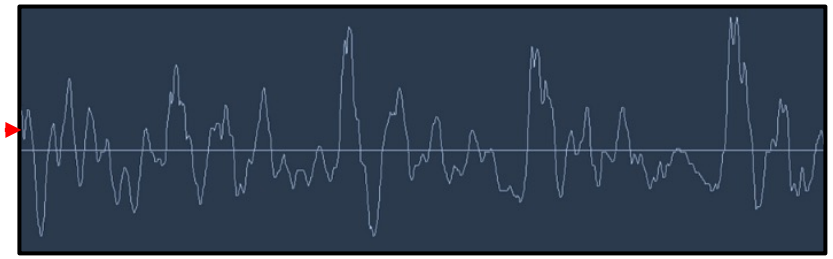
문제 정의 → 데이터 수집 → 데이터 전처리 → 모델링 → 해석 및 시각화

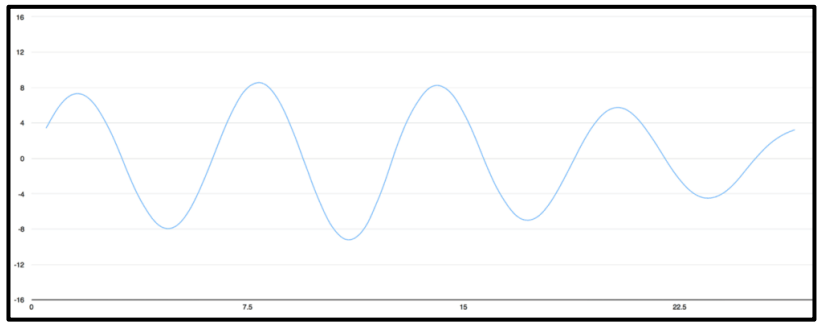
- 수도배관에서 녹음한 음향 데이터를 이용하여 배관 누수 여부를 탐지.
- 한국수자원공사에서 제공된 음성데이터(경상남도 거제시) 정상데이터 4195개, 누수데이터 1450개 사용.

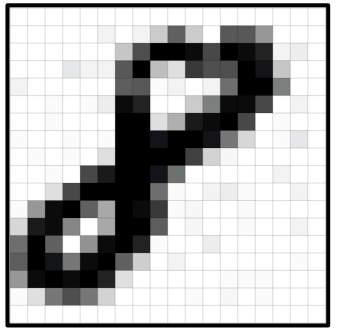
연구내용 (음성데이터 전처리)

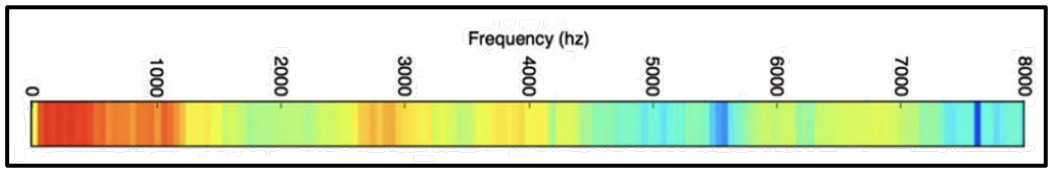
- 음향 데이터 원본 파일에 대한 정보
  - ✓ 파일 포맷 : .wav
  - ✓ 샘플링 타임: 22050HZ
  - ✓ 음원 길이 3~7초
- 전체 원본데이터의 길이 정규화
  - ✓ 80% 원본데이터가 5초 이하의 길이를 가진다.
  - ✓ 모든 데이터의 길이를 5초로 정규화
    - 5 이상의 데이터 : 0~5초 데이터만 자른다
    - 5초 이하의 데이터: 길이가 5초가 되도록 반복한다.







