In this study, the RF and DT models are implemented using scikit-learn tools. The CIS algorithm and ordinary search algorithm are implemented using MATLAB R2014a. Table 1 shows the optimized results for leakage detection models.

Table 1 Optimized results for leakage detection models

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Data | Model | Parameters | LPC | LSP | Time or frequency domain features |
| Raw  data | RF | *n\_estimators* | 1000 | 400 | 300 |
| *max\_features* | ‘auto’ | ‘auto’ | ‘auto’ |
| *criterion* | ‘entropy’ | ‘entropy’ | ‘entropy’ |
| DT | *max\_depth* | 50 | 50 | 50 |
| *min\_samples\_split* | 8 | 2 | 2 |
| *min\_samples\_leaf* | 2 | 1 | 1 |
| *criterion* | ‘entropy’ | ‘entropy’ | ‘entropy’ |
| -5 dB  data | RF | *n\_estimators* | 1000 | 300 | 300 |
| *max\_features* | ‘auto’ | ‘auto’ | ‘auto’ |
| *criterion* | ‘entropy’ | ‘entropy’ | ‘entropy’ |
| DT | *max\_depth* | 50 | 50 | 50 |
| *min\_samples\_split* | 2 | 2 | 8 |
| *min\_samples\_leaf* | 1 | 1 | 1 |
| *criterion* | ‘entropy’ | ‘entropy’ | ‘entropy’ |

The file named 'LPC.py' is related to LCP-based detection models using raw datasets.

The file named 'LPC\_noise.py' is related to LCP-based detection models using -5dB datasets. The file named 'LPC.py' is related to LPC-based detection models using raw datasets.

The file named 'LPC\_noise.py' is related to LPC-based detection models using -5dB datasets.

The file named 'Time or frequency features.py' is related to time or frequency domain features-based detection models using raw datasets.

The file named 'Time or frequency features\_noise.py' is related to time or frequency domain features-based detection models using -5dB datasets.

The file named 'OS\_algorithm.m' is related to the ordinary search algorithm in experiment 1.

The file named ' CIS\_algorithm.m' is related to the CIS algorithm in experiment 1.

Algorithm I is the CIS localization algorithm proposed in this paper. Algorithm II is the ordinary search algorithm proposed by Kang et al. (2018).

|  |  |
| --- | --- |
| **Algorithm I: Cubic interpolation search localization** | |
| 1 | **Input:** leakage signals |
| 2 | *, .* |
| 3 | **Output**: leak location |
| 4 | **for** *i* =1: *M* **do** |
| 5 | = // *M* is the number of nodes in pipe networks.  // is the distance from node *i* to measurement points. |
| 6 | **End** |
| 7 | = select node with the minimum . |
| 8 | = select the pipeline connected to node as a new search space, node is the start node, and node *k* is the end node. |
| 9 | **if (**<, , **)** |
| 10 | **return**  = |
| 11 | **if (**<, , **)** |
| 12 | **return**  = |
| 13 | **if (**<, , **)** |
| 14 | **while (** or **)** |
| 15 |  |
| 16 |  |
| 17 |  |
| 18 |  |
| 19 | **if ()** |
| 20 | **return**  = |
| 21 | **if ()** |
| 22 | **return**  = |
| 23 | **return**  = |
| 24 | **//** , , , are the distances from corresponds nodes  // to the start node, and is the search accuracy. |
| **Algorithm II: Ordinary search localization** | |
| 1 | **Input:** leakage signals |
| 2 | *, .* |
| 3 | **Output**: leak location |
| 4-8 | This is the same as Algorithm I. |
| 9 |  |
| 10 | // node *m* is the middle point between node and node . |
| 11 | **while ()** |
| 12 | **if (** and **)**  **//** *n* is the number of iterations |
| 13 | **return** , , , |
| 14 | = |
| 15 | **if (** and **)** |
| 16 | **return** , , |
| 17 | = |
| 18 | **if (** and **)** |
| 19 | **return** , |
| 20 | = = |
| 21 | **//** , , , are the distances from corresponds nodes  // to the start node, and is the search accuracy. |