**Water Leak Detection AI Model Performance Report**

# 1. Introduction

AI model for water leak detection using sound spectrograms.

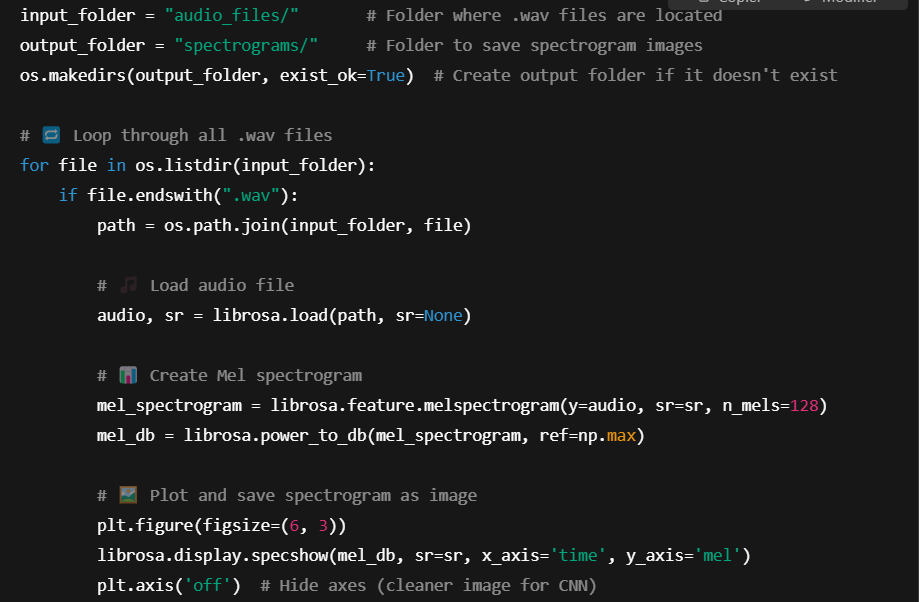
🔗 **Data Sources**

1. **Mendeley Dataset**
   * [Acoustic Leak Detection Dataset – Mendeley](https://data.mendeley.com/datasets/hkn8mxcjyz/1)
   * Contains labeled audio recordings (.wav format) of **"leak"** and **"no leak"** scenarios from various pipeline conditions.
2. **YouTube Audio Scraping**
   * Collected additional real-world leakage sounds by **scraping YouTube videos** using Python scripts

Exemple :

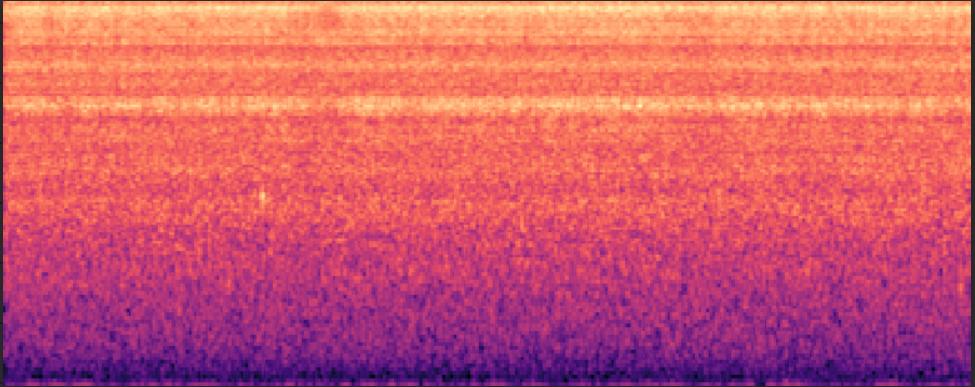
* ffmpeg -i "Water Leak Sound ｜ Leaking Pipe ｜ 1 HOUR [1YaQMq1NbnQ].wav" -f segment -segment\_time 10 -c copy "C:\Users\Lenovo-Thinkpad\Desktop\sonor\output\_%03d.wav"