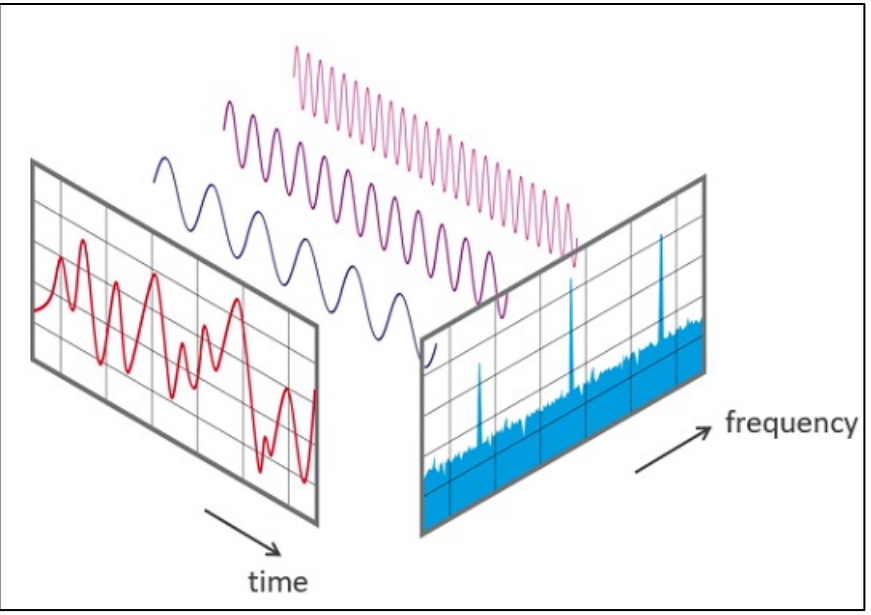




연구내용 (음성데이터 전처리)

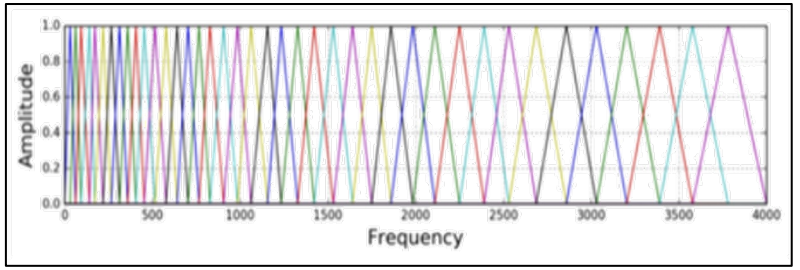
Fourier Transform

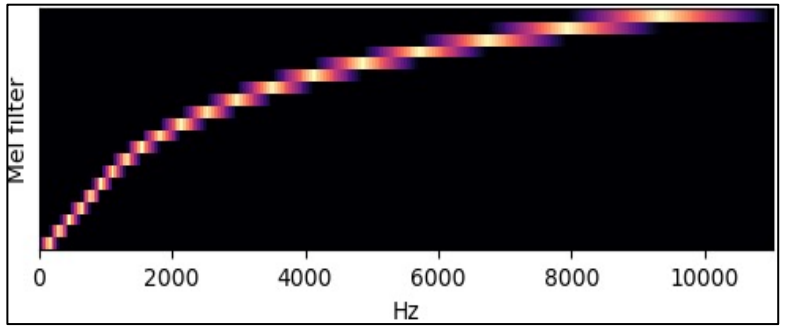
- 소리의 파형으로 표현되는 음향 데이터를 주파수별 세기 데이터로 변환
  - ✓ Fast Fourier Transform : 고속푸리에 변환을 사용하여 진폭데이터로부터 주파수 영역으로 변환
  - ✓ Short-Time Fourier Transform : 일정한 시간 간격으로 고속푸리에 변환을 적용하여 주파수 영역대의 변화를 표현



Mel Spectrogram

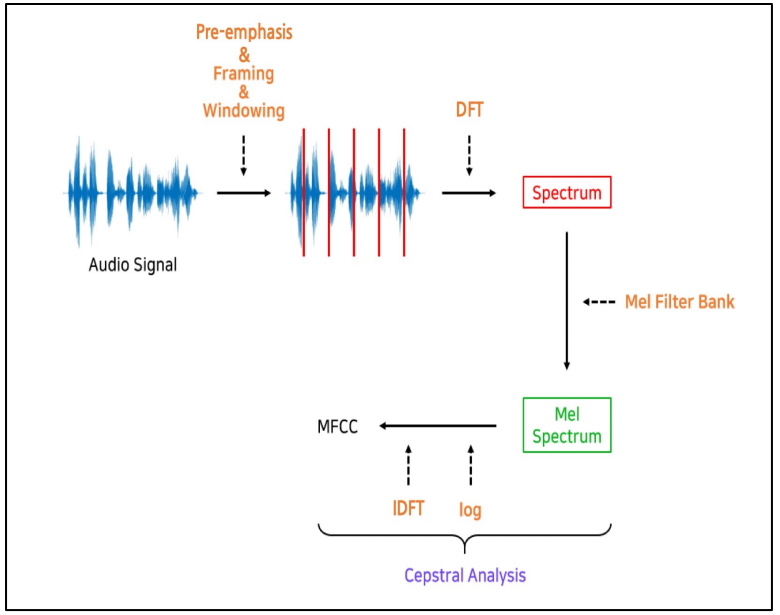
- Short-Time Fourier Transform을 거친 데이터는 주파수별 세기를 나타내는 시계열 데이터
- 사람이 듣는 가청 주파수 대역을 강조하는 MEL 필터를 적용할 수 있다.





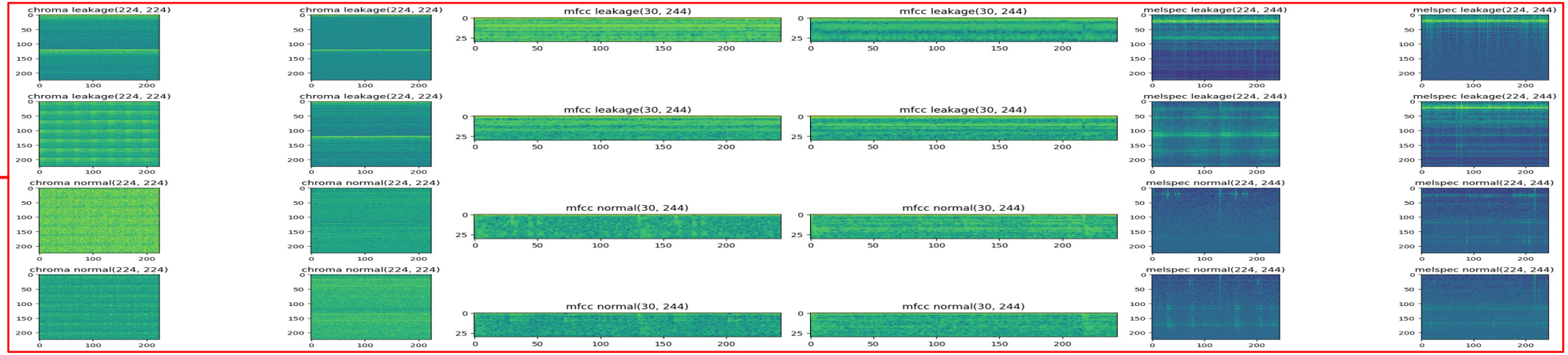
MFCC

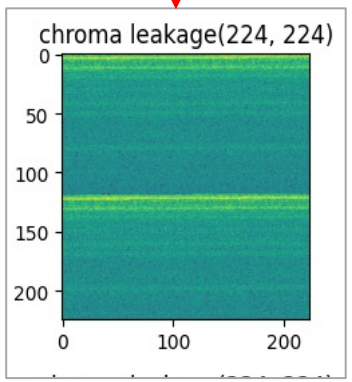
- Mel-Spectrogram에 Discrete Cosine Transform (DCT)를 적용하여 음향의 특성에 대해 행렬을 압축하는 알고리즘
- 주로 음성인식을 위한 음성 데이터 분석에 사용된다.

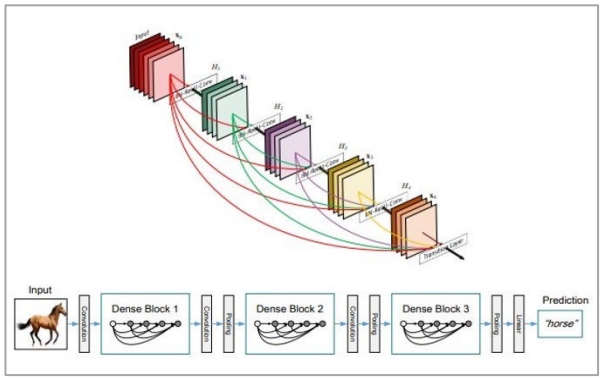


연구내용 (학습모델 설계)