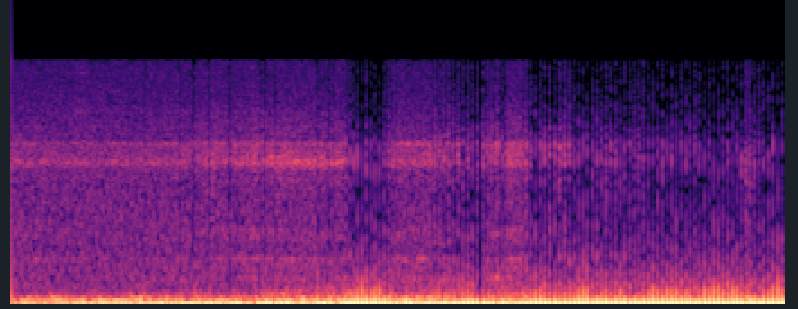
->his helps increase **diversity** and **realism**

**Process of changing .wav files into spectrograms :**

*Library Librosa*  : is a **Python library** used for **audio processing and analysis**. it provides easy-to-use tools for **loading, modifying, and analyzing audio data**.

Exemple of a spectrogram





**2. Model Architecture & Training**

The model follows a deep learning pipeline, including data preprocessing, class balancing, and enhanced regularization techniques.

## 2.1 Model Architecture

The model is a **4-layer CNN** with batch normalization, dropout layers, and max-pooling for effective feature extraction. It utilizes the **Focal Loss** function to address class imbalance and **AdamW** optimizer with Cosine Annealing LR scheduling.

**Model Summary:**

* **Conv Layers:** 4 convolutional layers with **ReLU** activation.
* **Dropout:** Incremental dropout (0.2 -> 0.5) to reduce overfitting.
* **FC Layers:** Two fully connected layers (384 neurons and output layer with 2 neurons).
* **Loss Function:** Focal Loss (handles class imbalance).
* **Optimizer**: AdamW with weight decay (L2 regularization).
* **Learning Rate Scheduler:** Cosine Annealing with Warm Restarts.
* **Regularization Techniques:** Dropout, weight decay, batch normalization.

## 2.2 Training & Validation Strategy

* **Train/Validation/Test Split:** 80% train, 10% validation, 10% test.
* **Balanced Training Set:** The dataset is balanced using oversampling
* **Early Stopping:** Stops training when validation loss stops improving.

**3. Model Performance Metrics**

**Train Loss:** 0.0057 **| Train Accuracy:** 97.38%

**Val Loss:** 0.0041 | **Val Accuracy:** 98.69%

# Test Loss: 0.0039 | Test Accuracy: 98.70%

**Observations:** The test accuracy is slightly higher than the training accuracy, indicating **no overfitting.**

**3.2 Classification Report (Test Set)**

**|** Class **|** Precision **|** Recall **|** F1-Score **|** Support **|**

# |------------|-----------|--------|----------|---------|

**|** No Leak  **|** 1.00  **|** 0.95 **|** 0.98 **|** 44 **|**

**|** Leak **|** 0.98 **|** 1.00 **|** 0.99 **|** 110 **|**

**|** Overall Accuracy **|** 98.70% **| | |** 154 **|**

**3.3 Confusion Matrix**

**[[ 42 2]**

**[ 0 110]]**

|  | **Predicted: No Leak** | **Predicted: Leak** |
| --- | --- | --- |
| **Actual: No Leak** | 42 (**True Negatives**) | 2 (**False Positives**) |
| **Actual: Leak** | 0 (**False Negatives**) | 110 (**True Positives**) |

# 4. Overfitting Analysis .

Indicators of overfitting include:

* **High training accuracy but low-test accuracy** -> **Not observed** in this model.
* **Large gap between train and validation loss** -> **Not observed.**

## 4.1 Regularization Techniques Used to Prevent Overfitting

**Dropout**: Applied progressively (0.2 -> 0.5) to prevent neuron dependency.

**Batch Normalization**: Normalizes activations to improve generalization.

**Weight Decay (L2 Regularization):** Reduces model complexity.

**Early Stopping**: Stops training when validation loss stops improving.

**Balanced Training**: Addresses class imbalance.

# 5. Conclusion & Final Verdict

* The model has high accuracy (98.70%) on the test set.
* Precision, recall, and F1-score are near perfect, proving robustness.
* There is no overfitting, as evidenced by minimal train-test accuracy gap.
* Regularization techniques were effectively applied, ensuring generalization.