- 이진 분류 모델의 Input 값으로 사용하기 위해 음원데이터를 이미지화
- 이미지 이진 분류 모델 설계
  - ✓ 이미지) ➔ DenseNet121 ➔ 임베딩 레이어 ➔ 주성분 추출(PCA) ➔ 이진 분류(SVM)

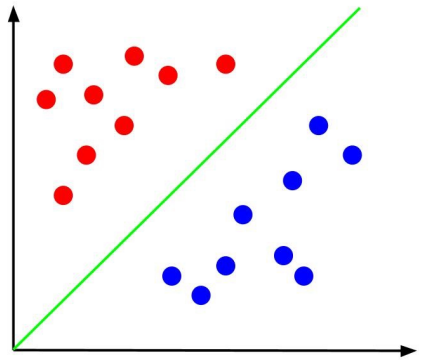






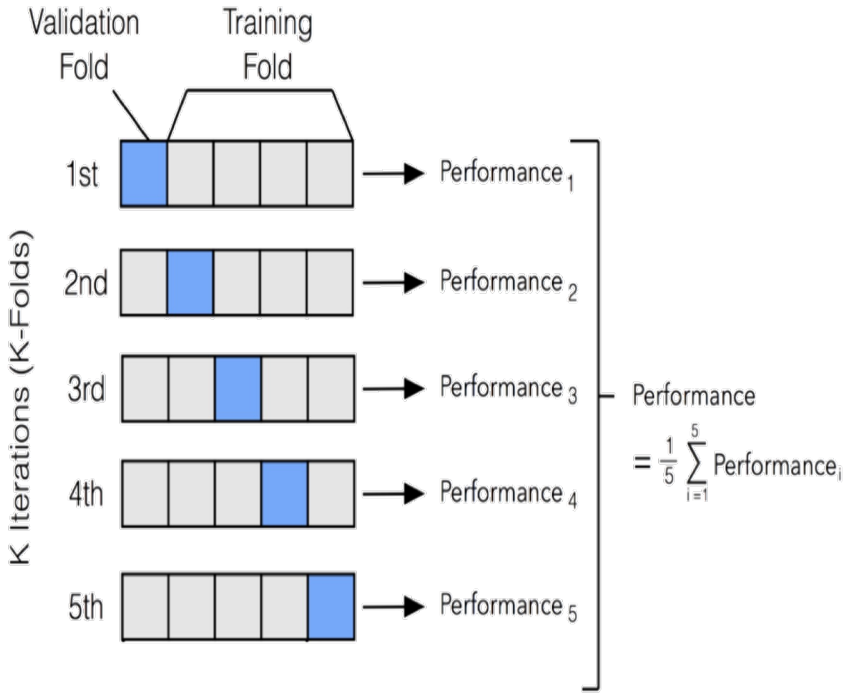
Embedding

PCA



결과 (전처리 방법에 따른 성능 분석)

- 각 Fold 에는 1600개의 학습데이터와 400개의 테스트 데이터가 포함
- 학습 및 테스트 데이터 셋 구축 (Kfold 교차 검증)
  - ✓ 전체 데이터셋을 5개의 폴드로 나누어 각 폴드마다 4개의 폴드를 학습에 사용하고 남은 1개의 폴드를 테스트에 활용
  - ✓ 데이터 불균형을 처리하기 위해 각 폴드에는 같은 수의 정상데이터와 비정상데이터를 사용



	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
Spectrogram	83.9%	94.5%	93.6%	62.8%	93.5%	85.7%
Mel	80.8%	66.8%	66.5%	66.8%	79.2%	72.0%
MFCC	67.8%	65.0%	69.3%	69.0%	71.5%	68.3%
FFT+1dCNN	76.6%	80%	84.5%	85.2%	82%	81.55%

결론 및 기대효과

- **결과분석**
  - ✓ 전문인력이 직접 청진 장비를 이용해 누수를 판별할 경우 80% 수준이며 Spectrogram 모델의 경우 **85%**로, 보다 높은 정확도 구축.
  - ✓ 현장의 녹음 환경에 따라 잡음의 비율이 높을 경우, 정확도가 떨어지는 한계점이 있으나, 필터를 통해 극복.
  - ✓ 현재 모델의 누수 탐지 정확도를 더 높이기 위해서는 더 많은 수의 학습데이터가 필요할 것이라고 판단됨.
  - ✓ 전처리 방법에 따라 성능 차이가 크게 나타나며, spectrogram 과 MFCC를 비교했을때 **17%** 차이가 발생함.
- **기대효과**
  - ✓ 누수가 발생하는 배관에 대해 신속하게 보수작업을 할 수 있는 기반이 될 AI 모델을 개발함.
  - ✓ 전문인력의 판단 오류를 방지하고, 수월한 업무 보조역할로 활용될 것으로 기대됨.

참고문헌

- Heon-cheol Oh et al, 2003, ‘A Feasibility Study on the Detection of Water Leakage using a Ground-Penetrating Radar’, Korean Institute of Electromagnetic Engineering and Science Journal
- Rangsarit Vanijjirattikhan(2022). AI-based acoustic leak detection in water distribution systems. Results in Engineering

부록

- Parameters
  - ✓ stft
    - n_fft=512
    - hop_length=512
  - ✓ Mel
    - n_mels=224
  - ✓ MFCC
    - n_mfcc=32
  - ✓ Epochs=150

```
for i, (train_index, test_index) in enumerate(folds):
  print('fold:',str(i))

  train = np.array([combine[i] for i in train_index])
  test = np.array([combine[i] for i in test_index])
  train_label = np.array([label[i] for i in train_index])
  test_label = np.array([label[i] for i in test_index])
  print('학습 데이터:',len(train),'테스트 데이터:',len(test))
  result = train_and_test(train,train_label,test,test_label)
  print(result)
```

```
fold: 0
학습 데이터: 1600 테스트 데이터: 400
Epoch 1/150
50/50 [==============================] - 1s 12ms/step - loss: 0.5875 - accuracy: 0.7069 - val_loss: 0.5141 - val_accuracy: 0.7525
Epoch 2/150
50/50 [==============================] - 0s 8ms/step - loss: 0.5170 - accuracy: 0.7681 - val_loss: 0.4841 - val_accuracy: 0.7650
Epoch 3/150
50/50 [==============================] - 0s 8ms/step - loss: 0.5015 - accuracy: 0.7769 - val_loss: 0.4751 - val_accuracy: 0.7750
Epoch 4/150
50/50 [==============================] - 0s 9ms/step - loss: 0.4921 - accuracy: 0.7756 - val_loss: 0.4804 - val_accuracy: 0.7650
Epoch 5/150
50/50 [==============================] - 1s 12ms/step - loss: 0.4780 - accuracy: 0.7862 - val_loss: 0.4683 - val_accuracy: 0.7750
Epoch 6/150
50/50 [==============================] - 1s 12ms/step - loss: 0.4689 - accuracy: 0.7894 - val_loss: 0.4729 - val_accuracy: 0.7550
Epoch 7/150
50/50 [==============================] - 1s 12ms/step - loss: 0.4649 - accuracy: 0.7944 - val_loss: 0.4869 - val_accuracy: 0.7750
Epoch 8/150
50/50 [==============================] - 1s 14ms/step - loss: 0.4630 - accuracy: 0.7950 - val_loss: 0.4645 - val_accuracy: 0.7675
Epoch 9/150
50/50 [==============================] - 1s 11ms/step - loss: 0.4551 - accuracy: 0.7994 - val_loss: 0.4662 - val_accuracy: 0.7725
Epoch 10/150
50/50 [==============================] - 1s 12ms/step - loss: 0.4513 - accuracy: 0.8025 - val_loss: 0.4655 - val_accuracy: 0.7650
Epoch 11/150
```

```
50/50 [==============================] - 0s 8ms/step - loss: 0.2785 - accuracy: 0.8900 - val_loss: 0.5261 - val_accuracy: 0.7625
13/13 [==============================] - 0s 3ms/step - loss: 0.5261 - accuracy: 0.7625
Test loss: 0.5261
Test accuracy: 0.7625
None
```